LogiCORE IP SelectIO Interface Wizard v4.1

Getting Started Guide

UG700 April 24, 2012

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
09/16/09	1.1	Initial Xilinx release.
12/02/09	1.2	Updated tool versions. Added Phase Detector Interface Ports to Table 3-2. Added Phase Detector in Chapter 4. Replaced the following: IODELAY with IODELAY2; ISERDES with ISERDES2; OSERDES with OSERDES2; IDDR with IDDR2; ODDR with ODDR2.
04/19/10	1.3	Updated Wizard and tool versions. Miscellaneous edits for clarification.
07/23/10	1.4	Revised LogiCORE IP Facts table's format and content. Added support for Virtex-6 FGPAs. Added Chapter 5, Generating the Core - Zynq-7000, 7 Series and Virtex-6 FPGAs.
12/14/10	1.5	Updated Wizard and tool versions. In Chapter 4, Generating the Core for Spartan-6 FPGAs: Added Interface for I/O Configuration, page 16, expanded Data Bus Direction, page 16, Clock Forwarding, page 18, and added Summary Page, page 22. In Chapter 5, Generating the Core - Zynq-7000, 7 Series and Virtex-6 FPGAs: Added Interface for IO Configuration, page 26, expanded Data Bus Direction, page 26, Clock Forwarding, page 28, and added Summary Page, page 31.
03/01/11	2.0	Added support for Virtex-7 and Kintex-7 device support. Updated GUI screens. Support for ISE software and tools v13.1.
06/22/11	3.0	Updated GUI screens. Added support for ISE software and tools v13.2.
01/18/12	4.0	Updated for core version 3.3. Added IDDR information and support for ISE software and tools v13.4.
04/24/12	5.0	Updated for core version 4.1 and ISE software version 14.1. Added support for Zynq-7000 and 7 series devices. Added constraints files to descriptions of Directory and File Contents, page 32. Added IODELAYE2 to Delay Type, page 19.

Table of Contents

Chapter 6: Detailed Example Design

XILINX

Chapter 1

Introduction

This chapter introduces the LogiCORE™ IP SelectIO™ Interface Wizard core and provides related information, including recommended design experience, additional resources, technical support, and submitting feedback to Xilinx. The SelectIO Interface Wizard core generates source code to implement an I/O circuit matched to your requirements and is designed to support both Verilog and VHDL design environments. In addition, the example design delivered with the core is provided in both Verilog and VHDL.

About the Core

The SelectIO Interface Wizard core is an ISE® CORE Generator™ IP core that automates the configuration of the SelectIO resources in Zynq™-7000, 7 series, Virtex®-6, and Spartan®-6 FPGAs.

Recommended Design Experience

The SelectIO Interface Wizard is designed to be used by those will some level of experience with Xilinx FPGA I/Os. It is the easiest way to create your I/O circuit. Advanced users may choose to modify the generated source code directly. Although the SelectIO Interface Wizard provides a fully verified solution, understanding the Xilinx I/O primitives will help in making design trade-off decisions.

Related Xilinx Documents

- UG471: *7 Series FPGAs SelectIO Resources User Guide*
- UG472: *7 Series FPGAs Clocking Resources User Guide*
- UG361: *Virtex-6 FPGA SelectIO Resources User Guide*
- UG362: *Virtex-6 FPGA Clocking Resources User Guide*
- UG381: *Spartan-6 FPGA SelectIO Resources User Guide*
- UG382: *Spartan-6 FPGA Clocking Resources User Guide*
- DS709: *LogiCORE IP Clocking Wizard Data Sheet*
- UG521: *LogiCORE IP Clocking Wizard Getting Started Guide*
- ISE® Design Suite documentation

Additional Core Resources

For detailed information and updates about the SelectIO Interface Wizard core, see the following documents, located on the Architecture Wizards product page:

- DS746: *LogiCORE IP SelectIO Interface Wizard Data Sheet*
- SelectIO Interface Wizard Release Notes

Technical Support

For technical support, go to www.xilinx.com/support. Questions are routed to a team with expertise using the SelectIO Interface Wizard core.

Xilinx will provide technical support for use of this product as described in this guide. Xilinx cannot guarantee timing, functionality, or support of this product for designs that do not follow these guidelines.

Ordering Information

The LogiCORE IP SelectIO Wizard is provided free of charge under the terms of the Xilinx End User License Agreement. The Wizard can be generated using the Xilinx ISE CORE Generator software, which is a standard component of the Xilinx ISE Design Suite. This version of the core can be generated using the ISE CORE Generator system v14.1. For more information, please visit the Architecture Wizards web page.

Information about additional Xilinx LogiCORE modules is available at the Xilinx IP Center. For pricing and availability of other Xilinx LogiCORE modules and software, please contact your local Xilinx sales representative.

Feedback

Xilinx welcomes comments and suggestions about the SelectIO Interface Wizard core and the accompanying documentation.

SelectIO Interface Wizard

For comments or suggestions about the SelectIO Interface Wizard core, please submit a WebCase from www.xilinx.com/support/clearexpress/websupport.htm. Be sure to include the following information:

- Product name
- Core version number
- Explanation of your comments

Document

For comments or suggestions about the SelectIO Interface Wizard core, please submit a WebCase from www.xilinx.com/support/clearexpress/websupport.htm. Be sure to include the following information:

- Document title
- Document number
- Page number(s) to which your comments refer

Chapter 2

Installation

This chapter provides information about installing the LogiCORE™ IP SelectIO™ Interface Wizard. It is not necessary to obtain a license to use the Wizard.

Supported Tools and System Requirements

For a list of System Requirements, see the *ISE Design Suite 14: Release Notes Guide* at the web page for [14.1 Release Notes/Known Issues].

Before You Begin

Before installing the Wizard, you must have an account. To create an account, Click **Login** at the top of the Xilinx.com home page then follow the on screen instructions to create an account.

