WWW.ESISTREAM.COM

ESIstream

The **E**fficient **S**erial **I**nterface

Migrate the ESIstream_Xilinx_KU060_V2-2 IP package to a KU115 FPGA
Application Note – AN221

Version 1.0

ESIstream is license-free high efficiency serial interface protocol based on 14b/16b encoding.
Its main benefits are low overhead and ease of hardware implementation.

Introduction

This document aims at explaining how to migrate VHDL projects provided in the ESIstream_Xilinx_KU060_V2-2 IP package dedicated for the KU060 to a KU115. KU115 and KU060 are parts of the same Xilinx Kintex UltraScale FPGA family.

To simplify and to speed up ESIstream IP implementation, this an ESIstream IP package provides:

- VHDL sources.
- Fully implementable projects.
- Test-bench for simulation.

Although each ESIstream IP package is specific to a Xilinx FPGA, the modular architecture allows an easy migration to a different Xilinx target replacing or upgrading the FPGA manufacturer specific IPs.

For technical support, please get the team involved and contact us using ESIstream contact web page or at GRE-HOTLINE-BDC@Teledyne.com

Package supported by this document

ESIstream_Xilinx_KU060_V2-2, tested on ADAS-SDEV-KIT2 KU060 FPGA evaluation kit.

Reference documents

ESIstream_Xilinx_KU060_V2-2 IP package design and user guide: download link
ESIstream Protocol specification V2.0: www.ESIstream.com
EV12AQ600 datasheet: website link, download link
Xilinx UltraScale FPGA Product Tables and Product Selection Guide: download link

ESIstream protocol initiated by Teledyne-e2v is born from a severe need of the following combination:

- Increase rate of useful data, when linking data converters operating at GSPS speeds with FPGAs on a serial interface, reducing data overhead on serial links, as low as possible.
- Simplified hardware implementation, simple enough to be built on RF SiGe technologies.

ESIstream provides an efficient High-Speed serial 14B/16B interface. It is **license-free** and supports in particular serial communication between FPGAs and High-Speed data converters.

An ESIstream system is made up of the following elements.

- A transmitter can be an ADC or an FPGA or an ASIC
- A receiver can be a DAC or an FPGA or an ASIC
- A number of lanes (L ≥ 1) to transmit serial data
- A synchronization signal (sync) to initialize the communication.

There is no clock lane in a serial interface. For each lane, the receiver should recover the clock from the data.

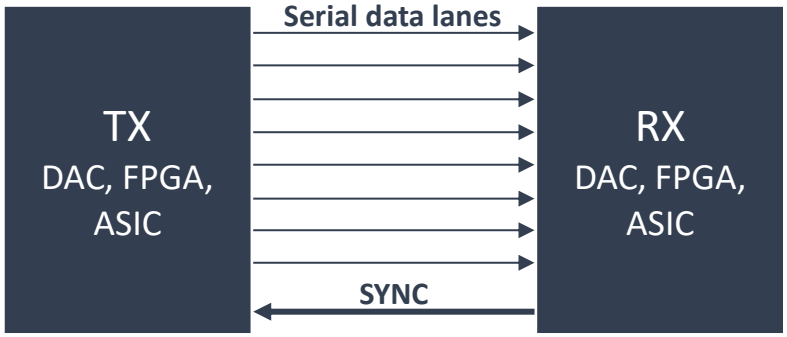


Figure 1: Basic ESIstream system

The main key benefits are:

- FLEXIBILITY: License free
- EFFICIENCY: 87.5%, with 14-bit of useful data and 2-bit overhead (clock bit and disparity bit).
- SIMPLICITY: Minimal hardware implementation.

The main key features of ESIstream are:

- Deterministic latency
- Multi-device synchronization
- Demonstrated lane rate up to 12.8Gbps (on Kintex Ultrascale KU040), depending on device abilities
- Multi-lanes synchronization
- Guaranteed DC balance transmission, ±16 bit running disparity
- Synchronization monitoring, using the clock bit (overhead bit)
- Sufficient number of transitions for CDR, max run length of 32.

Disclaimer

This is free and unencumbered software released into the public domain.

Anyone is free to copy, modify, publish, use, compile, sell, or distribute this software, either in source code form or as a compiled bitstream, for any purpose, commercial or non-commercial, and by any means.
In jurisdictions that recognize copyright laws, the author or authors of this software dedicate any and all copyright interest in the software to the public domain. We make this dedication for the benefit of the public at large and to the detriment of our heirs and successors. We intend this dedication to be an overt act of relinquishment in perpetuity of all present and future rights to this software under copyright law.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
THIS DISCLAIMER MUST BE RETAINED AS PART OF THIS FILE AT ALL TIMES.

