XPHY I/O Source Synchronous Interfaces

XAPP1350 (v1.0) February 4, 2021

Summary

This application note and the associated reference designs describe how to construct source synchronous high-speed I/O interfaces using the Advanced I/O Wizard (AIOW) on Versal™ devices. The wizard instantiates and configures I/O and clocking logic, such as XPHY nibbles and XPLL blocks, that are included in the physical-side interface (PHY) architecture. The designs in this application note are not hardware tested, but are verified through behavioral simulation.

Download the reference design files from the Xilinx® website. For detailed information about the design files, see the single-bank source synchronous design Reference Design section and the multi-bank source synchronous design Reference Design section.

Introduction

This application note covers two designs for a source synchronous application using the AIOW:

- Single-bank source synchronous design
- Multi-bank source synchronous design

The AIOW provides the option to choose the number of banks, but not to exceed three banks. The wizard creates one bank instance for each bank. Both designs use low-voltage differential signaling (LVDS) for data transmission speeds at 1800 Mb/s. See the *Versal AI Core Series Data Sheet: DC and AC Switching Characteristics* (DS957) for the speeds supported that can transmit and receive the LVDS standard. The underlying I/O and XPLL clocking architecture for these designs can be found in the *Versal ACAP SelectIO Resources Architecture Manual* (AM010) and *Versal ACAP Clocking Resources Architecture Manual* (AM003), respectively.

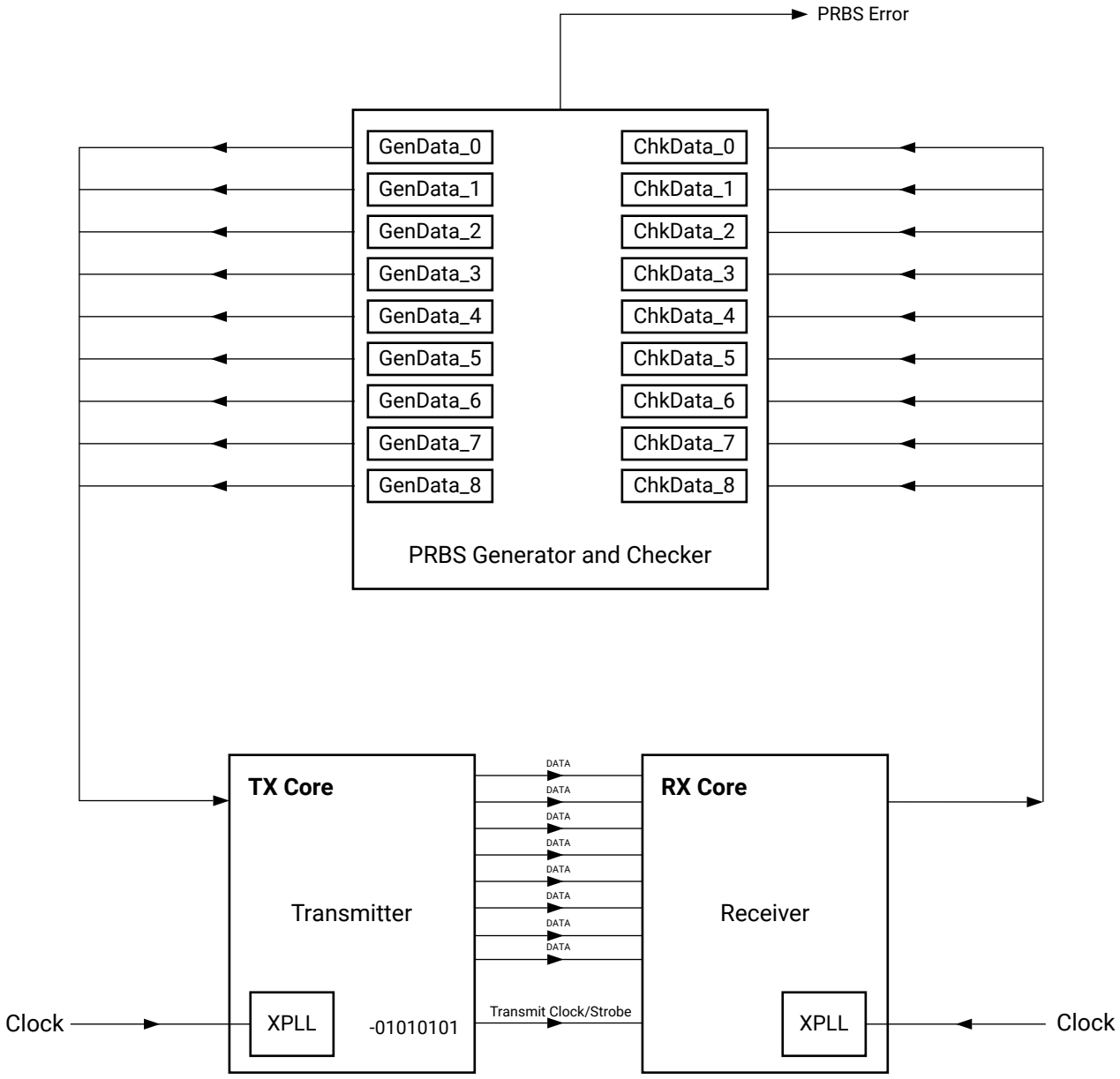
Single-Bank Source Synchronous Design

In the single-bank source synchronous reference design, all nine XPHY nibbles of the XPIO bank are used. Each XPHY nibble contains six XPHY NIBBLESLICEs that can transmit and receive data from six individual I/O pins, for a total of 54 pins/bank. The transmit clock can be forwarded from the transmit core either by the clock forward pins or by the transmit data pins from the bank. In this design, the transmit clock is forwarded by the clock forward pins. Because the design uses the LVDS standard for the I/Os, data and clock are available in pairs of I/O pins.

The wizard configures clocking circuitry using an XPLL that is needed to support these configurations. In this design, the XPLL is instantiated in the core and the wizard is used to configure the XPLL clock frequency. The XPLL input clock is fed through a global clock (GC) input pin. The core uses one differential pair of I/O as the PLL input clock for RX/TX, one differential pair of I/O for the transmit/capture clock, and the rest for the data. Consequently, the Transmit and Receive core has 25 pairs for data transmission and reception. See the *Advanced I/O Wizard LogiCORE IP Product Guide* (PG320) to understand how to use the wizard beyond the scope of this application note.

The reference design uses the PRBS generator and checker to exercise the I/Os. The design files for the PRBS generator and checker are provided in the design suite. The generator and checker are instantiated in the top-level source files. The PRBS generator generates the data and feeds the TX core, which serializes it and transmits it to the RX core via an external loopback in the test bench. The RX core feeds the data to the PRBS checker after deserializing it. Because the design is a simulation-only design, the external loopback is achieved via wires in the test bench. The checker flags an error if it detects any mismatch. The block diagram of the reference design is shown in the following figure. The transmit clock is generated by feeding a `01010101` pattern to the corresponding clock forwarding pins.

Figure 1: **Single-Bank Design**



Note: The design has three PRBS generators/checkers per XPHY nibble, one for each pair of XPHY NIBBLESLICEs transmitting and receiving data.

X24590-122320

Pre-Core Generation Setup

Before following the procedure in this section, download the reference design files from the Xilinx website. For detailed information about the design files, see Reference Design.

The following steps describe how to configure and set up the project before building the TX and RX cores using the AIOW.

1. Browse to the folder where the zipped file is downloaded.

2. Unzip the file and open the top-level `xapp1350` folder. Check the `Single_Bank_Ssync_Loopback_Design` folder, which has all the necessary design files under the `Sources`, `Constraints`, and `Testbench` folders.
3. Create a separate directory named `Versal_Ssync_RxTx_Intrfce_SB` to build the new project.
4. Launch the Vivado® tools 2020.1 or later from the newly created directory.
5. Under Quick Start, select **Create Project**.
6. Click **Next** for the prompt to **Create a New Vivado Project** and use `Versal_Ssync_RxTx_Intrfce_SB` for the name of the project. Deselect **Create a project subdirectory**.
7. Click **Next**. For Project Type, select **RTL project**. Deselect **Do not specify sources at this time**.
8. The next step is to add the sources. Add the source files from the `Sources` folder under the `Single_Bank_Ssync_Loopback_Design` folder.
9. Add the files `toplevel_sb.sv`, `Prbs_Any.vhd`, `Prbs_RxTx.vhd`, and `sync_cell.sv`. Make sure the Library is set to `xil_defaultlib` and the files are used for synthesis and simulation by setting it under HDL Source For.
10. Similarly, add the file `toplevel_testbench_sb.sv` from the `Testbench` folder under the `Single_Bank_Ssync_Loopback_Design` folder. Make sure the Library is set to `xil_defaultlib` and the file is used just for simulation by setting it under HDL Source For.
11. Select **Scan and add RTL include files into project** and **Copy sources into project**. Set the Target Language to **Verilog** and the Simulator Language to **Mixed**.
12. Click **Next** to proceed to adding the constraint files.
13. Add the file `toplevel_sb.xdc` from the `Constraints` folder under the `Single_Bank_Ssync_Loopback_Design` folder. Select **Copy constraints files into project**.
14. Click **Next** to select the part for the project. Select part **xcvc1902-vsva2197-2MP-e-S-eS1** for the reference designs and then click **Next**.
15. On the summary page for the project, make sure all the details match the settings and then click **Finish**.
16. The Vivado tools should create a project and display the hierarchy of the files under the `Sources` folder.

Introduction to Deserialization and Data Reception

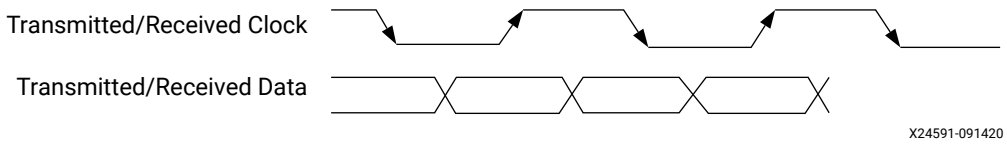
Data Reception

In this source synchronous design, the capture clock is the same as the transmit clock, which is looped back from the TX to the RX core. The transmit clock is forwarded with the data by the TX core. The clock-to-data relation in this design is center alignment as shown in the following figure. The capture clock should be received on the package pin pair assigned in the constraints file. It should be received on NIBBLESLICE[0] because it is the only NIBBLESLICE that has the clock forwarding abilities.

The XPLL in the RX core needs a PLL input clock that is used by the XPHY for calibration (BISC). The PLL input clock (CLKIN port of the XPLL) should only be received on a GC pin. External global user clocks must be brought into the Versal device on (GC) inputs. There are four GC pin pairs in each bank. GC inputs provide dedicated, high-speed access to the internal global and regional clock resources. GC inputs use dedicated routing and must be used for clock inputs where the timing of various clocking features is imperative.

The data received at the RX core interface is forwarded through the bank instance to the programmable logic where it gets checked by the PRBS checker.

Figure 2: **Center DDR**



Configure and Generate a RX Advanced I/O Wizard Core

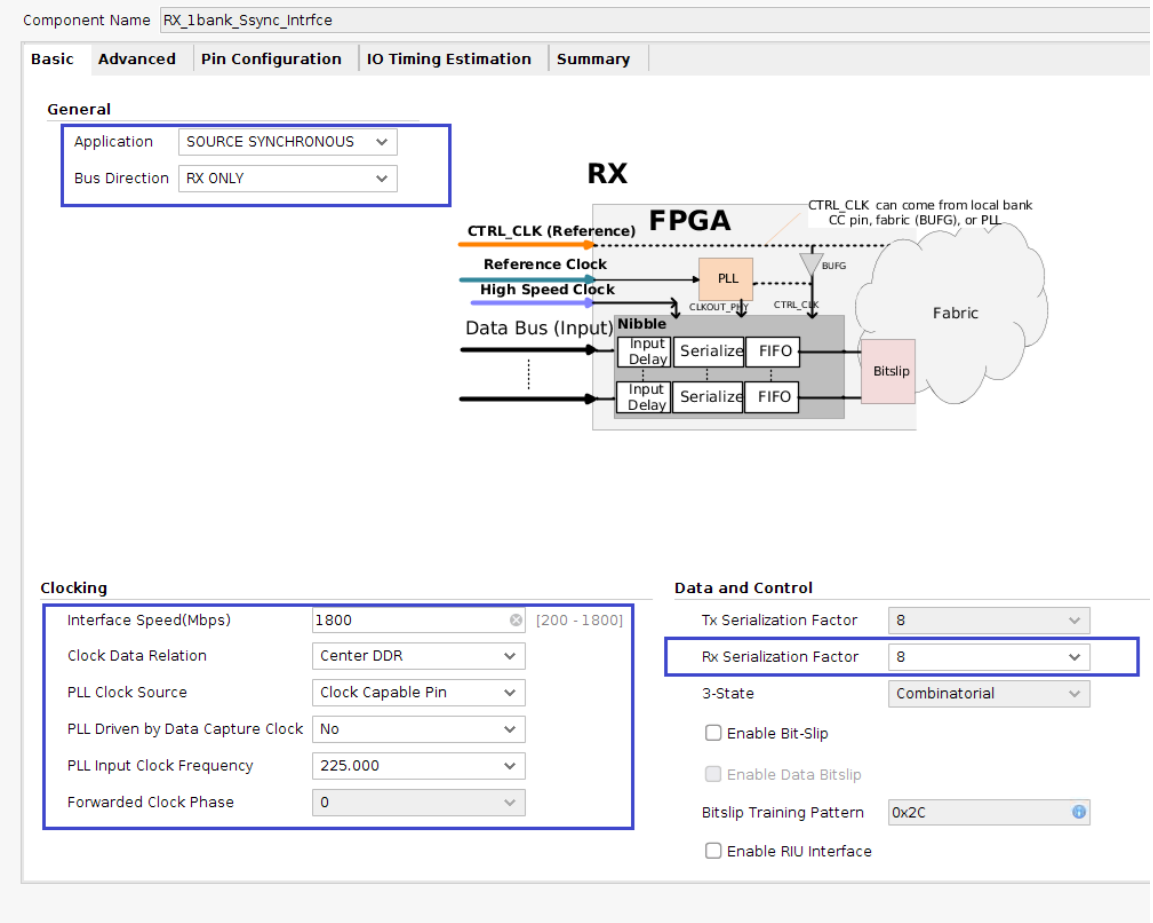
Generating a RX Core

After following the previous steps to generate the top-level design, the next step is to generate a TX and RX core for operation. Follow these steps to generate the RX core using the AIOW. See the figures in this section for reference.

1. To start generating a core for RX, open the IP catalog and search for Advanced I/O Wizard. Double-click **Advanced I/O Wizard** from the catalog to open the Customize IP window.
2. For Component Name, enter `RX_1bank_Ssync_Intrfce`, which is used in the reference design.

