



This document addresses known errata and documentation issues for the Nios® II Embedded Design Suite (EDS) version 7.1. Errata are functional defects or errors, which might cause the product to deviate from published specifications. Documentation issues include errors, unclear descriptions, or omissions from current published specifications or product documents. Errata items discovered after the release of Nios II EDS version 7.1 are marked with the date the items were added to this document.

Table of Contents:

Nios II Processor Core.....	4
Level 4 Debug in Stratix III Designs	4
VHDL Sensitivity List Warnings During Quartus II Synthesis	4
Double-Precision Floating-Point Operations With Floating-Point Custom Instructions.....	4
Peripherals.....	4
TSE MAC Designs Require a Manual Reset on Subsequent Software Downloads.....	4
Non System Wide Reset Can Cause Improper Initialization of Mailbox Core.....	5
DMA Controller Always Busy in Burst Mode If Transfers Less Than Full Width.....	5
Using the DEV_CLRn Signal Can Cause JTAG UART Instability After Device-Wide Reset	5
Host Platform.....	5
Linux: Process Fails To Terminate While Debugging With the Nios II ISS Target	5
Linux: The Quartus II Stand-Alone Programmer is Not Supported.....	6
Windows: Frisk Antivirus Software Causes SOPC Builder and Nios II Command Shell to Be Unresponsive	6
Device.....	6
Stratix II EP2S60 ES Devices Cannot Use MRAM Byte Enables.....	6
Nios II IDE	6
Building Projects	6
Build Option Not Functional for BSPs Created Using the Software Build Flow	6
Previous 7.1 Beta Install Might Generate Message 'Internal error occurred during "Launching"	7
Different Address Mapping for Master Ports Connected to Dual-Port Memories Can Cause Linker Errors	7
Nios II IDE Does Not Support Settings for Projects Imported From the Nios II Software Build Tools Design Flow	7
Using a URL Location for Projects Might Cause an Error Creating Project.....	7
Conflicting File Names Can Create Illegal Project Location Error When Creating New Project	7
Choosing Separate Exception Stack Option Might Cause Compilation error "UNDEFINED VARIABLE %STACK_POINTER%" When Building a Project	8
When Building a Project, the Nios II IDE Reports Problems, But the Build Output in the Console Does Not Contain Any Errors	8
Makefile reports incorrect number of bytes free for the stack and heap, if the heap is in a different memory region than the stack	8
Nios II IDE Unnecessarily Updates the SOPC Builder System File (.ptf)	8
Incorrect Address Assignment for Dual-Port Memory Mastered by Two Different Master Ports	8
Build Errors After Changing Component Names in SOPC Builder.....	8
Debugging Projects.....	9
GDB Error 256 When Launching Debugger.....	9
Previous 7.1 Beta Install Might Cause Internal Error When Launching FS2 in IDE	9
Extra Arguments in Debugger for Multiple Launch Configurations.....	9
Console Window Is Not Updated After ISS Error	9
Cannot Locate Source Code in Driver Files Shared by Multiple Projects	9
Memory Window Sets Control Register Values Incorrectly.....	10
ISS Fails on Designs Containing TSE MAC or SG-DMA Components.....	10

"Run as ModelSim" in the Nios II IDE Fails	10
Unable to Display Driver Source Code When Debugging a Custom Component in the Nios II IDE	10
The Restart Command on the Run Menu Does Not Work	11
Programs That Interact With a Terminal Console on Windows Do Not Work	11
"Step failed. Target is not responding (timed out)" Error Message	11
Incorrect Breakpoint Filtering on Threads	11
Uninitialized Memory Error When Executing From ISS	11
nios2-gdb-server Fails to Terminate After Setting a Watch Point	11
Watchpoints Do Not Work on a Variable Whose Size Is Not 32 Bits	12
Debugger Cannot Step Into __sflags(), and Continues Execution Instead	12
Missing Address and Data in the Trace View	12
Cannot Use Watchpoints in the Nios II IDE When the FS2 Console Is Open	12
Breakpoints on Adjacent Lines of Assembly Fail to Halt the Processor	12
Navigating Projects	13
Nios II IDE Import System Library Dialog Box Does Not Close After Import	13
C/C++ Search Returns Nothing	13
Internal Error When Double-Clicking on a Large Objdump File	13
After Switching Workspaces, the Nios II IDE Appears to Freeze While Displaying the Splash Screen	13
C/C++ Scanner Does Not Understand Certain C/C++ Constructs	13
Nios II Software Build Tools	14
Nios II IDE import settings for Nios II software build tools projects	14
The add_sw_setting Character Tcl Command Produces Improper Quote Format in system.h	14
Upper-Case File Extensions Not Supported	14
"make download-elf" Fails for Stratix II 2S60 RoHS Design Examples	14
C2H Compiler	15
C2H Regenerates Hardware Accelerator Even Though No Logic Changed	15
Error: c2h_accelerator_base_addresses.h: No Such File Or Directory	15
Pointer Dereferences to Volatiles	15
Java Heap Space Exception if Quartus II Compilation is Enabled	16
C2H Accelerators Time Out Immediately When Untethered	16
Accelerator Generation Failure If Tools Are Installed in Path With Spaces	16
C2H Compiler Out-of-Memory Error	16
C2H Compiler Does Not Accelerate Subfunctions Located in a Separate File	17
Incorrect Results From Logical or Conditional Operation With Side-Effects	17
Closed System Library While Working With the C2H Compiler	17
Launch SOPC Builder Button in C2H View	17
Build Clean Causes Build Failure	17
Multiple Clock Domains Causes Hardware Accelerator to Fail	17
Array Elements of structs Do Not Copy Correctly	18
Simulating on ISS Not Supported	18
Hardware Accelerators Remain After Deleting the Software Project	18
Changing Build Configurations Produces Unexpected Results	19
Flash Programmer	19
Delay When Creating New Flash Programmer Configuration	19
"No such file or directory" Error for a Project Stored in a Path Containing Spaces	19
elf2flash File Size Limit	19
When Used With a Multiprocessor Nios II System, the Nios II Flash Programmer Might Produce an Error	19
Download Cables & Debug Hardware	20
Communication Errors During Run/Debug Sessions Using Older Download Cables	20
Development Boards	20
Intermittent Failures While Accessing CompactFlash Card	20

Toolchain (gcc, gdb, etc.)	21
Breakpoints in C++ Constructors Fail to Halt the Processor	21
Target Software	21
Lightweight IP (lwIP) Failure When Using DHCP	21
stdin, stdout, and stderr Do Not Work in MicroC/OS-II Applications When Using Small C Library Option	21
malloc(), realloc() Failures With MicroC/OS-II	22
cout From MicroC/OS-II Task Will Not Send Data to stdout	22
Problems Using HAL Drivers With Toshiba Flash	22
Creating New Custom HAL Components in the Nios II IDE	22
Legacy SDK	22
SOPC Builder and Quartus II Software	22
Example Designs	23
Hardware Designs	23
Incorrect SSRAM & SDRAM PLL Phase Shift in Stratix II 2s60 RoHS Standard Design	23
Incorrect SSRAM PLL Phase Shift in Stratix II 2s60 RoHS and Cyclone II 2C35 TSE SG-DMA Designs	24
Software Designs	24
Dhrystone BSP Example Fails When Run on the Fast Hardware Example	24
Software Template Error With Non-Ethernet Hardware Design	25
RAM Test Failure When Running Memory Test Software Template on the ISS	26
Networking Examples	26
Hardware Simulation	26
Error: "UNC paths are not supported. Defaulting to Windows directory."	26
Simulation Failure if Reset Address is Set to EPCS	26
Uninitialized BSS Variables in Simulation	26
ModelSim Fails to Load Large Memory Models	27
Documentation Issues	27
Hardware Development Tutorial Example Does Not Run on Nios Cyclone 1C20 Board	27
Nios II IDE Help Topic - About Nios II IDE Projects on Linux Mozilla	27
Arria GX Support Incorrectly Listed in Documentation HTML Page	28
GNU Assembler Does Not Accept the --defsymb Flag	28
Contact Information	28
Revision History	28



For the most up-to-date errata for this release, refer to the errata sheet on the Altera® website:
http://www.altera.com/literature/es/es_nios2eds_71.pdf

Nios II Processor Core

This section lists all issues related to the Nios II processor cores.

