Peter the Great St. Petersburg Polytechnic University
Institute of Computer Science and Cybersecurity
Graduate School of Computer Technologies and Information Systems

Lecture: Nios II Software Development

Subject: Automation of discrete device design (in English)

Completed by student of group 5130901/10101 _____ Nepomnyaschiy M.T.
(signature)

Lecturer _____ Antonov A.P.
(signature)

Saint Petersburg
2024

Table of contents

List of illustrations

Creating a System Design with Platform Designer

This lecture provides information on the Nios II Software and Design Flow, detailing the tools and processes for developing software and designing systems around the Nios II processor.

Part 3 – Nios II Embedded Design Suite

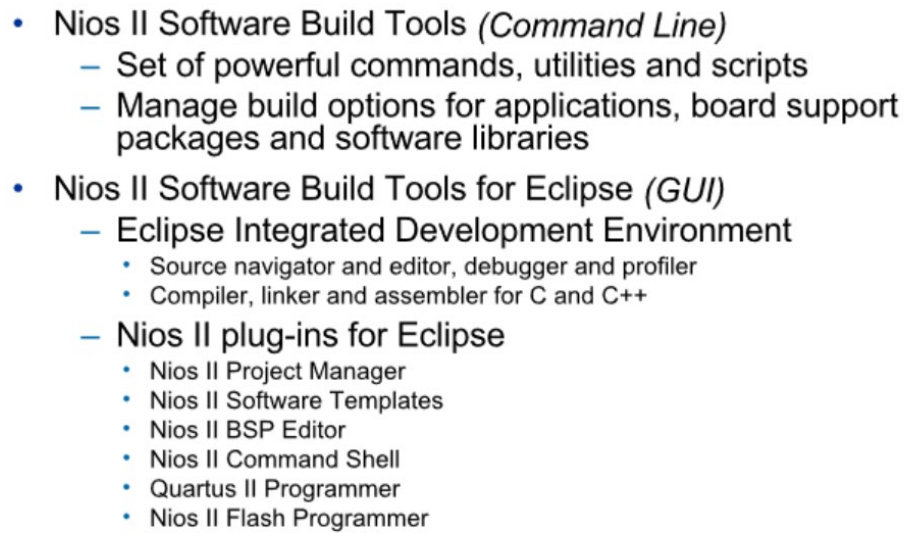

Figure 1 – Nios II Embedded Design Suite

The Nios II Embedded Design Suite (EDS) is a set of advanced software tools, like computer programs, that help to make designs quickly. It has things like tools for building software, which is the programs that run on a device, and drivers, which are like translators that help the device talk to other parts.

In this suite, there are tools that work inside a program called Eclipse, which is very popular among software developers. These tools help to write, test, and debug programs. They also help with organizing project and making it easier to work with big teams.

The suite includes a bunch of other useful things too, like templates for starting new projects, a tool for managing how program works with specific device, and even tools for directly programming the memory of your device. You can get all of this stuff for free from the Altera website.

