40. System ID Peripheral Core

40.1. Core Overview

The system ID core with Avalon interface is a simple read-only device that provides Platform Designer systems with a unique identifier. Nios II processor systems use the system ID core to verify that an executable program was compiled targeting the actual hardware image configured in the target FPGA. If the expected ID in the executable does not match the system ID core in the FPGA, it is possible that the software will not execute correctly.

40.2. Functional Description

The system ID core provides a read-only Avalon Memory-Mapped (Avalon-MM) slave interface. This interface has two 32-bit registers, as shown in the table below. The value of each register is determined at system generation time, and always returns a constant value.

Table 419. System ID Core Register Map

Offset	Register Name	R/W	Description
0	`id`	R	A unique 32-bit value that is based on the contents of the Platform Designer system. The id is similar to a check-sum value; Platform Designer systems with different components, different configuration options, or both, produce different id values.
1	`timestamp`	R	A unique 32-bit value that is based on the system generation time. The value is equivalent to the number of seconds after Jan. 1, 1970.

There are two basic ways to use the system ID core:

- Verify the system ID before downloading new software to a system. This method is used by software development tools, such as the Nios II integrated development environment (IDE). There is little point in downloading a program to a target hardware system, if the program is compiled for different hardware. Therefore, the Nios II IDE checks that the system ID core in hardware matches the expected system ID of the software before downloading a program to run or debug.
- Check system ID after reset. If a program is running on hardware other than the expected Platform Designer system, the program may fail to function altogether. If the program does not crash, it can behave erroneously in subtle ways that are difficult to debug. To protect against this case, a program can compare the expected system ID against the system ID core, and report an error if they do not match.

40.3. Configuration

The `id` and `timestamp` register values are determined at system generation time based on the configuration of the Platform Designer system and the current time. You can add only one system ID core to an Platform Designer system, and its name is always `sysid`.

After system generation, you can examine the values stored in the `id` and `timestamp` registers by opening the IP Parameter Editor for the System ID core.

Since a unique `timestamp` value is added to the System ID HDL file each time you generate the Platform Designer system, the Intel Quartus Prime software recompiles the entire system if you have added the system as a design partition.

Note: In Intel Quartus Prime Pro Edition, the Platform Designer generation process needs an additional TCL script for manual execution to have a unique `timestamp` value.

Follow these steps to have a unique `timestamp` value using the Intel Quartus Prime Pro Edition version:

1. Create a Platform Designer system with the System ID core.
2. Go to **View**>**System Scripting**. Click on **User Scripts** and direct to `soceds/examples/hardware/<*_ghrd folder>/update_sysid.tcl` (you may also copy the script into your local directory and point to it).
3. Click on **Run Script**.
4. After you run the script, the IP Parameter Editor window appears. Ensure that no error message is shown under **System Scripting Messages** box.
5. To save changes to your `.ip` file, select **File** > **Save** option.
6. Select **File** > **Open** to open your main QSYS system. Switch to**System** tab of the Open window, navigate and select your main QSYS system file in the **Platform designer system** field (on the bottom).

   Note: You must change **Files of Type** field to *All Files (*.qsys)*, then click **Open**.
7. Click on **Sync System Info** in Platform Designer.
8. Click on **Generate HDL**.

40.4. Software Programming Model

This section describes the software programming model for the system ID core. For Nios II processor users, Intel provides the HAL system library header file that defines the System ID core registers.

The System ID core comes with the following software files. These files provide low-level access to the hardware. Application developers should not modify these files.

- `alt_avalon_sysid_regs.h`—Defines the interface to the hardware registers.
- `alt_avalon_sysid.c, alt_avalon_sysid.h`—Header and source files defining the hardware access functions.

  Intel provides one access routine, `alt_avalon_sysid_test()`, that returns a value indicating whether the system ID expected by software matches the system ID core.

40.4.1. alt_avalon_sysid_test()

Prototype:	alt_32 alt_avalon_sysid_test(void)
Thread-safe:	No.
Available from ISR:	Yes.
Include:	<altera_avalon_sysid.h>
Description:	Returns 0 if the values stored in the hardware registers match the values expected by software. Returns 1 if the hardware timestamp is greater than the software timestamp. Returns -1 if the software timestamp is greater than the hardware timestamp.

40.5. System ID Core Revision History

Document Version	Intel Quartus Prime Version	Changes
2018.07.30	18.0	Corrected the steps to have a unique `timestamp` value in *Configuration* section.
2018.05.07	18.0	Added the steps to have a unique `timestamp` value in *Configuration* section.

Date	Version	Changes
July 2014	2014.07.24	Removed mention of SOPC Builder, updated to Platform Designer
December 2010	v10.1.0	Removed the "Device Support", "Instantiating the Core in SOPC Builder", and "Referenced Documents" sections.
July 2010	v10.0.0	No change from previous release.
November 2009	v9.1.0	Added description to the Instantiating the Core in SOPC Builder section.
March 2009	v9.0.0	No change from previous release.
November 2008	v8.1.0	Changed to 8-1/2 x 11 page size. No change to content.
May 2008	v8.0.0	No change from previous release.