Summary
This application note explains the steps required to validate the Xilinx LogiCORE™ Aurora
64B/66B IP core on the Kintex®-7 FPGA KC705 Evaluation Kit. Aurora 64B/66B is a scalable,
lightweight, high data rate, link-layer protocol for high-speed serial communication. Aurora is
designed to enable easy implementation of Xilinx transceivers using an intuitive wizard
interface. The Aurora protocol specification is open and available upon request. The Aurora
core is available free of charge in the Vivado® IP catalog and is licensed for use in Xilinx silicon
devices.

Aurora is typically used in applications where other industry standard serial interfaces are
either too complex or resource intensive. Aurora delivers a low-cost, high data rate, scalable
and flexible means to build a serial data channel. Its simple framing structure can be used to
encapsulate data from existing protocols, and electrical requirements are compatible with
commodity equipment. Aurora can be used to provide increased performance without high
FPGA resource costs, software redevelopment, or exotic physical infrastructure.

The reference design is targeted for the Xilinx Kintex-7 FPGA KC705 evaluation board.

Included Systems
The reference design is created and built using the Vivado Design Suite: System Edition
2014.1. The Vivado Design Suite helps simplify the task of instantiating, configuring, and
connecting IP blocks to form complex integrated systems. The design also includes VIO and
ILA cores to probe the signals.

Introduction
This application note details the steps required to configure the Aurora 64B/66B core with
Vivado Design Suite and to validate the operation of the core in simplex mode using the VIO
and ILA cores to probe various signals.

The example presented is a single-lane simplex configuration using two platforms (see
Figure 1). The completed example design can be used to form a building block for more
complex systems.

The example test setup uses two clock sources to generate the 156.25 MHz clock signals. Any
suitable conditioned 156.25 MHz clock source can be used to replicate these examples.
Hardware Requirements

The single-lane simplex configuration requires these hardware components:

- Two Kintex-7 FPGA KC705 evaluation boards
- Two KC705 Universal 12v power adapters
- Two suitable clock generators to generate 156.25 MHz
- Two JTAG platform USB cables
- Four SMA to SMA connector cables (for reference clock)
- Two SMA to SMA connector cables (for serial data)

Software Requirements

Software requirements for the Aurora 64B/66B simplex example design:

- Vivado Design Suite 2014.1

Building Hardware

Simplex Example Design

Customizing the Aurora Core

Follow these steps to customize and generate the Aurora 64B/66B core for the simplex example design:

1. Launch Vivado Design Suite.
2. Select Create New Project and click Next (Figure 2).
3. Select the project name and path and click **Next** (Figure 3).

4. Select **RTL Project** to permit running the example design and check **Do not specify sources at this time** (Figure 4). Click **Next**.
5. Click `xc7k325tffg900-2` or, select the **Boards** option and then click **Kintex-7 KC705 Evaluation platform** (Figure 5).

6. Click **Next**, then click **Finish**.

7. Under Project Manager in the Flow Navigator panel, select **IP catalog** and search for **Aurora 64B66B**. The Aurora cores can be found under **Communication & Networking > Serial Interfaces** (Figure 6).
8. Right-click **Aurora 64B/66B** and select **Customize IP**.

*Figure 6: Aurora 64B/66B Core in IP Catalog*
9. In the **Core Options** tab of the Customize IP window, set these options (see Figure 8):
   - Set **Line Rate (Gbps)** to **3.125** and **GT Refclk (MHz)** to **156.250**
   - Set **Dataflow Mode** to **TX-only Simplex** or **RX-only Simplex**, depending on the platform being configured
   - Set **Interface** to **Framing** and **Flow Control** to **None**.
   - Select the **Vivado Lab Tools** option.
10. Click the **GT Selections** tab.

11. Change the default setting in the lower list box for GTXQ0 from **1** to **X**.

12. Change the lower list box setting for GTXQ2 from **X** to **1** *(Figure 9)*.

**Note:** The GTXQ2 transceiver is the only transceiver pinned out to SMA connectors on the KC705 board. When placing the cursor over the list box setting, a tooltip appears to verify the location of the selected transceiver.
13. Options on the **Shared Logic** tab should remain at default values. Click **OK**.
14. In the Generate Output Products window, click **Generate**.

**Synthesizing the Example Design**

1. When product generation is complete, in the Project Manager section of the Vivado IDE, right-click the core name and select **Open IP Example Design** (see Figure 10).

![Figure 10: Open IP Example Design](image)

2. Click **OK** to overwrite the existing example design.

3. In the newly-opened Vivado IDE window, expand the Constraints entry in the Sources panel of the Project Manager section. Right-click the constraints file (aurora_64b66b_0_exdes.xdc) and select **Open file** (Figure 11).
4. Locate the two 50 MHz board clock constraints (see Figure 11).