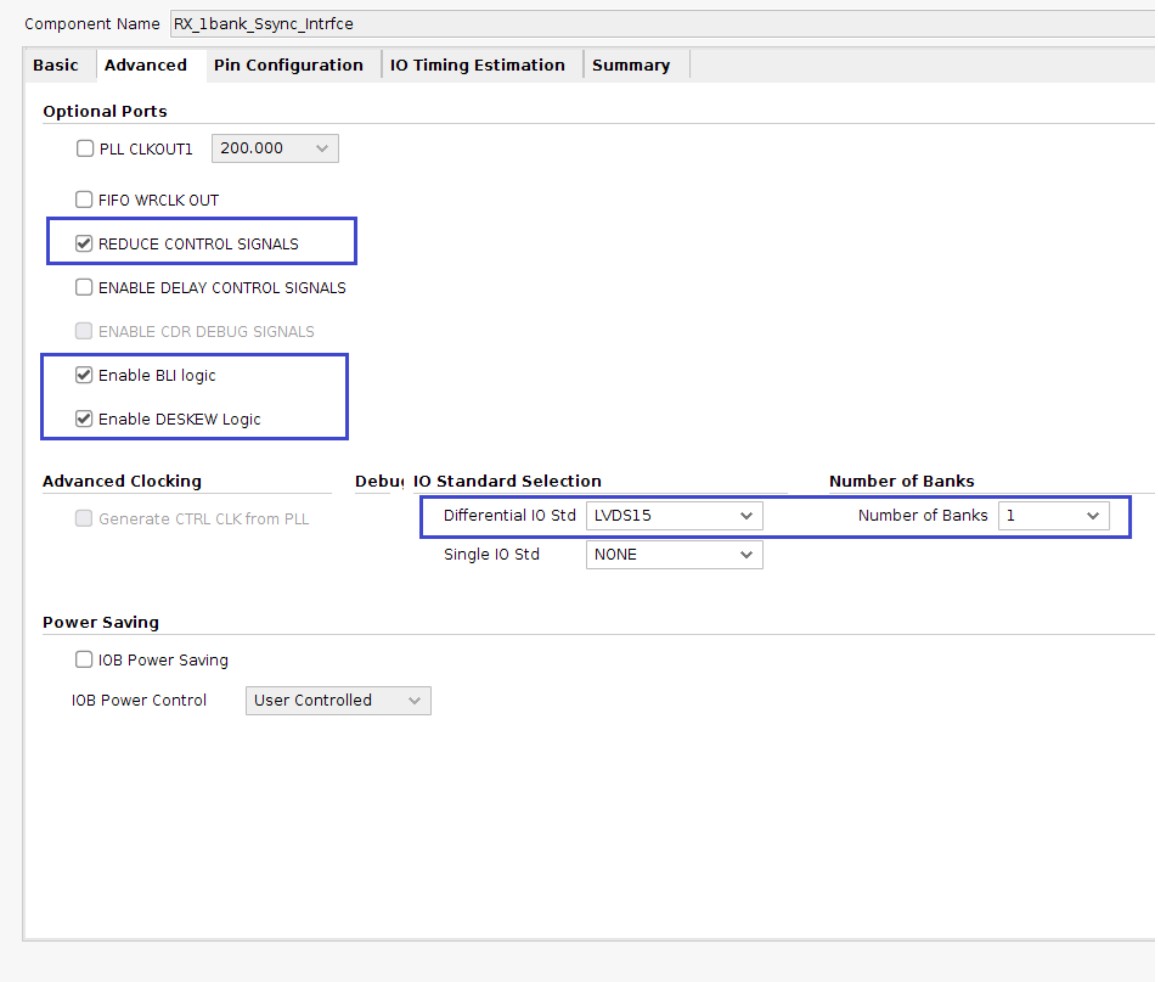
   ***Note*:** The component name should match the module name used in the top-level design.
3. Under the Basic Tab, set Application to **SOURCE SYNCHRONOUS** from the drop-down list. Set Bus Direction to **RX ONLY**.
4. On the same tab, set the following:
   a. Interface Speed: **1800 Mb/s**.
   b. Clock Data Relation: **Center DDR**.
   c. PLL Clock Source: **Clock Capable Pin**.
   d. PLL Input Clock Frequency: **225**.
   e. RX Serialization Factor: **8**.
   f. The remaining options can be set to the default.

Figure 3: **Single-Bank RX AIOW Basic Configuration**



5. In the Advanced tab, set the following:
   a. Select **REDUCE CONTROL SIGNALS**, **Enable BLI logic**, and **Enable DESKEW Logic**. When BLI logic is enabled, the BLI registers between fabric and XPHY can be used to help with timing closure. When deskew logic is enabled, the instantiated XPLLs are deskew enabled.
   b. Differential I/O Std: **LVDS15**.
   c. Number of Banks: **1** (because this is a single-bank design).

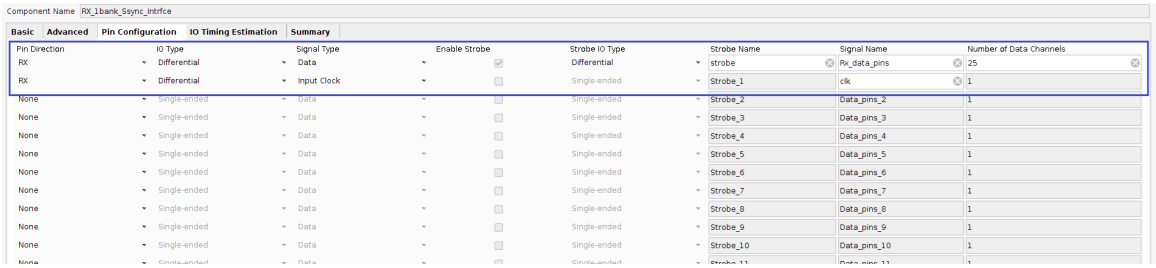
Figure 4: **Single-Bank RX AIOW Advanced Configuration**



6. In the Pin Configuration tab, make two entries in the table. One entry is for the data with the capture clock and the other is for the PLL input clock. Twenty five pairs for data and 1 pair for the PLL input clock are configured under this tab. When selecting 25 pairs for data, the wizard adds 1 pair for the capture clock.
   - Data and the Strobe Setting
     - Pin Direction = RX
     - I/O Type = Differential
     - Signal Type = Data
     - Strobe I/O Type = Differential
     - Strobe Name = Strobe
     - Signal Name = Rx_data_pins
     - Number of Data Channels = 25
   - PLL Input Clock Setting
     - Pin Direction = RX

- I/O Type = Differential
- Signal Type = Input Clock
- Signal Name = clk
- Number of Data Channels = 1

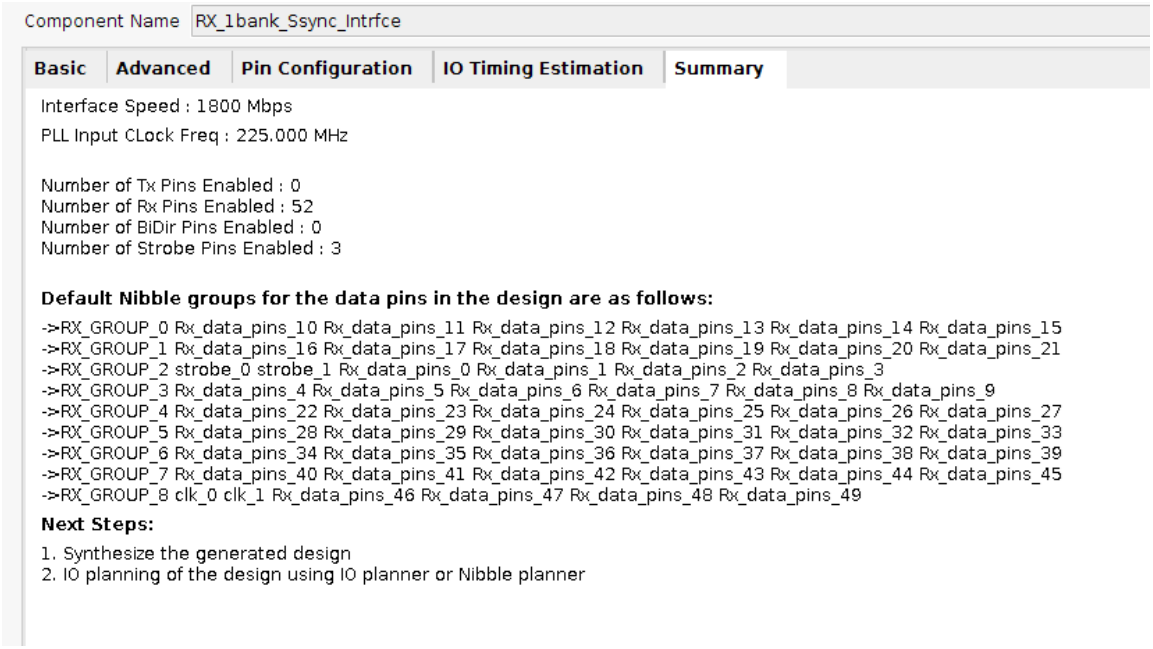
Figure 5: **Single-Bank RX AIOW Pin Configuration**



7. Check the Summary tab. It should show 52 RX pins enabled (25 pairs of data + 1 pair of strobes). One pair is used by the PLL input clock that feeds the CLKIN port of the XPLL.

   ***Note*:** In the following figure, the number of strobe pins enabled is incorrect. The correct value is 2.

Figure 6: **Single-Bank RX AIOW Summary**



8. Click **OK** after reviewing the settings. The IP is now customized and the Generate the Output Products prompt appears. Set the Synthesis Option to **Out of context per IP** and then click **Generate** to launch the design run for the newly generated RX core. See the following table.

Table 1: **Receiver Requirements**

	Requirement
Component name	RX_1bank_Ssync_Intrfce
Bus direction	RX_ONLY
RX serialization factor	8
Interface speed (Mb/s)	1800 Mb/s
Clock data relation	Center DDR
PLL clock source	Clock capable pin
PLL input clock frequency	225 MHz
Include PLL in core	Yes
PLL CLKOUT1	No
FIFO WRCLK OUT	No
Reduce control signals	Yes
Enable delay control signals	No
Enable BLI logic	Yes
Enable deskew logic	Yes
Differential I/O Std	LVDS15
Number of banks	1
Pin configuration	The number of data channels is set to 25 pairs. One pair is added for the capture clock and one pair is assigned to the input clock for the XPLL.

Receiver Design Considerations

This RX core is set up to work for a data rate of 1800 Mb/s. Also, the core is configured for LVDS15 in the reference design. This single-bank design uses a clock capable pin as a PLL clock source and is configured for a center aligned DDR system. An XPIO bank has 54 pins and the design uses all 54 pins in the form of 25 pairs of data pins, 1 pair of capture clock pins, and 1 pair of PLL input clock pins to the RX XPLL. The wizard provides the options to set the frequency for the CLKOUT1 port of the XPLL, which must be provided. The design constrains the ports for the receive interfaces and the wizard takes care of the placement.

Introduction to Serialization and Data Transmission

Data Transmission

In the reference design, the TX core sends out the transmit clock along with the data. The data in this design is generated using the PRBS generator and the transmit clock is generated by feeding the pattern `01010101` to the clock forwarding pins. The data generated by the PRBS generator is fed into the TX core from the programmable logic, which follows the TX datapath through a serializer and output delay within the PHY. The serializer supports 8:1, 4:1, and 2:1 serialization. In the design, an 8:1 serialization factor is used. The data is transmitted through the TX data pins of the core. To understand the data flow operation inside the TX core, see *Versal ACAP SelectIO Resources Architecture Manual* (AM010).

The PLL input clock to the TX core is fed to the XPLL through a clock capable GC/XCC pin within the same bank. In the design, the PLL input clock is received on the bank0_pll_clkin port of the TX core instantiated by the wizard. The transmit clock can be forwarded either through the data pins or the clock forwarding pins on the TX core. The design uses the clock forwarding pins to support the center DDR alignment. The wizard provides an option to set the phase on the forwarded clock to 90, which aligns the data-to-clock relation to support the center DDR on the RX core. The transmit clock is transmitted on NIBBLESLICE[0] and NIBBLESLICE[1] to the RX core.

Configure and Generate a TX Advanced I/O Wizard Core

Generating a TX Core

Follow these steps to generate the TX core using the AIOW. See the figures in this section for reference.

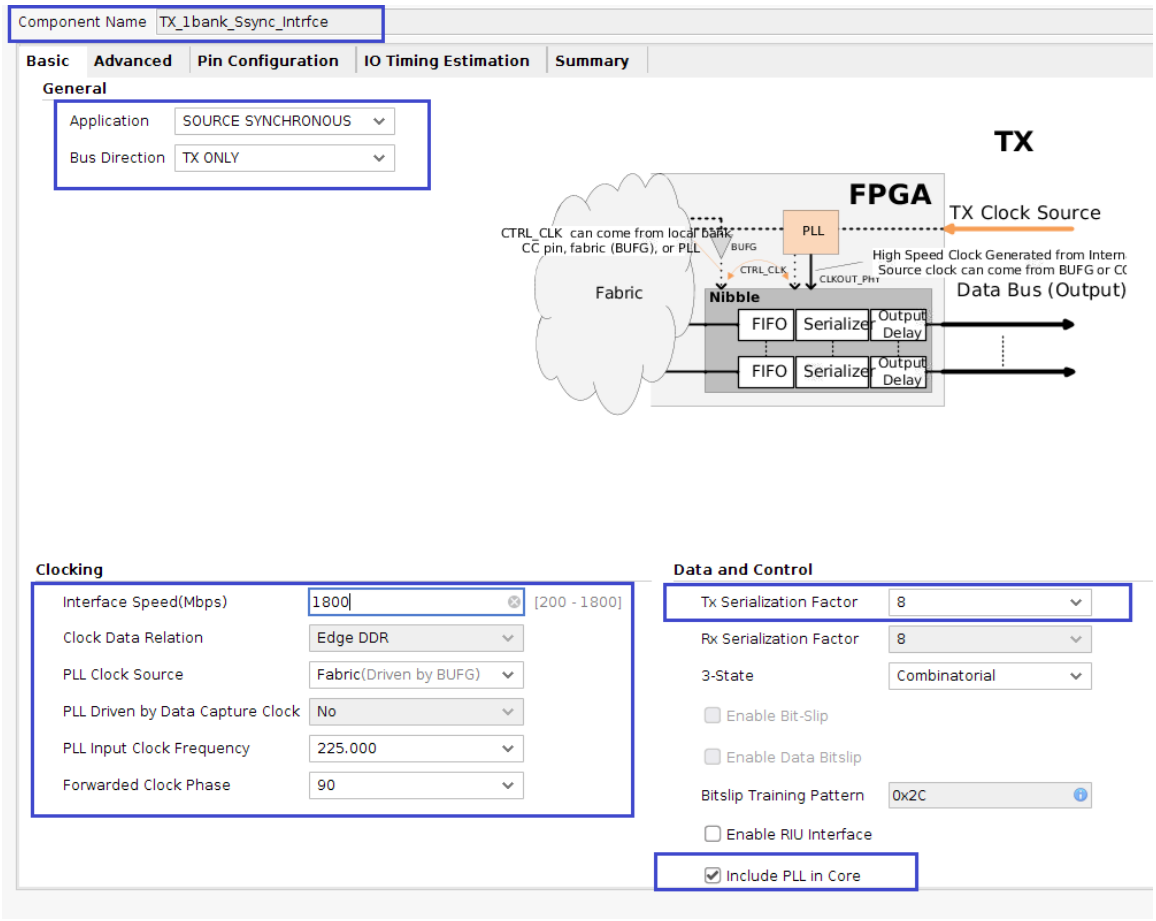
1. To start generating a core for TX, open the IP catalog and search for Advanced I/O Wizard. Double-click **Advanced I/O Wizard** to open the Customize IP window for the wizard.
2. For Component Name, enter `TX_1bank_Ssync_Intrfce`, which is used in the reference design.

   ***Note*:** The component name should match the module name used in the top-level design.
3. Under the Basic Tab, set Application to **SOURCE SYNCHRONOUS** from the drop-down list, and set Bus Direction to **TX ONLY**.
4. On the same tab, set the following:
   a. Interface Speed: **1800 Mb/s**.
   b. Clock Data Relation: **Edge DDR**.
   c. PLL Clock Source: **Fabric (Driven by BUFG)**.
   d. PLL Input Clock Frequency: **225**.
   e. Forwarded Clock Phase to **90**.
   f. TX Serialization Factor: **8**

g. Select **Include PLL in Core**.

h. The remaining options can be set to the default.