Installing the Wizard

The SelectIO Interface Wizard is included with the 11.3 and later versions of ISE software and can be accessed from the ISE CORE Generator tool.

For detailed ISE software installation instructions, see the ISE Design Suite Release Notes and Installation Guide available in the ISE software section of the Documentation Center under "Design Tools" at www.xilinx.com/support/documentation.

Verifying Your Installation

Use the following procedure to verify that you have successfully installed the LogiCORE IP SelectIO Interface Wizard in the CORE Generator tool.

1. Start the CORE Generator tool.
2. The IP core functional categories appear at the left side of the window, as shown in Figure 2-1.

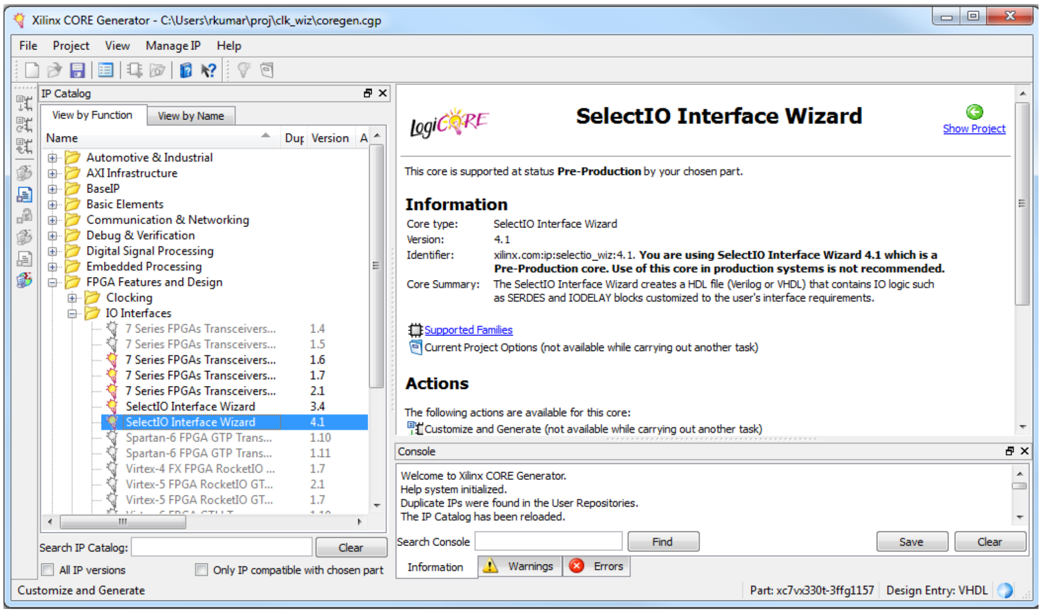

Figure 2-1: **CORE Generator Tool Window**

3. Click to expand or collapse the view of individual functional categories, or click the **View by Name** tab at the top of the list to see an alphabetical list of all cores in all categories.
4. Determine if the installation was successful by verifying that SelectIO Interface Wizard v4.1 appears at the following location in the Functional Categories list:
`/FPGA Features and Design/IO Interfaces`

Core Architecture

The SelectIO™ Interface Wizard provides source HDL that implements an I/O circuit for an input, output or bidirectional bus, including the buffer, any required delay elements, ISERDES and OSERDES elements, registers, and the I/O clock driver. The circuit is designed in two major components: clock buffering and manipulation, and datapath, which is implemented per-pin.

Clock Buffering and Manipulation

The wizard supports the use of a BUFG, BUFIO, BUFIO2, or BUFPLL for clocking the I/O logic. An example circuit illustrating a BUFIO2 primitive with input data is illustrated in Figure 3-1.

Insertion delay can be added for the input clock (except in the case of a BUFPLL, which is driven from a PLL_BASE in fabric).

For serialization or deserialization of the datapath, the slower divided fabric clock is created and/or aligned to the input clock on behalf of the user (except in the case of a BUFG, which does not support serialized/deserialized data).

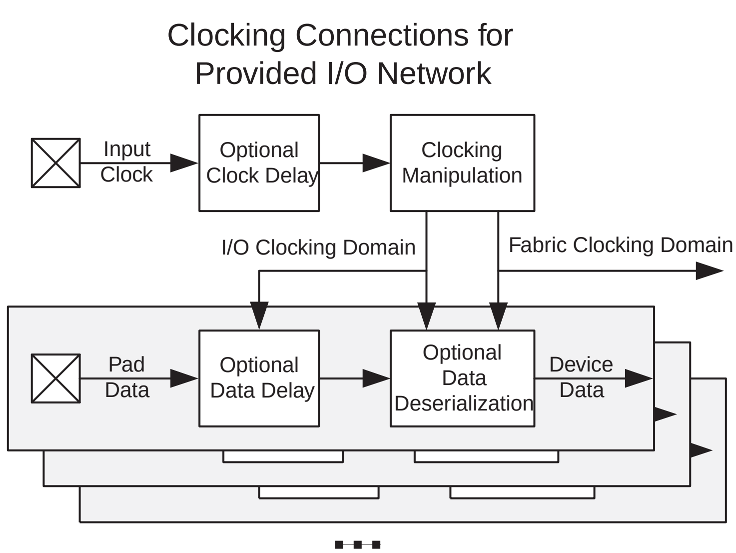

Figure 3-1: **Provided I/O Circuit**

Datapath

The wizard assists the user in instantiating and configuring the components within the I/O interconnect block.

The user can choose to:

- Use or bypass the delay insertion functionality
- Use serialization/deserialization through use of Input SERDES or Output SERDES
- Register double data-rate data
- Use the I/O registers for single rate data
- Drive directly into the fabric

The dataflow graph for an input bus is shown in Figure 3-2. For an output bus, the components will be similar, but the data will flow in the other direction. For a bidirectional bus, there will be both an input and output path, although there is only one IODELAY2 or IODELAYE1 element.