TABLE OF CONTENTS

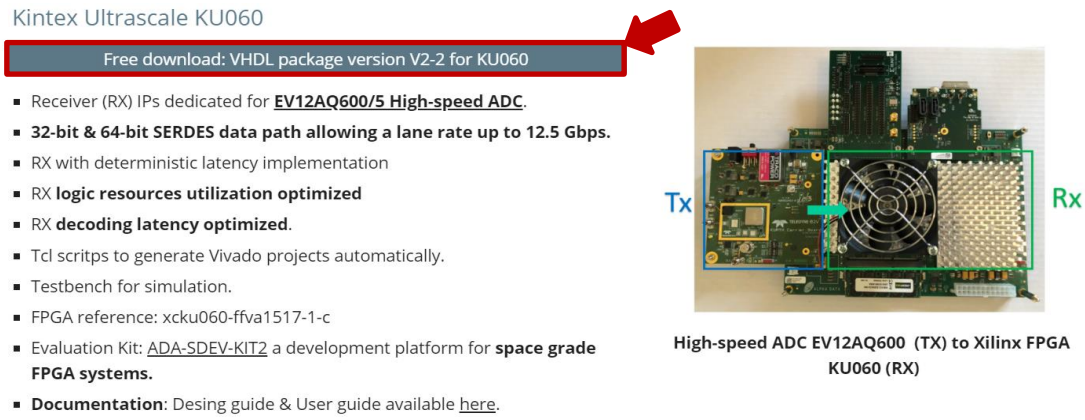
FIGURES

Documents Amendment

Issue	Date	Comments
1.0	May 2020	Creation and publication

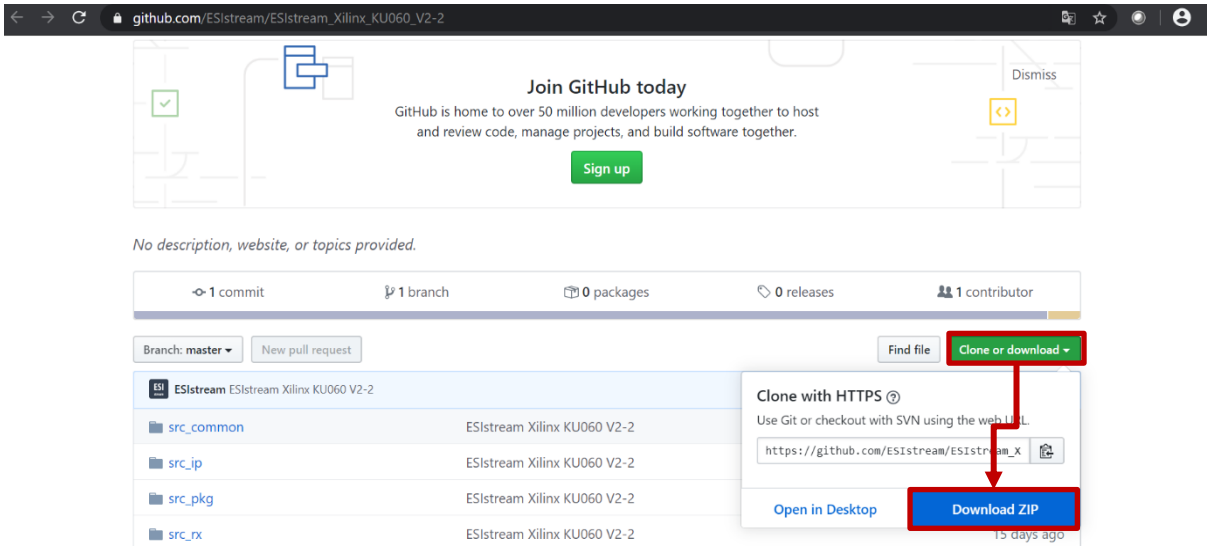
1. PACKAGE MIGRATION PROCEDURE

1. Download the latest ESIstream KU060 package version on ESIstream website:
   A. https://www.esistream.com/ip-package
   B. Click on the ESIstream “Free download link”:



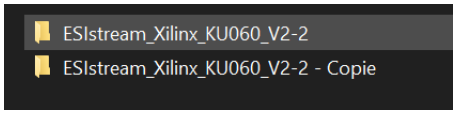
High-speed ADC EV12AQ600 (TX) to Xilinx FPGA KU060 (RX)

