lwIP for CrossCore Embedded Studio 1.0.1
User's Guide

Revision 1.1, October 2012

Part Number
82-100119-01

Analog Devices, Inc.
One Technology Way
Norwood, Mass. 02062-9106

Copyright Information

Printed in the USA.

Disclaimer

Analog Devices, Inc. reserves the right to change this product without prior notice. Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use; nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under the patent rights of Analog Devices, Inc.

Trademark and Service Mark Notice

CONTENTS

PREFACE

ABOUT THE LWIP ADD-IN

PREPARING FOR LWIP APPLICATIONS

CONFIGURING LWIP APPLICATIONS

LWIP APPLICATION CONFIGURATION EDITOR

CONFIGURING LWIP LIBRARIES

LWIP LIBRARY CONFIGURATION EDITOR

PREFACE

Thank you for purchasing the lwIP (light-weight Internet Protocol) for CrossCore® Embedded Studio (CCES) 1.0.1 add-in, the Analog Devices implementation of the open-source TCP/IP stack for embedded platforms.

Purpose

The *lwIP for CrossCore Embedded Studio 1.0.1 User's Guide* describes how to use the lwIP add-in software.

Intended Audience

The primary audience for this manual is a programmer who is familiar with Analog Devices processors and TCP/IP protocol suite. This manual assumes that the audience has a working knowledge of the appropriate processor architecture and instruction set. Programmers who are unfamiliar with Analog Devices processors can use this manual, but should supplement it with other texts (such as the appropriate hardware reference and programming reference manuals) that describe your target architecture.

Manual Contents

The manual consists of:

What's New in this Manual

This is the second revision of the *lwIP for CrossCore Embedded Studio 1.0.1 User's Guide.*

Technical Support

You can reach Analog Devices processors and DSP technical support in the following ways:

- Post your questions in the processors and DSP support community at EngineerZone®:
  `http://ez.analog.com/community/dsp`
- Submit your questions to technical support directly at:
  `http://www.analog.com/support`
- E-mail your questions about processors, DSPs, and tools development software from **CrossCore Embedded Studio** or **VisualDSP++®**:

Choose **Help > Email Support**. This creates an e-mail to processor.tools.support@analog.com and automatically attaches your **CrossCore Embedded Studio** or **VisualDSP++** version information and license.dat file.

- E-mail your questions about processors and processor applications to:
  processor.support@analog.com or
  processor.china@analog.com (Greater China support)
- In the **USA only**, call **1-800-ANALOGD** (1-800-262-5643)
- Contact your Analog Devices sales office or authorized distributor. Locate one at:
  www.analog.com/adi-sales
- Send questions by mail to:
  Processors and DSP Technical Support
  Analog Devices, Inc.
  Three Technology Way
  P.O. Box 9106
  Norwood, MA 02062-9106
  USA

Product Information

Product information can be obtained from the Analog Devices Web site and CrossCore Embedded Studio online help system.

Analog Devices Web Site

The Analog Devices Web site, www.analog.com, provides information about a broad range of products—analog integrated circuits, amplifiers, converters, and digital signal processors.

To access a complete technical library for each processor family, go to http://www.analog.com/processors/technical_library. The manuals selection opens a list of current manuals related to the product as well as a link to the previous revisions of the manuals. When locating your manual title, note a possible errata check mark next to the title that leads to the current correction report against the manual.

Also note, MyAnalog.com is a free feature of the Analog Devices Web site that allows customization of a Web page to display only the latest information about products you are interested in. You can choose to receive weekly e-mail notifications containing updates to the Web pages that meet your interests, including documentation errata against all manuals. MyAnalog.com provides access to books, application notes, data sheets, code examples, and more.

Visit MyAnalog.com to sign up. If you are a registered user, just log on. Your user name is your e-mail address.

EngineerZone

EngineerZone is a technical support forum from Analog Devices. It allows you direct access to ADI technical support engineers. You can search FAQs and technical information to get quick answers to your embedded processing and DSP design questions.

Use EngineerZone to connect with other DSP developers who face similar design challenges. You can also use this open forum to share knowledge and collaborate with the ADI support team and your peers. Visit http://ez.analog.com to sign up.

Notation Conventions

Text conventions used in this manual are identified and described as follows.