Figure 7: **Single-Bank TX AIOW Basic Configuration**



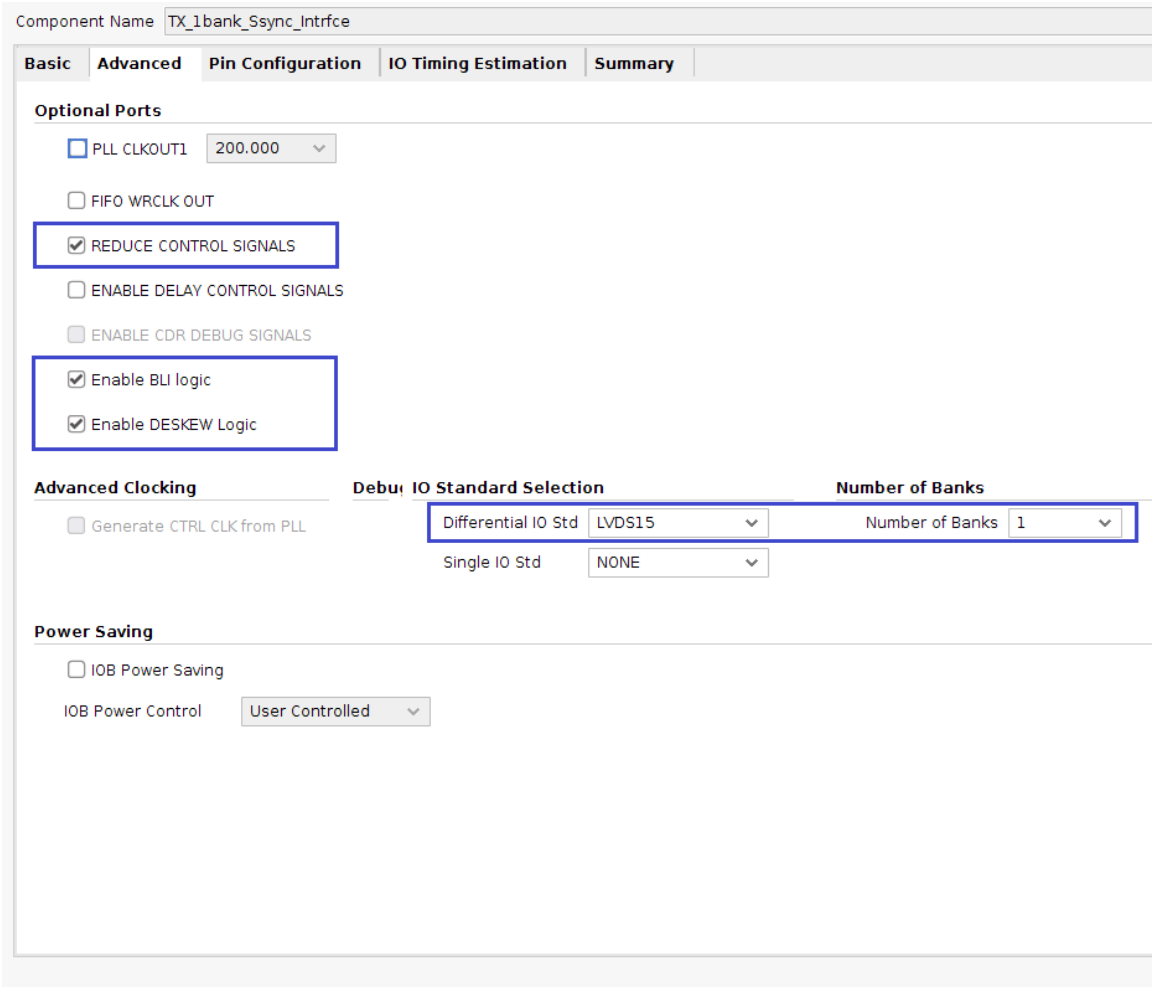
5. In the Advanced tab, set the following:

   a. Select **REDUCE CONTROL SIGNALS**, **Enable BLI logic**, and **Enable DESKEW Logic**. When BLI logic is enabled, the BLI registers between fabric and XPHY can be used to help with timing closure. When deskew logic is enabled, the instantiated XPLLs are deskew enabled.

   b. Differential I/O Std: **LVDS15**

   c. Number of Banks: **1** (because this is a single-bank design).

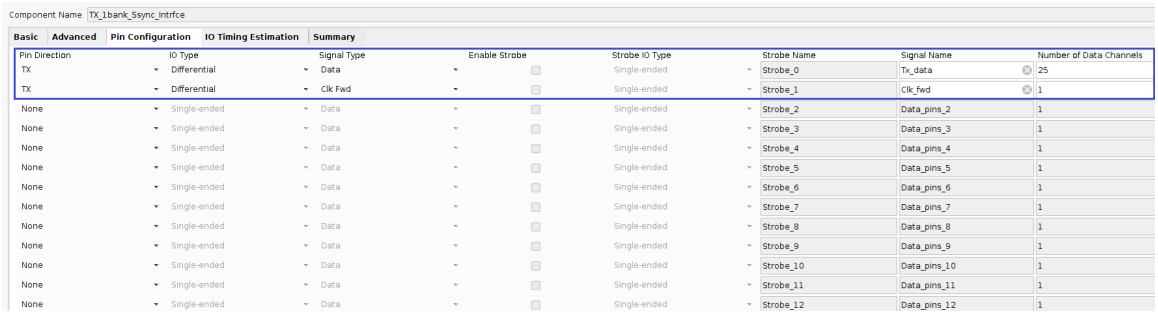
Figure 8: **Single-Bank TX AIOW Advanced Configuration**



6. In the Pin Configuration tab, add two entries in the table. One entry is for the data with I/O type as differential, signal type as data, and signal name as Tx_data_pins. The other entry is for the clock forwarding pins with I/O type as differential, signal type as Clk Fwd, and signal name as Clk_fwd.
   - Data Setting
     - Pin Direction = TX
     - I/O Type = Differential
     - Signal Type = Data
     - Signal Name = Tx_data
     - Number of Data Channels = 25
   - Data Setting
     - Pin Direction = TX
     - I/O Type = Differential
     - Signal Type = Clk Fwd

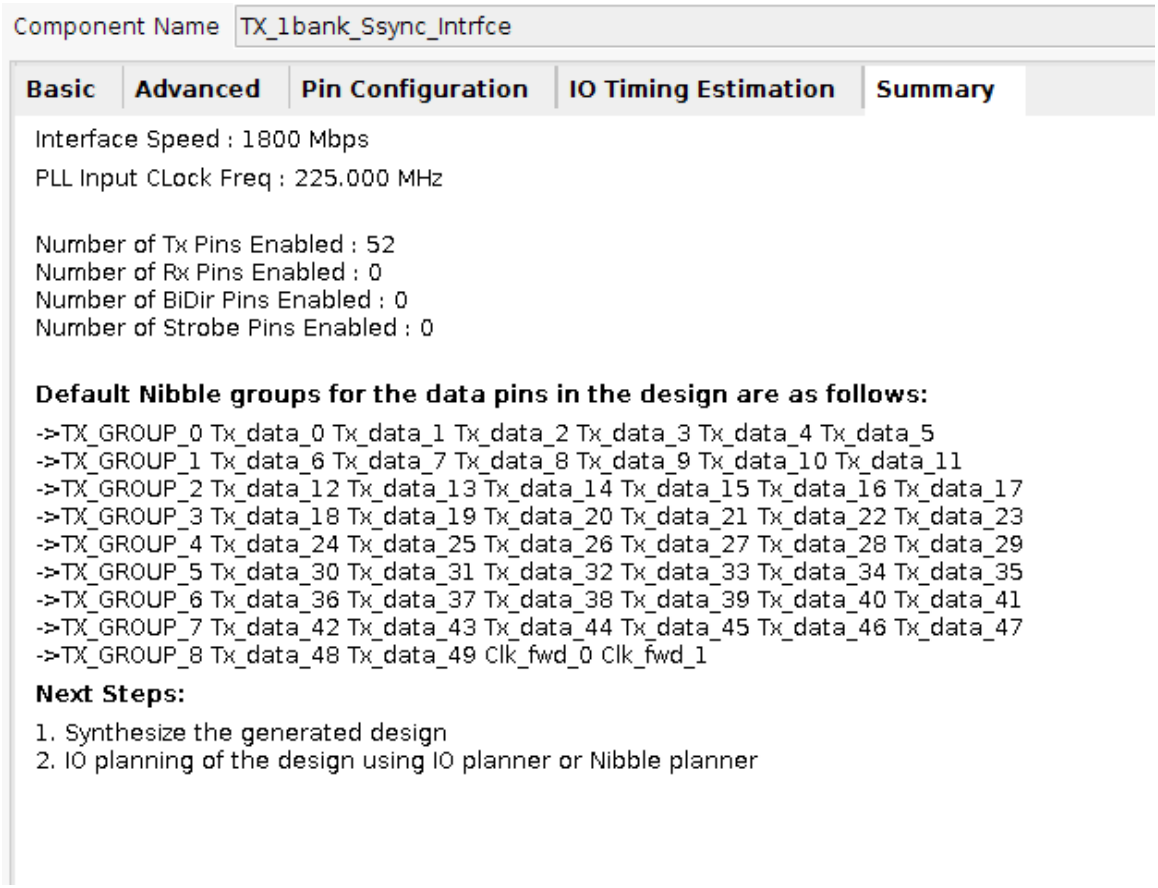
- Signal Name = Clk_fwd
- Number of Data Channels = 1

Figure 9: **Single-Bank TX AIOW Pin Configuration**



7. Check the Summary tab. It should show 52 TX pins enabled (25 pairs of data + 1 pair of forwarded clock). The PLL input clock for the TX is driven from the programmable logic as selected under the Basic tab while configuring the TX core.

Figure 10: **Single-Bank TX AIOW Summary**



8. Click **OK** after reviewing the settings. The IP is now customized and the Generate the Output Products prompt appears. Set the Synthesis Option to **Out of context per IP** and then click **Generate** to launch the design run for the newly generated TX core. See the following table.

Table 2: **Transmitter Requirements**

	Requirement
Component name	TX_1bank_Ssync_Intrfce
Bus direction	TX_ONLY
TX serialization factor	8
Interface speed (Mb/s)	1800 Mb/s
Clock data relation	Edge DDR
PLL clock source	Fabric (driven by BUFG)
Forwarded clock phase	90
PLL input clock frequency	225 MHz
Include PLL in core	Yes
PLL CLKOUT1	No
FIFO WRCLK OUT	No
Reduce control signals	Yes
Enable delay control signals	No
Enable BLI logic	Yes
Enable deskew logic	Yes
Differential I/O Std	LVDS15
Number of banks	1
Pin configuration	The number of data channels is set to 25 pairs. One pair is reserved for the transmit clock via the clock forwarding pins.

Transmitter Design Considerations

This TX core is set up for a data rate of 1800 Mb/s. Also, the core is configured and tested for LVDS15 in the reference design. This single bank design uses the same clock to feed the XPLL of both the RX and TX cores. The PLL input clock that feeds the TX core is driven through the IBUFDS and then passed into the core. An XPIO bank has 54 pins and the design uses 52 pins in the form of 25 pairs of data pins and 1 pair of transmit clock/strobe via the clock forwarding pins. One pair is reserved for the PLL input clock to the RX core, and consequently, is not used for TX core. The design constrains the ports for the transmit interfaces and the placement is taken care of by the Vivado tools.

Top-Level using TX and RX Advanced I/O Wizard Cores

The top-level design file (`toplevel_sb.sv`) includes the `toplevel_sb` module. This module helps connect the interfaces such as clocks, debug ports, I/O ports, etc., with the appropriate sources. The top-level design houses the instantiation of both the RX and the TX cores. To test the design, the PRBS patterns from the custom PRBS generator provided in the design suite can be used to generate and check the received data.

The PRBS generator generates 8-bit data for each pair of pins and feeds it to the TX core, which in turn transmits it through the TX pins. The PRBS generator also houses an error injecting mechanism. The data is received on the I/O ports of the RX core through external loopback. The RX core forwards the deserialized data through the PHY to the programmable logic. This deserialized data is then fed into the PRBS checker to check for any failures.

The top-level design file includes a constraints file (`toplevel_sb.xdc`). This file is used to create clocks for the design, assign locations or pins to all the I/O ports in the design, and set attributes if needed. The reference design constrains the design to optimally support high data rates. The user must constrain the TX and RX ports. The Vivado tools can assign the XPHY nibbles to the XPHY sites. The Advanced I/O Planner can be used to help with pin assignments (see *Advanced I/O Wizard LogiCORE IP Product Guide* (PG320)) when using the AIOW.

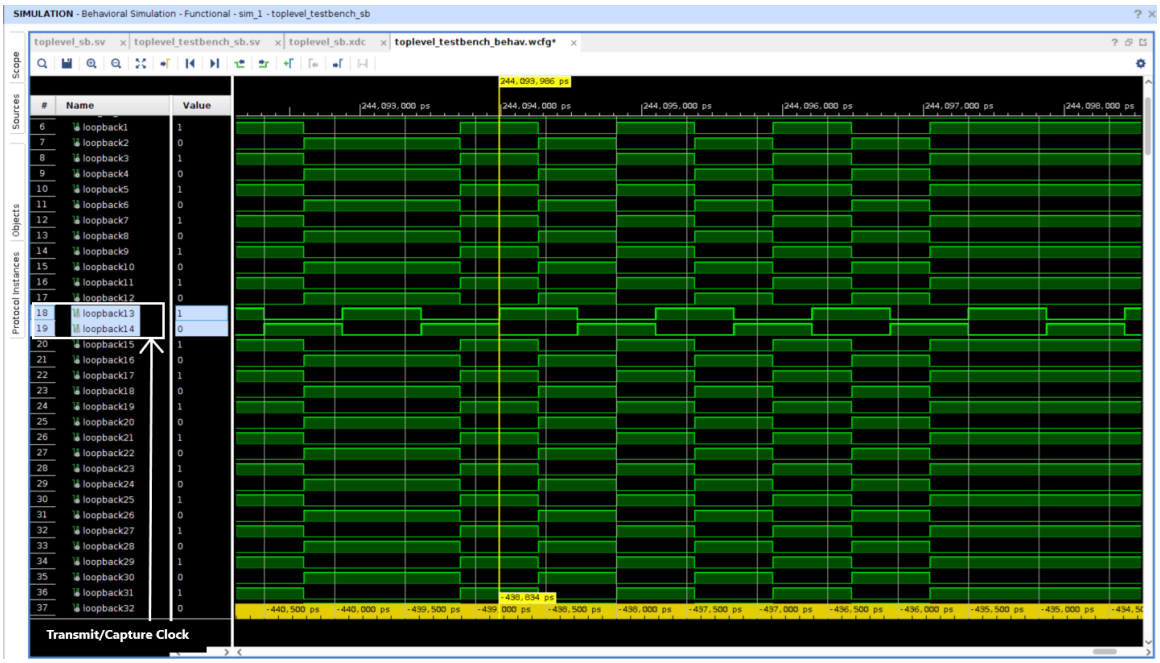
In the reference design, bank 707 for part xcvc1902-vsva2197-2MP-e-S-eS1 is used for the TX core and bank 704 is used for the RX core. The RX core has one pair reserved for the PLL input clock. The design assigns IO_L9P_GC_XCC_N3P0_M1P72_704_BE31 and IO_L9N_GC_XCC_N3P1_M1P73_704_BD32 in the constraints file. Similarly, IO_L6P_GC_XCC_N2P0_M1P66_704_BC31 and IO_L6N_GC_XCC_N2P1_M1P67_704_BC30 are assigned to the strobe. The corresponding pins in bank 707 are assigned to the PLL input clock and forwarded clock for the TX core. Check the board schematics and the `toplevel_sb.xdc` constraints file to understand the pin assignments.