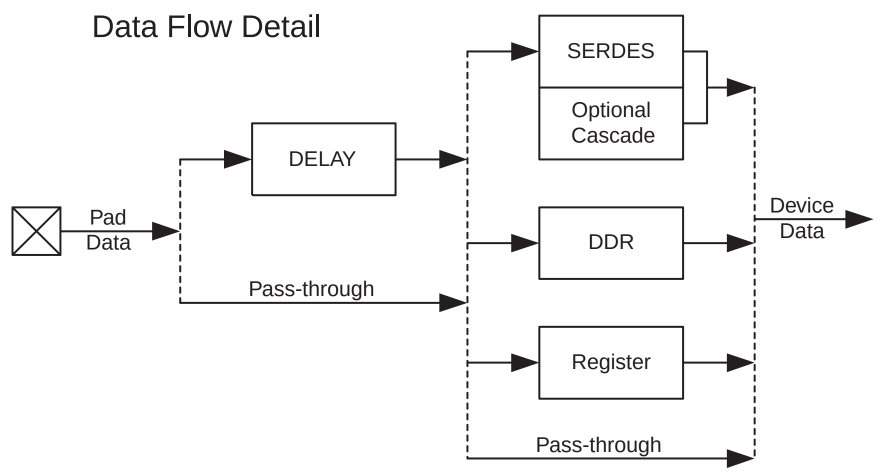

Figure 3-2: **Flow in the I/O Input Datapath**

I/O Signals

Table 3-2 describes the input and output ports provided by the I/O circuit. All ports are optional, although there will be at least one input clock, one signal tied to a pin connection, and one signal tied to a device connection. Availability of the ports is controlled by user-selected parameters. For example, when a variable delay is selected, the delay programming ports are exposed to the user.

Table 3-1 provides a list of resources for specific I/O interconnect and clock primitives.

Table 3-1: **SelectIO and Clock Resources**

I/O Primitives	Document
Interconnect	*Spartan-6 FPGA SelectIO Resources User Guide* *Virtex-6 FPGA SelectIO Resources User Guide* *7-Series FPGAs SelectIO Resources User Guide*
Clock	*7 Series Clocking Resources User Guide* *Virtex-6 FPGA Clocking Resources User Guide* *Spartan-6 FPGA Clocking Resources User Guide*

Table 3-2 defines the I/O circuit input and output ports.

Table 3-2: **I/O Circuit Input and Output Port Descriptions**

Port	I/O	Description
Clock Ports (1)		
CLK_IN	Input	Clock in: Single-ended input clock. Available when a single-ended clock is selected.
CLK_IN_P CLK_IN_N	Input	Clock in Positive and Negative. Available when a differential clock source is selected.
CLK_OUT	Output	Clock out: Buffered and/or delayed output clock to connect to fabric. Available when no serialization is selected, and the clock primitive is not a BUFPLL or MMCM.
CLK_DIV_IN	Input	Clock divided in: Input clock for serialization in the I/O Logic. Available when serialization is chosen, and the clock primitive is BUFPLL/MMCM.
CLK_DIV_OUT	Output	Clock divided out: Buffered and divided output clock to connect to fabric. Available when serialization is selected, and the clock primitive is a BUFIO2 or BUFIO.
Reset Ports		
CLK_RESET	Input	Clock reset: Reset connected to clocking elements in the circuit.
IO_RESET	Input	I/O reset: Reset connected to all other elements in the circuit.

Table 3-2: **I/O Circuit Input and Output Port Descriptions *(Cont'd)***

Port	I/O	Description
SYNC_RESET	Input	Sync reset: Reset is connected to IDDR when IDDR reset type is set to SYNC.
Pin Data Bus Ports		
DATA_IN_FROM_PINS	Input	Data in from pins: Single-ended input bus on the side of the pins.
DATA_IN_FROM_PINS_P DATA_IN_FROM_PINS_N	Input	Data in from pins positive and negative: Differential input bus on the side of the pins.
DATA_OUT_TO_PINS	Output	Data out to pins: Single-ended output bus on the side of the pins.
DATA_OUT_TO_PINS_P DATA_OUT_TO_PINS_N	Output	Data out to pins positive and negative: Differential output bus on the side of the pins.
DATA_TO_AND_FROM_PINS	Input/ Output	Data to and from pins: Single-ended bidirectional data bus on the side of the pins
DATA_TO_AND_FROM_PINS_P DATA_TO_AND_FROM_PINS_N	Input/ Output	Data to and from pins positive and negative: Differential bidirectional data bus on the side of the pins.
Device Data Bus Ports		
DATA_IN_TO_DEVICE	Output	Data in to device: Input bus on the side of the device.
DATA_OUT_FROM_DEVICE	Input	Data out from device: Output bus on the side of the device.
Control and Status Ports		
BITSLIP	Input	Bit slip: Enable bit slip functionality on input data. Available on a input datapath and when enabled. For 7 series and Virtex-6 based designs, this functionality is present for ISERDES in NETWORKING mode.
TRAIN	Input	Train: Enable the training pattern. The train function is a means of specifying a fixed output pattern that can be used to calibrate the receiver of the signal. This port allows the FPGA logic to control whether the output is the fixed training pattern or the output data from the pins. Available on a output datapath and when enabled. This is available for Spartan-6 only.
TRISTATE_OUTPUT	Input	3-state Output: Disables the output path. This signal is synchronized with the input data. Available with a bidirectional datapath.
LOCKED_IN	Input	Locked In: The input clock generator has locked. Available with a BUFPLL. Connect to locked indicator from the PLL in the fabric.