Example	Description
Close command (**File** menu)	Titles in bold style reference sections indicate the location of an item within the CrossCore Embedded Studio environment's menu system (for example, the **Close** command appears on the **File** menu).
`{this | that}`	Alternative required items in syntax descriptions appear within curly brackets and separated by vertical bars; read the example as `this` or `that`. One or the other is required.
`[this | that]`	Optional items in syntax descriptions appear within brackets and separated by vertical bars; read the example as an optional `this` or `that`.
`[this,…]`	Optional item lists in syntax descriptions appear within brackets delimited by commas and terminated with an ellipsis; read the example as an optional comma-separated list of `this`.
`.SECTION`	Commands, directives, keywords, and feature names are in text with `letter gothic` font.
filename	Non-keyword placeholders appear in text with italic style format.
ⓘ	**Note:** For correct operation, ... A Note provides supplementary information on a related topic. In the online version of this book, the word **Note** appears instead of this symbol.
⚡	**Caution:** Incorrect device operation may result if ... **Caution:** Device damage may result if ... A Caution identifies conditions or inappropriate usage of the product that could lead to undesirable results or product damage. In the online version of this book, the word **Caution** appears instead of this symbol.
🛇	**Warning:** Injury to device users may result if ... A Warning identifies conditions or inappropriate usage of the product that could lead to conditions that are potentially hazardous for devices users. In the online version of this book, the word **Warning** appears instead of this symbol.

ABOUT THE LWIP ADD-IN

The lwIP Lightweight TCP/IP Stack for CrossCore Embedded Studio add-in implements lwIP (light-weight Internet Protocol) for Analog Devices' family of Blackfin embedded processors. The ported stack uses a standard Ethernet device driver interface, which allows drivers to interface with different Ethernet controllers. All of the drivers adhere to Analog Devices' driver model and use CrossCore Embedded Studio's System Services Libraries (SSL).

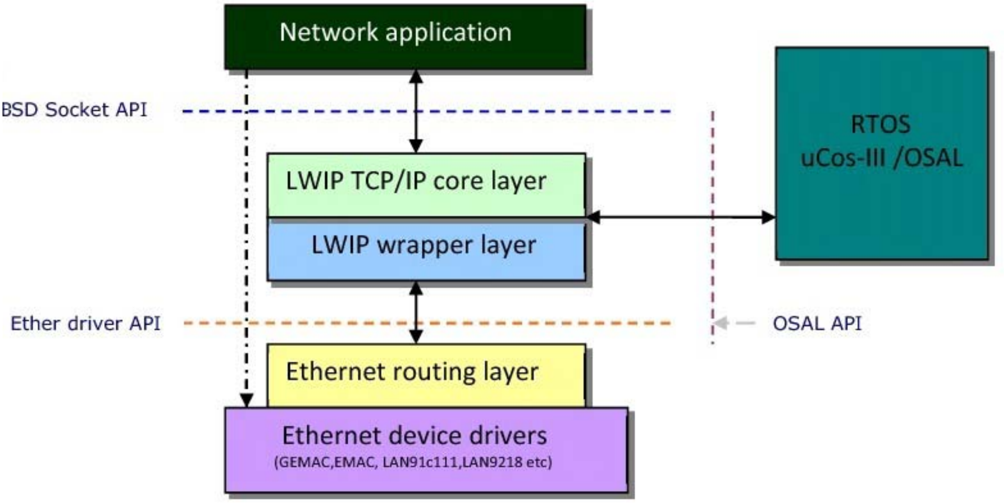
lwIP supports most of the standard protocols in the TCP/IP suite and applications conforming to the Berkeley-alike socket (BSD) interface. Supported protocols include:

- Internet Protocol (IP)
- Internet Control Message Protocol (ICMP)
- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)
- Dynamic Host Configuration Protocol (DHCP)
- Address Resolution Protocol (ARP)
- Berkeley-alike Socket API

Note: For more information on the lwIP stack, see `http://www.sics.se/~adam/lwIP/`. The stack sources are maintained at `http://savannah.nongnu.org/cgi-bin/viewcvs/lwIP/lwIP/`.

lwIP Add-in Architecture

The following is a simplified view of lwIP add-in architecture.



The lwIP add-in software comprises the following:

- Component added to the system configuration of a CrossCore project. (Note that a distinct project is implemented for each of the cores of a dual-core processor.)
- Template code for developing network applications.
- Support package that includes networking examples and online documentation.

Note: On-chip Ethernet device drivers are installed with the CrossCore Embedded Studio framework. Off-chip Ethernet drivers are distributed with products' Extender card packages. If an lwIP application uses an off-chip or Extender card, the associated board support software must be installed. See Adding Board Support for details.

Note: The lwIP TCP/IP stack with a socket-level interface requires an operating system. The lwIP add-in uses the Micrium-based **µC/OS-III Real-Time Kernel** add-in for CrossCore Embedded Studio. See the online documentation for the µC/OS-III add-in for details.

lwIP Core and Wrapper Layers

The lwIP component added to the system configuration of a CrossCore project consists of the:

- Core TCP/IP protocol layer
- Low-level wrapper layer

Core Layer

The lwIP core layer supports TCP/IP protocols. The lwIP core is distributed as a library with default protocol options. Its project directory is *lwipInstallationPath*\lwip\blackfin\lib\lwip\contrib\ports\blackfin\projects\lwipv4lib.