Part 4 – Nios II Software Built Tools for Eclipse

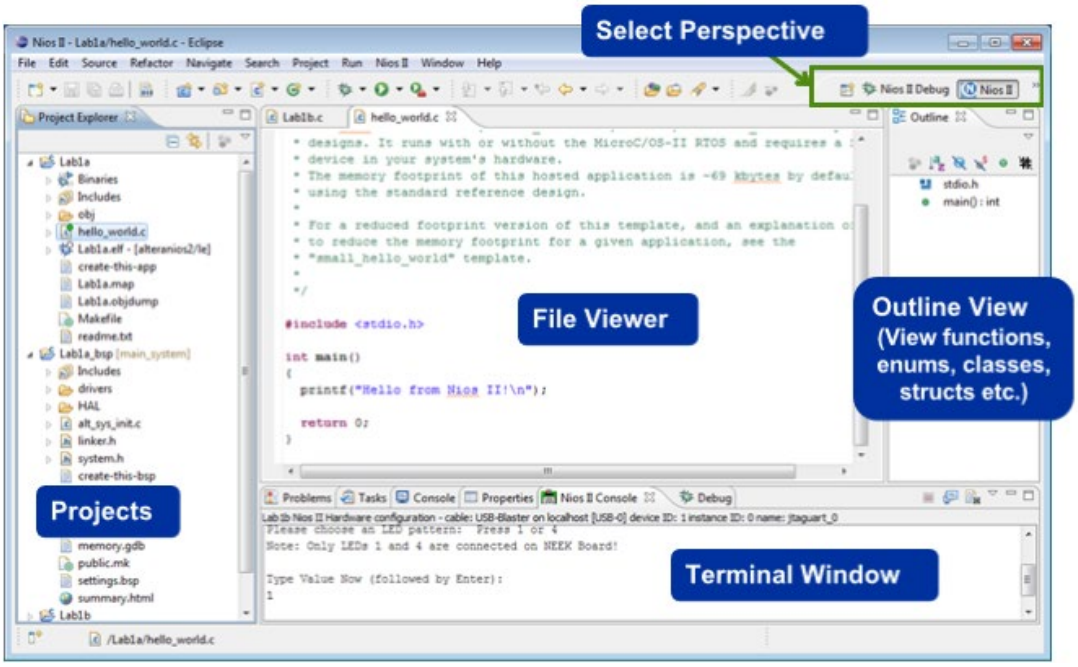

Figure 2 – Nios II Software Built Tools for Eclipse

The Nios II SBT for Eclipse can be started either from Qsys or from the Windows program menu. When you open it, you'll see a graphical interface that looks like this:

- Project Explorer: Here you'll see all the projects you've created.
- File Viewer: This shows your source files, and you can edit them directly here.
- Outline View: This gives you an overview of your project's functions, types, and other important stuff.
- Terminal Window: This displays messages from the build process.

If you're using C++ files, they need to end with ".epp". Also, if you're using inline assembly code, you should put it between "asm();" markers.

Part 5 – Creating a Nios II Application and BSP

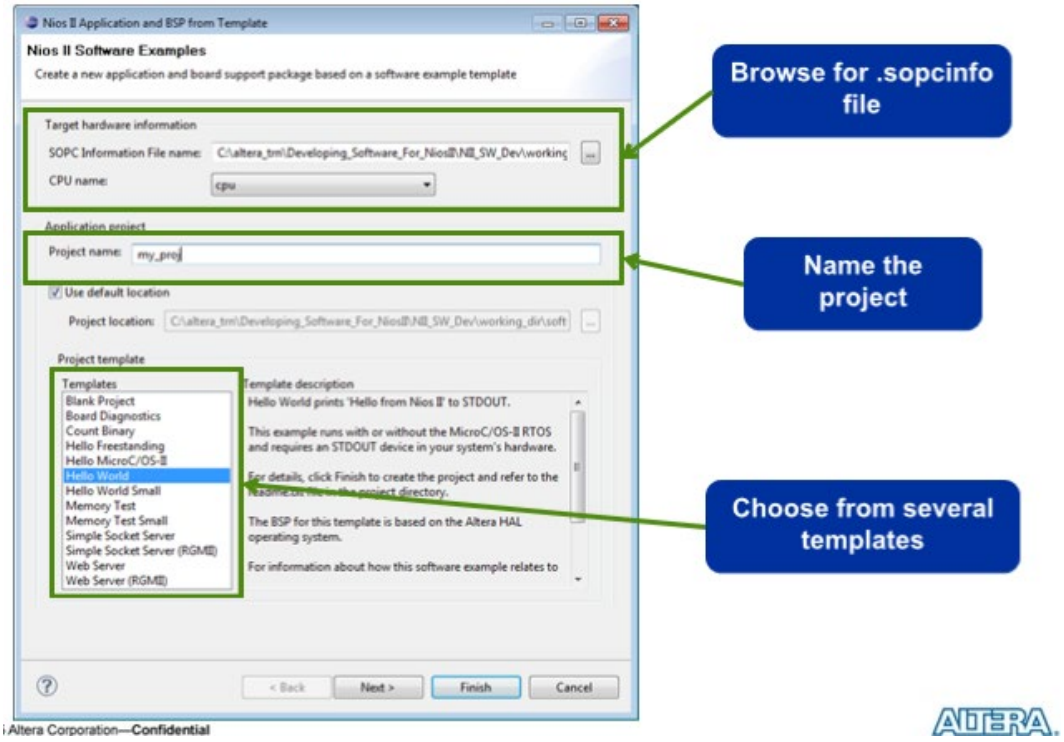

Figure 3 – Creating a Nios II Application and BSP

Creating a project is simple, especially if you use templates. When you want to make a new project, you can select the option to create a new application and BSP from a template.