Level 4 Debug in Stratix III Designs

You will get the following error during Quartus® II compilation if you are using Level 4 debug in the Nios II core and are targeting a Stratix® III device. Level 4 debug is not currently supported in designs targeting Stratix III devices.

```
Error: MGL_INTERNAL_ERROR: Port object altpll|clk of width 6 is being
assigned the port altpll|stratixii_pll inst pll1|clk of width 10 which is
illegal, as port widths dont match nor are multiples..
```

Workaround: There is no workaround available at this time. This issue will be fixed in a future version of the Nios II core.

VHDL Sensitivity List Warnings During Quartus II Synthesis

You might receive warnings similar to the following in the Quartus II software when compiling a VHDL Nios II system containing the JTAG debug module.

```
Warning (10492): VHDL Process Statement warning at
cpuname_jtag_debug_module.vhd(254): signal "usr1" is read inside the Process
Statement but isn't in the Process Statement's sensitivity list
Warning (10492): VHDL Process Statement warning at
cpuname_jtag_debug_module.vhd(254): signal "ena" is read inside the Process
Statement but isn't in the Process Statement's sensitivity list
```

These warnings are benign and can be ignored. The warnings are a result of the `ena` and `usr1` signals not being included in the debug module's sensitivity list.

Double-Precision Floating-Point Operations With Floating-Point Custom Instructions

Calls to double-precision floating-point functions in `math.h` return less-precise results on Nios II processors using the floating-point custom instruction. Floating-point constants are forced to single-precision when the floating-point custom instruction is present, which affects the constants for the double-precision floating-point functions in `libm`.

Workaround: There is no workaround available at this time.

Peripherals

This section lists all issues related to the Altera embedded peripherals included in the Quartus II software.

TSE MAC Designs Require a Manual Reset on Subsequent Software Downloads

You may experience issues in designs containing the Altera Triple Speed Ethernet (TSE MAC) MAC which may prevent the TSE MAC from operating properly after repeated downloads of the Nios II processor's executable linkable file (`.elf`). When this bug occurs, even though the Nios II processor executes the correct initialization

sequence to reset the TSE MAC component, a complete reset never occurs. The result is that the TSE MAC is unable to transmit or receive Ethernet data.

Workaround: Before downloading the software image file to the Nios II processor, push the board's reset button. This action will reset the Altera Triple Speed Ethernet MAC into a known-good state.

Non System Wide Reset Can Cause Improper Initialization of Mailbox Core

The `altera_avalon_mailbox` peripheral may not be initialized properly when a soft (non-system-wide) reset occurs. Examples: the `resetrequest` signal resets a specific Nios II processor that masters the mailbox; a processor mastering the mailbox restarts execution from the start address. In this condition, mailbox contents (read and write pointers) will not be reinitialized and may show potentially stale data.

Workaround: Ensure that a system-wide reset event occurs by asserting the `reset_n` input to the SOPC Builder system containing the mailbox. This will reset all peripherals and Nios II processors in the system. Alternatively, ensure that any messages posted to the mailbox are cleared before issuing a soft reset.

DMA Controller Always Busy in Burst Mode If Transfers Less Than Full Width

The DMA controller component (`altera_avalon_dma`), when enabled for burst transactions, will not do transfers at widths other than its full data width. The DMA controller is always busy.

Workaround: When bursting is enabled, the DMA controller must be programmed to do transactions at its full data width.

Using the DEV_CLRn Signal Can Cause JTAG UART Instability After Device-Wide Reset

If the `DEV_CLRn` pin on the FPGA input has been assigned (in Quartus II software) to generate a device-wide reset, and the FPGA is reset while the JTAG UART is active, then the JTAG UART might become unstable.

Workaround: Do not use the `DEV_CLRn` function in designs with the JTAG UART. Turn off the **Enable device wide reset (DEV_CLRn)** setting in Quartus II software.

Host Platform

This section lists all issues related specifically to the host platform.

Linux: Process Fails To Terminate While Debugging With the Nios II ISS Target

If you try to interrupt or terminate a debug session targeting the Nios II instruction set simulator (ISS), you might see an error message **Interrupt Failed or Terminate Failed**. This means that the `nios2-iss` process failed to terminate. The debug session appears to have terminated in the IDE, but the `nios2-iss` process still continues running.

Workaround: Open a command shell and kill the `nios2-iss` process as follows:

1. Type:

```
jobs
```

to get a list of process IDs.

2. Type

```
kill -9 <nios2-iss process ID>
```

to terminate the process.

Linux: The Quartus II Stand-Alone Programmer is Not Supported

There is no Quartus II stand-alone programmer for Linux. As a result, in the Nios II IDE the Quartus II Programmer command on the Tools menu has no effect. When you attempt to download software to a board without the expected hardware image, the IDE does not launch the programmer.

Workaround: Launch the Quartus II software to run the Quartus II Programmer.

Windows: Frisk Antivirus Software Causes SOPC Builder and Nios II Command Shell to Be Unresponsive

The SOPC Builder and Nios II Command Shell might become unresponsive if run while the Frisk antivirus software is running.

Workaround: Turn off the Dynamic Virus Checking feature of the Frisk software before running SOPC Builder or the Nios II Command Shell.

Device

This section lists any device-related issues.

Stratix II EP2S60 ES Devices Cannot Use MRAM Byte Enables

Early shipments of the Nios II Development Kit, Stratix II Edition include an EP2S60 engineering sample (ES) device. Stratix II EP2S60 ES devices have a silicon problem that prevents the use of byte enables on MRAM blocks.

Workaround: Refer to the *Stratix II FPGA Family Errata Sheet* for details.

Nios II IDE

This section lists all issues relating to the Nios II IDE.

Building Projects

Build Option Not Functional for BSPs Created Using the Software Build Flow

The build option in the Nios II IDE menu does not rebuild BSPs imported into the IDE.

Workaround: To build the BSP, build the associated application project.

Previous 7.1 Beta Install Might Generate Message 'Internal error occurred during "Launching"'

If you installed a Beta release of Nios II version 7.1 prior to installing the full release, you might encounter the error **An internal error occurred during "Launching"**, when building, running or debugging a software application project in Nios II IDE.

Workaround: Close Nios II IDE, open a Nios II command shell, and type `nios2-ide -clean`. The Nios II IDE launches again, and the error does not recur. Launching Nios II IDE with the `-clean` option does not affect any IDE or project settings you have made.

Altera strongly recommends that you uninstall Beta versions of the 7.1 tools before installing the released version of the Nios II EDS v7.1. Also, remove or rename the existing installation directories.

Different Address Mapping for Master Ports Connected to Dual-Port Memories Can Cause Linker Errors

If your instruction master and data master ports are connected to the same dual-port memory and the ports have different addresses, your code fails to run or you experience a linker error. The Nios II IDE does not warn you of the addressing violation.