Note: Advanced users can configure protocol options through library plug-in software.

Wrapper Layer

The lwIP wrapper layer is the interface between the lwIP core and Ethernet device drivers. Its project directory is *lwipInstallationPath*\lwip\blackfin\lib\lwip\contrib\ports\blackfin\projects\lwIPwrapperlib.

The wrapper layer provides the functionality for initializing and starting the stack. As part of the initialization process, the wrapper supplies buffers to the Ethernet driver. The stack callback handler is part of the wrapper function, which receives frames from the underlying Ethernet driver and queues it for further processing by the stack. Similarly, the stack-supplied frames are transmitted through the Ethernet driver.

The wrapper exports initialization APIs that an application calls to set up the lwIP subsystem. Exported functions include:

- int **adi_lwip_Initstack** (const unsigned int *buffer_length*, char **buffer_start*);
- int **adi_lwip_Startstack** (void);
- void **adi_lwip_Stopstack** (void);
- int **adi_lwip_Setdrvhandles** (int *num_services*, ADI_ETHER_HANDLE **pdd_handles*);
- void **adi_lwip_Stackcallback** (void **arg1*, unsigned int *event*, void* *pack_list*);

lwIP Template Code

The lwIP templates facilitate development of network application by providing code that initializes and starts the lwIP subsystem. The following template code files are added to a CrossCore project along with the lwIP component.

File	Description
lwip_sysboot_task.c	Provides core routines and the lwIP system boot task for starting the lwIP subsystem.
lwip_sysboot_task.h	Provides global configuration settings for the lwIP subsystem and exported functions.
lwip_app.c	Provides application-specific configuration settings that can also be controlled through the lwIP Application Configuration editor.
softswitch_cfg.c	Provides processor-specific switch configuration routines.

The lwIP system boot task is responsible for opening and configuring the Ethernet drivers, and initialization, and starting the network stack. The `adi_lwip_Init()` function, which creates the task, is called by the `adi_init_Components()` function.

Note: When the lwIP component is added to the system configuration of a project, the **System Configuration** utility automatically adds the call to `adi_lwip_Init()` to `adi_init_Components()`, which is typically called by the project's `main()` function. However, to start **µC/OS-III**, you must manually add the call to `OSStart()` to the `main()` function.

Upon successful initialization, the lwIP system boot task prints the IP address obtained by default from the Dynamic Host Configuration Protocol (DHCP) server.

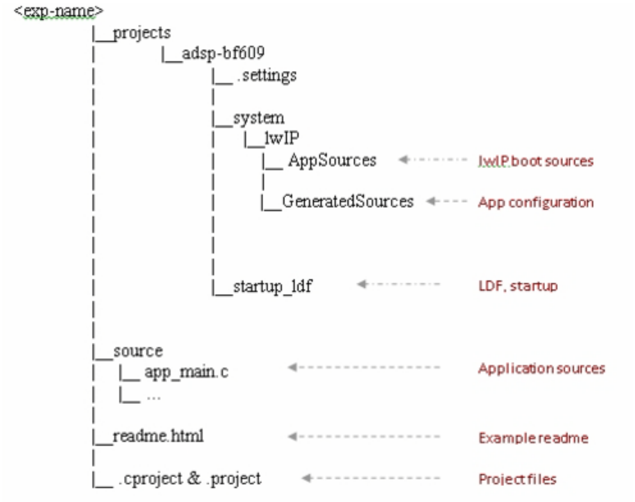
Note: If the default is not changed, you must connect hardware to a network that has an available DHCP server by configuring a static IP address.

The lwIP system boot task also posts the `g_semLWIPBootComplete` semaphore. Other network-dependent application boot tasks can pend on this semaphore for the network boot to complete. All tasks pending on this boot semaphore are released. In case of a network connection problem, the lwIP system boot task pends on a semaphore and periodically checks for a network link.

lwIP Examples

The lwIP examples are distributed with the add-in and installed in directory `lwipInstallationPath\lwip\Blackfin\examples`. The examples:

- Have the common directory structure shown in the following figure.
- Use a pre-built demo version of the Micrium µC/OS-III libraries, for which the µC/OS-III add-in must be installed.



Examples include the following; refer to the `readme.html` file provided with each example for details. You can open any example using the Example Browser in the CrossCore Embedded Studio IDE, or by manual import.

Note: The readmes explain how to convert an example to use Micrium µC/OS-III add-in and sources instead of pre-built demo libraries.