You'll need to tell the system which hardware system you're using, and you do this by specifying the "sopcinfo" file. Then, in a dialog box, you give your project a name and choose the template you want to use. For example, you might choose the "hello world" template.

Part 6 – Application and BSP Project Folders

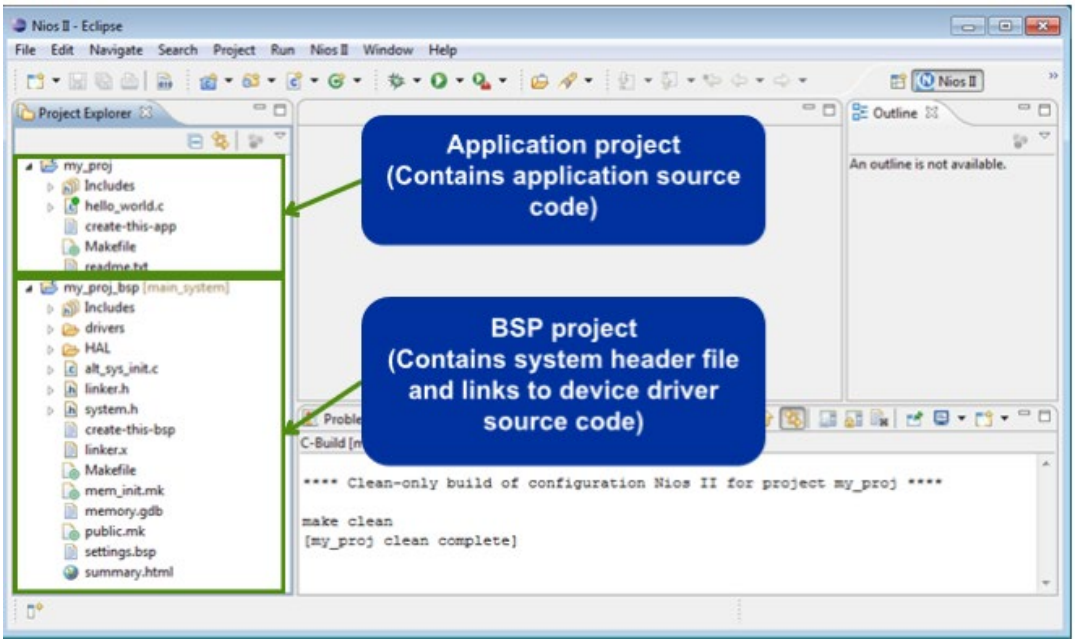

Figure 4 – Application and BSP Project Folders

After creating the project, you'll notice two separate directories in the project explorer: one for the application and the other for the BSP.

- The application project holds all the source code for your project.
- The BSP project includes the system header file and links to the device driver code.