5. To accommodate the onboard 200 MHz clock, change the clock period from 20 ns to 5 ns. The corrected constraint statements should appear as:

   ```
   create_clock -name TS_INIT_CLK -period 5 [get_ports INIT_CLK_P]
   create_clock -name TS_INIT_CLK -period 5 [get_ports INIT_CLK_N]
   ```

6. Assign the pin locations for the Aurora core ports to those shown in Table 1 (see Figure 12).
7. This example contains unconstrained pins. To permit bitsream file generation, add this line to the end of the constraints file (Figure 12):

```vhdl
set_property BITSTREAM.General.UnconstrainedPins {Allow} [current_design]
```

**Caution!** Spelling is critical. Double-check changes to the constraints file before proceeding.

8. Right-click within the constraints file editor window and select **Save File**. Close the constraints file editor window.

9. Select **Generate Bitstream** from the Flow Navigator panel.

---

**Figure 12:** Aurora 64B/66B Simplex LOC Constraints

**Table 1:** Aurora 64B/66B Simplex Constraints

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Loc Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>INIT_CLK_N</td>
<td>AD11</td>
</tr>
<tr>
<td>INIT_CLK_P</td>
<td>AD12</td>
</tr>
<tr>
<td>RESET</td>
<td>AG5</td>
</tr>
<tr>
<td>PMA_INIT</td>
<td>AC6</td>
</tr>
<tr>
<td>TX_CHANNEL_UP/RX_CHANNEL_UP</td>
<td>AA8</td>
</tr>
<tr>
<td>TX_LANE_UP/RX_LANE_UP</td>
<td>AB8</td>
</tr>
<tr>
<td>GTXQ2_N</td>
<td>J7</td>
</tr>
<tr>
<td>GTXQ2_P</td>
<td>J8</td>
</tr>
</tbody>
</table>
Executing the Reference Design in Hardware

10. Click Yes to launch Synthesis and Implementation and proceed with bitstream file generation.

11. Repeat the steps under Customizing the Aurora Core and Synthesizing the Example Design to generate the bitstream file for the alternate platform:
   - Set Dataflow Mode to TX-only Simplex for the transmit platform
   - Set Dataflow Mode to RX-only Simplex for the receive platform

Setting up the Simplex Example Design

This example illustrates a single-lane Aurora 64B/66B simplex connection between two platforms (see Figure 1, page 2). The platforms consist of two Kintex-7 FPGA KC705 Evaluation Kit boards shown in Figure 13.

Figure 13: KC705 Board Features

In these instructions, numbers in parentheses correspond to callout numbers in Figure 13. Make these connections using the SMA to SMA connector cables.

- Connect TXP from board 1 (4) to RXP of board 2 (5).
- Connect TXN from board 1 (7) to RXN of board 2 (6).
- Connect CLKP from clock source 1 to MGT CLK P of board 1 (2).
- Connect CLKN from clock source 1 to MGT CLK N of board 1 (3).
- Connect CLKP from clock source 2 to MGT CLK P of board 2 (2).
- Connect CLKN from clock source 2 to MGT CLK N of board 2 (3).
- Connect a JTAG platform USB cable from the host PC to the platform cable header of board 1 (1).
- Connect a JTAG platform USB cable from the host PC to the platform cable header of board 2 (1).
• Connect a KC705 Universal 12v power adapter cable to the power connector (9) of both boards.

• Set the power switch (8) of both boards to the ON position.

The completed setup should resemble that shown in Figure 14.

**Note:** Separate clock sources should be used for each board.

---

**Figure 14: Aurora 64B/66B Simplex Setup**

**Setting Up the Simplex Example Hardware Session**

**Programming the Devices**

1. On completion of bitstream generation, select **Flow > Open Hardware Manager** (Figure 15).
2. At the top of the Hardware Manager panel (see Figure 16), click **Open a new hardware target** and Click **Next**.

**Figure 15**: Open Hardware Manager
3. Select **Local server** and click **Next** (Figure 17).

**Note:** This operation assumes the hardware target is connected to the host PC running Vivado Design Suite. It is possible to connect the hardware target to a second, networked host PC using the Vivado CSE Server application. For details, see the *Vivado Design Suite User Guide: Programming and Debugging* (UG908), [Ref 4].
4. On the Select Hardware Target page, set the JTAG Clock Frequency for both boards to 750000 Hz (Figure 18).

5. Highlight the target board to be programmed and click Next, then Finish.

6. In the Hardware panel, click the active device, XC7K325T_0(0) (Active).
7. In the Hardware Device Properties panel, set Programming file to the bitstream file name for the receive platform (aurora_64b66b_0_exdes.bit) and set Probes file to the appropriate .ltx probes file name (debug_nets.ltx). See Figure 19.

8. Right-click the device in the Hardware list and select Program Device... (Figure 20). Ensure that the bitstream file path and name are correct and click OK.
9. When programming completes, right-click the programmed target device in the Hardware list and select **Close Target** (Figure 21).

10. Right-click the second target platform in the Hardware list and select **Open Target** (Figure 22).

---

**Figure 21:** Close Target

**Figure 22:** Open Target
11. Repeat step 6 and step 7 using the bitstream file name for the transmit platform and the appropriate .ltx probes file name.