   C. Then, on github website, download the .zip file.

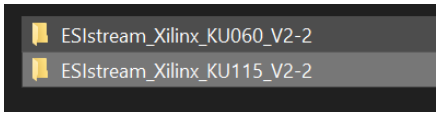


   D. Unzip the ESIstream IP package.

2. Copy paste the KU060 IP package:



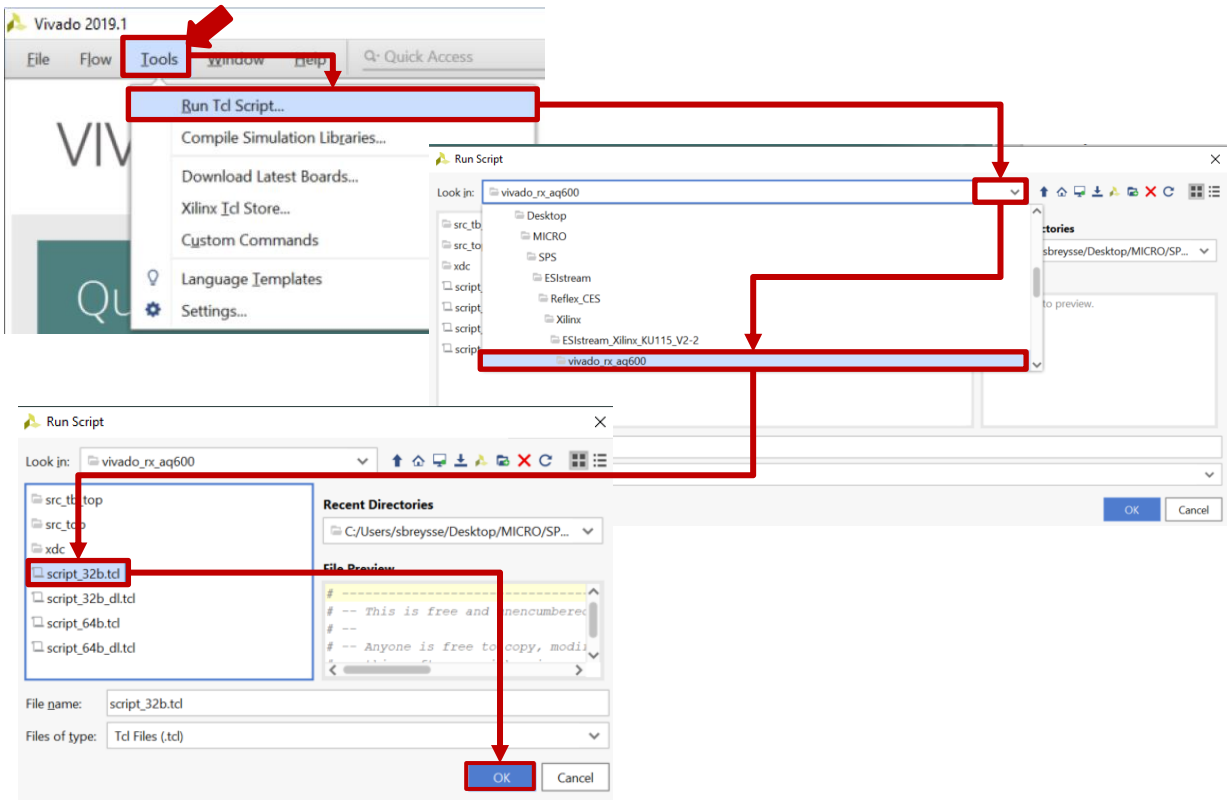
3. Rename the folder package, for instance in *ESIstream_Xilinx_KU115_V2-2*:



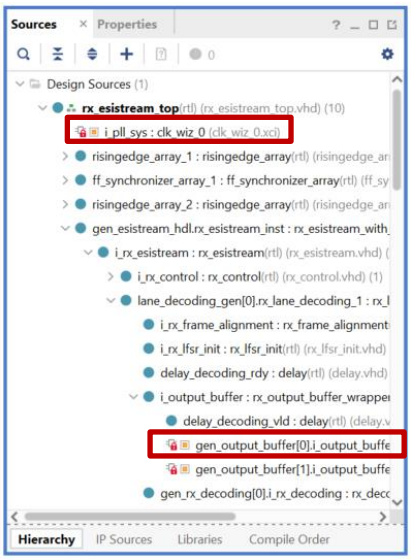
4. In *ESIstream_Xilinx_KU115_V2-2\vivado_rx_aq600* folder
    A. Select and open one of the TCL script depending on your need:
        I. Low logic resources: prefer 32-bit transceiver user data path (script_32b…).
        II. Low latency: prefer 32-bit transceiver user data path (script_32b…).
        III. Deterministic latency: script_32b_dl.tcl or script_32b_dl.tcl
        IV. Slow frame clock frequency to relax design timing constraints: choose a 64-bit transceiver user data path implementation (script_64b…).
    B. In this migration example we choose the TCL script script_32b.tcl.
    C. Open the file in a text editor:
        I. Change the FPGA reference: variable ***fpga_ref***. We select the xcku115-flva1517-1-c KU115 reference.
        II. Change the vivado workspace directory location: variable ***path_project***. For instance replacing ku060 by ku115 in the existing path definition.