Part 7 – BSP Project Properties

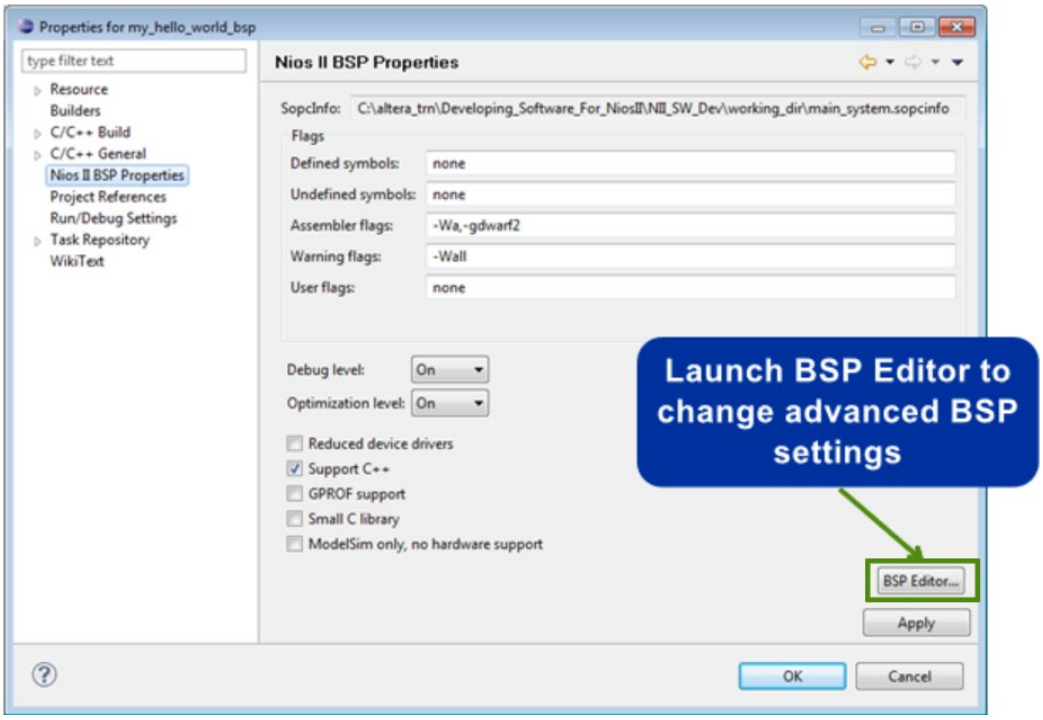

Figure 5 – BSP Project Properties

If you right-click on the BSP project and choose "Properties," you can change optimization levels for the board support package files. You can also select options like reduced size device drivers, a small C library, and ModelSim simulation support from here.

Additionally, you can launch the BSP editor by pressing the BSP Editor button in the BSP project properties. Similarly, settings such as debug level and optimization level are available for the application project.

Part 8 – BSP Editor – Main Tab

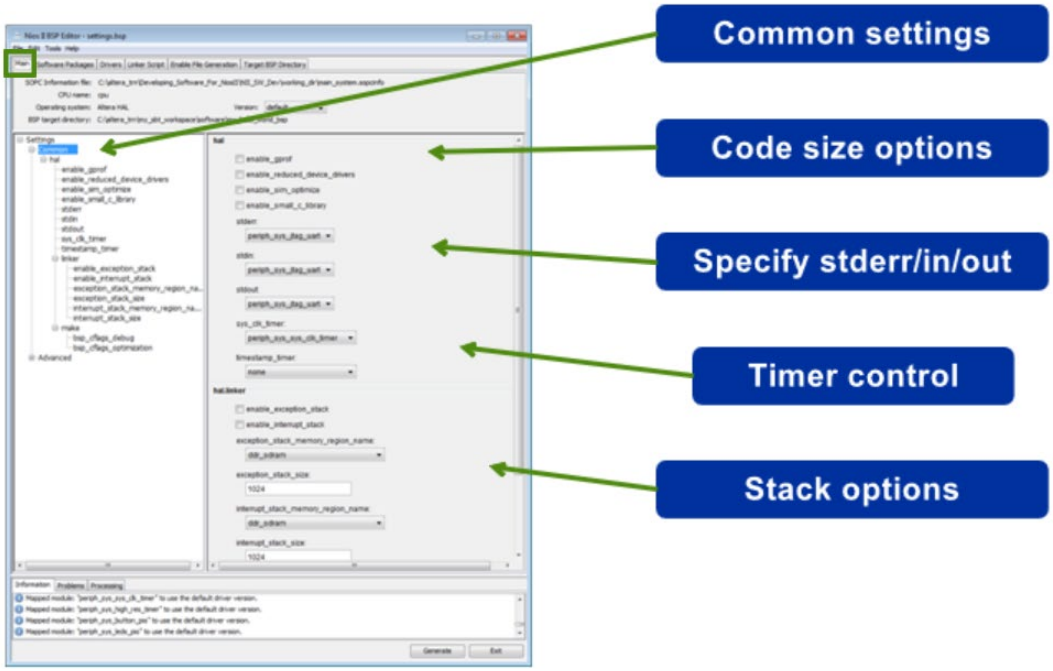

Figure 6 – BSP Editor – Main Tab

The BSP Editor allows to adjust more detailed settings in the board support package. It's organized into tabs at the top, letting you switch between different categories.