12. Repeat step 8 to program the device.

13. When programming completes, right-click the programmed target device in the Hardware list and select **Refresh Device** (Figure 23).
Executing the Design

1. Right-click the device in the Hardware list and select Run Trigger (Figure 24).

Figure 23: Refresh Device
2. In the waveform window that appears, observe a High state on the `lane_up_vio_usrclk` and `tx_channel_up_i` signals.

3. Control-click to select these signals in the Debug Probes list under `hw_vio_1`:
   - `channel_up_in_initclk`
   - `lane_up_vio_i`
   - `gtreset_from_vio_i`
   - `sysreset_from_vio_i`

4. Right-click a highlighted signal and select Add Probes to VIO Window (Figure 25).
5. Toggle the reset signals by clicking the value field for each signal (See Figure 26). Enter 1 or 0 and click OK.

6. The `channel_up_in_initclk` and `lane_up_vio_i` signals should go Low, then return High after each reset signal is toggled.
Follow these steps to view the results of the reset signals in the waveform display:

1. Set one of the reset signals High.
2. Right-click the device in the Hardware list and select Run Trigger.
3. Click the waveform display tab and observe the results of the reset signal shown in Figure 27.
4. Repeat step 2 and step 3 after each change to the reset signals to observe the results.

The preceding steps attempt to demonstrate that when either `sysreset_from_vio_i` or `gtreset_from_vio_i` are asserted, both `channel_up_in_initclk` and `lane_up_vio_i` go Low as the core (or transceiver) is in reset state. However, when both `sysreset_from_vio_i` and `gtreset_from_vio_i` are Low, the core is out of reset state and both `channel_up_in_initclk` and `lane_up_vio_i` are High.

**Figure 27:  Reset Signal Results in Waveform**
Table 2 shows the reference design checklist.

### Table 2: Reference Design Checklist

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td></td>
</tr>
<tr>
<td>Target devices (stepping level, ES, production, speed grades)</td>
<td>Kintex-7 XC7K325T-2FFG900</td>
</tr>
<tr>
<td>Source code provided</td>
<td>Yes</td>
</tr>
<tr>
<td>Source code format</td>
<td>VHDL/Verilog (some sources encrypted)</td>
</tr>
<tr>
<td>Design uses code/IP from existing Xilinx application note/reference designs, Vivado IP Catalog, or third party</td>
<td>Reference design provided by Aurora core generated from Vivado IP catalog</td>
</tr>
<tr>
<td><strong>Simulation</strong></td>
<td></td>
</tr>
<tr>
<td>Functional simulation performed</td>
<td>No</td>
</tr>
<tr>
<td>Timing simulation performed</td>
<td>No</td>
</tr>
<tr>
<td>Test bench used for functional and timing simulations</td>
<td>N/A</td>
</tr>
<tr>
<td>Test bench format</td>
<td>N/A</td>
</tr>
<tr>
<td>Simulator software/version used</td>
<td>N/A</td>
</tr>
<tr>
<td>SPICE/IBIS simulations</td>
<td>No</td>
</tr>
<tr>
<td><strong>Implementation</strong></td>
<td></td>
</tr>
<tr>
<td>Synthesis software tools/version used</td>
<td>Vivado Design Suite 2014.1</td>
</tr>
<tr>
<td>Implementation software tools/versions used</td>
<td>Vivado Design Suite 2014.1</td>
</tr>
<tr>
<td>Static timing analysis performed</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Hardware Verification</strong></td>
<td></td>
</tr>
<tr>
<td>Hardware verified</td>
<td>Yes</td>
</tr>
<tr>
<td>Hardware platform used for verification</td>
<td>Kintex-7 FPGA KC705 evaluation kit</td>
</tr>
</tbody>
</table>

### Conclusion

The Kintex-7 FPGA KC705 Evaluation Kit provides an excellent platform to implement and test the LogiCORE IP Aurora 64B/66B core. Following the procedure outlined in this application note, Aurora 64B/66B simplex designs can be verified and extended for specific applications. Various configurations can be quickly evaluated using only the KC705 board, a clock source and the Vivado Design Suite.

### References

This application note uses these references:

1. LogiCORE IP Aurora 64B/66B Product Guide (PG074)
2. Kintex-7 FPGA KC705 Evaluation Kit Getting Started Guide (UG883)
Revision History

The following table shows the revision history for this document.

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Description of Revisions</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/01/2015</td>
<td>1.0.1</td>
<td>Corrected “Aurora 8B/10B” to be “Aurora 64B/66B” in the Conclusion.</td>
</tr>
<tr>
<td>01/09/2015</td>
<td>1.0</td>
<td>Initial Xilinx release.</td>
</tr>
</tbody>
</table>

Notice of Disclaimer

The information disclosed to you hereunder (the “Materials”) is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at [http://www.xilinx.com/warranty.htm](http://www.xilinx.com/warranty.htm); IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: [http://www.xilinx.com/warranty.htm#critapps](http://www.xilinx.com/warranty.htm#critapps).

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.