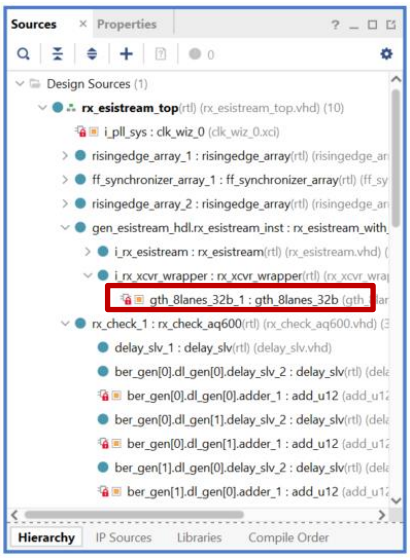
```
# --
# -- THIS DISCLAIMER MUST BE RETAINED AS PART OF THIS FILE AT ALL T
# ------------------------------------------------------------------
set gt_udw 32b
set fpga_ref xcku115-flva1517-1-c
set project_name vivado_rx_aq600_32b
set path_file [ dict get [ info frame 0 ] file ]
set path_src [string trimright $path_file "/script_32b.tcl"]
set path_project C:/vw/xilinx_ku115_v2-2/$project_name
set path_src_ip $path_src/../src_ip
set path_src_common $path_src/../src_common
set path_src_pkg $path_src/../src_pkg
set path_src_rx $path_src/../src_rx
```

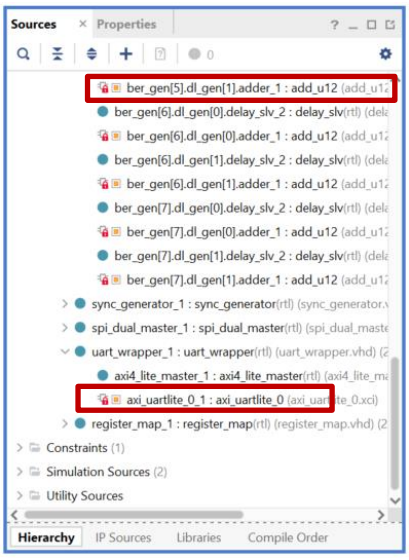
5. Open Vivado 2019.1 and launch the modified TCL script.
    A. Tools > Run Tcl Script…
    B. Browse and select the TCL script and click OK.
    C. At the end of the project generation, check there is no error in the Vivado TCL console.