The "Settings" tab, shown in Figure 6, lets you tweak hardware resources for standard input, output, and error devices, as well as timers like sys clk and timestamp timers. You can also specify the size and location of the stack here. Remember to click the "Generate" button after making changes to create a new BSP.

Part 9 – Software Compilation (“Build”)

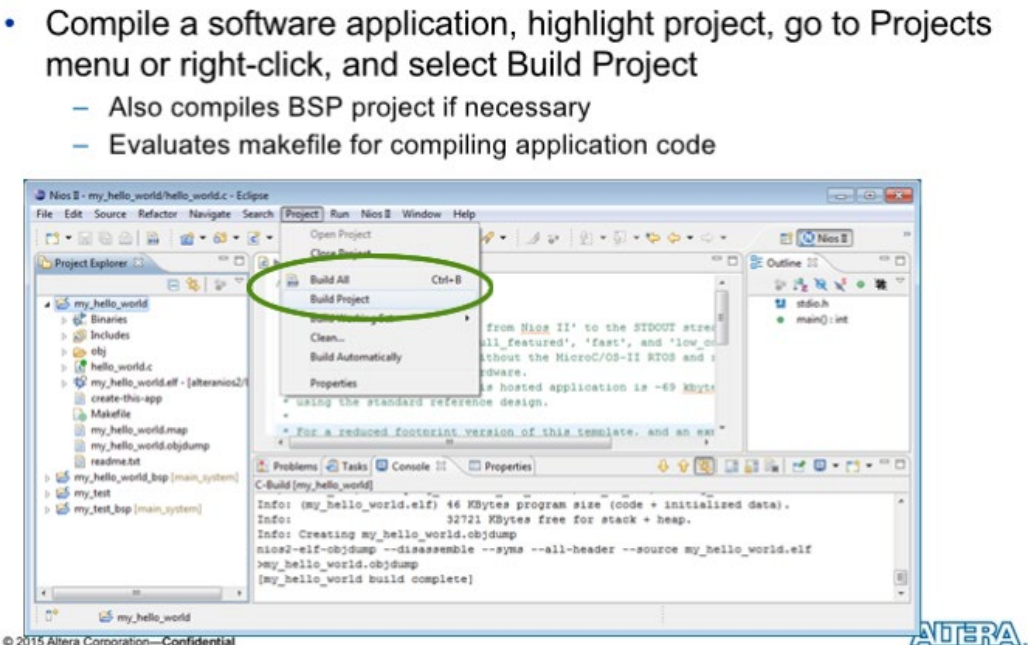

Figure 7 – Software Compilation (“Build”)

To compile or build your project, you have a couple of options. You can either:

1. Highlight the application project, right-click, and choose "Build Project..."
2. Alternatively, you can go to the project menu and select "Build Project."

Either way, this will also compile the associated BSP project if necessary.