Workaround: Assign the same address to both ports of the dual-port memory.

Nios II IDE Does Not Support Settings for Projects Imported From the Nios II Software Build Tools Design Flow

For projects created using the new software build flow and imported into the Nios II IDE, the IDE configuration settings have no effect. For example, `objdump`, `compiler`, and `linker` settings made in the IDE are ignored. This behavior occurs because Nios II software build tools projects are not IDE-managed projects. In addition, the make-related options (at **Window >>> Preferences >>> Nios II** and **Window >>> Preferences >>> Nios II >>> New projects**) do not pertain to imported Nios II software build tools projects. The IDE ignores these options during the build process.

Workaround: Make these settings in the project's makefile.

Using a URL Location for Projects Might Cause an Error Creating Project

You might see an error dialog box saying **Project cannot be created. Reason: Internal Error** when you try to create a new project while pointing to an existing workspace. This error might occur if the path to any project in the workspace is a URL location, for example `file:/F:/Design`. To view the path, right-click on the project and select Properties.

Workaround: Import your existing application and system library projects into a new workspace.

Conflicting File Names Can Create Illegal Project Location Error When Creating New Project

You might receive an **Illegal project location** error message in the IDE if you use the default project name when creating a project in a new workspace. If the project name exists in another workspace, the IDE might not account for that in the new workspace.

Workaround: Change the project name to a name other than the default.

Choosing Separate Exception Stack Option Might Cause Compilation error "UNDEFINED VARIABLE %STACK_POINTER%" When Building a Project

This error occurs if the **Use a separate exception stack option** is turned on, and the exception stack is larger than the memory available for it.

Workaround: On the system library properties page for the project, turn off the separate exception stack or reduce the **Maximum exception stack size** setting.

When Building a Project, the Nios II IDE Reports Problems, But the Build Output in the Console Does Not Contain Any Errors

The Nios II IDE incorrectly reports some linker warnings as errors, with a dialog box saying **Errors exist in a required project**. The Dhrystone software example exhibits this behavior, and recompiling the project again makes the issue go away.

Workaround: If the Console output does not contain errors, then the project actually built fine. On subsequent builds, the linker step is skipped and the errors do not appear, because the project built without error previously.

Makefile reports incorrect number of bytes free for the stack and heap, if the heap is in a different memory region than the stack

Workaround: Do not trust the heap and stack memory report from the makefile if you have placed the heap and stack in different memory regions.

Nios II IDE Unnecessarily Updates the SOPC Builder System File (.ptf)

The Nios II IDE opens the SOPC Builder system file (**.ptf**) by invoking SOPC Builder during certain operations, which might cause SOPC Builder to change the date stamp of the file even though the system is not modified. This behavior might cause problems if you are using a version control system.

Workaround: If you are not using the Nios II C2H Compiler, you can change the PTF file properties to read-only to prevent the IDE from changing the file.

Incorrect Address Assignment for Dual-Port Memory Mastered by Two Different Master Ports

If you have a dual-port memory in your Nios II system, and only the second slave port is mastered by the processor, you might see an overlapping section error during the linking stage of building your software.

Workaround: In SOPC Builder, ensure that the first slave port of the dual-port memory is mastered by the Nios II processor. The processor does not have to master the second port.

Build Errors After Changing Component Names in SOPC Builder

If you rename components in the SOPC Builder system and then regenerate the SOPC Builder system, Nios II IDE system library projects based on that system have build errors.

Workaround: Delete the system library project from the workspace without deleting the contents from the file system, and then import the project again, selecting the regenerated SOPC Builder system. Alternatively, after regenerating the SOPC Builder system, create a new system library project for the SOPC Builder system.

Debugging Projects

GDB Error 256 When Launching Debugger

You might see an `ERROR 256` message when launching the debugger if you have environment variables that are longer than 1200 characters.

Workaround: Ensure that you do not have any environment variable that exceed 1200 characters.

Previous 7.1 Beta Install Might Cause Internal Error When Launching FS2 in IDE

If you have a Beta version of Nios II EDS 7.1 installed on your machine along with the release version of 7.1, you might experience an FS2 launch error. If you have selected **Use FS2 console window for trace and watchpoint support** in the Nios II IDE debugger settings window, you might receive an internal error while launching.

Workaround: Uninstall any Beta version of the Nios II EDS version 7.1. When installing a released version of the software, uninstall any Beta version first, and remove or rename the Beta installation directories before installing the released version.

Extra Arguments in Debugger for Multiple Launch Configurations

If you have multiple debug launch configuration, you might have issues with the **Additional nios2-download arguments** field. If you delete the contents of the **Additional nios2-download arguments** field in one debugger launch configuration, view a second launch configuration, with a nonempty string, in the **Additional nios2-download arguments** text box, and return to the first launch configuration, the value from the second configuration appear in the first configuration.

Workaround: To remove all arguments, use a single space rather than deleting the entire contents of the field.

Console Window Is Not Updated After ISS Error

After performing a **Run as ISS**, if you receive an ISS error in the console window, the console is not updated subsequently.

Workaround: Close the console window after receiving an ISS error. A new console window opens when a new message is available.

Cannot Locate Source Code in Driver Files Shared by Multiple Projects

If you hit a breakpoint in a driver file, and that driver file is shared with another project that is closed, the Nios II IDE might indicate that it cannot locate the source code.

Workaround: Open the closed system library project and resume debugging.

Memory Window Sets Control Register Values Incorrectly

The memory window might incorrectly set values in memory-mapped control registers. For example, writing 0x1234 to a byte addressed register results in the value 0x3434 being stored in the register. The memory window will show this incorrect value.

Workaround: Use the GDB console window in the IDE, instead of the memory window, to write to the registers. For example, type `set {int} <register address>=0x1234` in the GDB console window.



You must refresh the memory window in order for it to correctly display the target value.

ISS Fails on Designs Containing TSE MAC or SG-DMA Components

You will receive an Internal Error when attempting to perform an ISS simulation of designs containing the TSE MAC or SG-SMA components because the Nios II ISS does not support these components.

Workaround: Remove the TSE MAC and SG-DMA components from your system and perform ISS simulation on the simplified system. You can also simulate the design in ModelSim[®] or test it on hardware.

"Run as ModelSim" in the Nios II IDE Fails

The **Run as ModelSim** command might fail on Run configurations created in prior versions of the IDE. This problem will not occur for new Run configurations.

Workaround: select a location for the ModelSim tool from the launch configuration dialog box. You can use the Browse button next to the **ModelSim path** group, or type in a path to the ModelSim directory (e.g. `c:/altera/71/modelsim_ae/win32aloem`).

Unable to Display Driver Source Code When Debugging a Custom Component in the Nios II IDE

When debugging a custom component in the Nios II IDE, the debugger might not find the software driver files for that component.

Workaround: You must add a linked folder to an IDE project so that the IDE can find source code files associated with the component. The locations which you must add appear in a generated file called **install2.ptf** in the directory containing the **install.ptf** for the hardware design. Examine this file to find the directory path which you must add. For example, a custom component called **altera_avalon_pwm** is stored in **/data/pwm_component_test**. An excerpt from **install2.ptf** follows:

```
PACKAGE install2
{
  COMPONENT altera_avalon_pwm
  {
    VERSION 2.0
    {
      local = "/data/pwm_component_test";
    }
  }
}
```

To add the path **/data/pwm_component_test** to the IDE, pick the system library for your design, and right-click the system library project, point to **New >>> Other >>> General >>> Folder**, and click **Next**.

