



Nios II Embedded Design Suite 6.0 Service Pack 1 Release Notes

June 2006, Version 6.0 SP1

Release Notes

This document lists the release notes for the Nios® II Embedded Design Suite (EDS) version 6.0 Service Pack 1 (6.0 SP1).

Table of Contents:

New Features & Enhancements	2
Device & Host Support	2
Installation and Licensing Instructions	2
Installing the Nios II Embedded Design Suite Version 6.0 SP1 on Windows	2
Installing the Nios II Embedded Design Suite version 6.0 SP1 on Linux	3
Installing the USB-Blaster Download Cable on Linux	4
Using Previously Installed Versions of the Nios II EDS	4
Licensing	5
Nios II Processor Cores	5
SOPC Builder	5
Nios II IDE	5
Dual-ported memories	5
Nios II C-to-Hardware Acceleration (C2H) Compiler	5
Pipelined memory accesses.....	5
Using the C2H compiler on software projects from previous versions.....	6
Incorrect results when using the C2H Compiler with VHDL designs	6
Signed division by a power of two.....	6
Signed modulo by a power of two	6
Return value cache coherency	7
Flash Programmer	7
Corrected benign errors when booting from Flash with Stratix II.....	7
Target Software	7
Example Designs	7
Hardware Example Designs.....	7
Software Example Designs.....	8

New Features & Enhancements

The Nios II EDS version 6.0 SP1 addresses issues found in previous releases, and adds the following primary feature:

- Example designs for the RoHS-compliant Nios II Development Board, Stratix II Edition.

The sections below provide a detailed list of all product updates.

Device & Host Support

This release supports the following Altera® FPGA families:

- Stratix® II
- Stratix
- Cyclone™ II
- Cyclone

This release supports the following host environments:

- Quartus® II software version 6.0 SP1
- Windows XP Professional, Windows 2000, 32-bit Linux 8.0, and Enterprise 3 (64-bit not supported)
- ModelSim® versions supported on Windows: 6.0e OEM, 6.1d OEM, 6.1b SE, 6.1c SE, 6.1d SE
- ModelSim versions supported on Linux: 6.0e OEM, 6.1d OEM

Installation and Licensing Instructions

This section describes how to install the service pack. There are two modes of delivery for the service pack:

- A service pack download, which must be installed on top of the Nios II EDS 6.0, is available on the Altera website.
- A CD-ROM, which contains the complete installation of the Nios II EDS 6.0 SP1, is included in Nios II development kits.

Installing the Nios II Embedded Design Suite Version 6.0 SP1 on Windows

To install the service pack on a Windows computer, you must have administrative privileges.

Installing the Service Pack Download

To install the service pack on a Windows computer using the installation file downloaded from www.altera.com, perform the following steps. You must first have the Quartus II software version 6.0 SP1 and Nios II EDS v6.0 installed.

1. Exit the Quartus II software and Nios II IDE before continuing.
2. Download the service pack installation file to your hard drive.
3. Run the executable service pack file from your hard drive.
4. Follow the on-screen instructions to install the service pack.

Installing from CD-ROM

To install the Nios II software on a Windows computer using the Nios II Embedded Design Suite CD-ROM, perform the following steps:

1. Insert the compact disc into your CD-ROM drive. The Nios II setup program appears automatically. If the Nios II setup program does not start automatically, browse to the CD-ROM drive in the Windows Explorer, and run the program **launcher.exe** at the top level of the CD.
2. Click **Install Nios II EDS**. The Nios II EDS installer starts and guides you through the installation process.

Installing the Nios II Embedded Design Suite version 6.0 SP1 on Linux

To install the service pack on a Linux workstation, you must have administrative privileges.

Before installing the Nios development tools on Linux, ensure that the shell has the `DISPLAY` environment variable pointing to a valid X server. Otherwise, the installer generates the following error:

```
Updating SOPC Builder components... Exception in thread "main"
java.lang.InternalError: Can't connect to X11 window server using ':0.0' as the value
of the DISPLAY variable.
at sun.awt.X11GraphicsEnvironment.initDisplay(Native Method)
at sun.awt.X11GraphicsEnvironment.<clinit>(Unknown Source)
at java.lang.Class.forName0(Native Method)
at java.lang.Class.forName(Unknown Source)
at java.awt.GraphicsEnvironment.getLocalGraphicsEnvironment(Unknown Source)
at java.awt.Font.initializeFont(Unknown Source)
at java.awt.Font.<init>(Unknown Source)
at socp_wizard.sopc_ui.<clinit>(sopc_ui.java:31)
```

Installing the Service Pack Download

To install the service pack on a Linux workstation using the installation file downloaded from www.altera.com, perform the following steps. You must first have the Quartus II software v6.0 SP 1 and Nios II EDS v6.0 installed.