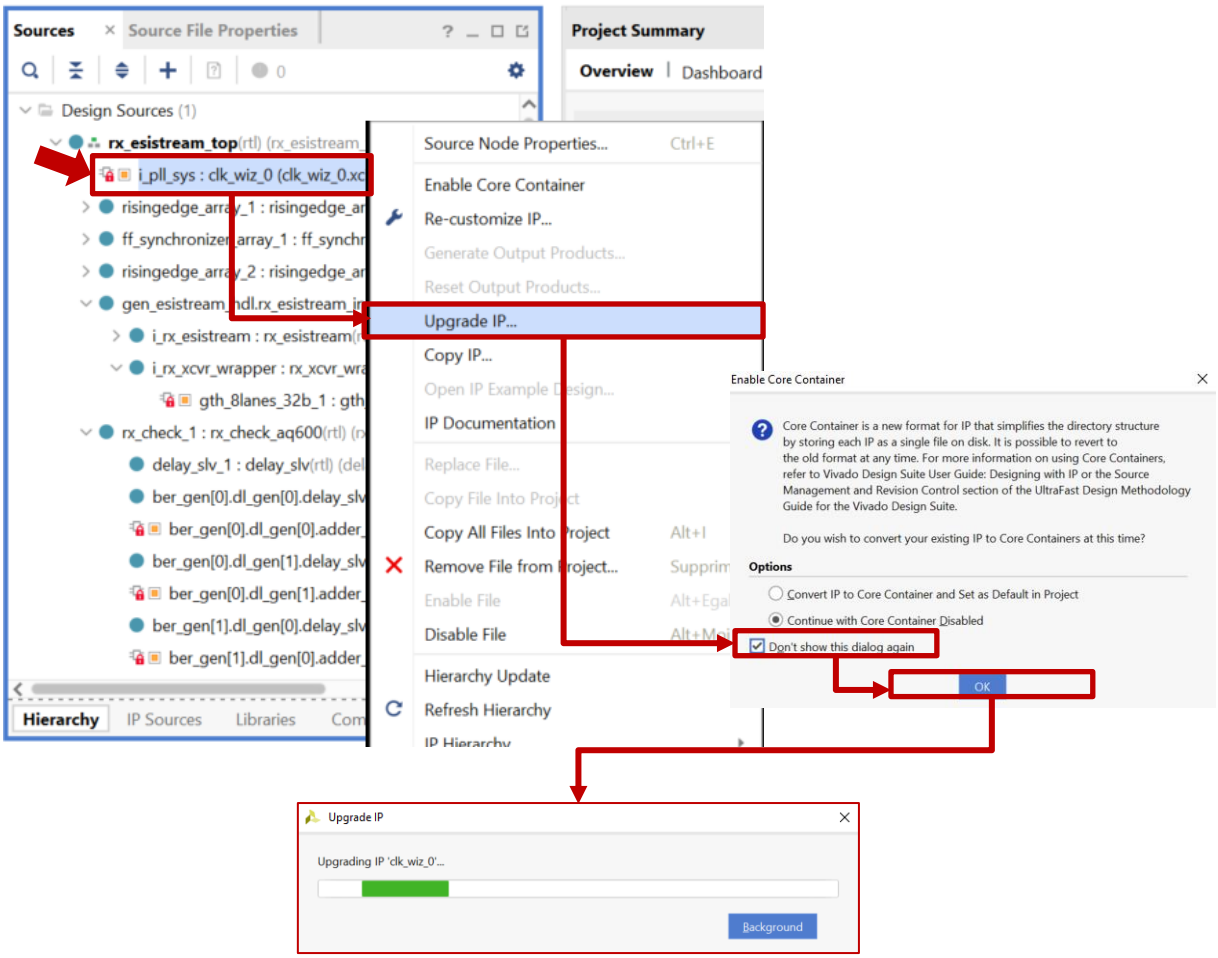
6. Ugrade each Xilinx IPs listed below:
    A. clk_wiz_0
    B. gth_8lanes_32b
    C. output_buffer (upgrade only one instance in the design is enough).
    D. add_u12 (upgrade only one instance in the design).
    E. axi_uartlite_0
    F. Optional: When it is a _64b TCL script: gth_8lanes_64b instead of gth_8lanes_32b
    G. Optional: When it is a _dl TCL script: clk_wiz_frame_clk



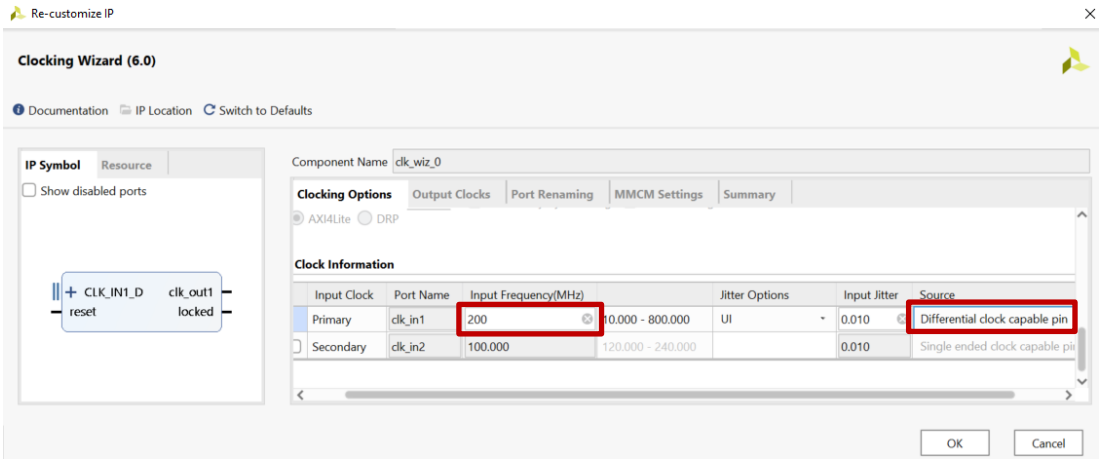




    H. Right click on each IP
        I. Upgrade IP
        II. OK > OK > Generate (Generate Output Products)
        III. Repeat H. step for each IP.