Specify the parent project (e.g. **syslib**) in the **Enter or select the parent folder** field.

Click **Advanced**, turn on **Link to folder in the file system**, and browse to or type in the path to the component.

Click **Finish** to link the folder.

Note: Do not link a custom component to more than one system library in the IDE. The breakpoint manager sometimes fails to handle the aliases, which results in breakpoints being hit even when removed from the breakpoint manager.

The Restart Command on the Run Menu Does Not Work

Workaround: Stop the program, then debug it again. If the debugger is hung in an endless loop, use the following bash alias to break the target, then stop it:

```
alias break="kill -2 `ps -a | grep nios2-elf-gdb | cut -f6 -d' '\`"
```

Programs That Interact With a Terminal Console on Windows Do Not Work

Programs with this behavior work in v6.0 and earlier, but do not work in Nios II IDE v6.1 and later.

The Eclipse platform in v6.1 and later of the IDE (on Windows only) sends the string `\r\n` instead of just `\n` when running or debugging using the Terminal. This behavior can break existing software designs, and it is inconsistent with `nios2-terminal`, which still just sends `\n`.

Workaround: Change the software running on the Nios II processor to parse for `\r` as well as `\n`.

"Step failed. Target is not responding (timed out)" Error Message

The Nios II IDE debugger might hang and report this message if your code contains large arrays declared as local variables on the stack.

Workaround: Place the array and any other large buffers on the heap rather than on the stack.

Incorrect Breakpoint Filtering on Threads

If you enable breakpoint filtering for a thread and later turn off filtering for the thread, the debugger might incorrectly continue to filter the thread.

Workaround: There is no workaround available at this time.

Uninitialized Memory Error When Executing From ISS

In some cases the ISS does not ignore uninitialized memory reads, even when **Uninitialized memory reads** is set to **Ignore** on the ISS Settings tab of the run configuration.

Workaround: There is no known workaround.

nios2-gdb-server Fails to Terminate After Setting a Watch Point

You might be unable to terminate **nios2-gdb-server** after setting a watchpoint in the Nios II IDE debugger and resuming execution past the end of main. You will see an error **Terminate failed**. You will not be able to start the

debugger again; you will see a message reading **Another application is using the target processor...** in the Console view.

Workaround: Terminate the **nios2-gdb-server** process manually using the Windows Task Manager.

Watchpoints Do Not Work on a Variable Whose Size Is Not 32 Bits

Workaround: Change the types of global and static local variables to `int`, `long`, or `unsigned long` before setting watchpoints on them.

Debugger Cannot Step Into `__sflags()`, and Continues Execution Instead

The Nios II IDE debugger is unable to step into some low-level C library functions, such as `__sflags()` (`__sflags()` is called from `_fopen_r()`, called from `fopen()`). If you try to step into such a function, the debugger proceeds as if you had selected **Resume execution**.

Workaround:

- Step over such functions.
- If execution continues after trying to step in, click **Suspend** on the Run menu.

Missing Address and Data in the Trace View

If the trace options **Include load addresses**, **Include store addresses** or **Include data values** are enabled during debug, the load and store address and data will not appear at the first breakpoint in the debugging session. They will appear at subsequent breakpoints.

Workaround: To see load or store addresses and data in the instruction trace prior to `main()`, turn on **Break at `alt_main()`** located on the **Debugger** tab for your debug configuration.

Cannot Use Watchpoints in the Nios II IDE When the FS2 Console Is Open

Hardware watchpoints do not work in the Nios II IDE when the **Use FS2 console window for trace and watchpoint support** setting is turned on in the **Debugger** tab of the Debug configuration. You will see an error message **The execution of program is suspended because of error.** with details indicating that hardware watchpoints could not be inserted and deleted.

Workaround: If the FS2 console is open, you must use FS2 to control watchpoints. For details, see the FS2 documentation located in `<Nios II EDS install path>\nios2eds\bin\fs2\doc`.

Breakpoints on Adjacent Lines of Assembly Fail to Halt the Processor

Setting breakpoints on adjacent lines of assembly code might cause the Nios II processor to stop responding to the debugger. This issue does not affect Nios II cores that do not have hardware breakpoints enabled in the JTAG debug module.

Workaround: When debugging in mixed mode or disassembly view, separate breakpoints by at least one assembly instruction.

Navigating Projects

Nios II IDE Import System Library Dialog Box Does Not Close After Import

When importing a system library project, the Nios II IDE wizard does not close after clicking the **Finish** button. The wizard imports the project successfully after clicking **Finish** once. Clicking **Finish** again results in an error message stating that the project is already created.

Workaround: Click **Cancel** in the **Import Wizard** dialog box after clicking **Finish**.

C/C++ Search Returns Nothing

The C/C++ Search in the Nios II IDE does not return results because a search index is not created at start up.

Workaround: Manually evoke indexing for projects you need to search. Refer to the *Indexing and Searching Project Content* IDE online help topic for instructions.

Internal Error When Double-Clicking on a Large Objdump File

On a Windows PC when opening a large **objdump** file in the Nios II IDE, you might get the following error message: **Unable to create this part due to an internal error. Reason for the failure: Editor could not be initialized.**

Workaround: Adjust the Windows launch arguments for the Nios II IDE editor. Perform the following steps:

1. On the Windows Start menu, browse to the **Nios II EDS 7.1** program icon, right click it, then click **Properties**. The **Windows Properties** dialog box appears.
2. In the **Target** field, append `vmargs -Xmx1024m` to the end of the path to the Nios II IDE executable. For example:

```
C:\altera\70\nios2eds\bin\eclipse\nios2-ide.exe -vmargs -Xmx1024m
```

After Switching Workspaces, the Nios II IDE Appears to Freeze While Displaying the Splash Screen

After clicking **Switch Workspace** on the File menu on a Windows machine, a Nios II IDE splash screen appears. Unfortunately, this splash screen obscures a dialog box which asks you to specify the new workspace.

Workaround: Press Alt-Tab to switch applications. Two relevant application icons appear: an Eclipse icon associated with the splash screen and a Nios II IDE icon associated with the **Workspace** dialog box. Select the Nios II icon to bring the dialog box to the foreground.

C/C++ Scanner Does Not Understand Certain C/C++ Constructs

The C/C++ scanner performs C/C++ Search, navigation, open declaration, and parts of content assist. Due to limitations of the C/C++ scanner, these features do not work with the following code constructs:

- Kernighan & Ritchie-style C
- Functions that take a function pointer as an argument

Workaround: If the C/C++ Search fails, use the File Search facility.

Nios II Software Build Tools

This section lists any issues related to the Nios II software build tools. These are new tools introduced in Nios II EDS version 7.1.

Nios II IDE import settings for Nios II software build tools projects

For projects created using the Nios II software build tools and imported into Nios II IDE projects, the IDE configuration settings are invalid. For example, objdump, compiler, and linker settings made in the IDE have no effect. This behavior occurs because Nios II software build tools projects are not IDE-managed projects. In addition, the make-related options on **Window > Preferences > Nios II and Window > Preferences > Nios II > New projects** do not pertain to imported Nios II software build tools projects, and the IDE ignores them during the build process.

Workaround: Specify these settings in the Nios II software build tools project's makefile.

The add_sw_setting Character Tcl Command Produces Improper Quote Format in system.h

The Nios II software build tools tcl command

```
add_sw_setting character system_h_define character_system_h character_system_h a "character_system_h"
```

generates a **system.h** file with contents in double quotes i.e.: "a", instead of the proper single quotes, i.e.: 'a'.