1. Exit the Quartus II software and Nios II IDE before continuing.
2. Download the platform-specific service pack installation file into a temporary directory.
3. Change directory into the temporary directory.
4. Type the following commands at a command shell:

```
tar -xf niosii_60sp1_linux.tar
cd niosii_60sp1_linux
./install
```

The installation script guides you through the installation process.

Installing from CD-ROM

To install the Nios II EDS on a Linux workstation using the Nios II Embedded Design Suite CD-ROM, perform the following steps:

1. Insert the compact disc into your CD-ROM drive. If the computer has problems reading the CD-ROM, you might have to mount the CD-ROM manually, by typing the following at a command shell:

```
mount /mnt/cdrom
```

2. Type the following commands at a command shell:

```
cd /mnt/cdrom
./install
```

The Nios II EDS installer starts and guides you through the installation process.

Installing the USB-Blaster Download Cable on Linux

To use the USB-Blaster download cable on Linux systems, you need to set up the permissions by adding the following lines to **/etc/hotplug/usb.usermap**. You need to do this before plugging in your USB-Blaster.

```
#
# Altera USB-Blaster
#
usbblaster 0x03 0x09fb 0x6001 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
usbblaster 0x03 0x09fb 0x6002 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
```

Then add the following script as **/etc/hotplug/usb/usbblaster**.

```
#!/bin/sh
# USB-Blaster hotplug script.
# Allow any user to access the cable
chmod 666 $DEVICE
```

This script sets up your USB-Blaster permissions so that any user can access it. Type `chmod +x /etc/hotplug/usb/usbblaster` to make the script executable.

Using Previously Installed Versions of the Nios II EDS

SOPC Builder and the Nios II IDE refer to the most recently installed version of components (such as the Nios II processor and peripherals) and their software drivers. To revert to a prior version of the Nios II development tools, you can reinstall the previous version of tools or modify the following environment variables.

- `SOPC_BUILDER_PATH` - Ensure that `SOPC_BUILDER_PATH` points to the installation directory of the desired Nios II version and no other Nios II versions.
- `SOPC_BUILDER_PATH_<version>` - Ensure that `SOPC_BUILDER_PATH_<version>` points to the installation directory of the corresponding Nios II version.
- `SOPC_KIT_NIOS2` - Ensure `SOPC_KIT_NIOS2` points to the installation directory of the desired Nios II version and no other Nios II versions.

If you have multiple versions of the Quartus II software installed, launch the supported version of Quartus II to ensure that the `QUARTUS_ROOTDIR` environment variable is updated.

Licensing

You can create, compile and generate time-limited FPGA programming files for Nios II hardware systems without obtaining a license file. To generate non-time-limited FPGA programming files and flash programming files, you must obtain a license for the Nios II processor core and the Quartus II software. See the getting started material included with the Nios II Embedded Design Suite. You do not need a license if you will only develop software using the Nios II IDE.

Nios II Processor Cores

This section describes changes to the Nios II processor cores.

There are no updates.

SOPC Builder

For complete revision history of SOPC Builder and the Quartus II software, refer to the release notes for the Quartus II software version 6.0 SP1. The *Quartus II Handbook, Volume 4: SOPC Builder* contains complete documentation for SOPC Builder.

Nios II IDE

This section describes changes to the Nios II integrated development environment (IDE).

Dual-ported memories

In version 6.0, dual-ported memories were treated as if they had only one slave port. The '_S1' and '_S2' portions of generated labels were dropped. This impacted the generated file, **system.h**. In systems containing multiple Nios II CPUs where the CPUs are connected to separate ports of the dual-port memory, **system.h** reported the S1 base address for both CPUs. This issue has been addressed in this service pack so that different base addresses can be used for the CPUs connected to the dual-port memory ports.

Note that if you have a dual-port memory in your SOPC Builder design where only the second slave is mastered by the Nios II CPU, you will see an overlapping section error when linking an ELF. If you ensure that the first slave is the one mastered by the CPU (the second does not have to be) then the linker will work properly.