At this point, all the files are added to the project and both the RX and TX cores are generated. The design is ready to be synthesized and implemented.

Simulation

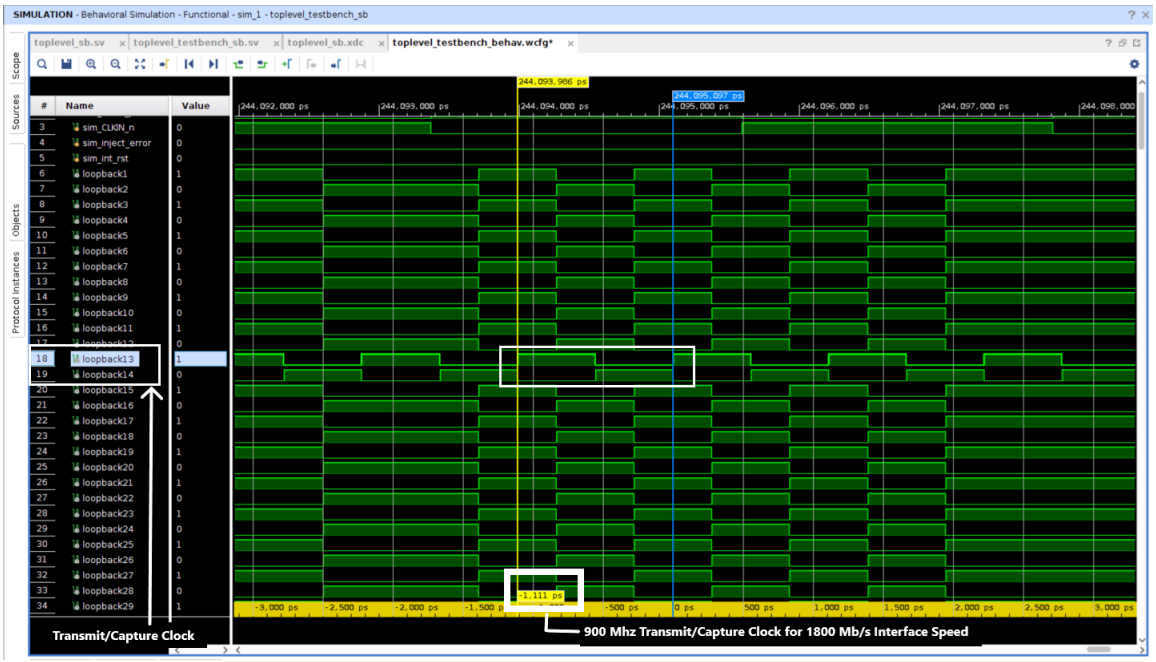
The reference design uses the `toplevel_testbench_sb.sv` file to create a simple test bench. This test bench connects the TX core to the RX core via loopback<num> connections (wires). The transmit clock is transmitted on loopback13 and loopback14 loopback wires. All the other loopback connections are used to transmit and receive the data. The clock-to-data relationship is center aligned as shown in the following figure captured from the simulation.

Figure 11: **Clock-to-Data Relation - Center Alignment**



The test bench provides the necessary clock and resets to the design and triggers its operation. The PLL input clock is provided at 4.444 ns (225 MHz) to the RX and TX cores. The transmit/capture clock toggles at 900 MHz (1.111 ns period) as shown in the following figure. Because the system is double data rate, the interface operates at 1800 Mb/s.

Figure 12: **Single Bank Design - Interface Speed 1800 Mb/s**



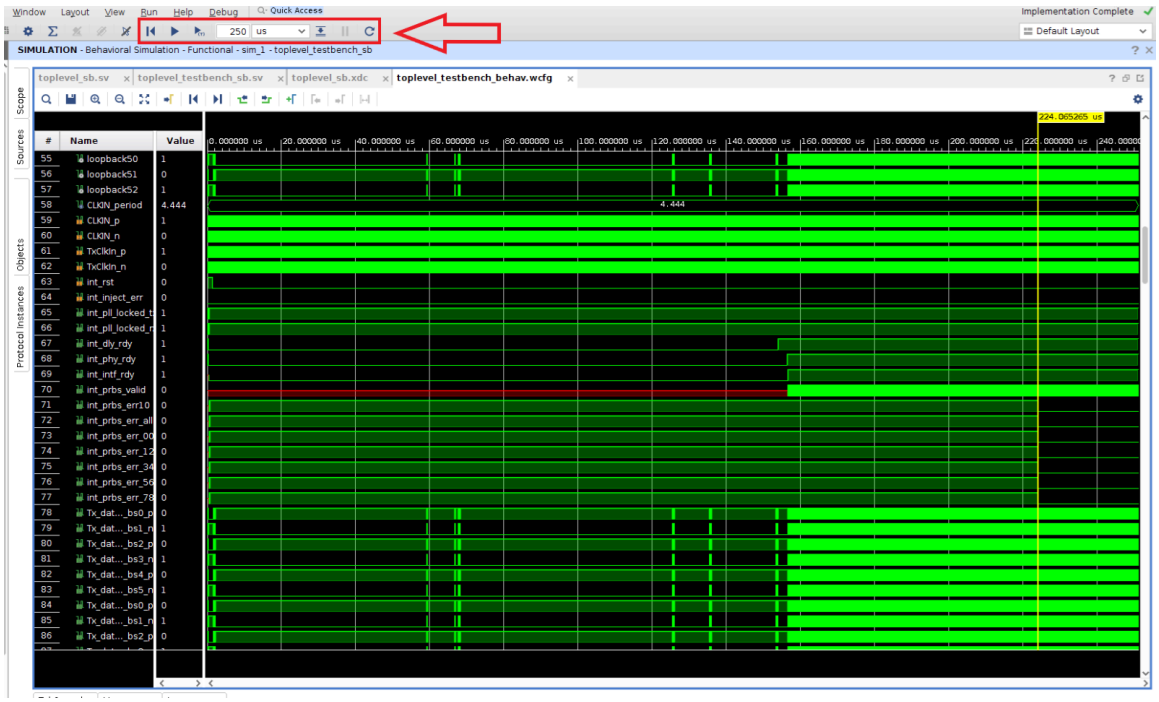
Note: The Vivado 2020.1 tools allow only behavioral simulation when using the AIOW.

Simulating the Design

The design is tested with the Vivado Simulator 2020.1. This section describes how to launch the simulation. Assuming the Vivado project is already created for the design, follow these steps to simulate the design.

1. Click **Run Simulation** under Simulation from the Flow Navigator.
2. From the listed options, select **Run Behavioral Simulation**, which elaborates the design and launches the simulation.
3. The run time by default is 1 ns for the simulation. It takes about 230 µs for the intf_rdy to be asserted for the RX core. Only after its assertion and the next Prbs_Valid, should the data be compared. Consequently, launch the simulation for a duration of more than 230 µs using the box highlighted in the following figure. A waveform config file `toplevel_testbench_behav.wcfg` is provided under the `Testbench` directory to add a list of signals for the simulation.

Figure 13: **Single-Bank Design - Changing the Run Time for Simulation**



4. Once the behavioral simulation is finished, check for any errors by observing error flags for the PRBS generator and checker. For example, int_prbs_err<num> reports the errors for each instantiation of the PRBS generator and checker. Int_prbs_err24 denotes the error flag for the PRBS module instantiated for NIBBLE[2], NIBBLESLICE[4]. Int_prbs_err_00 denotes the error for any NIBBLESLICEs on NIBBLE[2]. Int_prbs_err_12 denotes the error for any NIBBLESLICEs on NIBBLE[1] and NIBBLE[2]. Int_prbs_err_all denotes an error on any NIBBLESLICEs across all the nibbles.

Reference Design

Download the reference design files from the Xilinx website. Unzip this file and browse into the top-level folder `xapp1350`. Check the `Single_Bank_Ssync_Loopback_Design` folder, which has all the source files under `Sources`, `Constraints`, and `Testbench`.

Table 3: **Source Files for Single-Bank Design**

Folder	File Name	Description
Sources	`toplevel_sb.sv`	The top-level design file that instantiates the TX and RX cores and connects the design
	`Prbs_RxTx.vhd`	Outer wrapper for the custom PRBS generator and checker
	`Prbs_Any.vhd`	Contains the modules for the PRBS generator and checker
Constraints	`toplevel_sb.xdc`	The constraints file to assign pin locations to ports of the design and set the necessary attributes
Test Bench	`toplevel_testbench_sb.sv`	The top-level test bench file to test the design
	`toplevel_testbench_behav.wcfg`	Waveform configuration file

Reference Design Matrix

The following checklist indicates the procedures used for the provided reference design.

Table 4: **Reference Design Matrix**

Parameter	Description
General	
Developer name	Xilinx
Target devices	Versal ACAP
Source code provided?	Yes
Source code format (if provided)	Verilog and VHDL
Design uses code or IP from existing reference design, application note, third party or Vivado software? If yes, list.	Yes. Uses the Advanced I/O Wizard from the IP catalog.
Simulation	
Functional simulation performed	Yes
Timing simulation performed?	No
Test bench provided for functional and timing simulation?	Yes
Test bench format	Verilog
Simulator software and version	Vivado simulator 2020.1
SPICE/IBIS simulations	No
Implementation	
Synthesis software tools/versions used	Vivado synthesis 2020.1
Implementation software tool(s) and version	Vivado 2020.1 implementation
Static timing analysis performed?	Yes
Hardware Verification	
Hardware verified?	No
Platform used for verification	N/A

Multi-Bank Source Synchronous Design

In the multi-bank source synchronous reference design, all three available banks are used, which is the maximum width of a single I/O interface supported by the AIOW. This design uses all nine XPHY nibbles from one XPIO bank, with each XPHY nibble containing six XPHY NIBBLESLICEs that transmit and receive data from six individual I/O pins, for a total of 54 pins/bank. This design has three banks, and consequently, uses a total of 27 XPHY nibbles. The wizard requires the banks to be adjacent to each other. The transmit clock can be forwarded from the transmit core either via the clock forward pins or the transmit data pins from the bank. In the reference design, the transmit clock is forwarded via the transmit data pins. The design uses the LVDS standard for the I/Os, and consequently, the data, transmit/capture clock, and PLL input clock use differential I/O pin pairs.

The wizard configures clocking circuitry using the XPLL that is needed to support these configurations. In this design, the XPLL is instantiated in the core and the wizard configures the XPLL clock frequencies. The XPLL input clock is fed through a global clock (GC) input pin. As a result, the core reserves a pair of I/Os for the PLL input clock for RX, one pair of I/Os for the capture clock, and the remainder for the data in each bank. Consequently, the transmit and receive cores have 25 pairs for data transmission and reception per bank. See *Advanced I/O Wizard LogiCORE IP Product Guide* (PG320) for more information on the AIOW. Each bank has a transmit clock of its own that the transmitter transmits. Similarly, the receiver receives a capture clock on each bank.

Clocking in the multi-bank source synchronous reference design is similar to the single-bank design. Instead of one instantiation of the XPLL as in the single-bank design, this design has three instantiations of the XPLLs, one for each bank. A common PLL input clock for all the XPLLs is fed into the RX core via the clock capable pins. This clock drives the CLKIN of the XPLLs for each bank. The TX core has three separate inputs for the PLL input clock, which is fed to the bank<0/1/2>_pll_clkin ports of the TX core. It uses the same clock as the RX core except that it passes through an IBUFDS to create a single-ended version that is passed to the respective CLKIN ports of each bank on the TX core.

In this design, the TX core uses the "Fabric (driven by BUFG)" option as the PLL clock source in the AIOW. These clocks are received on the GC pins and, consequently, do not need any BUFGs added with the IBUFDS. The clocks need BUFGs added with the IBUFDS if they are not received on the GC pins. The following figures illustrate how the clocks are connected to the TX and RX cores, respectively.

Figure 14: **Clock Connection for TX Core**

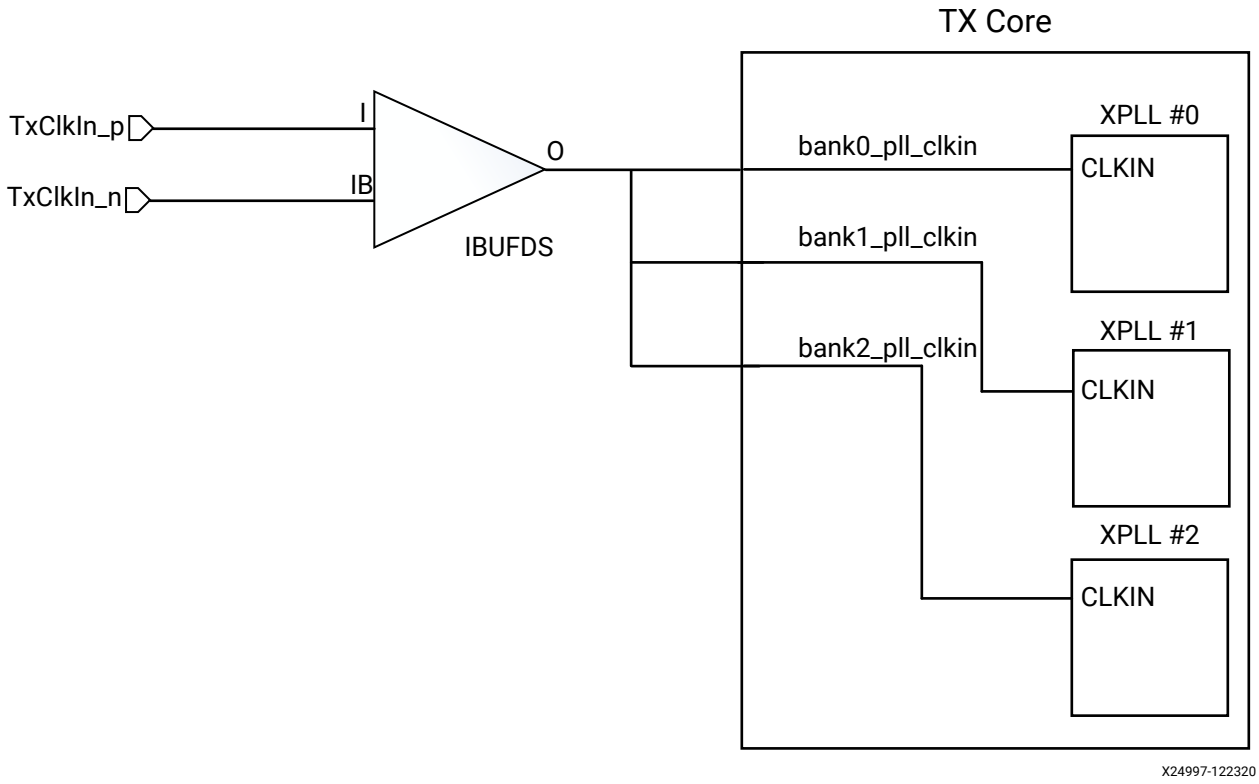
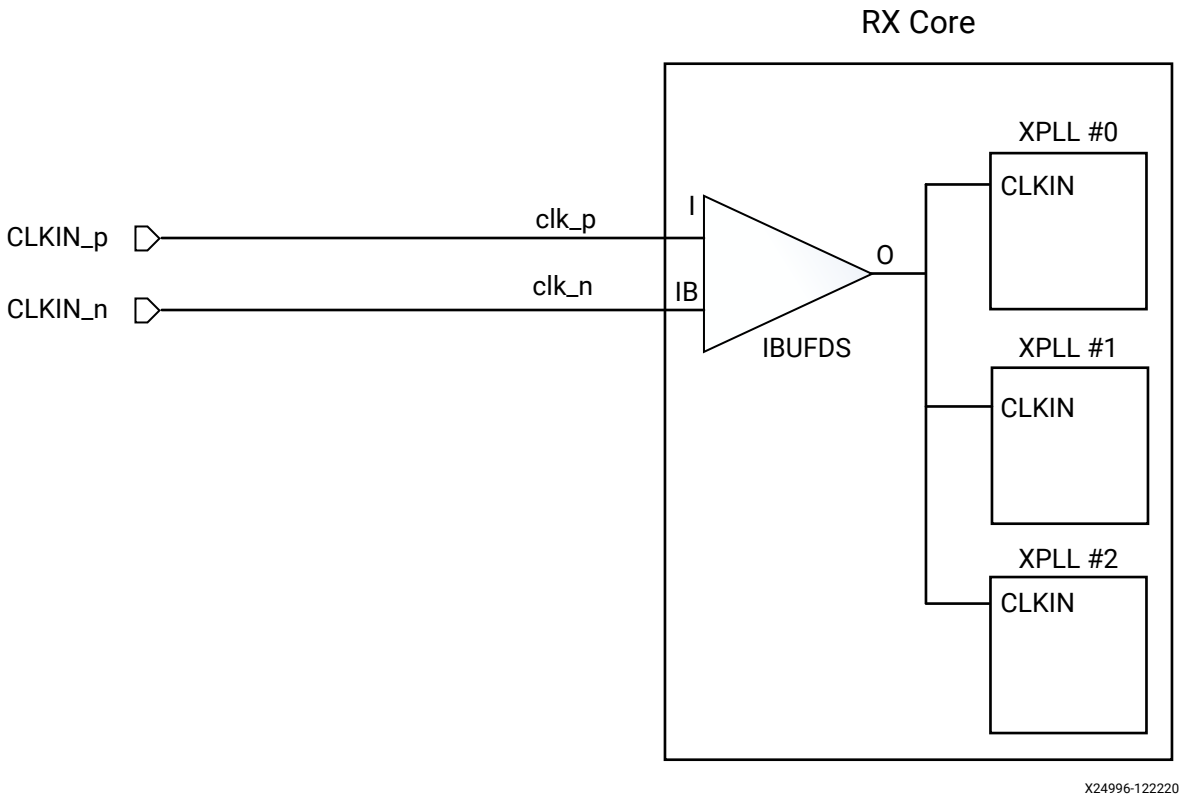