Example	Description
inetd	Demonstrates connections between three server ports and a client. The server ports include: character generator (port #19); echo (port #9); and discard (port #7). Upon startup, the application starts the three servers and waits for the connections between the named ports and the client. The servers spawn a task for handling each client connection. The application uses TCP sockets and calls TCP socket APIs.
dns_client	Demonstrates Domain Name System (DNS) client support. The DNS client running on a Blackfin processor obtains the IP address of domain name `www.analog.com`. The example uses UDP sockets.
fileserver	Demonstrates Windows host support for enabling a standard C/C++ file input/output from the Blackfin processor over a TCP/IP connection. The application consists of a Windows MFC-based host program and a Blackfin fileserver program, and uses TCP sockets.
multicast	Demonstrates how one node can send data to many destinations by making a single call on the multicast transport service. The multicast application consists of a Windows MFC-based host program, multicast receiver, and a multicast sender. The multicast sender is a Blackfin processor-based application that sends out a `Hello World` message to the multicast IP address `225.0.0.37` at port `12345`. The Windows application (multicast receiver) receives the message from the same multicast IP address, and runs and receives the message simultaneously on multiple machines.

Ethernet Device Drivers

Every Ethernet controller has a hardware-specific device driver. On-chip device drivers are distributed with CrossCore Embedded Studio software. Off-chip and EZ-Extender-based Ethernet controller drivers are distributed with the model-specific board support packages identified in Supported Platforms and Dependencies.

lwIP product requires additional board support packages or BSPs. These are distributed as separate installable modules. Typically off-chip ethernet drivers are distributed via a board support package. On-chip Ethernet drivers are distributed which CrossCore Embedded Studio so they do not require a BSP for Ethernet.

All Ethernet drivers interact with the lwIP subsystem through a standard driver interface. A driver exports the entry-point that lwIP subsystem uses to access its functionality. During system initialization, applications open the underlying device driver and supply the handle to the lwIP subsystem. The Ethernet driver entry-point includes the following functions, most of which are used by lwIP subsystems.

Note: Drivers can export additional functions for applications to use. Such functions are accessed directly by applications without interfacing through the entry-point.

Note: Typical applications call at least the `adi_ether_Open`, `adi_ether_GetLinkStatus`, and `adi_ether_EnableMAC` functions.

Function	Description
adi_ether_Open	Opens the Ethernet driver.
adi_ether_Read	Supplies receive buffers to the driver.
adi_ether_Write	Transmits data in the supplied buffer.
adi_ether_Close	Closes the device driver.

adi_ether_GetLinkStatus	Returns the network link status.
adi_ether_AddMulticastFilter	Enables multicast for the given group address.
adi_ether_DelMulticastFilter	Disables multicast for the given group address.
adi_ether_GetBufferPrefix	Returns the buffer prefix of underlying driver.
adi_ether_GetMACAddress	Returns the MAC address.
adi_ether_SetMACAddress	Sets the MAC address.
adi_ether_EnableMAC	Enables the MAC (starts the Ethernet driver).

Supported Platforms and Dependencies

The following table identifies platforms supported by the lwIP add-in and their dependencies.

Processor	**EZ-KIT Lite/EZ-Extender Boards**	**Software Dependencies**
ADSP-BF609	ADSP-BF609 EZ-Board	Micrium µC/OS-III 1.0.0 or later CCES 1.0.0 or later
ADSP-BF537	ADSP-BF537 EZ-KIT Lite	Micrium µC/OS-III 1.0.0 or later CCES 1.0.1 or later
ADSP-BF527	ADSP-BF527 EZ-KIT Lite	Micrium µC/OS-III 1.0.0 or later CCES 1.0.1 or later
ADSP-BF518	ADSP-BF518 EZ-Board	Micrium µC/OS-III 1.0.0 or later CCES 1.0.1 or later
ADSP-BF526	ADSP-BF526 EZ-Board	Micrium µC/OS-III 1.0.0 or later CCES 1.0.1 or later
ADSP-BF533	ADSP-BF533 EZ-KIT Lite Blackfin USB-LAN EZ-Extender	Micrium µC/OS-III 1.0.0 or later CCES 1.0.1 or later Blackfin USB-LAN EZ-Extender Board Support Package 1.0.0 or later

ADSP-BF561	ADSP-BF561 EZ-KIT Lite Blackfin USB-LAN EZ-Extender	Micrium µC/OS-III 1.0.0 or later CCES 1.0.1 or later Blackfin USB-LAN EZ-Extender Board Support Package 1.0.0 or later
ADSP-BF548	ADSP-BF548 EZ-KIT Lite	Micrium µC/OS-III 1.0.0 or later CCES 1.0.1 or later ADSP-BF548 Board Support Package 1.0.0 or later

PREPARING FOR LWIP APPLICATIONS

To prepare for configuring an application of the lwIP add-in in a CrossCore project:

1. It is assumed that:
   - The `main()` function exists in the project.
     Note: This can be accomplished by selecting **Add a source file with a default main() function** on the **Template Code** page of the **New CrossCore Project** wizard.
   - System configuration of the project includes the **µC/OS-III** add-in.
     Note: The operating system must be started by calling the `OSStart()` function from the `main()` function, and this call must be added manually.
   - System configuration of the project includes the **Startup Code/LDF** add-in.
   - lwIP Lightweight TCP/IP Stack for CrossCore Embedded Studio add-in software is installed on the computer.
2. Add the lwIP component to the system configuration of the project.
3. Modify lwIP template code for the project.
4. Configure the project linker description for lwIP.
5. Enable processor-specific Ethernet pins in the project.
6. Add board support to the project.

Adding the lwIP Component to Projects

To add the lwIP component to the system configuration of a CrossCore project:

1. In a navigation view, double-click the `system.svc` file of the project.
   The System Configuration utility appears.
2. Click **Add**.
   The **Add New Components** dialog box appears.
3. From the **Middleware** branch of the tree control, select **Analog Devices' lwIP TCP/IP Stack**.
   Note: If necessary, also select **µC/OS-III for Blackfin**.
4. Click **Next** and **Finish**; see "*Adding an Add-in to a System Configuration*" in the CrossCore Embedded Studio help for details.
   The lwIP component, lwIP template code, lwIP libraries, and include paths required for network applications are added to the project.

Note: When a project is being created, you can select **Analog Devices' lwIP TCP/IP Stack** and **µC/OS-III for Blackfin**on the **Add-in Selection** page of the **New CrossCore Project** wizard. See "*Creating a CrossCore Executable Project*" or "*Creating a CrossCore Static Library Project*" in the CrossCore Embedded Studio help for details.

Modifying lwIP Template Code

To modify configuration parameters in lwIP template code in support of project-specific requirements:

1. In `lwip_sysboot_task.c`, modify the core routines and lwIP system boot task for starting the lwIP subsystem.
2. In `lwip_sysboot_task.h`, modify global configuration settings for the lwIP subsystem and exported functions. The critical configuration settings are:

   ```
   /*! LWIP task stack size */
   #define APP_OS_CFG_LWIP_TASK_STK_SIZE
   (2048)

   /*! LWIP task priority */
   #define APP_OS_CFG_LWIP_TASK_PRIO (6)

   /*! Number of receive DMA descriptors */
   #define EMAC_NUM_RECV_DESC (10)

   /*! Number of transmit DMA descriptors */
   #define EMAC_NUM_XMIT_DESC (10)
   ```

   Note: For DMA-based Ethernet drivers, an application can increase the number of supplied DMA descriptors to the driver by changing the `EMAC_NUM_RECV_DESC` and `EMAC_NUM_XMIT_DESC` macros.

3. (Optional) In `softswitch_cfg.c`, modify processor-specific switch configuration routines.
4. (Optional) Modify application-specific network configuration settings through the xxx lwIP user interface.

Note: Refer to the CrossCore Embedded Studio help for information about editing source files in the IDE.

Configuring the Linker Description File for lwIP

To configure the linker description for a CrossCore project that implements an lwIP application:

1. It is assumed that the **Startup Code/LDF** component is added to the project.
2. In a navigation view, double-click the `system.svc` file of the project.
   The **System Configuration** utility appears.
3. Select the **Startup Code/LDF** tab.
   The **Startup Code/LDF** page appears.
4. In the navigation pane on the left side of the tab, click **LDF**.
   The **LDF** configuration page appears.
5. Select the **Use external memory** (SDRAM) check box.
6. Select the **Customize the system heap** check box.
7. In **Custom system heap memory type**, select **L3 external memory** (SDRAM).
8. In **Custom system heap size**, select 4 **MegaBytes Minimum**.
9. Choose **File > Save**.
   The IDE regenerates the linker description file for the project as required for lwIP applications.

Enabling Ethernet Pins

lwIP applications must enable processor-specific Ethernet peripheral pins. The Pin Multiplexing add-in can easily generate required code.

Note: If you do not use the Pin Multiplexing add-in, you must manually:

- Create `pinmux_config.c`, which contains muxing code, in the \\projectName\system\pinmux\GeneratedSources folder of the project.
- Call `adi_initpinmux()` from `adi_initComponents()`, which is called at system startup.

Note: Custom boards may require changes or additions to default pinmmux configurations and generated sources.