Part 10 – Project Directory Structure

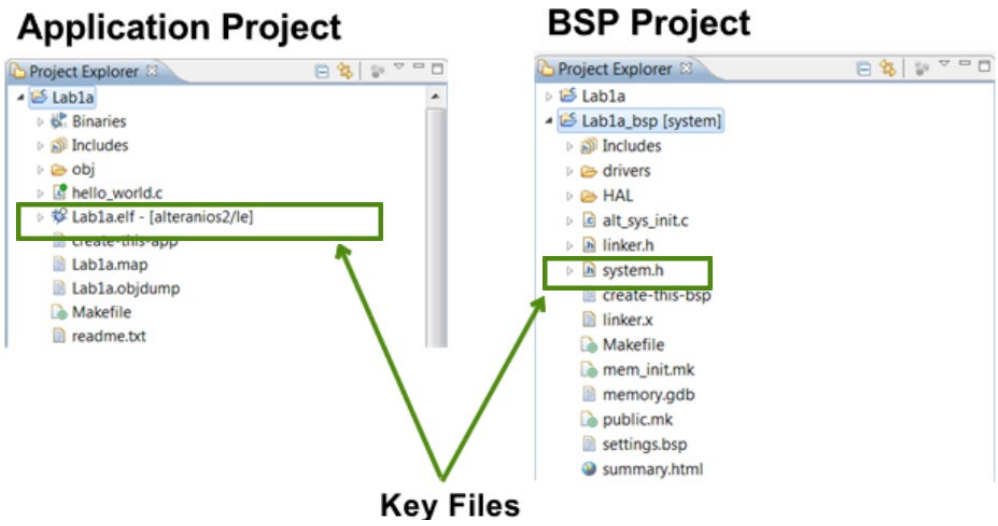

Figure 8 – Project Directory Structure

After compiling or building your projects, you'll notice several files created inside both your application project folder and your BSP project folder. Here are the most important files:

In the application project:

- The .elf file: This contains all your compiled code and device drivers. It's what gets downloaded to the processor.

In the BSP project:

- The system.h file: This is the key file to remember. It contains mappings between the Qsys peripheral names and their base addresses in the memory map. Using macros in this file allows you to call out your peripherals by name rather than specific hex addresses.

Part 11 – Running Code on a Target

- stdio messages appear in Nios II Console

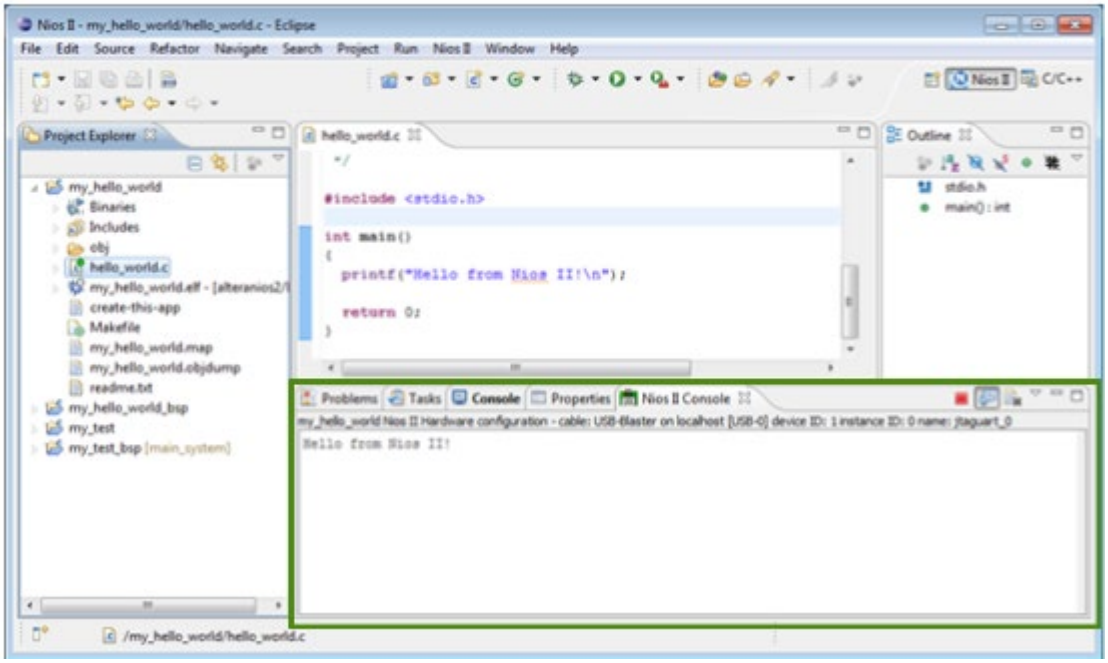

Figure 9 – Running Code on a Target

Once you've built your project, you'll want to download the code to the processor on your FPGA to run or debug it. To run the code on the target, follow these steps:

1. Right-click on your application project.
2. Select "Run As Nios II Hardware."

This action will download the code through the FPGA programming cable, into the FPGA via the JTAG ports. Then, the processor will start running the code.