Table 3-2: **I/O Circuit Input and Output Port Descriptions *(Cont'd)***

Port	I/O	Description
LOCKED_OUT	Output	Locked Out: The BUFPLL has locked. Use this signal as the PLL locked indicator.
Variable Delay Ports		
DELAY_BUSY	Output	Delay busy: The variable delay circuity is still busy- don't change current state.
DELAY_CLK	Input	Delay clock: The clock used to control the variable delay circuitry. Most designs will have this connected to the divided/buffered clock for the I/O logic.
DELAY_DATA_CAL	Input	Delay data calibrate: Trigger calibration on the delay for the datapath.
DELAY_DATA_CE	Input	Delay data clock enable: Enable a delay change event for the datapath. In case of Virtex-6 family devices, this pin is provided for each of the IODELAYE1 components.
DELAY_DATA_INC	Input	Delay data increment: Controls whether the delay is incremented (when asserted) or decremented (when deasserted) when the delay clock is enabled. In case of Virtex-6 family devices, this pin is provided for each of the IODELAYE1 components.
DELAY_RESET	Input	IODELAYE1 reset signal: Controls the loading of initial delay value
DELAY_TAP_IN [4:0]	Input	IODELAYE1 tap in signal: Counter value from FPGA logic for dynamically loadable tap value (CNTVALUEIN). This is provided for each of the IODELAYE1 components.
DELAY_TAP_OUT[4:0]	Output	IODELAYE1 tap out signal: Counter value going to FPGA logic for monitoring tap value (CNTVALUEOUT). This is provided for each of the IODELAYE1 components.

Notes:

1. Only a single-ended or differential input clock is required. For a BUFG or BUFIO2, this comes from a pin. For a BUFPLL, this comes from fabric.

Chapter 4

Generating the Core for Spartan-6 FPGAs

This chapter describes the GUI and follows the same flow required to set up the I/O circuit, using a Spartan-6 FPGA as the target device. Tool tips are available in the GUI for most features. To access them, place your mouse over the relevant text.

For information about configuration options for 7 series and Virtex-6 devices, see Chapter 5, Generating the Core - Zynq-7000, 7 Series and Virtex-6 FPGAs.

Data Bus Setup

Page 1 of the GUI (Figure 4-1) allows you to set up some general features for the data bus.

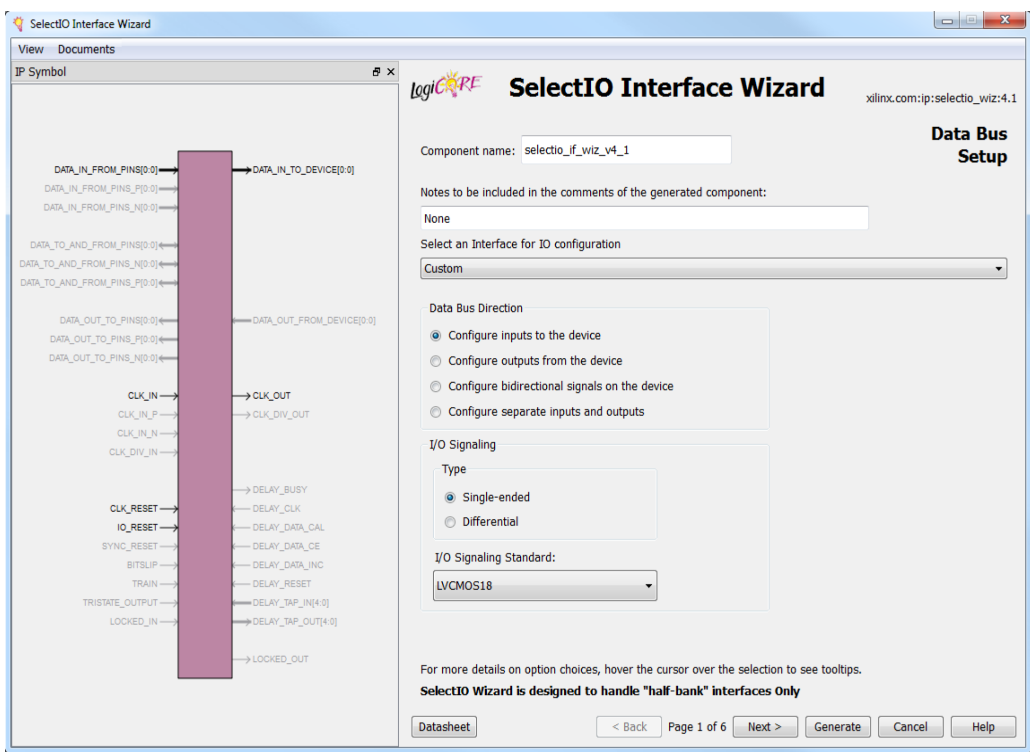

Figure 4-1: **Main Screen- Data Bus Setup - Page 1**

Component Name

The component name is user selectable. Component names must not contain any reserved words in Verilog or VHDL. Blank spaces are not allowed; use an underscore character instead as a separator between multiple words specified as the component name.

Interface for I/O Configuration

SelectIO Wizard has some pre-configured I/O interfaces. Choosing one of these interfaces from the drop-down menu automatically sets the necessary parameters such data bus direction, I/O signalling, and serialization factor.

Currently the wizard supports SGMII, DVI receiver, DVI transmitter, Camera link receiver, Camera link transmitter and Chip-to-Chip interface. SelectIO Interface Wizard would only configure the data pins for all the interfaces mentioned above.

The listed options vary based on the device family selected.

Data Bus Direction

The direction of the bus can be chosen here. Only choose bidirectional if you need your bus to be bidirectional: selecting it will cause restrictions later on in the configuration process. SelectIO Wizard supports Inputs, Outputs, Bidirectional and Separate IO buses.

Separate Inputs and Outputs create independent Inputs and Outputs pins. Other configurable settings such as Serialization factor, data width, delays are common to both Inputs and Outputs i.e. if Separate Inputs and Outputs is chosen and the serialization factor is set to "5" then this serialization factor of "5" would apply to both Input SERDES as well as Output SERDES.