To generate pinmux sources for lwIP applications:

1. If necessary, add the Pin Multiplexing add-in to the system configuration of the CrossCore project.
2. Select the required signal for the lwIP application.
   The following selections apply to Analog Devices' supported EZ-KIT Lites and EZ-Boards:
   - ADSP-BF537, ADSP-BF527, ADSP-BF518, ADSP-BF526 -- MAC[MII/RMII (MAC) Ethernet
   - ADSP-BF609 -- ETH0 (EMAC Module) and ETH (EMAC module)

Adding Board Support

To add board support to an lwIP application:

1. If the application uses an off-chip driver, install the required board support software.
   Note: Board support for on-chip drivers is distributed and installed with CrossCore Embedded Studio software.
2. Add the board support package to the system configuration of the CrossCore project.

CONFIGURING LWIP APPLICATIONS

To configure an lwIP application for a CrossCore project:

1. It is assumed that the project is prepared for configuring lwIP.
2. In a navigation view, double-click the `system.svc` file of the project.
   The System Configuration utility appears.
3. Choose the **lwIP** tab.
   The lwIP Application Configuration page appears.
4. To configure the default network:
   1. Select **Network 0** and click **Edit**.
      The Editing Network dialog box appears.
   2. (Optional) Select or clear the **DHCP** check box to control whether the Dynamic Host Configuration Protocol is enabled.
   3. (Optional) If **DHCP** is disabled, configure static addressing by modifying any of **IP address**, **Subnet mask**, or **Gateway**.
   4. (Optional) Modify the value of **Mac address** (**hex**).
   5. (Optional) In **Number of Ethernet driver receive buffers** and **Size** (**bytes**), modify the number or size of receive buffers.

    6. (Optional) In **Number of Ethernet driver transmit buffers** and **Size (bytes)**, modify the number or size of transmit buffers.
    7. Click **OK**.
       The dialog box disappears, and the properties of **Network 0** are updated.
5. (Optional) To add a second network:
    1. Click **New**.
       The Editing Network dialog box appears.
    2. (Optional) Modify field values as in step 4.
    3. Click **OK**.
       Network 1 is added to the application configuration.
6. Choose **File > Save**.
7. Build and run the network application.

LWIP APPLICATION CONFIGURATION EDITOR

The **lwIP Application Configuration** editor provides the graphical interface for configuring network and driver configurations for an lwIP applications. For example, using the editor you can select dynamic addressing or configure a static IP address for an lwIP application. Properties of Ethernet driver receive or transmit buffers can also be configured.

Note: If you increase the number or sizes of driver receive or transmit buffers from the default settings, then the `ETHER_STACK_SIZE` macro defined in lwip_sysboot_task.c must be changed accordingly.

The **lwIP Application Configuration** editor consists of the lwIP Application Configuration page and Editing Network dialog box.

lwIP Application Configuration Page

Use the **lwIP Application Configuration** page to configure an lwIP network stack for a Blackfin project.

To access the **lwIP Application Configuration** page, attach the lwIP add-in to the system configuration of a project and choose the **lwIP** tab.

The following table describes the **lwIP Application Configuration** page

Item	**Description**	**Default**
Name	Names of the one or two lwIP networks configured for the project.	Network 0
New	If only one network is configured, opens the Editing Network dialog box from which to add a second network to the project.	N/A
Edit	Opens the Editing Network dialog box from which to modify the configuration of the selected network.	N/A
Delete	Removes the selected network from the project configuration. **Note:** If Network 0 is removed, Network 1 becomes Network 0.	N/A
Properties	Provides the values of the following parameters for the selected network: Using DHCP; MAC address; PLI receive buffers; and PLI transmit buffers.	Same as for the Editing Network dialog box

Editing Network Dialog Box

Use the **Editing Network** dialog box to configure lwIP networks.

To access the **Editing Network** dialog box, click **New** or **Edit** from the lwIP Application Configuration page.

The following tables describe the **Editing Network** dialog box.

Addressing

Item	Description	Default
Use DHCP	If selected, the Dynamic Host Configuration Protocol is enabled.	Selected
IP address	Provides the IP address of the network.	0.0.0.0
Subnet mask	Provides the subnet mask of the network.	0.0.0.0
Gateway	Provides the gateway of the network.	0.0.0.0
MAC address (hex)	Provides the MAC address of the network.	000000000000

Ethernet Driver Buffers

Item	Description	Default
Number of Ethernet driver receive buffers	Provides the number of Ethernet driver receive buffers.	60
Size (bytes)	Provides the size of Ethernet driver receive buffers.	1600

Number of Ethernet driver transmit buffers	Provides the number of Ethernet driver transmit buffers.	40
Size (bytes)	Provides the size of Ethernet driver transmit buffers.	1548

CONFIGURING LWIP LIBRARIES

To modify the local lwIP library configuration for a processor:

1. Make a copy of the default lwIP library configuration whose folder is:
   `*lwipInstallationRoot*\lwip\blackfin\lib\bfxxx_rev_any\liblwipipv4.dlb`
   where bf*xxx* is the processor type.
2. Import the library project whose path is:
   `*lwipInstallationRoot*\lwip\blackfin\lib\lwip\contrib\ports\blackfin\projects\lwipv4lib\liblwipv4_bf*xxx*`
   where bf*xxx* is the same processor type.
3. Double-click the imported project's **Config\Configuration.lwip** file.
   The lwIP Library Configuration editor appears.
4. (Optional) Modify settings on the General Configuration Options page.
5. (Optional) Modify settings on the UDP/ARP Configuration Options page.
6. (Optional) Modify settings on the TCP Protocol Options page.
7. (Optional) Modify settings on the Memory Configuration Options page.
8. (Optional) Modify settings on the Statistical Collection and Debug Options page.

9. Rebuild and save the library project.
   Note: Rebuilding the project overwrites its `liblwipipv4.dlb` file.
10. Rebuild lwIP applications of the affected processor in local workspaces.

LWIP LIBRARY CONFIGURATION EDITOR

The **lwIP Library Configuration** editor provides advanced users with the interface for modifying local lwIP libraries. The editor is accessed through processor type-specific library projects. Rebuilding a library project overwrites the default lwIP library for its type.

The **lwIP Library Configuration** editor includes one page for each major category of modifiable parameters. The pages are:

- General Configuration Options
- UDP/ARP Configuration Options
- TCP Protocol Options
- Memory Configuration Options
- Statistical Collection and Debug Options

lwIP Library Configuration Editor, General Configuration Options Page

Use the **General Configuration Options** page to configure general library options for applications of the lwIP add-in in the workspace.

To access the **General Configuration Options** page, open the lwIP Library Configuration editor.

The following tables describe the **General Configuration Options** page.

Protocols

Item	Description	Default
TCP	If selected, use of the Transmission Control Protocol is enabled.	Selected
UCP	If selected, use of the User Datagram Protocol is enabled.	Selected
ICMP	If selected, use of the Internet Control Message Protocol is enabled.	Selected
IGMP	If selected, use of the Internet Group Management Protocol is enabled.	Selected
DNS	If selected, use of the Domain Name System is enabled.	Not selected
SNMP	If selected, use of the Simple Network Management Protocol is enabled.	Selected
Raw	If selected, use of the lwIP RAW access layer is enabled.	Selected

IP Configurations

Item	Description	Default
IP Forward	If selected, forwarding IP packets on multiple interfaces is enabled.	Selected
Reassembly	If selected, reassembling incoming fragmented IP packets is enabled.	Selected
IP Options	If selected, accepting packets with IP options is enabled.	Selected

IP Fragmentation	If selected, fragmenting packets if size exceeds MTU is enabled.	Selected
IP SOF Broadcast	If selected, UDP broadcast on transmit is enabled.	Not selected
IP SOF Broadcast Receive	If selected with **IP SOF Broadcast**, UDP broadcast on receive is enabled.	Not selected
Maximum reassembly wait time (seconds)	Provides the maximum reassembly wait time.	3
Default time to live (seconds)	Provides the default time to live.	255
Maximum reassembly buffers	Provides the maximum number of reassembly buffers.	10

Common

Item	**Description**	**Default**
Restore Defaults	Returns all settings to their default values.	N/A

lwIP Library Configuration Editor, UDP/ARP Configuration Options Page

Use the **UDP/ARP Configuration Options** page to configure UDP and ARP library options for applications of the lwIP add-in in the workspace.

To access the **UDP/ARP Configuration Options** page, open the lwIP Library Configuration editor and choose **UDP\ARP**.

The following tables describe the **UDP/ARP Configuration Options** page.

lwIP Library Configuration Editor, UDP/ARP Configuration Options Page

UDP Options

Item	Description	Default
UDP time to live (seconds)	Provides the UDP time to live.	255
Number of protocol blocks	Provides the number of protocol blocks.	8

ARP Options

Item	Description	Default
Table size	Provides the size of the ARP table.	10
Trust IP	If not selected, an ARP request is sent if an entry is not present in the ARP table.	Not selected
Virtual LAN support	If selected, virtual LAN support is enabled.	Not selected
Queuing	If selected, ARP queuing is enabled.	Selected

Common

Item	Description	Default
Restore Defaults	Returns all settings to their default values.	N/A

lwIP Library Configuration Editor, TCP Protocol Options Page

Use the **TCP Protocol Options** page to configure TCP protocol library options for applications of the lwIP add-in in the workspace.

To access the **TCP Protocol Options** page, open the lwIP Library Configuration editor and choose **TCP**.

The following table describes the **TCP Protocol Options** page.