Nios II C-to-Hardware Acceleration (C2H) Compiler

Pipelined memory accesses

In version 6.0, pipelined memory accesses might stall the pipeline by one cycle causing performance to degrade. This service pack changes the FIFO implementation of the C2H Compiler to correct the problem.

Using the C2H compiler on software projects from previous versions

This service pack fixes an issue with the C2H Compiler hanging during compilation of a project that compiled successfully in a previous (pre-6.0) version.

Incorrect results when using the C2H Compiler with VHDL designs

This service pack fixes an issue with the instantiation of `lpm_clshift` which caused incorrect results when using the C2H Compiler with VHDL designs.

Signed division by a power of two

In version 6.0, if code performing signed division by a power of two was accelerated, incorrect results were generated if the numerator was negative and a non-power of two. This is due to the arithmetic shift right operation being used for division which can lead to incorrect rounding. For example assume that `data_in` is -15. The following code will return the value of -2 instead of the correct result of -1:

```
int foo (int data_in) {
    return (data_in / 8);
}
```

This issue is fixed in this service pack by using a full division circuit. It is recommended to use unsigned numerators if possible, because the full division circuit is only used for signed division by powers of two. If this cannot be avoided, a simple workaround is to convert signed values to unsigned first, perform the division by a power of two, then restore the quotient back to a negative value if required.

The following code will return -1 if `data_in` is -15. The optimizations below will avoid the creation of a full division circuit since the numerator is an unsigned data type:

```
int foo (int data_in) {
    unsigned int temp_data_in;

    temp_data_in = (data_in < 0) ? (0 - data_in) : data_in;
    return ((data_in < 0) ? (0 - (temp_data_in / 8)) : (temp_data_in / 8));
}
```

Signed modulo by a power of two

In version 6.0, if code performing signed modulo by a power of two was accelerated, incorrect results were generated if the numerator was negative. This is due to the result not being sign extended. For example assume that `data_in` is -15. The following code will return the value of 1 instead of the correct result of -7:

```
int foo (int data_in) {
    return (data_in % 8);
}
```

This issue is fixed in this service pack by using a full division circuit. It is recommended to use unsigned numerators if possible, because the full division circuit is only used for signed modulo by powers of two. If this cannot be avoided a simple workaround is convert signed values to unsigned first, perform the modulo by a power of two, then restore the result back to a negative value if required. The following is an example of a signed modulo operation of `data_in % 8`:

```
int foo (int data_in) {
    unsigned int temp_data_in;
```

```
temp_data_in = (data_in<0) ? (0-data_in) : data_in;
return ((data_in<0) ? (temp_data_in % 8) : (temp_data_in % 8));
}
```

Return value cache coherency

In version 6.0, if code with a return value is accelerated, it might return incorrect results under certain circumstances. This problem is due to the C wrapper file not using cache-bypassing reads for the Nios II processor to read the return value from the hardware accelerator. This service pack fixes this issue.

Flash Programmer

This section describes changes to the flash programmer in the Nios II IDE.

Corrected benign errors when booting from Flash with Stratix II

The following benign errors are no longer displayed by the Flash Programmer when targeting a Stratix II device and booting from flash.

```
(SEVERE) elf2flash: Read error: File not found: c:/alt
era/kits/nios2_60/components/altera_nios2
(SEVERE) elf2flash: Error reading boot copier
(SEVERE) elf2flash: Error generating Flash file, exiting
```

Target Software

This section describes changes to Altera-provided target software which runs on the Nios II processor, such as the hardware abstraction layer (HAL) system library.

There are no updates.

Example Designs

Hardware Example Designs

Examples added for lead-free Nios II Development Board, Stratix II Edition

Example hardware designs for the RoHS-compliant Nios Development Board, Stratix II Edition are included in the `<Nios II EDS install path>/examples/<verilog or vhdl>/niosII_stratixII_2s60_rohs` directory.

LAN component updated in example designs for the Nios Development Board, Cyclone II Edition

The example designs were updated to use the latest LAN91C111 Ethernet component. This new component reflects the following updated timing parameters:

```
Read_Wait_States = "20ns";
Write_Wait_States = "20ns";
Setup_Time = "20ns";
```

```
Hold_Time = "20ns";
```

Previous values:

```
Read_Wait_States = "175ns";  
Write_Wait_States = "175ns";  
Setup_Time = "10ns";  
Hold_Time = "5ns";
```

The examples for other development boards already contain the correct, updated component.

Software Example Designs

There are no updates.