7. Configure the clk_wiz_0 IP according to your system clock input frequency.
    A. clk_wiz_0 generates a 100 MHz system clock output from a 200 MHz clock input in each *ESIstream_Xilinx_KU060_V2-2* package project. The 100 MHz system clock is used by UART wrapper and register map modules.
    B. If your design use a different clock input IO standard and frequency:
        I. Change the IP configuration:

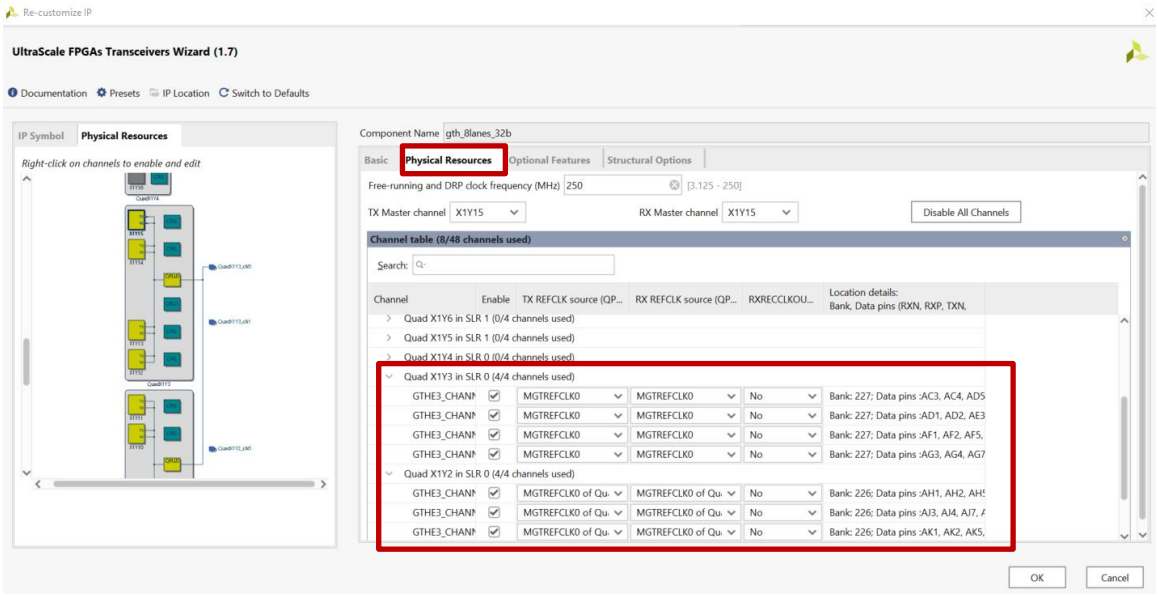


II. Modify the clock period and the IO standard values according to your design in the xdc file located in *ESIstream_Xilinx_KU115_V2-2\vivado_rx_aq600\xdc*.

Note: rx_esistream_top_32b.xdc for script_32.tcl.

```
117
118 # PL system clock:
119 set_property IOSTANDARD DIFF_HSTL_I [get_ports FABRIC_CLK_P]
120
121 set_property PACKAGE_PIN H19 [get_ports FABRIC_CLK_P]
122 set_property PACKAGE_PIN G19 [get_ports FABRIC_CLK_N]
123 set_property IOSTANDARD DIFF_HSTL_I [get_ports FABRIC_CLK_N]
124
125 create_clock -period 5.000 -name FABRIC_CLK_P [get_ports FABRIC_CLK_P]
126
```

8. Configure the gth_8lanes_32b IP according to your HSSL and SSO, Slow Synchronization Output provided by the ADC connected to the transceivers reference clock input, hardware configuration.



A. Re-generate the IP
B. Modify the xdc file according to the GTH new configuration:

Note: The file gth_8lanes_32b.xdc, located in vivado workspace directory…

```
52
53 # Commands for enabled transceiver GTHE3_CHANNEL_X1Y8
54 # ------------------------------------------------------------------------------------------------------------------
55
56 # Channel primitive location constraint
57 set_property LOC GTHE3_CHANNEL_X1Y8 [get_cells -hierarchical -filter {NAME =~ *gen_channel_container[26].*gen
58
59 # Channel primitive serial data pin location constraints
60 # (Provided as comments for your reference. The channel primitive location constraint is sufficient.)
61 #set_property package_pin AL3 [get_ports gthrxn_in[0]]
62 #set_property package_pin AL4 [get_ports gthrxp_in[0]]
63 #set_property package_pin AL7 [get_ports gthtxn_out[0]]
64 #set_property package_pin AL8 [get_ports gthtxp_out[0]]
65
```