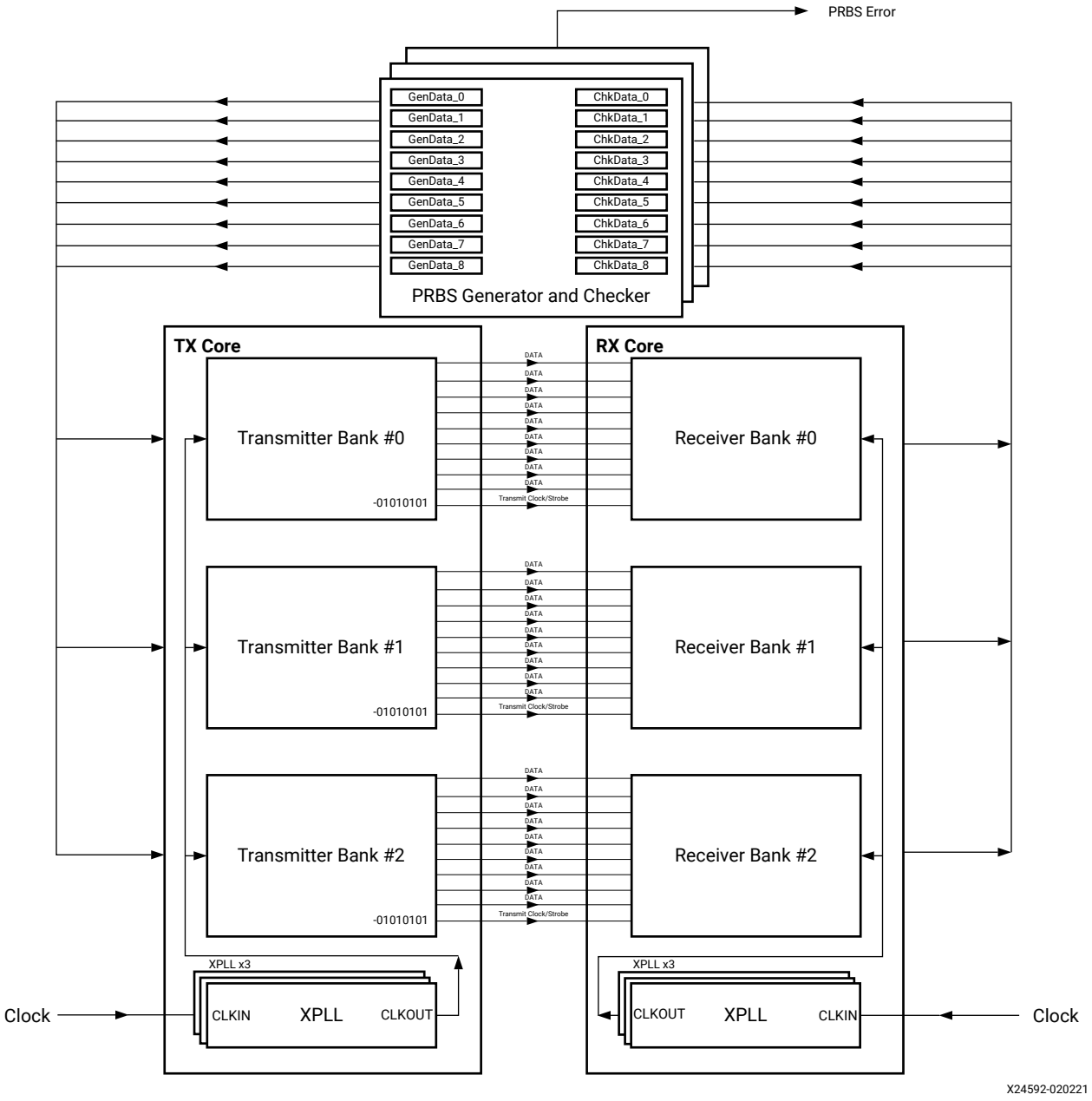
Figure 15: **Clock Connection for RX Core**



The reference design uses the PRBS generator and checker to exercise the I/Os. The design files for the PRBS generator and checker are provided in the design suite. It is instantiated in the top-level source file. The PRBS generator generates the data and feeds the TX core, which serializes it and transmits to the RX core via external loopback. The RX core feeds the data to the PRBS checker after deserializing it. It flags if it detects any mismatch. The block diagram of the reference design is shown in the following figure. The transmit clock or strobe is generated by feeding the pattern `01010101` to the corresponding TX NIBBLESLICEs for each bank.

***Note*:** The design has three PRBS generators/checkers per XPHY nibble, one for each pair of XPHY NIBBLESLICEs transmitting and receiving data.

Figure 16: **Multi-Bank Design**



Pre-Core Generation Setup

Before following the procedure in this section, download the reference design files from the Xilinx website. For detailed information about the design files, see Reference Design.

The following steps describe how to configure and set up the project before building the TX and RX cores using the AIOW.

1. Browse to the folder where the zipped file is downloaded.
2. Unzip the file and open the top-level `xapp1350` folder. Check the `Multi_Bank_Ssync_Loopback_Design` folder, which has all the necessary design files under the `Sources`, `Constraints`, and `Testbench` folders.
3. Create a separate directory named `Versal_Ssync_RxTx_Intrfce_MB` to build the new project.
4. Launch the Vivado tools 2020.1 or later from the newly created directory.
5. Under Quick Start, select **Create Project**.
6. Click **Next** for the prompt to **Create a New Vivado Project** and use `Versal_Ssync_RxTx_Intrfce_MB` for the name of the project. Deselect **Create a project subdirectory**.
7. Click **Next**. For Project Type, select **RTL project**. Deselect **Do not specify sources at this time**.
8. The next step is to add the sources. Add the source files from the `Sources` folder under the `Multi_Bank_Ssync_Loopback_Design` folder.
9. Add the files `toplevel_mb.sv`, `Prbs_Any.vhd`, `Prbs_RxTx.vhd`, and `sync_cell.sv`. Make sure the Library is set to `xil_defaultlib` and the files are used for synthesis and simulation by setting it under HDL Source For.
10. Similarly, add the file `toplevel_testbench_mb.sv` from the `Testbench` folder under the `Multi_Bank_Ssync_Loopback_Design` folder. Make sure the Library is set to `xil_defaultlib` and the file is used just for simulation by setting it under HDL Source For.
11. Select **Scan and add RTL include files into project** and **Copy sources into project**. Set the Target Language to **Verilog** and the Simulator Language to **Mixed**.
12. Click **Next** to proceed to adding the constraint files.
13. Add the files `toplevel_mb.xdc` and `attributes_mb.xdc` from the `Constraints` folder under the `Multi_Bank_Ssync_Loopback_Design` folder. Select **Copy constraints files into project**.
14. Click **Next** to proceed to select the part for the project. Select part **xcvc1902-vsva2197-2MP-e-S-eS1** for the reference designs and then click **Next**.
15. On the summary page for the project, make sure all the details match the settings and then click **Finish**.
16. The Vivado tools should create a project and display the hierarchy of the files under the `Sources` folder.

17. Under the `Sources` folder, right-click on `attributes_mb.xdc` and click **Source File Properties**. Under Source File Properties, open the tab for Properties and under USED_IN, select **opt_design_post**.

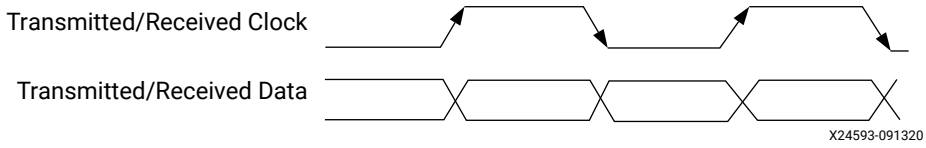
Introduction to Deserialization and Data Reception

Data Reception

In this source synchronous design, the capture clock is the same as the transmit clock, which is looped back from the TX to the RX core. The transmit clock is forwarded with the data by the TX core. The clock-to-data relation in this design is edge aligned and is shown in the following figure. The XPLL in the RX core needs a PLL input clock apart from the capture clock received with the data. This PLL input clock acts as the input to the XPLL (CLKIN port of the XPLL) and should only be received on a GC/XCC pin. Because this is a multi-bank design (three banks), the wizard instantiates three instances of the XPLLs for each bank. Consequently, the PLL input clock received on the GC/XCC pin is fed to the CLKIN ports of all the XPLL instantiations in the design. The capture clock is received on NIBBLESLICE[0] and NIBBLESLICE[1] of NIBBLE[2], which forwards it to other nibbles through inter-byte and inter-nibble clocking. NIBBLESLICE[0] and NIBBLESLICE[1] have clock forwarding abilities. See the "Inter-nibble and Inter-byte Clocking Within an XPIO Bank" figure in the *Versal ACAP SelectIO Resources Architecture Manual* (AM010).

The data received at the RX core interface is transmitted to the programmable logic via the PHY and the bank instances, where it gets checked by the PRBS checker.

Figure 17: **Edge DDR**



Configure and Generate a RX Advanced I/O Wizard Core

Generating a RX Core

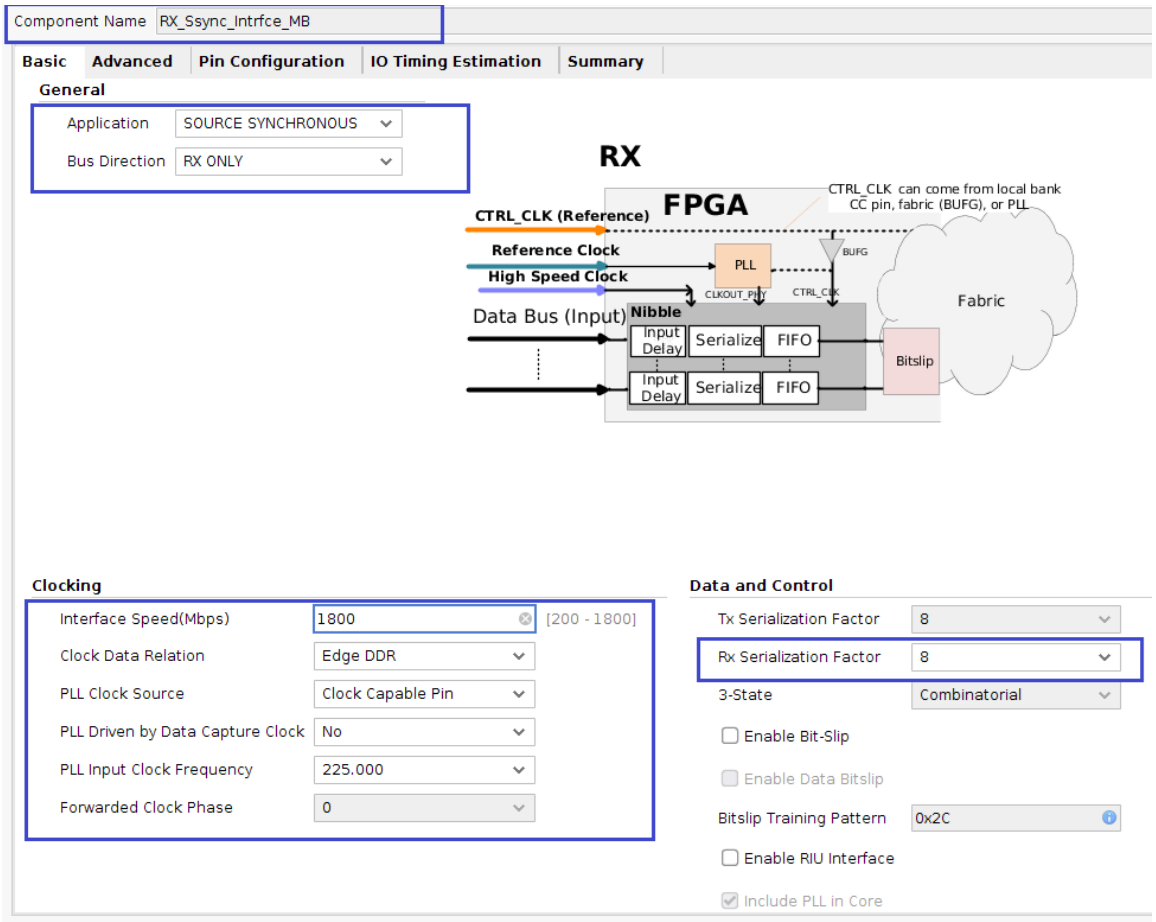
After following the previous steps to generate the top-level design, the next step is to generate a TX and RX core for operation. Follow these steps to generate the RX core using the AIOW. See the figures in this section for reference.

1. To start generating a core for RX, open the IP catalog and search for Advanced I/O Wizard. Double-click **Advanced I/O Wizard** from the catalog to open the Customize IP window.
2. For Component Name, enter `RX_Ssync_Intrfce_MB`, which is used in the reference design.

   ***Note*:** The component name should match the module name used in the top-level design.
3. Under the Basic Tab, set Application to **SOURCE SYNCHRONOUS** from the drop-down list. Set Bus Direction to **RX ONLY**.

4. On the same tab, set the following:
   a. Interface Speed: **1800 Mb/s**.
   b. Clock Data Relation: **Edge DDR**.
   c. PLL Clock Source: **Clock Capable Pin**.
   d. PLL Input Clock Frequency: **225**.
   e. RX Serialization Factor: **8**.
   f. The remaining options can be set to the default.