I/O Signaling

Choose whether your bus is single-ended or differential. Single-ended signals with a serialization factor of 4 or less will occupy half of an I/O pair. Single-ended signals with a serialization factor of 5 or more will occupy an entire I/O pair. Differential signals will be created as I/O pairs.

All I/O signaling standards are shown for the I/O signaling type that has been selected. This value will appear in the generated HDL code.

Data Bus Setup 2

Page 2 of the GUI (Figure 4-2) allows you to specify the configuration for the I/O interconnect datapath.

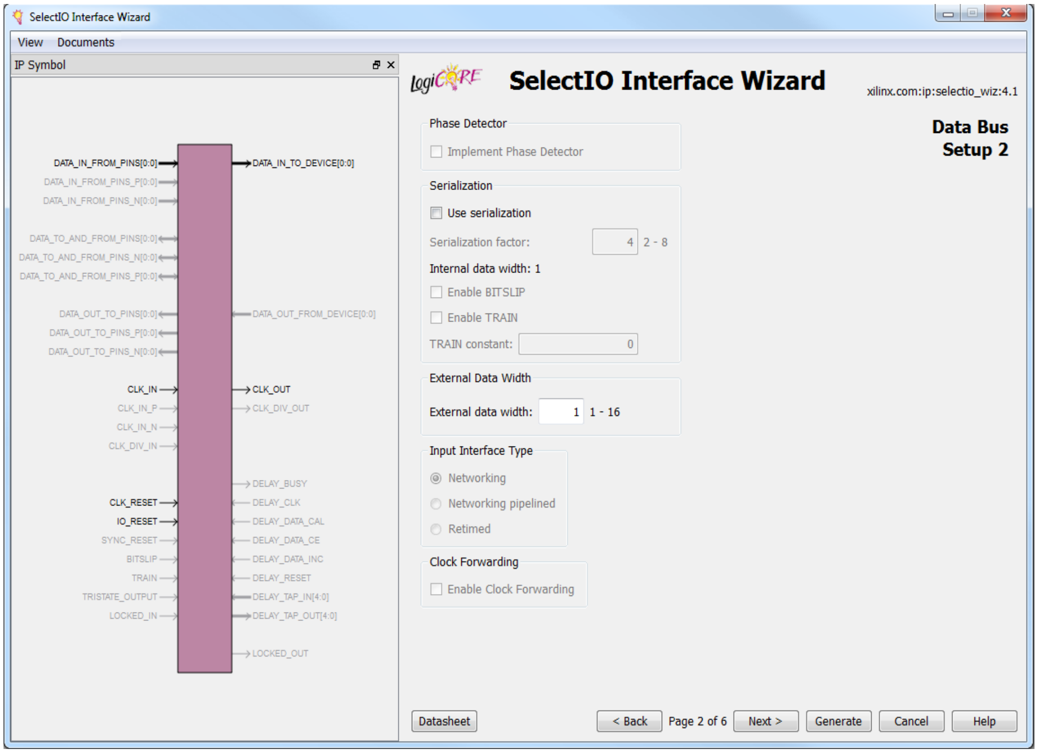

Figure 4-2: **Data Bus Setup - Page 2**

Phase Detector

If Implement Phase Detector is selected, the IODELAY2 and ISERDES2 will be configured for phase detector. The bus on the device side will increase by the serialization factor. Selecting this feature will instantiate a reference logic that would implement the Phase detector functionality. Selecting this option makes the wizard skip the Data Delay and Clock Delay pages. The Phase Detector box is available only when I/Os are configured as Inputs with differential I/O standard.

Serialization

If Use Serialization is selected, an ISERDES2 and/or OSERDES2 will be instantiated for the user. The bus on the device side will increase by the serialization factor. All data is collected by timeslice, then concatenated from right to left. For example, assume that the output data bus is 8-bits wide, with a serialization factor of 4. If the data is presented on the pins as: 00, 01, 02, and 03, the data presented to the device will be 03020100.

If a serialization factor of 5–8 is selected, two SERDES blocks per I/O will be instantiated for a user because each SERDES is capable of a maximum serialization of 4:1. Even if a single-ended bus was chosen, the entire I/O pair is now occupied. Once serialization is selected, BITSLIP and TRAIN can be chosen depending on the presence of an input or output datapath. The TRAIN constant will be configured in the source code.

If the Phase Detector interface is chosen, then both ISERDES2 are instantiated.

External Data Width

You can configure the number of bits on the system side, and this will automatically be set up on the device side. Note that differential signals will occupy two pins for each data bit.

Input Interface Type

If serialization is chosen, the interface type can be configured to set to specify the timing of the data on the device side.

Clock Forwarding

Select this option if you want the SelectIO wizard to generate a clock forwarding logic. This option is only available when bus direction is Output, Bidirectional or Separate Inputs and Outputs.

If any delay is set on the output data path, the same delay is assigned to the forwarded clock so that the data and clock remain in sync.

Data Delay

Page 3 of the GUI (Figure 4-3) allows you to specify the type of delay for the data bus.