Workaround: Manually edit the **system.h** file to fix the appropriate line by changing double quotes to single quotes.

Upper-Case File Extensions Not Supported

The Nios II software build tools for applications and libraries (nios2-app-generate-makefile and nios2-lib-generate-makefile commands) do not support source files with certain upper-case extensions. If a file with an upper-case extension is included, the make command stops with no descriptive warning.

Only Nios II assembly language files built by the C preprocessor can have upper-case file extension (.S). All C language files must have the extension .c or .h. C++ language source files must have the extension .cpp, .cxx, .cc, or .h.

Workaround: Rename all C language files with the extension .c or .h. Rename all C++ language files with the extension .cpp, .cxx, .cc, or .h.

"make download-elf" Fails for Stratix II 2S60 RoHS Design Examples

You might see error messages when running the 2S60 RoHS example applications with the Software Build Tools. These messages appear due to an incorrect system ID value in the BSP makefile. The error messages are similar to the following:

```
Reading System ID at address 0x021208B8:
  ID value does not match: read 0x072DA598; expected 0x7D1BD35D
  Timestamp value was not verified: value was not specified
The software you are downloading may not run on the system which is currently
configured into the device. Please download the correct SOF or recompile
```

Workaround: Remove the system ID check from the application makefile. To remove the system ID check, carry out the following steps:

1. Open the application **Makefile**, located in the application project directory.
2. Search for `download-elf` to find the section that needs modification.
3. Remove the `$(SOPC_SYSID_FLAG)` argument from the `$(DOWNLOAD)` command in the following section:

```
.PHONY : download-elf
download-elf : $(ELF)
    @echo Info: Downloading $(ELF)
    $(DOWNLOAD) --go --cpu_name=$(CPU_NAME) $(SOPC_SYSID_FLAG) $(WRITE_GMON_OPTION) $(ELF)
```

The resulting section looks like:

```
.PHONY : download-elf
download-elf : $(ELF)
    @echo Info: Downloading $(ELF)
    $(DOWNLOAD) --go --cpu_name=$(CPU_NAME) $(WRITE_GMON_OPTION) $(ELF)
```

Rebuild the application.

C2H Compiler

This section lists issues related to the Nios II C-to-Hardware Acceleration (C2H) Compiler.

C2H Regenerates Hardware Accelerator Even Though No Logic Changed

This problem can result from changes to files included by the C file containing the accelerated function. C2H fails to check that the HDL generated matches the previously generated HDL causing the system to be regenerated.

Workaround: To avoid this issue move the information in the include file specific to the hardware accelerator into a separate include file. This workaround prevents regeneration of the system when the HDL is unchanged.

Error: `c2h_accelerator_base_addresses.h`: No Such File Or Directory

When a C2H accelerator is compiled for the first time, the following compile-time error can result if the **Analyze all accelerators** option is selected: `c2h_accelerator_base_addresses.h`: No such file or directory.

Workaround: Click **Build software and generate SOPC Builder system** and build once before building with the **Analyze all accelerators** option.

Pointer Dereferences to Volatiles

The C2H compiler treats pointer dereferences to a volatile type as if they alias all other pointer dereferences. Pointers that are restrict-qualified are treated the same way. For example, in the cases below, the two loops cannot

be scheduled concurrently because the volatile qualification requires that a dependency exist between the two dereference might cause incorrect behavior.

```
volatile int * restrict fifo_rd = FIFO_RD_BASE;
volatile int * restrict fifo_wr = FIFO_WR_BASE;
for ()
{
    *fifo_wr = ....;
}
for ()
{
    ... = *fifo_rd;
}
```

Workaround: Divide the function into multiple IRQ-enabled accelerators that are launched concurrently from the CPU, and use FIFOs to communicate between them.

Java Heap Space Exception if Quartus II Compilation is Enabled

You might receive the following error during Quartus II compilation of your design containing a C2H accelerator if you selected the **Build software, generate SOPC Builder system, and run Quartus II compilation** in the C2H Compiler settings window.

```
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
make: *** [c2h_hdl-t] Error 1
```

Workaround: Select the **Build software and generate SOPC builder system** option in the C2H Compiler settings window and launch the Quartus II compilation manually afterward.

C2H Accelerators Time Out Immediately When Untethered

If you do not have a C2H Compiler license and close the OpenCore Plus status box while running a design containing a C2H accelerator, the design times out immediately rather than allowing the one hour grace period.

Workaround: Remain tethered during evaluation and do not close the OpenCore Plus status box. Alternatively, you can purchase a C2H Compiler license.

Accelerator Generation Failure If Tools Are Installed in Path With Spaces

If the path to your installation of the Altera Design Suite contains spaces, the C2H Compiler fails to generate the accelerators.

Workaround: Reinstall the tools to a path containing no spaces.

C2H Compiler Out-of-Memory Error

The C2H Compiler might generate an out-of-memory error at compile time for code that initializes a large local array.

Workaround: Initialize the memory with the Nios II processor and access it through pointers in the accelerator.

C2H Compiler Does Not Accelerate Subfunctions Located in a Separate File

When accelerating a function in a file, the C2H Compiler cannot link subfunctions that are defined in a different file.

Workaround: Include all subfunctions called by the accelerated function within the same source code file.

Incorrect Results From Logical or Conditional Operation With Side-Effects

The C2H Compiler always evaluates both operands of logical (&&, ||) and conditional (?:) operators. This is different from expected ANSI C behavior, for which operands are evaluated left-to-right, and unnecessary operands are skipped. For example, in the expression `(i-- && j--)`, if the value of `i` is zero, the right-hand-side (RHS) expression should not evaluate (i.e., `j` should not be decremented). However, this C2H Compiler erroneously evaluates both sides unconditionally, causing `j` to be decremented. The following example expressions could suffer from the same issue: `(i-- || j--)`, `(cond ? i-- : j--)`

Workaround: Use logical and conditional operations whose operators have no side effects. Side effects include pre-/post-fix increment operations (`++`, `--`), memory operations (`*`, `[]`, `.`, `->`), and function calls.

Closed System Library While Working With the C2H Compiler

The C2H Compiler requires the system library to obtain important details about the system, and cannot function if the system library is closed.

Workaround: Ensure that the system library project in the Nios II IDE is open prior to building an application project that contains a hardware accelerator.

Launch SOPC Builder Button in C2H View

When the Nios II IDE workspace contains multiple projects with multiple system libraries, the incorrect SOPC Builder system might open when you click the Launch SOPC Builder in the C2H view.

Workaround: Keep only one system library project open at a time while using the C2H Compiler.

Build Clean Causes Build Failure

Performing a clean build on a Nios II IDE project that contains a hardware accelerator can cause the next build to fail in the IDE, because the clean build erroneously deletes a file required by the C2H Compiler.

Workaround: Do not perform a clean build on projects that use hardware accelerators. If you have already performed a clean build, recompile with option **Build software, generate SOPC Builder system, and run Quartus II compilation** to regenerate the necessary files.

Multiple Clock Domains Causes Hardware Accelerator to Fail

If a hardware accelerator and the components connected to its master ports are in different clock domains, the accelerator might behave incorrectly.

Workaround: Assign a single clock to a hardware accelerator and all the slave ports it connects to. It is acceptable for the system to contain multiple clock domains.