Part 12 – System ID Peripheral Check

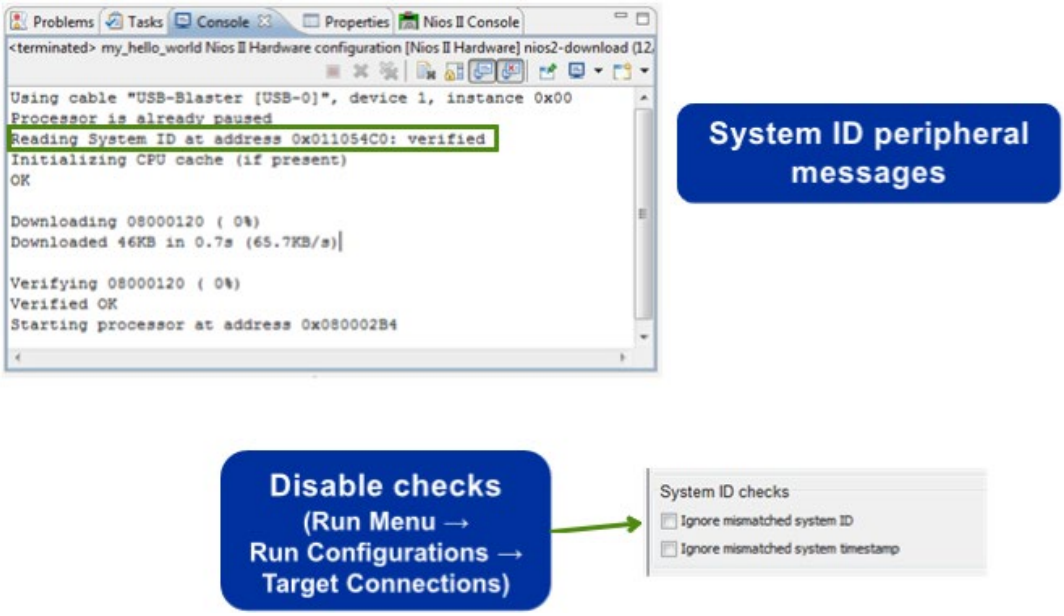

Figure 10 – System ID Peripheral Check

During runtime, the Nios II Eclipse platform ensures that the software matches the hardware on the FPGA. It does this by calculating expected system ID peripheral values from the .sopcinfo file and compares them to the ones in the .sof file.

If the computed ID values don't match the system ID variables stored on the target board, an error is flagged. You have the option to disable the system ID check in the run configuration dialog box, but it's recommended to keep it enabled. This way, if there's a mismatch, the system will flag an error and stop the download process, preventing potential issues.

Part 13 – Nios II Debug Perspective

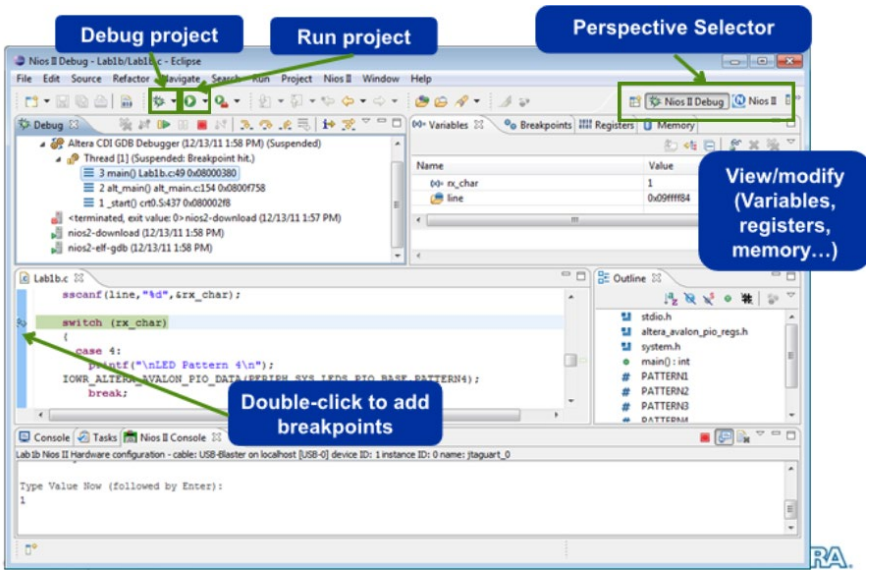

Figure 11 – Nios II Debug Perspective

The Nios II SBT for Eclipse has a debug perspective that's really handy for debugging your software. You can switch to the debug perspective by right-clicking on an application and selecting "Debug As Nios II Hardware." This action will download and run the code on the FPGA in debug mode.

