

Lab 2

Introduction to DE2 Embedded System

Abstract

Learn to use the Nios II IDE (programming environment) to implement an Ethernet chat client on the DE2 board.

Unlike the first lab, your job in this lab is just to develop software. We have supplied a hardware design for the DE2 that includes the Nios II processor, memory, an Ethernet controller, and a controller for the PS/2 keyboard.

Your job in this lab is to develop software that uses this hardware to implement an Ethernet-based chat client. This will function as a sort of terminal: the user should be able to type in a line of text using the attached keyboard. When s/he presses enter, the contents of the line should be sent as a UDP broadcast packet at an attached keyboard and the line s/he enters should be sent as a UDP broadcast packet through the Ethernet controller.

Similarly, when the board receives a UDP broadcast packet, it should print it on the screen. This assumes that any UDP broadcast packet the board receives comes from a similar project.

To help clarify who is saying what, the first few characters of each packet sent should be the user's name. Have the user type this first, save it, and send it automatically at the beginning of each packet.

To test these projects, use the ethernet hubs and patch cables in the lab. Use these to set up small networks to which only the DE2 boards are connected. Please do not connect them to the network for the computers when you are debugging your project since doing so will likely cause trouble for all the other computers.

Programming the board for this lab will take two steps. First, open the lab2 project in Quartus like you did for lab1. We already compiled this hardware design for you, so you can go directly to the programmer and download the .sof file to the board. This configures the hardware but leaves the software unconfigured. It will not do anything yet. You have to download and run software. Do this using the Nios II IDE, described below.

1 The Nios II IDE

First run the Nios II IDE software package. When it starts, it may give you a few icons to choose among. Select "Workbench" to get started.

First, set your workspace. Broadly, this is the directory in which the IDE will put all your files. Select File—>Switch

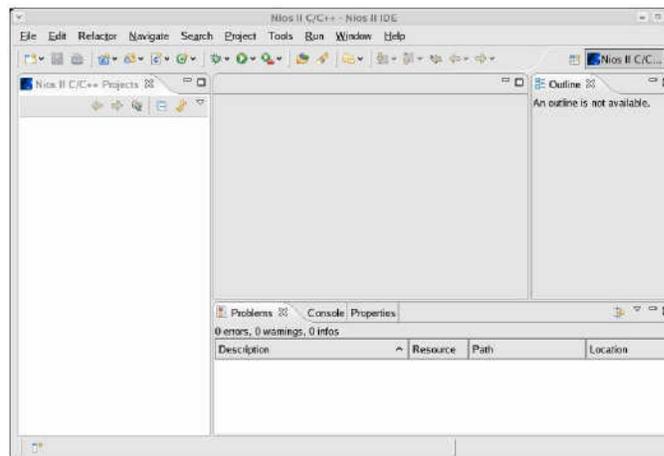


Figure 1: The Nios II IDE ready to accept a new project

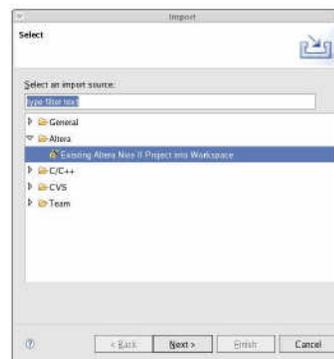


Figure 2: Importing projects into the IDE

Workspace and select the directory in which the Quartus project resides.

When you select a new workspace location, it should start up empty, such as in Figure 1.

Start by informing the IDE about the skeleton projects we provided to you.

To bring the software into the IDE, right-click on the "Nios II C/C++ Projects" window on the left side of the main window and select "Import." Under "Altera," select "Existing Altera Nios II Project into Workspace" (Figure 2). Click on Next >, then choose the lab2 directory in the project directory. Click "Finish" to import it.