Array Elements of structs Do Not Copy Correctly

C2H hardware accelerators do not correctly copy array elements of structs. For example:

```
typedef struct my_struct {
    int a;
    int b;
    int buf[BUF_SIZE];
}MY_STRUCT;

MY_STRUCT struct_a = {1, 2, {3, 3, 3, 3}};
MY_STRUCT struct_b = {9, 8, {7, 7, 7, 7}};

struct_a = struct_b;
```

In this example, the `a` and `b` elements of the struct copy correctly, but the `buf` element does not. After this assignment, `struct_a` equals `{9, 8, {3, 3, 3, 3}}`.

Workaround: Copy the array elements explicitly, as follows:

```
{
    int i=0;
    do
    {
        struct_a.buf[i] = struct_b.buf[i];
        i++;
    } while (i<LENGTH_OF_BUF_ELEMENT)
}
```

Simulating on ISS Not Supported

The Nios II instruction set simulator (ISS) does not support custom SOPC Builder components, and therefore cannot simulate systems that use hardware accelerators. You might get the following internal error if attempting to simulate using the ISS:

```
Internal Error (unhandled exception) in file cosim_main.cpp
```

Workaround: Run the program on a hardware system that includes the hardware accelerator.

Hardware Accelerators Remain After Deleting the Software Project

If a system contains C2H hardware accelerators, deleting the software project that defines the accelerators does not remove the accelerators from the hardware system, and the accelerator logic remains in the SOPC Builder system.

Workaround: To remove an accelerator from a system, delete the accelerator from the C2H view in the Nios II IDE first, and then recompile the software project. The C2H Compiler then removes the accelerator from the SOPC Builder system. Once the compilation is complete then the software application can be deleted from the workspace.

Changing Build Configurations Produces Unexpected Results

The C2H Compiler does not support multiple build configurations (e.g. Release or Debug) in the Nios II IDE. After creating one or more accelerators in a particular configuration, the C2H Compiler produces undefined results if you switch to a different build configurations and create more accelerators.

Workaround: For a specific SOPC Builder system and Nios II IDE project, specify C2H accelerators in only one build configuration. You can use multiple build configurations, as long as only one configuration specifies C2H Compiler settings.

Flash Programmer

This section lists any issues relating to the Nios II IDE flash programmer.

Delay When Creating New Flash Programmer Configuration

You might experience a delay of several seconds when creating a new Flash Programmer configuration in the IDE.

Workaround: There is no workaround available at this time.

"No such file or directory" Error for a Project Stored in a Path Containing Spaces

You might receive this spurious error message when programming flash for a project stored in a path containing spaces. The flash programmer does not correctly handle spaces in the directory path. However, this error is benign, because flash programming completes successfully.

Workaround: None required.

elf2flash File Size Limit

The elf2flash utility supports .elf files up to approximately 24 MBytes in size. The elf2flash utility might fail with error **java.lang.OutOfMemoryError** on files larger than 24 MBytes.

Workaround: You can either lower the number of symbols in your elf file by turning off debug symbols, or specify less initialized data in the application.

When Used With a Multiprocessor Nios II System, the Nios II Flash Programmer Might Produce an Error

The Nios II Flash Programmer might produce the following error:

```
There are two or more Nios II processors available which match the values
specified. Please use the configuration dialog to pick one, or specify the --
device and/or --instance parameters on the command line.
```

Workaround: In the Nios II IDE on the **Flash Programmer** dialog box, make sure you specify an appropriate `--instance=<correct instance value>` argument in **Additional nios2-flash-programmer arguments**. If the Quartus II project has been recompiled since the flash configuration's creation, click **Load JDI File**. If using command line mode, add `--instance=<correct instance value>` to the command line. There are two ways to

find the correct value of the instance ID for a processor. The easiest is to use the Nios II IDE to create a sample flash programmer script. Alternatively, open <Quartus II project name>.jdi, in the Quartus II project directory. Locate the Nios II processor node by finding a value of hpath containing <processor module name>=. The instance ID is specified as instance_id.



For further details, refer to the *Nios II Flash Programmer User Guide*.

Download Cables & Debug Hardware

This section lists any issues related to download cables and other debug hardware.

Communication Errors During Run/Debug Sessions Using Older Download Cables

Debugging with the following Altera download cables might fail, due to electrical noise-related JTAG communication failures: USB-Blaster™ Rev A, ByteBlaster™, ByteBlasterMV™, ByteBlaster II, and MasterBlaster™ cables.

Currently, the only fully supported cable for downloading, debugging, or communicating with Nios II systems is the USB-Blaster Rev B cable or later. Revision B cables are clearly labeled as Revision B. (Revision A cables have no revision label.)

Workaround: Use a USB-Blaster Rev B cable. Older cables can be used, but they might encounter JTAG communication failures.

Development Boards

This section lists any issues related to Altera development boards.

Intermittent Failures While Accessing CompactFlash Card

The Nios II Development Kit version 5.0 and higher includes a CompactFlash controller peripheral suitable for interfacing to CompactFlash cards in True IDE mode on Nios development boards. In order for True IDE mode to operate, CompactFlash cards require that the ATASEL_N input be driven to ground during power-up.

The CompactFlash controller peripheral includes a configurable power register used to power-cycle CompactFlash cards in Nios II software through a MOSFET on the Nios development boards. However, in certain development boards, power to the CompactFlash card does not turn off completely during this power cycle operation. Because of this condition, the CompactFlash might not sample the ATASEL_N pin during the power-cycle operation after FPGA configuration when this pin is driven to ground. Instead, the CompactFlash card might sample the ATASEL_N pin when power is first applied to the development board, when I/O are not yet driven by the FPGA (before FPGA configuration).

Workaround: If you encounter errors with CompactFlash when using the Nios development boards, try one of the following:

- Use a different CompactFlash card. Certain cards are more susceptible to the power-cycling issue than others.

- Modify the Nios development board. This is recommended for users who are familiar and comfortable with board-level modifications. Disconnect pin 9 (ATASEL_N) on the CompactFlash socket on your Nios development board and tie this pin to ground.



The CompactFlash socket uses a staggered numbering on the pins (starting from pin 1: 1, 26, 2, 27, ...); refer to the CompactFlash Association specification for right-angle surface-mount connectors for exact specifications on this connector. This modification permanently enables True-IDE mode operation.

Toolchain (gcc, gdb, etc.)

This section lists any issues related to the Nios II compiler toolchain, such as gcc and gdb.

Breakpoints in C++ Constructors Fail to Halt the Processor

Breakpoints set in a C++ constructor might not halt the processor due to a widespread GNU GCC, GDB issue. This is not a Nios II IDE-specific issue.

Workaround: You can work around this issue by moving all of your constructor source code into another class method, called `init`. Then invoke this method from within the constructor.

Target Software

This section lists any issues related to software or drivers that target the Nios II processor.

Lightweight IP (lwIP) Failure When Using DHCP

The Lightweight IP stack might experience intermittent failures when using DHCP to acquire an IP address. When the failure occurs, the following is printed to stdout:

```
Assertion "dhcp_create_request: dhcp->p_out == NULL" failed at line 1283 in
/cygdrive/c/altera/61b169/nios2eds/components/altera_lwip/UCOSII/src/downloads/lwi
p-1.1.0/src/core/dhcp.c
```

Workaround: If possible, Altera recommends switching to the NicheStack TCP/IP Stack - Nios II Edition introduced in Nios II 6.1. If not, the following workarounds can be employed.