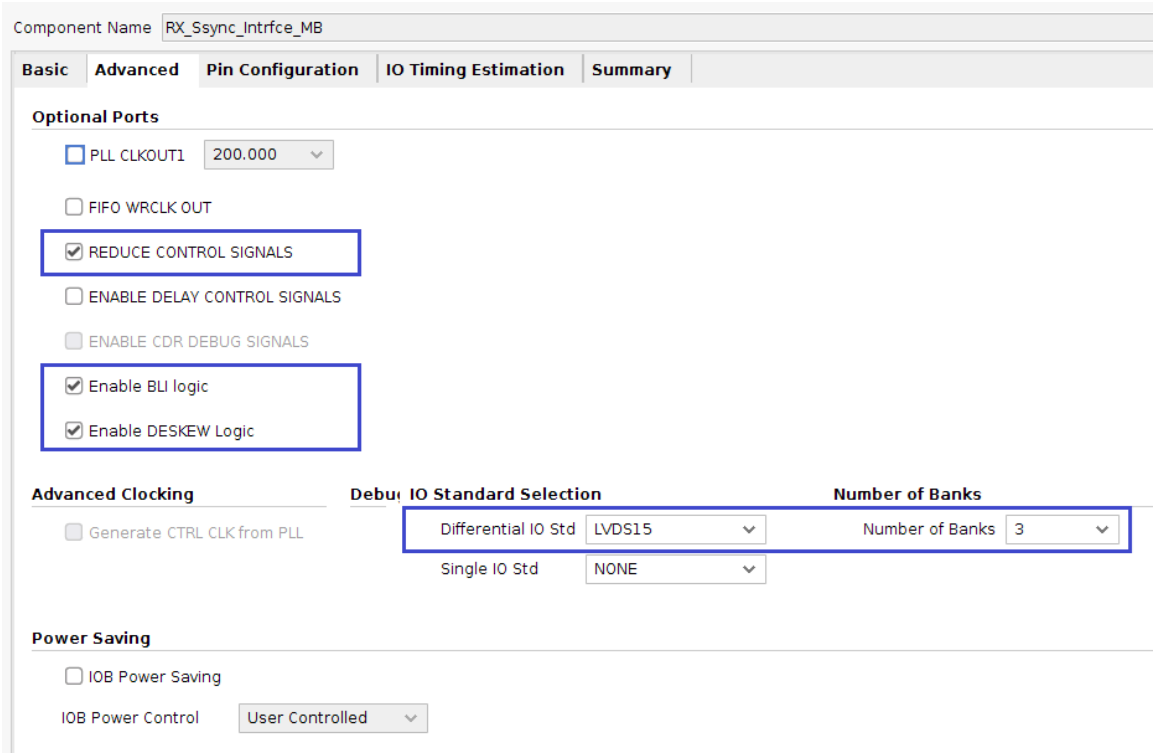
Figure 18: **Multi-Bank RX AIOW Basic Configuration**



5. In the Advanced tab, set the following:
   a. Select **REDUCE CONTROL SIGNALS**, **Enable BLI logic**, and **Enable DESKEW Logic**. When the BLI logic is enabled, the BLI registers between the fabric and the XPHY can be used to help with timing closure. When deskew logic is enabled, the instantiated XPLLs are deskew enabled.

      When Deskew Logic is enabled, XPLLs instantiated will be deskew enabled.
   b. Differential I/O Std: **LVDS15**.
   c. Number of Banks: **3** (because this is a multi-bank design).

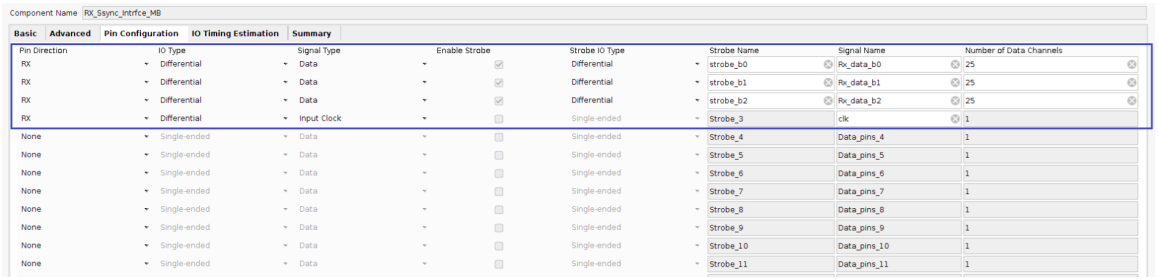
Figure 19: **Multi-Bank RX AIOW Advanced Configuration**



6. In the Pin Configuration tab, make four entries in the table. One entry is for the data and the strobe per bank and the other is for the common PLL input clock for all the banks.
   - Data and the Strobe Setting #1
     - Pin Direction = RX
     - I/O Type = Differential
     - Signal Type = Data
     - Strobe I/O Type = Differential
     - Strobe Name = strobe_b0
     - Signal Name = Rx_data_b0
     - Number of Data Channels = 25
   - Data and the Strobe Setting #2
     - Pin Direction = RX
     - I/O Type = Differential
     - Signal Type = Data
     - Strobe I/O Type = Differential
     - Strobe Name = strobe_b1
     - Signal Name = Rx_data_b1

    - Number of Data Channels = 25
- Data and the Strobe Setting #3
    - Pin Direction = RX
    - I/O Type = Differential
    - Signal Type = Data
    - Strobe I/O Type = Differential
    - Strobe Name = strobe_b2
    - Signal Name = Rx_data_b2
    - Number of Data Channels = 25
- PLL Input Clock Setting
    - Pin Direction = RX
    - I/O Type = Differential
    - Signal Type = Input Clock
    - Signal Name = clk
    - Number of Data Channels = 1

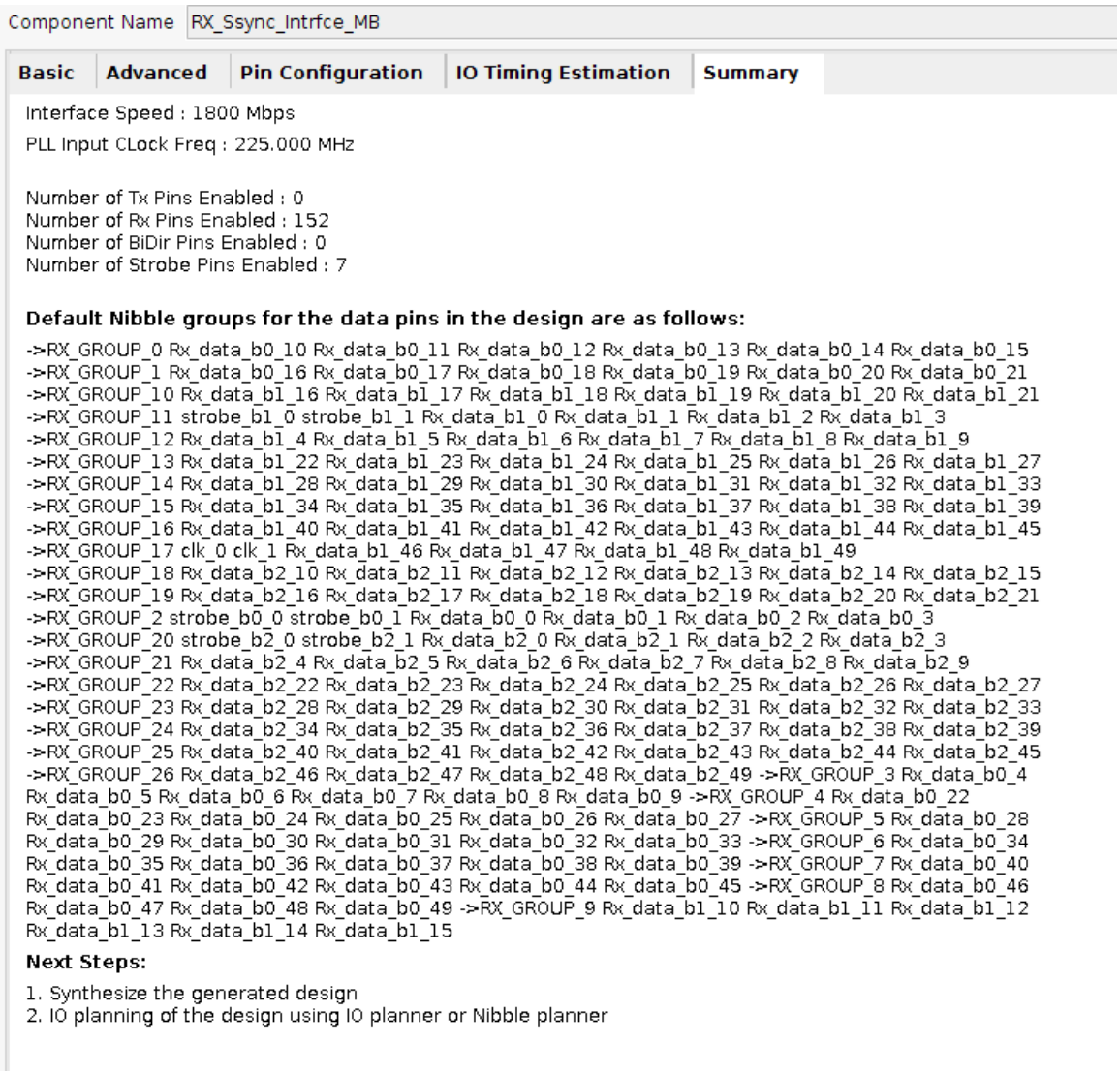
Figure 20: **Multi-Bank RX AIOW Pin Configuration**



7. Check the Summary tab. It should show 158 RX pins enabled (25 pairs of data x 3 banks + 1 pair of capture clocks x 3 banks + a pair for the PLL input clock).

   Note: In the following figure, the number of strobe pins enabled is incorrect. The correct value is 6.

Figure 21: **Multi-Bank RX AIOW Summary**



8. Click **OK** after reviewing the settings. The IP is now customized and the Generate the Output Products prompt appears. Set the Synthesis Option to **Out of context per IP** and then click **Generate** to launch the design run for the newly generated RX core. See the following table.

Table 5: **Receiver Requirements Multi-Bank**

	Requirement
Component name	RX_Ssync_Intrfce_MB
Bus direction	RX_ONLY
Serialization factor	8
Interface speed (Mb/s)	1800 Mb/s
Clock data relation	Edge DDR
PLL clock source	Clock capable pin
PLL input clock frequency	225 MHz
Include PLL in core	Yes
PLL CLKOUT1	No

Table 5: **Receiver Requirements Multi-Bank** *(cont'd)*

	Requirement
FIFO WRCLK OUT	No
Reduce control signals	Yes
Enable delay control signals	No
Enable BLI logic	Yes
Enable deskew logic	Yes
Differential I/O Std	LVDS15
Number of banks	3
Pin configuration	As mentioned above. The configuration occupies 158 pins across all three banks.

Receiver Design Considerations

This RX core is set up to work for a data rate of 1800 Mb/s. Also, the core is configured for LVDS15 in the reference design. This multi-bank design uses a clock capable pin as a PLL clock source and is configured for an edge aligned DDR system. An XPIO bank has 54 pins and the design uses all 54 pins in the form of 25 pairs of data pins, one pair for capture clock, and one pair of PLL input clock, in the bank receiving the input clock. The other two banks use 52 pins in the form of 25 pairs of data pins and one pair for capture clock. The PLL input clock is received by the core and then forwarded to all of the three instantiations of the XPLLs. The wizard allocates one XPLL per bank. The design constrains the ports for the receive interfaces and the wizard takes care of the placement.

Introduction to Serialization and Data Transmission

Data Transmission

In the reference design, the TX core sends out the strobe or the transmit clock along with the data. The data in this design is generated using the PRBS generator and the strobe is generated by feeding the pattern `01010101` to the NIBBLESLICEs forwarding the strobe. The data generated by the PRBS generator is fed into the TX core from the programmable logic, which follows the TX datapath through a serializer and output delay. The serializer supports 8:1, 4:1, and 2:1 serialization. This design uses 8:1 serialization. The data is transmitted through the TX data pins of the core. To understand the data flow operation inside the TX core, see the *Versal ACAP SelectIO Resources Architecture Manual* (AM010).

The TX core needs a PLL input clock that acts as the input to the XPLL (CLKIN port of the XPLL) and should only be received on the GC/XCC pin as in the RX core. This option is available if the clock capable (XCC) pin is selected for the XPLL clock source. The PLL input clock is received on the bank0_pll_clkin, bank1_pll_clkin, and bank2_pll_clkin ports of the TX core, which is passed to the XPLLs instantiated in the core for all three banks. The strobe should be transmitted on NIBBLESLICE[0] and NIBBLESLICE[1] to forward it to the RX core.

The PLL input clock to the TX core is fed to the XPLL through a bank<0/1/2>_pll_clkin port within each bank. In this design, the PLL input clock is provided similarly to the RX core, except that it is passed through an IBUFDS, and the single-ended clock is provided to the respective bank<0/1/2>_pll_clkin ports. The transmit clock can be forwarded either through the data pins or clock forwarding pins on the TX core. In both cases, a clock pattern of `01010101` needs to be fed to either the data pins or the clock forwarding pins. This results in the same output clock. The design uses the data pins to forward the transmit clock. The transmit clock is transmitted on NIBBLESLICE[0] and NIBBLESLICE[1] to the RX core.

Configure and Generate a TX Advanced I/O Wizard Core

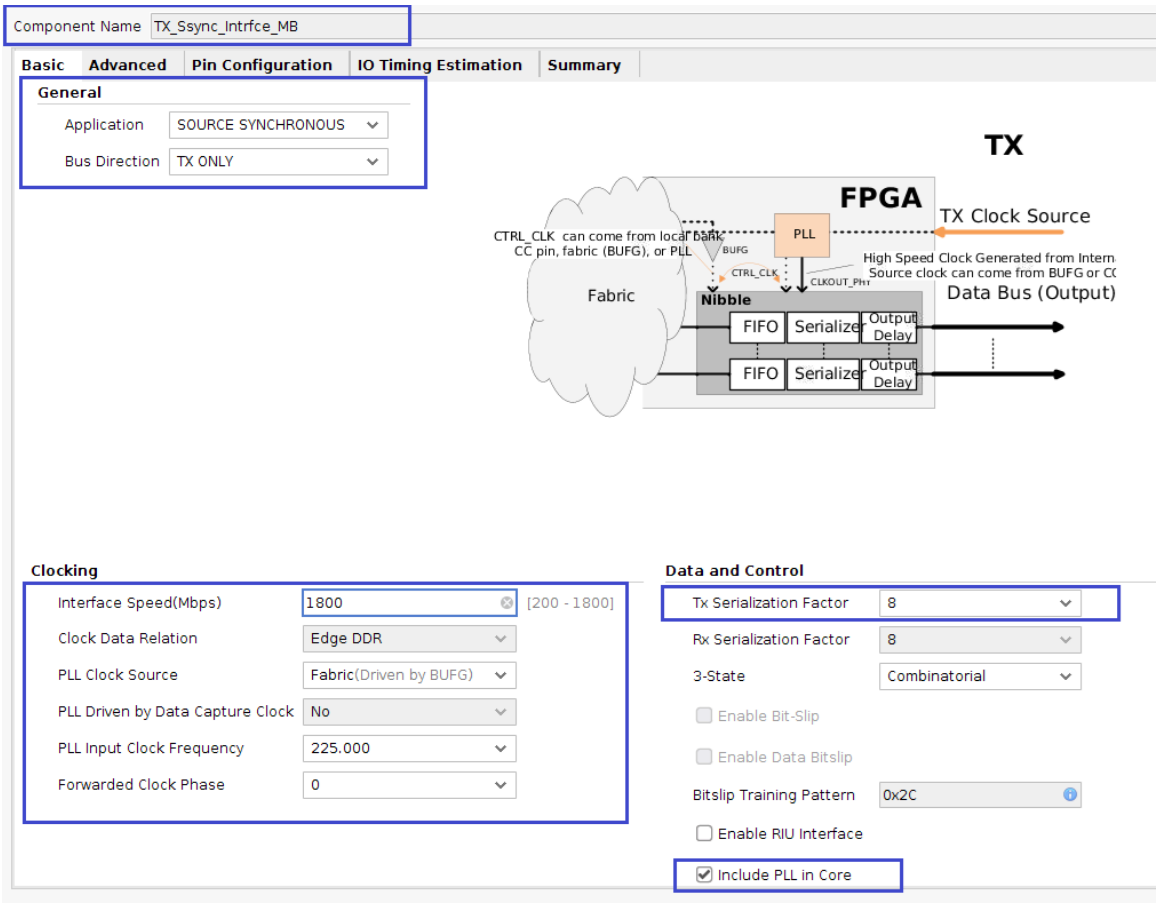
Generating a TX Core

Follow these steps to generate the TX core using the AIOW. See the figures in this section for reference.