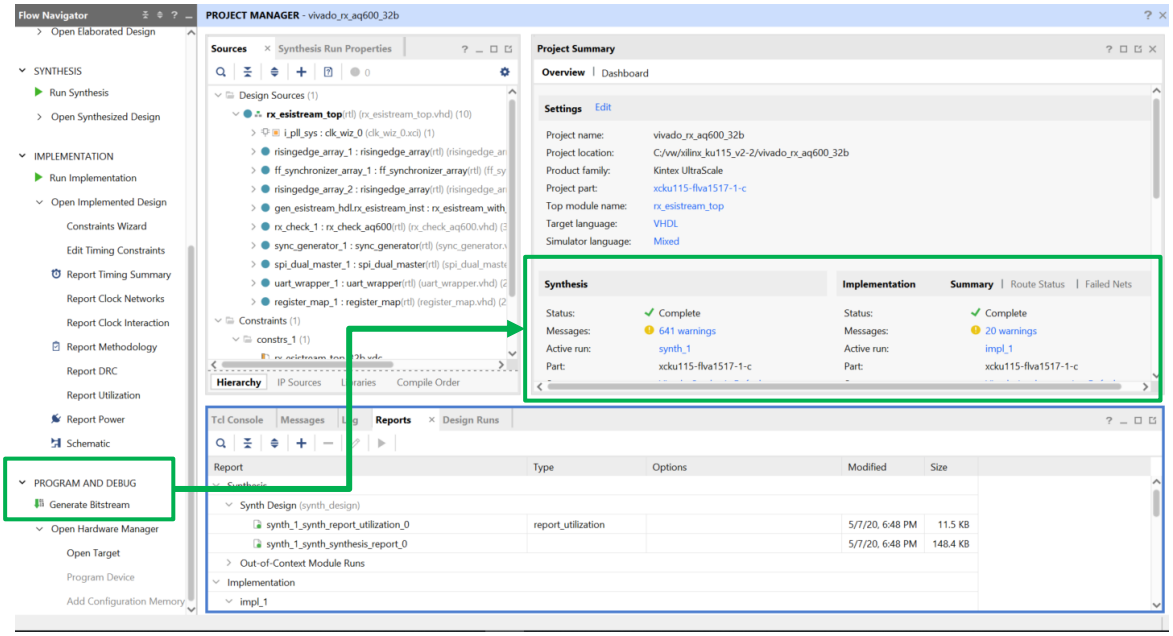
C:\vw\xilinx_ku115_v2-2\vivado_rx_aq600_32b\vivado_rx_aq600_32b.srcs\sources_1\ip\gth_8lanes_32b\synth
... can help to get FPGA balls reference according to the GTH IP lane index 0 to 7.
Example for channel 0:

9. Update all top entity signals ball reference and constraints according to your design in the xdc file located in *ESIstream_Xilinx_KU115_V2-2\vivado_rx_aq600\xdc*.

Note: *rx_esistream_top_32b.xdc* for *script_32.tcl.*

A. After this step and without additional modifications than previous steps, you should be ready to generate a bit stream or to modify the project files.

Note: Integrated Logic Analyzer debug nets may generate timings errors, critical warnings, due to the large number of nets running on the frame clock domain (390.625 MHz for a lane rate configured at 12.5 Gbps), using the script_32b.tcl. To resolve the issue, delete section from the line 157 to the end of the xdc file and add only needed debug nets.



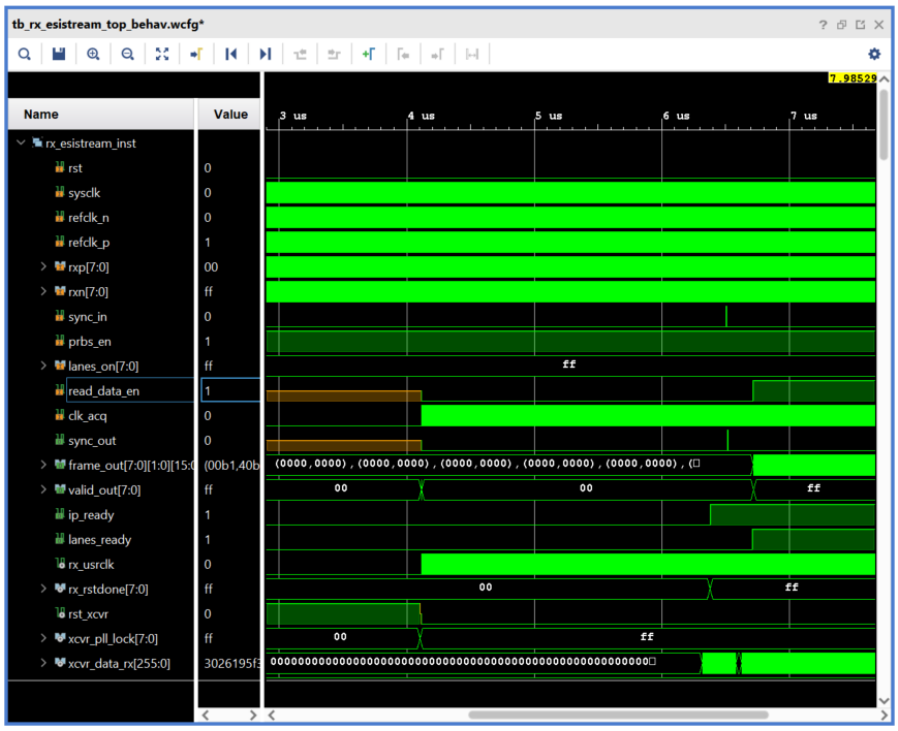
B. After this step and without additional modifications than previous steps, you should also be ready to simulate the design using vivado simulator.