Alternatively, you can set the debug perspective by clicking on the perspective window.

In the debug perspective, you can do a lot of things like examining the program stack, setting breakpoints, checking registers, and viewing memory. You can also control the debugger to pause or restart the program, or step into or over certain lines of code to better understand what's happening.

Part 14 – Also Available for Nios II Processor

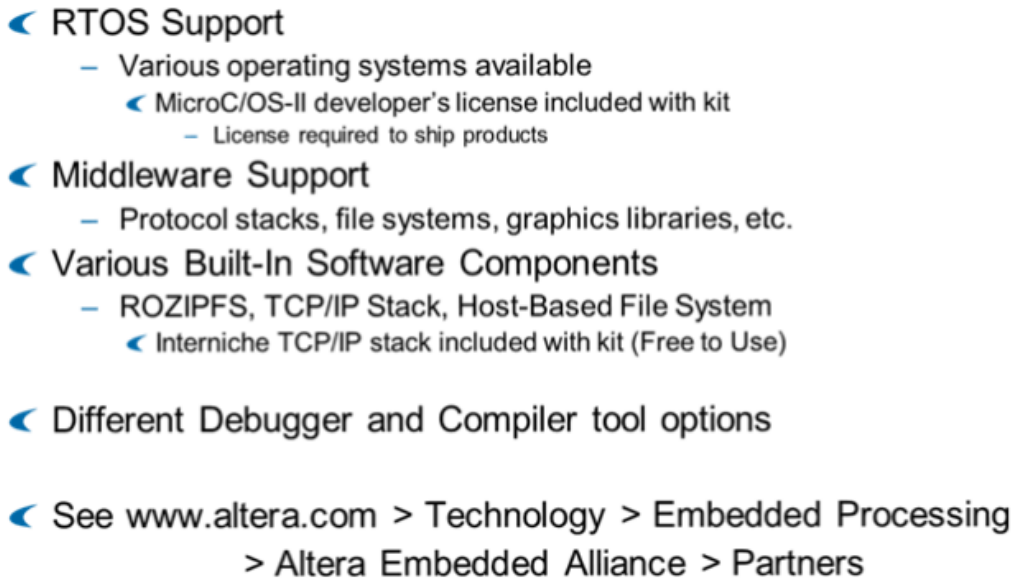

Figure 12 – Also Available for Nios II Processor

This presentation won't cover all the software components supported by Nios II. However, there are various real-time operating systems available, including uC/OS-II, which comes with the kit along with its full source code and developer's license.

Additionally, there are other operating systems, middleware, protocol stacks for TCP/IP, USB, and a read-only file system. The InterNiche TCP/IP stack is included with the software development tools and is free to use.

You can find a list of all software partners on altera.com by selecting the embedded processor link and then clicking on embedded software partners.

Conclusion

In this lecture, we explored the Nios II EDS, a comprehensive set of software tools aimed at accelerating the development of embedded systems. The EDS includes the Nios II Software Build Tools, Eclipse-based and command-line interfaces, facilitating efficient code development and debugging. Through the BSP Editor, users can fine-tune hardware settings and optimize system performance. Compiling projects integrates seamlessly with the BSP, ensuring synchronization between software and hardware configurations. Debugging capabilities, accessed through the Nios II SBT for Eclipse, enable thorough analysis of program execution, including stack examination and breakpoint setting. Moreover, the EDS supports various real-time operating systems, middleware, and protocol stacks, enhancing versatility and functionality. With resources like uC/OS-II and InterNiche TCP/IP stack, developers can leverage robust solutions for diverse embedded applications, ultimately streamlining the design-to-market process.