1. To start generating a core for TX, open the IP catalog and search for Advanced I/O Wizard. Double-click **Advanced I/O Wizard** to open the Customize IP window for the wizard.
2. For Component Name, enter `TX_Ssync_Intrfce_MB`, which is used in the reference design.

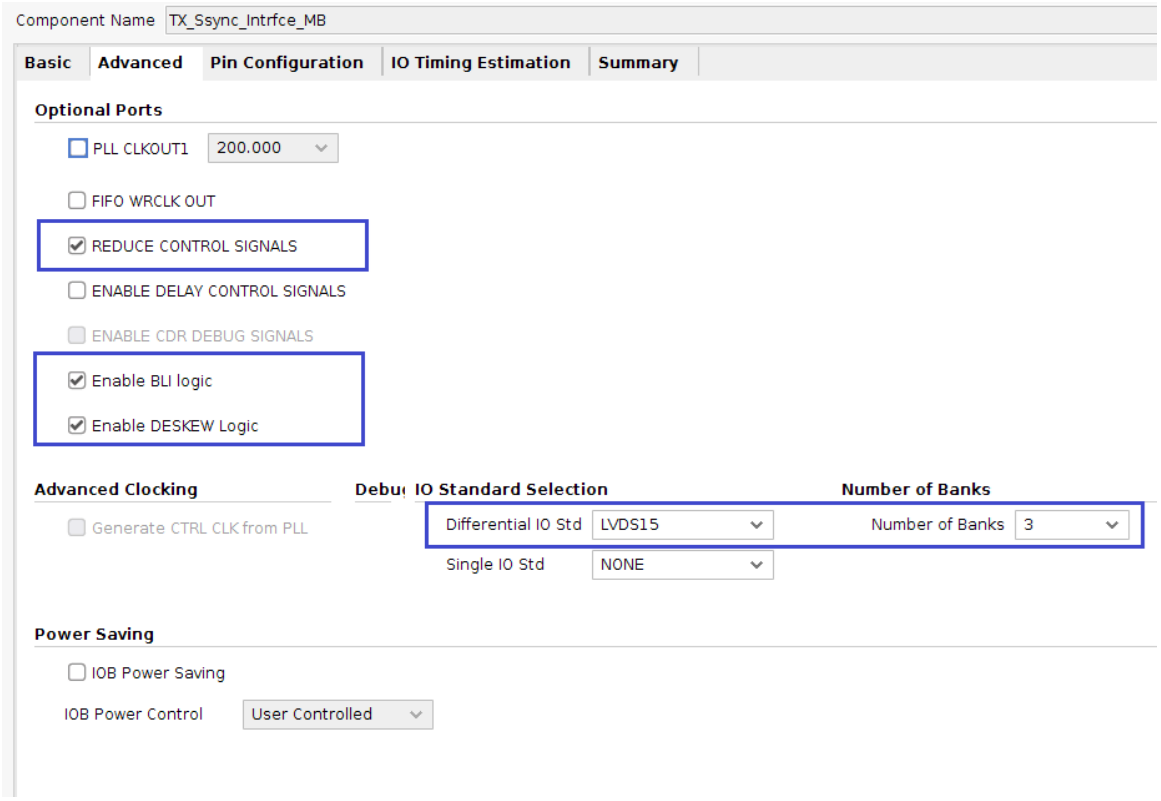
   ***Note*:** The component name should match the module name used in the top-level design.
3. Under the Basic Tab, set Application to **SOURCE SYNCHRONOUS** from the drop-down list, and set Bus Direction to **TX ONLY**.
4. On the same tab, set the following:
   a. Interface Speed: **1800 Mb/s**.
   b. Clock Data Relation: **Edge DDR**.
   c. PLL Clock Source: **Fabric (Driven by BUFG)**.
   d. PLL Input Clock Frequency: **225**.
   e. Forwarded Clock Phase to **0**.
   f. TX Serialization Factor: **8**.
   g. The remaining options can be set to the default.

Figure 22: **Multi-Bank RX AIOW Basic Configuration**



5. In the Advanced tab, set the following:
   a. PLL CLKOUT1: **225**
   b. Select **REDUCE CONTROL SIGNALS**, **Enable BLI logic**, and **Enable DESKEW Logic**. When the BLI logic is enabled, the BLI registers between the fabric and the XPHY can be used to help with timing closure. When the deskew logic is enabled, the instantiated XPLLs are deskew enabled.
   c. Differential I/O Std: **LVDS15**
   d. Number of Banks: **3** (because this is a multi-bank design).

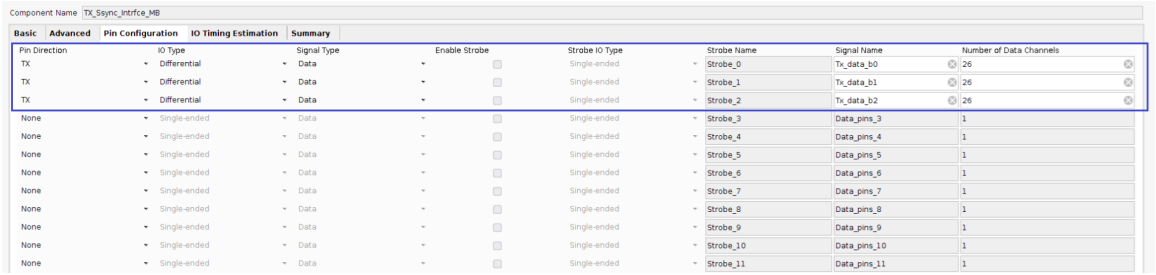
Figure 23: **Multi-Bank RX AIOW Advanced Configuration**



6. In the Pin Configuration tab, add three entries in the table. This entry is for the data with I/O type as differential, signal type as data, and mark the signal name as “Tx_data_b#”, where # is the bank number. This also accounts for the transmit clock.
   - Data Setting #1
     - Pin Direction = TX
     - I/O Type = Differential
     - Signal Type = Data
     - Signal Name = Tx_data_b0
     - Number of Data Channels = 26
   - Data Setting #2
     - Pin Direction = TX
     - I/O Type = Differential
     - Signal Type = Data
     - Signal Name = Tx_data_b1
     - Number of Data Channels = 26
   - Data Setting #3
     - Pin Direction = TX

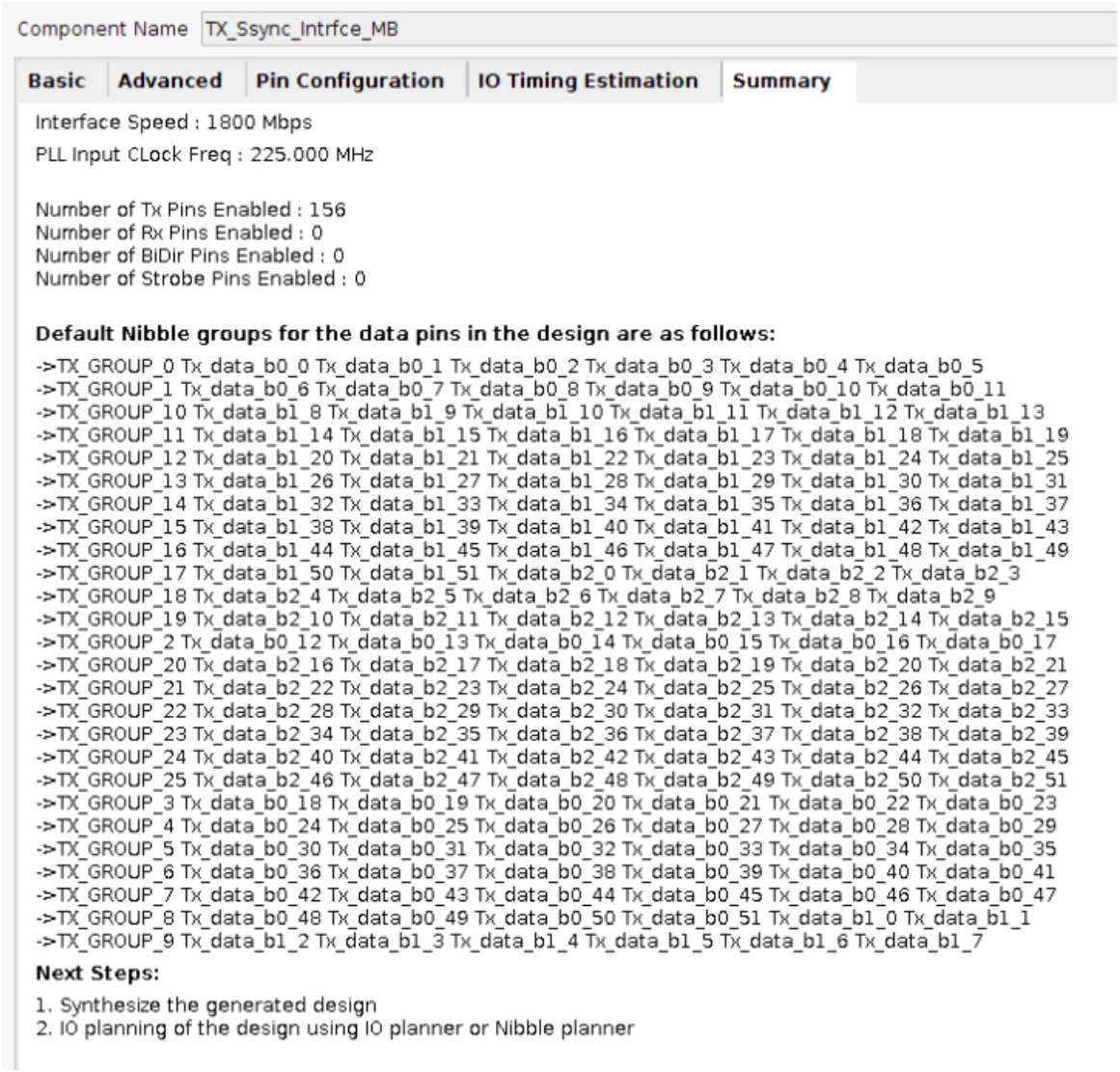
- I/O Type = Differential
- Signal Type = Data
- Signal Name = Tx_data_b2
- Number of Data Channels = 26

Figure 24: **Multi-Bank RX AIOW Pin Configuration**



7. Check the Summary tab. It should show 156 TX pins enabled (25 pairs of data x 3 banks + 1 pair of transmit clock x 3 banks).

Figure 25: **Multi-Bank RX AIOW Summary**



8. Click **OK** after reviewing the settings. The IP is now customized and the Generate the Output Products prompt appears. Set the Synthesis Option to **Out of context per IP** and then click **Generate** to launch the design run for the newly generated TX core.

Table 6: **Transmitter Requirements Multi-Bank**

	Requirement
Component name	TX_Ssync_Intrfce_MB
Bus direction	TX_ONLY
Serialization factor	8
Interface speed (Mb/s)	1800 Mb/s
Clock data relation	Edge DDR
PLL clock source	Fabric (driven by BUFG)
Forwarded clock phase	0
PLL input clock frequency	225 MHz
Include PLL in core	Yes

Table 6: **Transmitter Requirements Multi-Bank** *(cont'd)*

	Requirement
PLL CLKOUT1	No
FIFO WRCLK OUT	No
Reduce control signals	Yes
Enable delay control signals	No
Enable BLI logic	Yes
Enable deskew logic	Yes
Differential I/O Std	LVDS15
Number of banks	3
Pin configuration	The number of data channels is set to 26 pairs. One pair is reserved for the transmit clock.

Transmitter Design Considerations

This TX core is set up for a data rate of 1800 Mb/s. Also, the core is configured and tested for LVDS15 in this reference design. The PLL input clock to the TX core is fed to the XPLL through a bank<0/1/2>_pll_clkin port within each bank. In the design, the PLL input clock is provided similarly to the RX core, except that it is passed through an IBUFDS, and the single-ended clock is provided to the respective bank<0/1/2>_pll_clkin ports. An XPIO bank has 54 pins and the design uses 52 pins in the form of 25 pairs of data pins and one pair of strobe for each of the three bank instances. The RX core has a pair of pins reserved for the PLL input clock, and consequently, the TX core leaves an equivalent pair unused. The design constrains the ports for the transmit interfaces and the wizard takes care of the placement.

Top-Level using TX and RX Advanced I/O Wizard Cores

The top-level design file (`toplevel_mb.sv`) includes the `toplevel_mb` module. This module helps connect the interfaces such as clocks, debug ports, I/O ports, etc., with the appropriate sources. The top-level design houses the instantiation of both the RX and the TX cores. To test the design, the PRBS patterns from the custom PRBS generator provided in the design suite can be used to generate and check the received data.

The PRBS generator generates the 8-bit data for each pair and feeds it to the TX core, which in turn transmits it through the TX pins. The PRBS generator also houses an error injecting mechanism. The data is received on the I/O ports of the RX core through external loopback. The RX core forwards the deserialized data through the PHY to the programmable logic. This deserialized data is then fed into the PRBS checker to check for any failures.

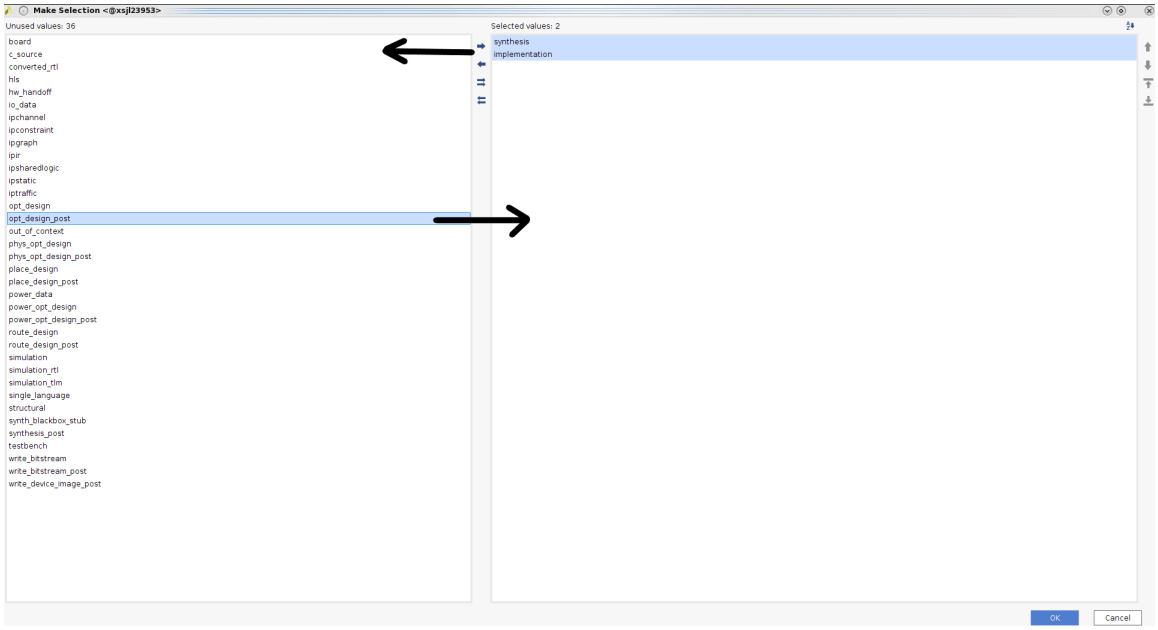
The top-level design files include two constraints files. The `toplevel_mb.xdc` file is used to create clocks for the design and assign a location to all the I/O ports in the design. The `attributes_mb.xdc` file is provided as a placeholder for any attributes to be added for non-default bank instances. The current design is provided as an empty file because it does not have any constraints for non-default bank instances. This file must be marked for post-opt usage. The

current release of the Vivado tools fails to recognize any bank instances before the optimization stage, and consequently, the design uses two separate constraints files. As a result, any constraints to modify the attributes of the non-default bank instance are recognized, but the ones for the non-default bank instances are ignored. Consequently, the second constraint file needs to be added and marked to be used for post-optimization.