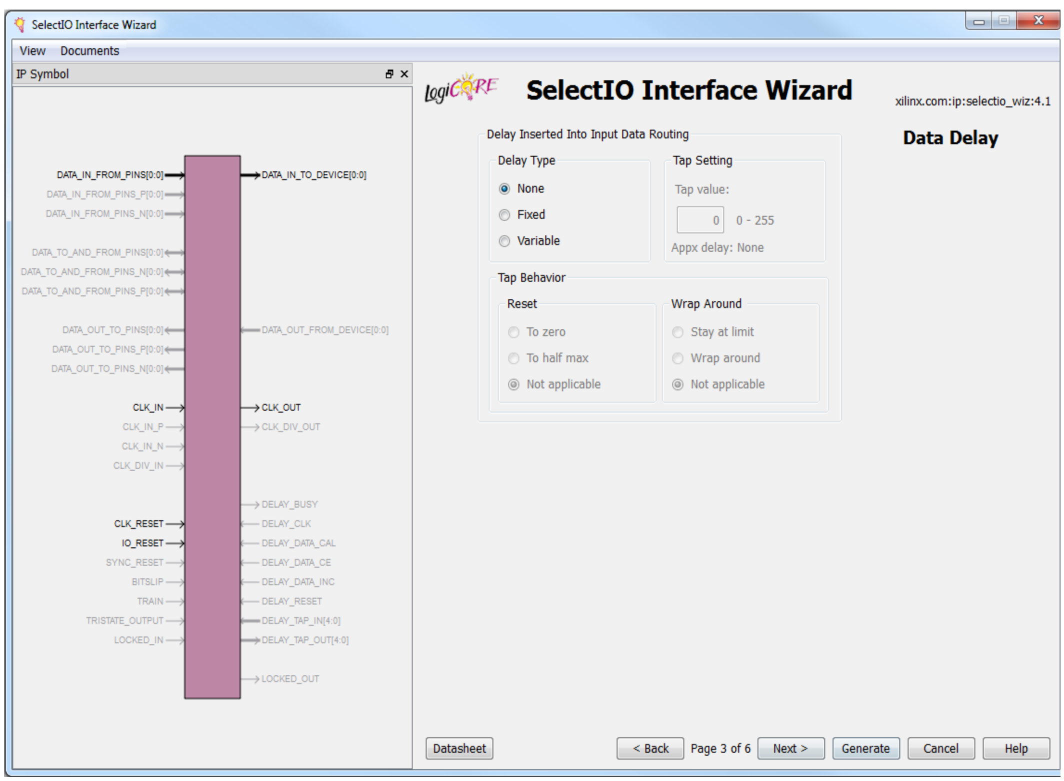

Figure 4-3: **Data Delay - Page 3**

Delay Type

An IODELAY2 will be instantiated if Fixed or Variable delay is chosen. Generally, if data is delayed with Fixed or Variable, delay will also be desired for the clock, given the high amount of insertion delay for the IODELAY2 primitive. The wizard will suggest delay settings for the clock based on the Data Delay settings. The tool will instantiate two IODELAY2 components when phase detector interface is chosen.

Tap Setting

If a delay is chosen, the tap value can be specified. A typical insertion delay for the value specified is show under the Tap value box. For a variable delay, the tap value is for the initial programming. The GUI also shows the approximate delay value for tap setting. The user is expected to refer to the trace report for exact values.

Tap Behavior

If a Variable delay is chosen, the user can specify the behavior during a reset/calibration sequence, and the behavior in the event the user attempts to go past the final value in the counter.

Clock Setup

Page 4 of the GUI (Figure 4-4) allows you to configure the behavior of the clock.

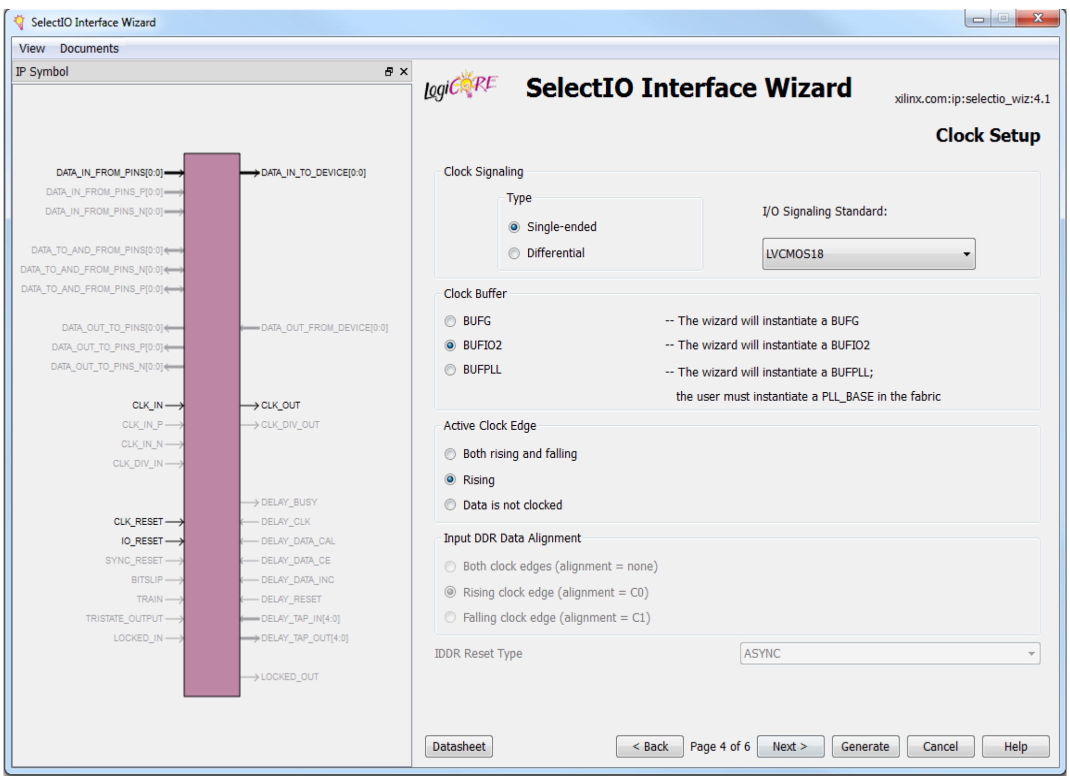

Figure 4-4: **Clock Setup - Page 4**

Clock Signaling

You can specify the signaling type and standard for the input clock. The I/O signaling standard will be embedded in the provided HDL source. Using double-data rate (DDR) data places some restrictions on clocks.