Repeat the process by importing the lab2_syslib project, which builds information about the hardware configuration. This requires you to specify the processor it targets by select-

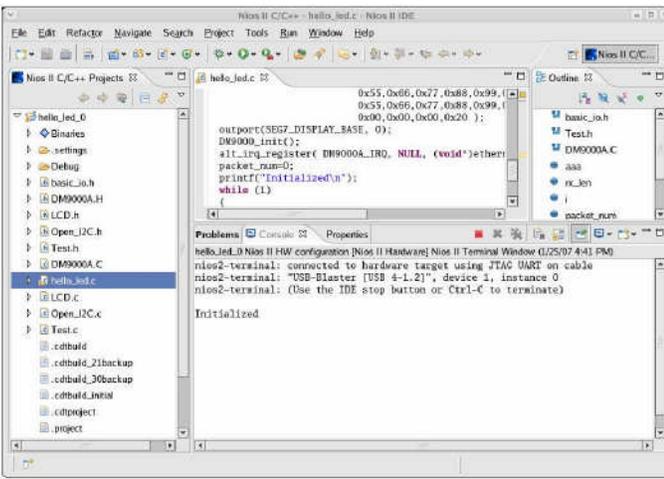


Figure 3: Running a program and observing output on the terminal

ing the `nios_0.ptf` file in the project directory. The lab2 project will not build without the `lab2_syslib` project.

To run the sample project we have provided, right-click on the project name (lab2) on the left and select “Run As...” and “Nios II Hardware.” Assuming you have downloaded the hardware configuration using the Quartus programmer, this should first compile the software, then download it to the hardware and start it running. You may need to set the USB-BLASTER as your target connection (Run-Run- Target Connection)

One of the most convenient aspects of the Nios II IDE is that after it runs your program, it connects a terminal to the DE2 board that allows you to print from your running C program and have it appear in the IDE. Furthermore, you can type in the terminal window and have it sent to your program. Figure 3 shows a running project with the terminal connection.

2 The Chat Application

The lab2 files have a partially-working skeleton for the application. Here is a list of things you need to change:

- Make the keyboard input work properly. Specifically,
 - Make the space bar work properly (display a space)
 - Turn off the debugging information for the unrecognized keycodes
 - Make the left and right arrow keys work
 - Make the backspace key work
 - Make both shift keys work (i.e., do upper and lower-case characters)
 - Ignore the other keys (e.g., tab, escape, print screen, the keypad, etc.).
- When the system starts, have the user enter his/her name before going into “chat” mode. Start the string sent by each packet with this name.

- Ensure the UDP packets are well-formed
 - Make sure the string sent in the UDP packet is always zero-terminated
 - Make sure the UDP packet length field is correct
 - Make the IP header checksum correct (include calculation for IP header, UDP header, and Data until terminated only) (verify with a neighbor)
- Make it so that when a packet arrives, it does not disrupt the line the user is creating. Do this by overwriting the current line with the new packet (clearing the rest of the line by counting characters), then print a new version of the line-under-construction immediately after the incoming packet’s contents. (See section 3 for better terminal support) E.g., layout before:

```
STEPHEN: Wow isn't this nice
Yes it is reall <Line being typed
```

and then after the “DAVID” packet arrives

```
STEPHEN: Wow isn't this nice
DAVID: Amazing that it works
Yes it is reall <Line being typed
```

3 Running nios2-terminal

The terminal provided in the Nios II IDE does not have good support for cursor-motion commands (e.g., carriage return backspace, etc.), so it is difficult to overwrite lines.

Instead, use the `nios2terminal` command-line program as follows:

1. Run your program as usual (right click on lab2, then Run As→Nios II hardware)
2. Once it starts running, click the red square button at the upper-left corner of the terminal window within the Nios II IDE. This disconnects the terminal, but does not stop your program from running.
3. Open a NIOS II command-line window (via Start-Altera menu) and run “`nios2-terminal --cable USB-BLASTER`”
4. Now, whatever you type into that terminal window is sent to your program and whatever your program prints will be displayed in that window rather than in the IDE window.