Item	Description	Default
Number of active connections	Provides the number of active connections.	16
Maximum listen protocol control blocks	Provides the maximum number of listen protocol control blocks.	8
Number of simultaneously queued segments	Provides the number of simultaneously queued segments.	32
Number of TCPIP messages	Provides the number of TCP/IP messages.	32
Time to live (seconds)	Provides the time to live.	255
Maximum segment size (seconds)	Provides the maximum segment size.	1460
Send buffer space (bytes)	Provides the size of the send buffer space.	11680
Sender buffer queue length (bytes)	Provides the length of the buffer queue.	32
Maximum number of data retransmissions	Provides the maximum number of data retransmission.	12
TCP window size (bytes)	Provides the size of the TCP window.	8192
Maximum number of sync retransmissions	Provides the maximum number of sync retransmissions.	12

TCP writable space (bytes)	Provides the size of the TCP writable space.	5840
Restore Defaults	Returns all settings to their default values.	N/A

lwIP Library Configuration Editor, Memory Configuration Options Page

Use the **Memory Configuration Options** page to configure memory library options for applications of the lwIP add-in in the workspace.

To access the **Memory Configuration Options** page, open the lwIP Library Configuration editor and choose **Memory**.

The following table describes the **Memory Configuration Options** page.

Item	**Description**	**Default**
RAM stack memory (bytes)	Provides the size of RAM stack memory.	65536
Number of packet buffers	Provides the number of packet buffers.	32
Number of active timeouts	Provides the number of active timeouts.	12
Number of network buffers	Provides the number of network buffers.	16
Number of network connections	Provides the number of network connections.	20
Number of API messages	Provides the number of API messages.	32
Number of buffers in packet buffer pool	Provides the number of buffers in the packet buffer pool.	128
Buffer size in packet pool (bytes)	Provides the buffer size for the packet pool.	512
Minimal standalone system	If selected, minimal standalone systems are enabled.	Not selected
Restore Defaults	Returns all settings to their default values.	N/A

lwIP Library Configuration Editor, Memory Configuration Options Page

lwIP Library Configuration Editor, Statistical Collection and Debug Options Page

Use the **Statistical Collection and Debug Options** page to statistics and debug library options for applications of the lwIP add-in in the workspace.

To access the **Statistical Collection and Debug Options** page, open the lwIP Library Configuration editor and choose **Stat\Debug**.

The following tables describe the **Statistical Collection and Debug Options** page.

Statistical Collection Options

Item	Description	Default
TCP	If selected, collection of Transmission Control Protocol statistics is enabled.	Not selected
UDP	If selected, collection of User Datagram Protocol statistics is enabled.	Not selected
IP	If selected, collection of Internet Protocol statistics is enabled.	Not selected
ICMP	If selected, collection of Internet Control Management Protocol statistics is enabled.	Not selected
Memory	If selected, collection of memory statistics is enabled.	Not selected
System	If selected, collection of system statistics is enabled.	Not selected
Link	If selected, collection of link statistics is enabled.	Not selected

TCP Debug Options

Item	Description	Default
TCP	If selected, debug of Transmission Control Protocol is enabled.	Not selected
UDP	If selected, debug of User Datagram Protocol is enabled.	Not selected
ICMP	If selected, debug of Internet Control Management Protocol is enabled.	Not selected

IGMP	If selected, debug of Internet Group Management Protocol is enabled.	Not selected
DNS	If selected, debug of Domain Name System is enabled.	Not selected
SNMP	If selected, debug of Simple Network Management Protocol is enabled.	Not selected
RAW	If selected, debug of lwIP RAW access layer is enabled.	Not selected
TCP Input Layer	If selected, debug of TCP input layer is enabled.	Not selected
TCP Fast Retransmit	If selected, debug of TCP fast retransmit is enabled.	Not selected
TCP Retransmit	If selected, debug of TCP retransmit is enabled.	Not selected
TCP Congestion	If selected, debug of TCP congestion is enabled.	Not selected
TCP Window	If selected, debug of TCP window is enabled.	Not selected
TCP Output	If selected, debug of TCP output is enabled.	Not selected
TCP Reset	If selected, debug of TCP reset is enabled.	Not selected
TCP Queue Length	If selected, debug of TCP queue length is enabled.	Not selected

TCPIP Thread	If selected, debug of TCP/IP threading is enabled.	Not selected
Debug Level	Debug level for the selected actions. Options include: • LWIP_DBG_LEVEL_OFF • LWIP_DBG_MASK_LEVEL • LWIP_DBG_LEVEL_SEVERE • LWIP_DBG_LEVEL_SERIOUS • LWIP_DBG_LEVEL_WARNING • LWIP_DBG_LEVEL_ALL	LWIP_DBG_LEVEL_OFF

Common

Item	**Description**	**Default**
Restore Defaults	Returns all settings to their default values.	N/A