To set the Xilinx design constraints (XDC) to be used in post_opt, open the Vivado tools and follow these steps.

1. Select the XDC file, right-click, and click **Source File Properties**.
2. Select **Properties** from the Source File Properties window.
3. Navigate to USED IN. Move `opt_design_post` from left to right and `synthesis` and `implementation` from right to left as shown in the following figure.

Figure 26: **Setting the XDC to be Used in Post_Opt**



4. Click **OK**.

The reference design constrains the design to optimally support high data rates. This file is used to create clocks for the design, assign locations or pins to all the I/O ports in the design, and set attributes if needed. The reference design constrains the design to optimally support high data rates. The user must constrain the TX and RX ports. The Vivado tools can assign the XPHY nibbles to the XPHY sites.

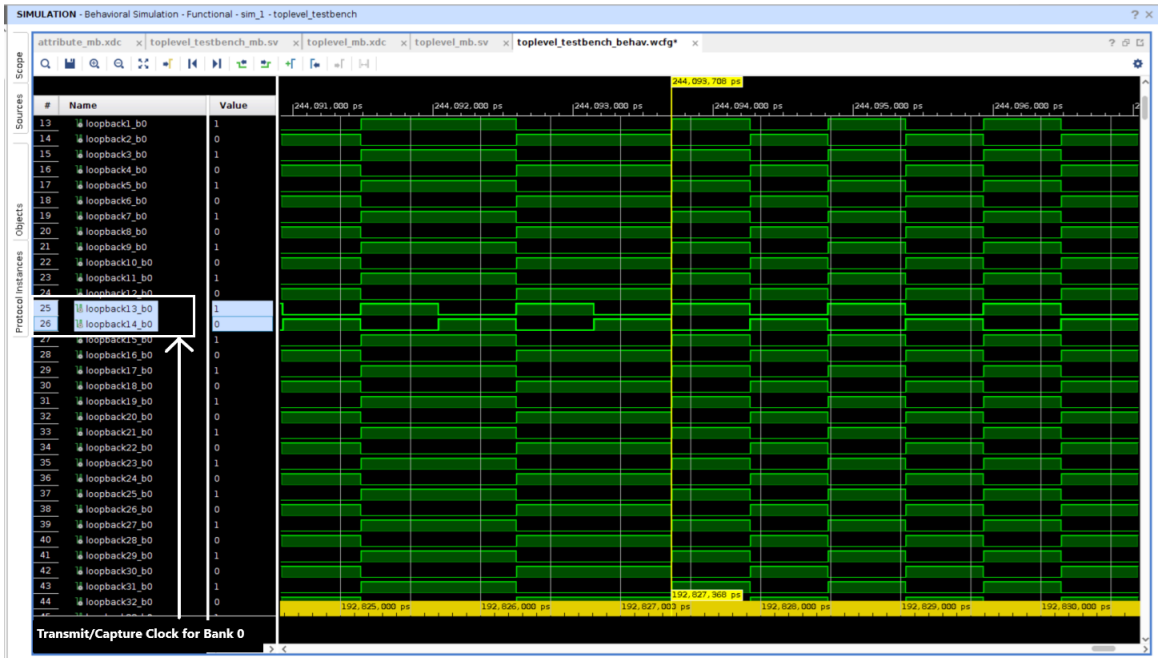
In the reference design, bank 706, bank 707, and bank 708 for part xcvc1902-vsva2197-2MP-e-S-eS1 are used for the TX core and bank 703, bank 704, and bank 705 are used for the RX core. The RX core has one pair reserved for the PLL input clock per bank, the design assigns IO_L9P_GC_XCC_N3P0_M1P72_704_BE31 and IO_L9N_GC_XCC_N3P1_M1P73_704_BD32 in the constraint file for bank 704. Similarly, IO_L6P_GC_XCC_N2P0_M1P66_704_BC31 and IO_L6N_GC_XCC_N2P1_M1P67_704_BC30 are assigned to the capture clock. The corresponding pins in other banks for both the TX core and RX core are assigned similarly.

At this point all the files are added to the project and both the RX and TX cores are generated. The design is ready to be synthesized and implemented.

Simulation

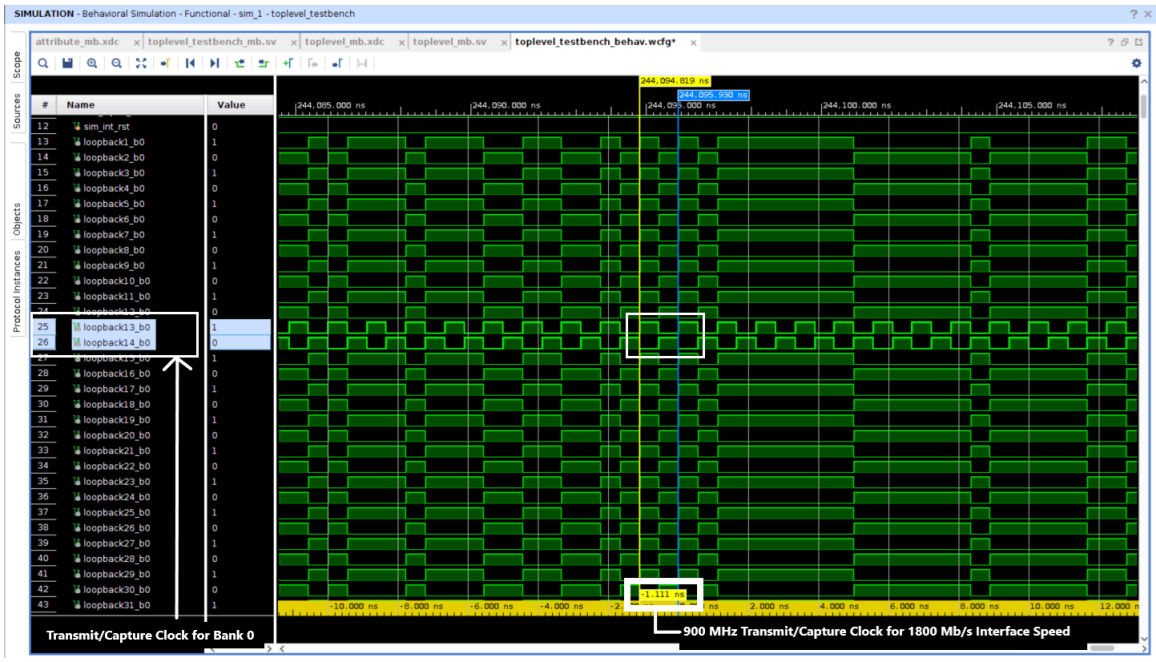
The design uses the `toplevel_testbench_mb.sv` file to create a simple test bench. This test bench connects the TX core to the RX core via loopback<num>_b<0/1/2> connections (wires). The transmit clock is transmitted on loopback13_b<0/1/2> and loopback14_b<0/1/2> loopback wires. All the other loopback connections are used to transmit and receive the data. The clock-to-data relationship is edge aligned as shown in the following figure captured from the simulation.

Figure 27: **Clock-to-Data Relation - Edge Alignment**



The test bench provides the necessary clock and resets to the design and triggers its operation. The PLL input clock is provided at 4.444 ns (225 MHz) to the RX and TX cores. The transmit/capture clock toggles at 900 Mb/s (1.111 ns period) as shown in the following figure. Because the system is double data rate, the interface operates at 1800 Mb/s.

Figure 28: **Multi-Bank Design - Interface Speed 1800 Mb/s**



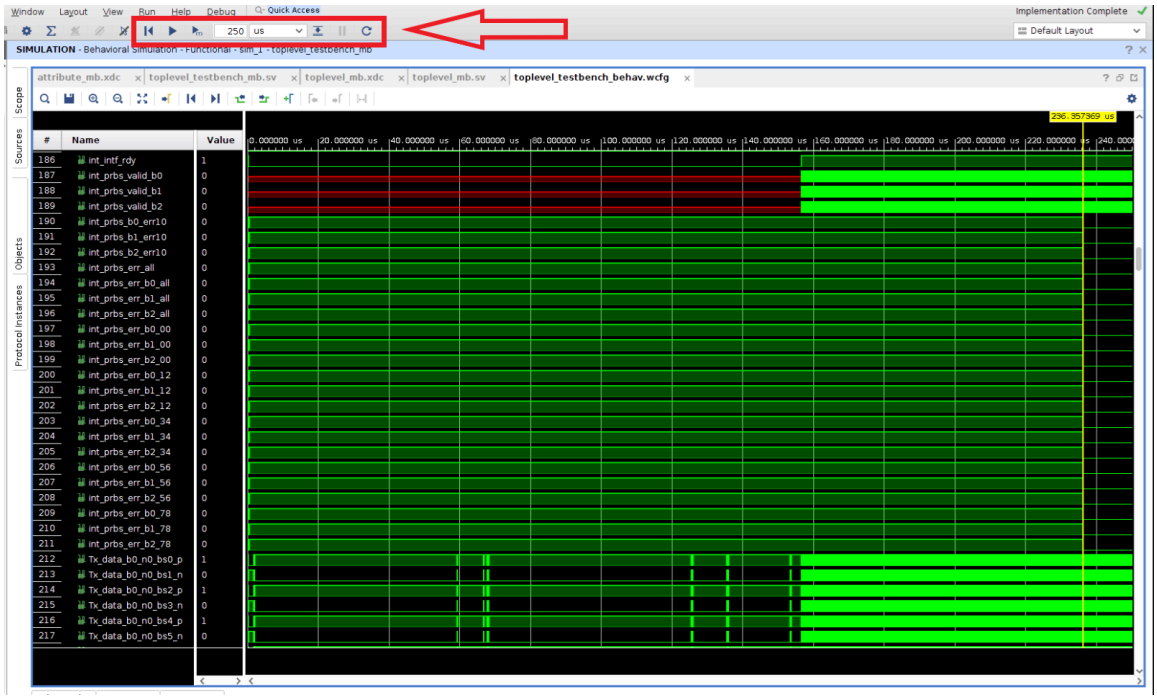
The Vivado 2020.1 tools allow only behavioral simulation when using the AIOW.

Simulating the Design

The design is tested with the Vivado Simulator 2020.1. This section describes how to launch the simulation. Assuming the Vivado project is already created for the design, follow these steps to simulate the design.

1. Click **Run Simulation** under Simulation from the Flow Navigator.
2. From the listed options, select **Run Behavioral Simulation**, which elaborates the design and launches the simulation.
3. The run time by default is 1 ns for the simulation. It takes about 240 µs for the intf_rdy to be asserted for the RX core. Only after its assertion and the next Prbs_Valid, should the data be compared. Consequently, launch the simulation for a duration of more than 240 µs using the box highlighted in the following figure. A waveform config file `toplevel_testbench_behav.wcfg` is provided under the `Testbench` directory to add a list of signals for the simulation.

Figure 29: **Multi-Bank Design - Changing the Simulation Time**



4. Once the behavioral simulation is finished, check for any errors by observing error flags for the PRBS generator and checker. For example, int_prbs_b<0/1/2>_err<num> reports the errors for each instantiation of the PRBS generator and checker. Int_prbs_b0_err24 denotes the error flag for the PRBS module instantiated for NIBBLE[2], NIBBLESLICE[4] of bank instance 0. Int_prbs_err_b<0/1/2>_00 denotes the error for any NIBBLESLICEs on NIBBLE[0]. Int_prbs_err_b<0/1/2>_12 denotes the error for any NIBBLESLICEs on NIBBLE[1] and NIBBLE[2]. Int_prbs_err_b0_all denotes an error on any NIBBLESLICE across all the nibbles in bank instance 0. Int_prbs_err_all denotes an error on any NIBBLESLICE across all the nibbles in any of the bank instances.

Reference Design

Download the reference design files from the Xilinx website. Unzip this file and browse into the top-level folder `xapp1350`. Check the `Multi_Bank_Ssync_Loopback_Design` folder, which has all the source files under `Sources`, `Constraints`, and `Testbench`.

Table 7: **Source Files for Multi-Bank Design**

Folder	File Name	Description
Sources	`toplevel_mb.sv`	The top-level design file that instantiates the TX and RX cores and connects the design
	`Prbs_RxTx.vhd`	Outer wrapper for the custom PRBS generator and checker
	`Prbs_Any.vhd`	Contains the modules for the PRBS generator and checker

Table 7: **Source Files for Multi-Bank Design** *(cont'd)*

Folder	File Name	Description
Constraints	toplevel_mb.xdc	The constraints file to assign pin locations to ports of the design for all bank instances and set the necessary attributes for bank instance 0 (Default bank instance)
	attributes_mb.xdc	The constraints file to set the attributes for bank instances 1 and 2 (Non-default bank instances). This file is marked to be used only for post optimization.
Test Bench	toplevel_testbench_mb.sv	The top-level test bench file to test the design
	toplevel_testbench_behav.wcfg	Waveform configuration file

Reference Design Matrix

The following checklist indicates the procedures used for the provided reference design.

Table 8: **Reference Design Matrix**

Parameter	Description
General	
Developer name	Xilinx
Target devices	Versal ACAP
Source code provided?	Yes
Source code format (if provided)	Verilog and VHDL
Design uses code or IP from existing reference design, application note, 3rd party or Vivado software? If yes, list.	Yes. Uses the Advanced I/O Wizard from the IP catalog.
Simulation	
Functional simulation performed	Yes
Timing simulation performed?	No
Test bench provided for functional and timing simulation?	Yes
Test bench format	Verilog
Simulator software and version	Vivado simulator 2020.1
SPICE/IBIS simulations	No
Implementation	
Synthesis software tools/versions used	Vivado synthesis 2020.1
Implementation software tool(s) and version	Vivado 2020.1 implementation
Static timing analysis performed?	Yes
Hardware Verification	
Hardware verified?	No
Platform used for verification	N/A

Conclusion

This application note confirms the use of the Advanced I/O Wizard for the source synchronous application. It uses two reference designs, a single-bank source synchronous design and a multi-bank source synchronous design to showcase how the wizard instantiates the bank instances. It can further be expanded to modify various attributes of the XPHY nibbles and XPLL for further applications.

References

These documents provide supplemental material useful with this guide:

1. *Versal Architecture and Product Data Sheet: Overview* (DS950)
2. *Versal ACAP Clocking Resources Architecture Manual* (AM003)
3. *Versal ACAP SelectIO Resources Architecture Manual* (AM010)
4. *Advanced I/O Wizard LogiCORE IP Product Guide* (PG320)

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
02/04/2021 Version 1.0	
Initial release.	N/A

Please Read: Important Legal Notices

AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

Copyright