Clock Buffer

If your clock comes from a pin, you should leave the input buffer as a BUFIO2 for the most flexible functionality. In the event your clock comes from fabric, you will want to choose a BUFPLL, but you will need to be sure to instantiate a PLL_BASE in fabric to drive the BUFPLL. See the LogiCORE IP Clocking Wizard core and the *LogiCORE IP Clocking Wizard Getting Started Guide* for assistance with PLL_BASE instantiation and configuration.

Active Clock Edge

If using DDR data, select Both Rising And Falling. If the data requires an asynchronous delay only, select Data Is Not Clocked. In all other circumstances, leave it at the default value of Rising. If a topology is not available, you will not be able to select it. See the *Spartan-6 FPGA SelectIO Resources User Guide* and the *Spartan-6 FPGA Clocking Resources User Guide* for more information on clocking requirements.

DDR Data Alignment

If serialization is not chosen, but DDR data is chosen, the ODDR2 and IDDR2 primitives can be configured to align data to the rising, falling, or both edges of the input clock. Note that the internal data width will double, and that data will be grouped by timeslice just as is it for serialization.

IDDR Reset

Input, Separate Bus Designs: The IDDR Reset type is a new parameter with two IDDR Reset type options: DDR Data and Serialization. IF DDR data is selected and the DDR data alignment is set to “both clock edges” you will have the option to select the reset type for the IDDR primitive. By default, the value is ASYNC. However, if you select SYNC as the reset type, the reset to the clock driving the IDDR primitive must be provided by the SYNC_RESET port.

Clock Delay

Page 5 of the GUI (Figure 4-5) allows you to specify the type of delay for the clock bus. Please see Data Delay for more information on the meanings of the fields within the clock delay configuration page. When using the Phase detector feature, the clock delay is set to FIXED with a tap value of 0.

- If there is no delay in the datapath, there generally should not be any delay in the clock path, choose None.
- If there is delay in the datapath, you'll generally want to match the insertion delay in the clock path, choose Fixed with a tap value of 0.

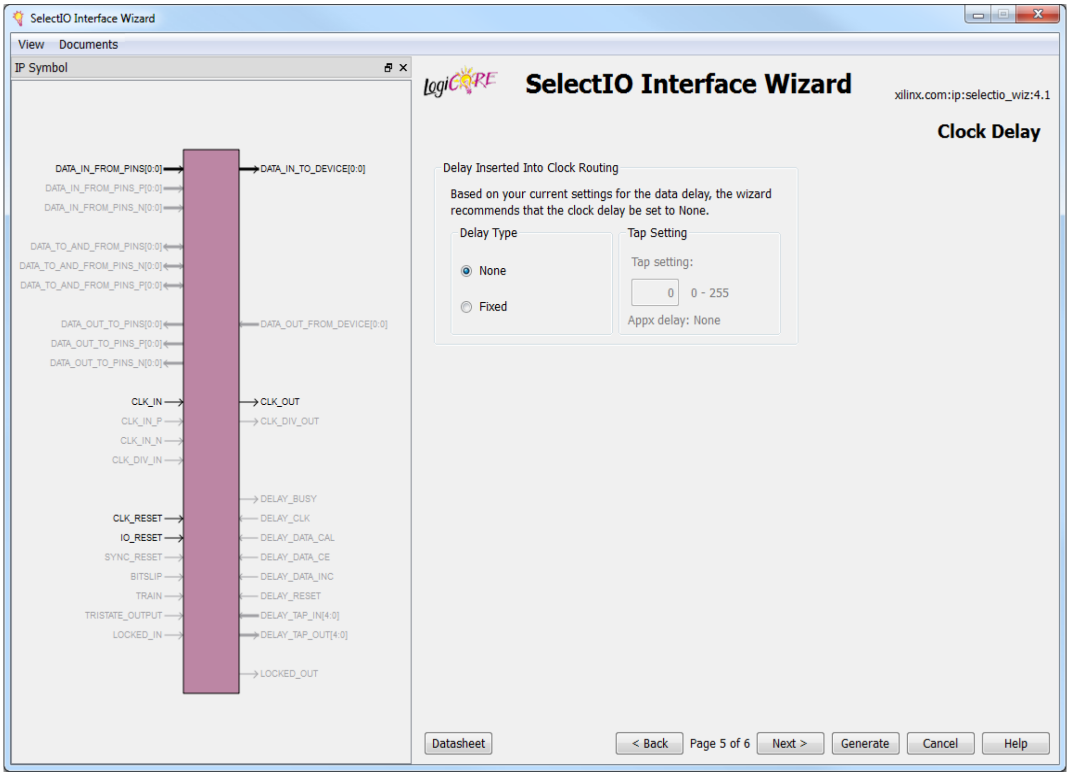

Figure 4-5: **Clock Delay - Page 5**

Summary Page

The summary page lists all the key parameters selected, such as the number of data I/Os, bus direction, serialization factor, buffers used, and the bus I/O standard.