- Power-cycle the target board and retry DHCP negotiation, which usually results in correct acquisition of an address from DHCP, assuming that the DHCP server is able to allocate IP addresses.
- Use a static IP address and disable DHCP.

stdin, stdout, and stderr Do Not Work in MicroC/OS-II Applications When Using Small C Library Option

MicroC/OS-II applications using `stdin`, `stdout`, and `stderr` fail to operate when using the small C library.

Workaround: Disable the small C library option.

malloc(), realloc() Failures With MicroC/OS-II

When using the MicroC/OS-II RTOS, calls to `malloc()` and `realloc()` might fail if successive calls to `malloc()` or `realloc()` within a MicroC/OS-II task occur after changing the task priority of the task in which a memory block is originally allocated.

Workarounds:

- Allocate and/or reallocate memory blocks outside of MicroC/OS-II tasks, before task switching starts. This makes it possible to change thread priorities at runtime.
- Allocate fixed areas of memory using arrays (rather than using `malloc()`) before task switching starts. This makes it possible to change thread priorities at runtime.
- Allocate memory using `malloc()` or `realloc()` from a MicroC/OS-II task. You can change task priorities at runtime, but only for tasks that have not used `malloc()` or `realloc()`.

cout From MicroC/OS-II Task Will Not Send Data to stdout

If neither `printf()` or `cout` is used from `main()` before tasks are started, `cout` does not work from a task.

Workaround: Add the following C++ code to the beginning of `main()`:

```
std::ios_base::sync_with_stdio(false);
```

Problems Using HAL Drivers With Toshiba Flash

The HAL CFI Flash driver might not work for Toshiba flash memory that claims to be CFI compliant.

Workaround: In the `altera_avalon_cfi_flash_table.c` file, change the `#define READ_ARRAY_MODE` from `(alt_u8)0xFF` to `(alt_u8)0xF0` and rebuild the project.

Creating New Custom HAL Components in the Nios II IDE

When you first create a component's `inc` directory or HAL header file, you might first need to perform a clean build (i.e., rebuild) of existing system library projects for the new files to be detected.

Legacy SDK

Support for the Legacy SDK mode was removed in version 6.0 of the Nios II Embedded Design Suite.

SOPC Builder and Quartus II Software

This section lists any issues related to the Quartus II software or SOPC Builder that specifically affect Nios II designers.



For further information on the Quartus II software, refer to the latest Quartus II release notes on the Altera web site:

<http://www.altera.com/literature/lit-qts.jsp>

Example Designs

This section lists issues related to the example designs included with the Nios II Embedded Design Suite.

Hardware Designs

Incorrect SSRAM & SDRAM PLL Phase Shift in Stratix II 2s60 RoHS Standard Design

In the standard design targeting the Nios II Stratix II 2S60 (RoHS) development boards, the PLL phase shift used to implement the SSRAM and DDR SDRAM clocks is incorrect. This problem might cause timing violations when reading from or writing to SSRAM or DDR.

If you wish to create a custom design that uses SSRAM on Nios II Stratix II 2S60 (RoHS) edition boards, Altera recommends that you set the PLL phase shift for generating the SSRAM clock to -3.38 ns in Normal PLL mode. If you wish to create a custom design that uses the legacy DDR SDRAM on Nios II Stratix II 2S60 (RoHS) edition boards, Altera recommends that you set the PLL phase shift for generating the DDR write clock to 270 degrees in Normal PLL mode.

To change the PLL settings, perform the following steps:

1. Open the standard design in SOPC Builder.
2. Double click the component instance named **pll** to launch the PLL MegaWizard interface.
3. Click **Launch Altera's ALTPLL MegaWizard** to launch the MegaWizard Plug-In Manager.
4. Select the **Output Clocks** page and then the **clk c1** page, used to generate the SSRAM clock.
5. Ensure that **Clock phase shift** is -3.38 ns.
6. Click the **clk c2** page, used to generate the DDR write clock.
7. Ensure that **Clock phase shift** is 270 degrees.
8. Click **Finish** to save changes and exit the PLL MegaWizard interface.
9. Click **Finish** to save settings to the PLL instance in SOPC Builder.
10. Regenerate the system in SOPC Builder and recompile in the Quartus II software. Refer to the **readme.txt** file in the design example folder for information on compiling designs containing the DDR controller.




For further information refer to **ssram_interface_readme.html**, located in the **<Quartus II installation directory>/sopc_builder/documents** folder. This document discusses the SSRAM timing analysis methodology in detail. Additionally, *AN 411: Understanding PLL Timing for Stratix II Devices* discusses clock phase shift calculations and assignments for PLLs in Stratix II devices.

Incorrect SSRAM PLL Phase Shift in Stratix II 2s60 RoHS and Cyclone II 2C35 TSE SG-DMA Designs

In the TSE SG-DMA design targeting the Nios II Cyclone™ II 2C35 and Nios II Stratix II 2S60 RoHS development boards, the PLL phase shift used to implement the SSRAM clock is incorrect. This problem might cause timing violations when reading from or writing to SSRAM if you increase the system clock speed above that of the example designs (85MHz), while using two clock cycles of read latency in the SSRAM interface.

While most Altera example designs targeting these boards run at 85Mhz, the TSE SG-DMA design targeting the Stratix II 2S60 RoHS board runs at 100MHz. Although no SSRAM failures have been observed with this board and example design in their current configurations, you must regenerate the design for reliable SSRAM operation.

 No action is required if you wish to continue using the example designs at their initial clock speed of 85MHz. However, if you wish to create a custom design that uses SSRAM and runs above 85Mhz, or if you wish to increase the clock speed of an Altera example design above 85MHz, Altera recommends that you adjust the PLL phase shift for generating the SSRAM clock. On Nios II Cyclone II 2C35 and Nios II Stratix II 2S60 RoHS edition boards, set the phase shift to -3.38 ns, in Normal PLL mode.

To change the PLL setting, perform the following steps:

1. Open the example design in SOPC Builder
2. Double click the component instance named **sys_pll** and launch the PLL MegaWizard interface.
3. Click **Launch Altera's ALTPLL MegaWizard** to launch the MegaWizard Plug-In Manager.
4. Select the **Output Clocks** page and then the **clk c2** page, used to generate the SSRAM clock.
5. Change **Clock phase shift** from 4.8 ns to -3.38 ns.
6. Click **Finish** to save changes and exit the PLL MegaWizard interface.
7. Click **Finish** to save settings to the PLL instance in SOPC Builder.
8. Regenerate the system in SOPC Builder and recompile in Quartus II.

For further information, refer to [ssram_interface_readme.html](#), located in the *<Quartus II installation directory>/sopc_builder/documents* folder. This document discusses the SSRAM timing analysis methodology in detail.



Additionally, *AN 411: Understanding PLL Timing for Stratix II Devices* discusses clock phase shift calculations and assignments.

Software Designs

Dhrystone BSP Example Fails When Run on the Fast Hardware Example

You might see the following output when running the Dhrystone BSP example in the IDE after creating and building it via command line.

```
Int_Comp:          36608
  should be:       18
Str_Comp:          a"
nios2-terminal: exiting due to ^D on remote
```

This error is caused by settings in the **create-this-bsp** script that cause the application to overrun the amount of memory available in the Fast design.

Workaround: To run the Dhrystone example successfully with the Software Build Tools, edit the **create-this-bsp** script located in the `<hardware example design>/software_examples/bsp/hal_dhrystone` folder. Carry out the following steps:

1. If the BSP already exists, remove the **settings.bsp** file and run `make clean` in the BSP folder.
2. Change the following line:

```
--set hal.enable_lightweight_device_driver_api true \
```

to:

```
--set hal.enable_reduced_device_drivers true \
```

3. Run the **create-this-bsp** script. (Alternatively, run the **create-this-app** script, found in the Dhrystone application folder.)