I. Before running the simulation, check that the testbench high speed clock (clk_bit) half period value is set according to the GTH transceiver lane rate configuration (40 ps for a lane rate of 12.5 Gbps).

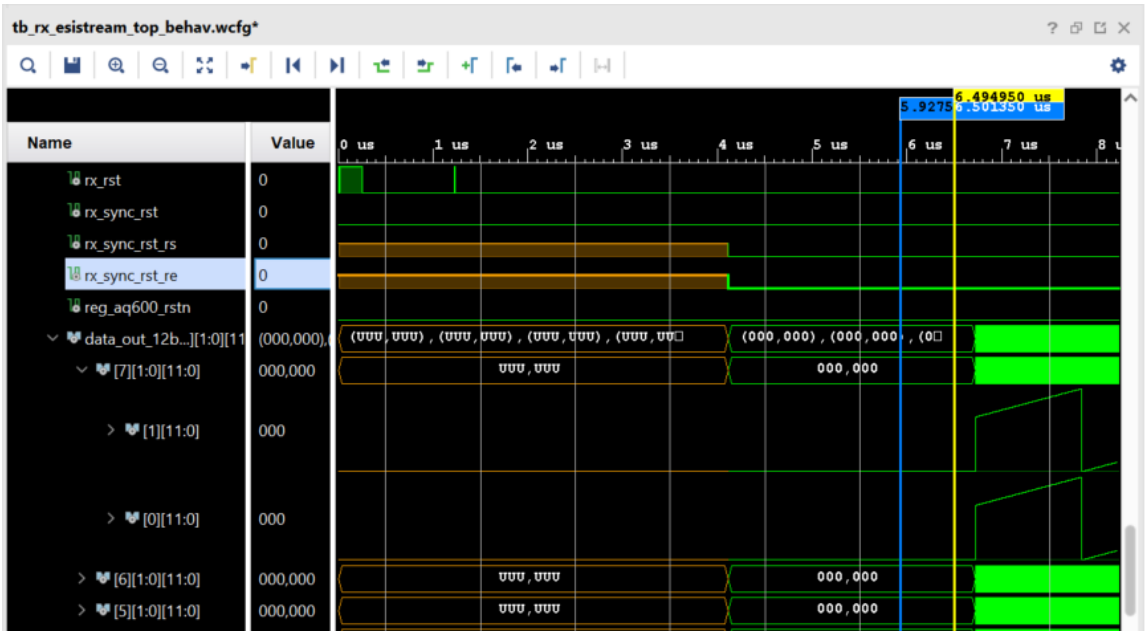
1. Testbench location: ESIstream_Xilinx_KU115_V2-2\vivado_rx_aq600\src_tb_top
2. Testbench file: tb_rx_esistream_top.vhd

```
392
393   gen_esistream_hdl : if GEN_ESISTREAM = true generate
394     -- esistream clock generation:
395     clk_bit <= not clk_bit after 40.0 ps;     -- @ 12.5 Gbps
396     --clk_bit <= not clk_bit after 39.0625 ps;    -- @ 12.8 Gbps
397     --clk_bit <= not clk_bit after 50.0 ps;      -- @ 10 Gbps
398
399     tx_esistream_emulator_1 : entity work.tx_emu_esistream_top
400       generic map (
401         NB_LANES => NB_LANES,
402         COMMA    => COMMA)
403       port map (
404         rst          => rst,
405         clk          => clk_bit,
```

II. Example of ESIstream signals analysis (sync, ip_ready, lanes_ready, valid data…)



III. Example of received data analysis, data_out_12b signals:



For technical support, please get the team involved and contact us using ESIstream contact web page or at GRE-HOTLINE-BDC@Teledyne.com