The BSP is regenerated with the new setting.

Software Template Error With Non-Ethernet Hardware Design

You might get an Internal Error if your hardware design does not contain an Ethernet MAC, and you try to create a software example from one of the following templates:

- `host_filesystem`
- `simple_socket_server`
- `web_server`
- `zip_filesystem`

These templates check for components in the system library and incorrectly test for the existence of a MAC.

Workaround: Create the system library project for your hardware design before creating your application project from one of the templates. In the system library properties, turn on the Host File System feature by carrying out the following steps:

1. Right-click the system library project and select **Properties**.
2. Click the **Software Components** button, located in the bottom left corner of the **System Library Properties** page.
3. Select **Altera Host Based File System**.
4. Turn on **Add this software component**.
5. Accept the default mount point shown in the **Mount-point** box.

When creating your application project, select this system library.

RAM Test Failure When Running Memory Test Software Template on the ISS

An issue in the instruction set simulator (ISS) model of the JTAG UART can cause a console communication error during the RAM test when running the Memory Test software template on the ISS.

Workaround: There is no workaround available at this time.

Networking Examples

If you are running a networking example design and you are asked for a nine-digit number after the letters **ASJ**, and your Nios II development board does not have a sticker with a nine-digit number after the letters **ASJ**, please enter a unique nine-digit number when prompted. Ensure that this number is unique to each Nios board connected to your network to avoid network address conflicts.

Hardware Simulation

This section lists issues related to simulating Nios II processor systems on an RTL simulator, such as the ModelSim simulator.

Error: "UNC paths are not supported. Defaulting to Windows directory."

If you launch ModelSim from a working directory that is mapped via a UNC path (a path that starts with // instead of drive letter), you receive this error message in SOPC Builder. This error occurs because ModelSim is calling a command shell, which does not support UNC paths.

Workaround: Map the UNC path to a drive letter and use the drive letter to reference the working directory in the launching shell.

Simulation Failure if Reset Address is Set to EPCS

Running ModelSim RTL simulation of a Nios II system fails if the reset address of the Nios II processor is set to an EPCS Serial Flash Controller because there is no simulation model for it.

Workaround: To simulate your system, temporarily set the Reset Address of the Nios II CPU to the memory in which your application code resides (for example, SDRAM), then regenerate the system in SOPC Builder and run RTL simulation again. Before booting the Nios II CPU from EPCS flash on your target board, change the Nios II Reset Address back to the EPCS Controller peripheral and regenerate the system in SOPC Builder and recompile in the Quartus II software to produce an updated FPGA configuration file with the Nios II CPU booting from EPCS flash.

Uninitialized BSS Variables in Simulation

If your program reads the value of an uninitialized BSS variable during HDL simulation, and the BSP (system library) is compiled with the **ModelSim only, no hardware support** property enabled in Nios II IDE, a warning appears about unfiltered data being 'x'. This warning appears because when this property is enabled, the code that clears the BSS memory region is omitted to speed up HDL simulation so this memory region is uninitialized. The BSS region contains global and static local variables that are not initialized by the application so they default to a value of zero. When the Nios II CPU reads uninitialized variables, it displays a warning and converts any of the bits of the uninitialized data to zero which correctly mimics the effect of the missing BSS clearing code. The

HAL code that executes before and after `main()` might use BSS variables, so these warnings might appear even if your application does not use the BSS.

ModelSim Fails to Load Large Memory Models

The ModelSim software might fail to load simulation models for memory arrays larger than 128M bytes, halfwords or words in size. If the sum of the following parameters is greater than 27, the ModelSim software fails to load:

- Address bits (i.e. 14)
- Column bits (i.e. 11)
- $\log_2(\text{banks})$ (number of banks is usually 4, so this term is usually 2)
- $\log_2(\text{chip selects})$ (number of chip selects is usually 1, so this term is usually 0)

Workaround: If you do not need to simulate the entire memory space, simulate using a smaller SDRAM than the SDRAM implemented in hardware.

Documentation Issues

This section lists errors, unclear descriptions, or omissions from current published specifications or product documents.

Hardware Development Tutorial Example Does Not Run on Nios Cyclone 1C20 Board

The `count_binary` application used in the tutorial hangs if run on the 1C20 development board.

Workaround: Modify the `count_binary.c` file by making lines 330 and 331 conditional, as follows:

```
#ifdef LCD_DISPLAY_NAME
    lcd = LCD_OPEN();
    lcd_init( lcd );
#endif
```

Rebuild and run the software again.

Nios II IDE Help Topic - About Nios II IDE Projects on Linux Mozilla

The expandable Help text for the **Nios II C/C++ Application** does not function correctly on Mozilla 1.4.3 under Linux.

Workaround: The missing information is as follows:

A Nios II C/C++ application project contains a C/C++ program, usually including a project's `main()` function. Building a Nios II C/C++ application project results in an executable file (`.elf`) that you can run on target hardware, the Nios II instruction set simulator (ISS), and the ModelSim hardware simulator. A Nios II C/C++ application project depends on a single system library project and might reference functions in a Nios II C/C++ library project.

A Nios II C/C++ application project is a Nios II IDE managed make project. The Nios II IDE creates the necessary makefiles and manages the project for you.



Refer to www.altera.com/literature/ug/ug_nios2_ide_help.pdf for complete text of the Nios II help system.

Arria GX Support Incorrectly Listed in Documentation HTML Page

In the Nios II 7.1 Documentation Launchpad, the **Example Designs** page incorrectly states that a fast design exists targeting the Arria GX family. Although Nios II release 7.1 supports the Arria GX family, it does not include a fast design for Arria GX devices.

Workaround: Use an existing fast design targeting a Stratix II device and change the device settings. For information about device settings, refer to the *Device Page* topic in the Quartus II software online help.

GNU Assembler Does Not Accept the --defsym Flag

According the GNU documentation, you can set an assembler definition by using the `--defsym` flag, but it does not work in the following form: `--defsym MY_VAR=1`

Contact Information

For more information, contact Altera's mySupport website at www.altera.com/mysupport. Click **Create New Service Request**, and click the **Product Related Request** form.

Revision History

Table 1 shows the revision history for the Nios II Embedded Design Suite v7.1 Errata Sheet.

Table 1. Nios II Embedded Design Suite v7.1 Errata Sheet Revision History

Version	Date	Summary
1.0	May 2007	First release
1.1	May 2007	<ul style="list-style-type: none"> • Build Option Not Functional for BSPs Created Using the Software Build Flow • GDB Error 256 When Launching Debugger • "make download-elf" fails for Stratix II 2S60 RoHS Design Examples • Incorrect SSRAM & SDRAM PLL Phase Shift in Stratix II 2s60 RoHS Standard Design • Incorrect SSRAM PLL Phase Shift in Stratix II 2s60 RoHS and Cyclone II 2C35 TSE SG-DMA Designs • Dhrystone BSP Example Fails When Run on the Fast Hardware Example • Internal Error Using Software Templates With Hardware Designs That Do Not Contain an Ethernet MAC • Hardware Development Tutorial Example Does Not Run on Nios Cyclone 1C20 Board • Nios II IDE Help Topic - About Nios II IDE Projects on Linux Mozilla • Arria GX Support Incorrectly Listed in Documentation HTML Page



101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
www.altera.com
Applications Hotline:
(800) 800-EPLD
Literature Services:
literature@altera.com

© 2007 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