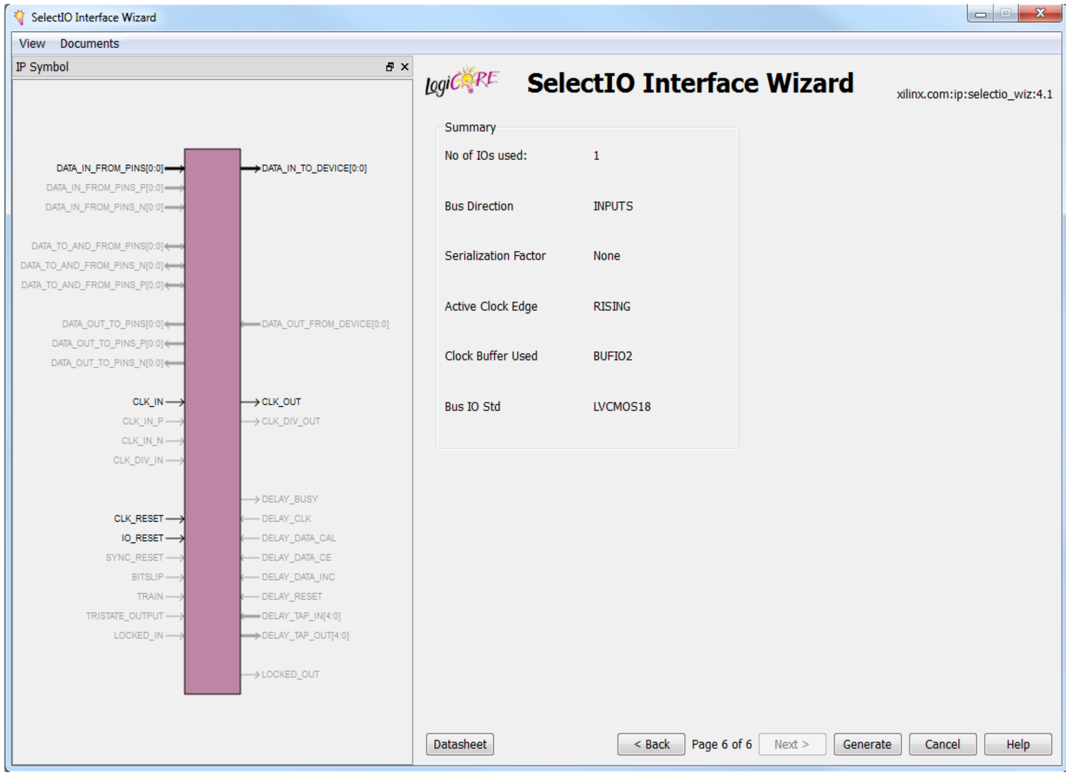

Figure 4-6: **Summary - Page 6**

Generating the Core

After the desired configuration parameters have been selected, you can generate the SelectIO Wizard Interface core. To do so, click the "Generate" button option that is located at the bottom of the Summary page.

![Xilinx logo]

Chapter 5

Generating the Core for Zynq-7000, 7 Series, and Virtex-6 Devices

This chapter describes the GUI and follows the same flow required to set up the I/O circuit for Zynq™-7000, 7 series and Virtex®-6 devices. Tool tips are available in the GUI for most features; simply place your mouse over the relevant text, and additional information is provided in a pop-up dialog.

Data Bus Setup

Page 1 of the GUI (Figure 5-1) allows you to set up some general features for the data bus.

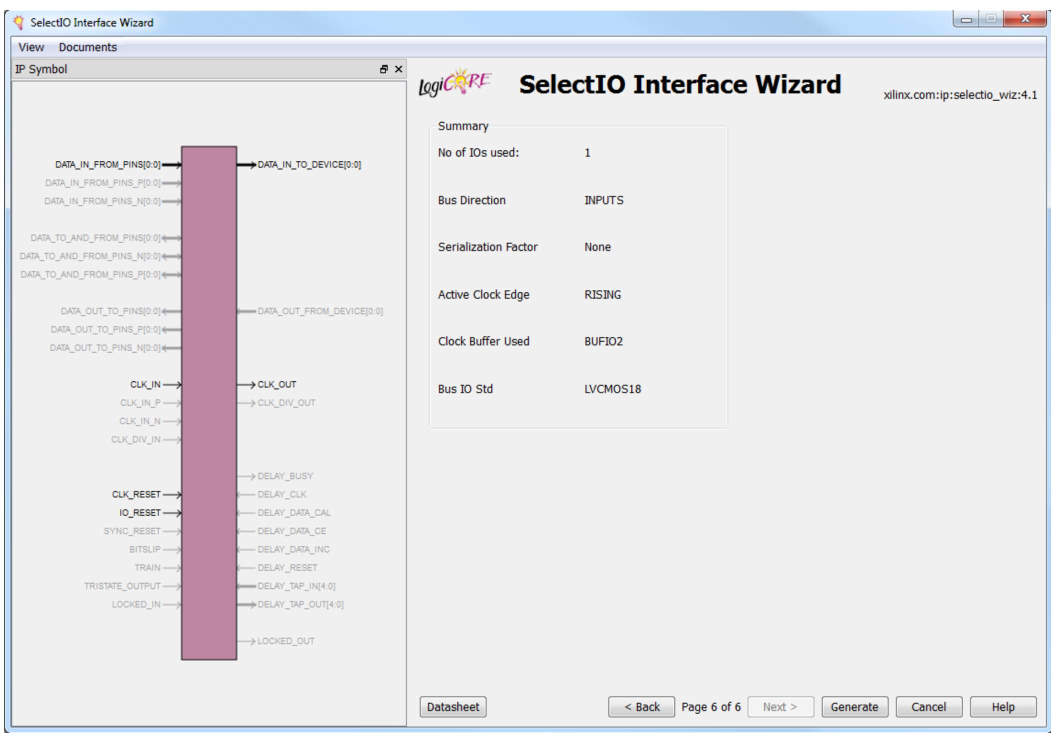

Figure 5-1: **Main Screen- Data Bus Setup - Page 1**

Component Name

User selectable component name and notes are available. Component names must not contain any reserved words in Verilog or VHDL. Notes must not include any spaces; instead use an underscore.

Interface for IO Configuration

SelectIO Wizard has some pre-configured IO interfaces. Choosing one of these interfaces from the drop-down menu will automatically set the necessary parameters such data bus direction, I/O signalling, serialization factor etc.

Currently the wizard supports SGMII, DVI receiver, DVI transmitter, Camera link receiver, Camera link transmitter and Chip-to-Chip interface. SelectIO Interface Wizard would only configure the data pins for all the interfaces mentioned above.

The listed options vary based on the device family selected.

Data Bus Direction

The direction of the bus can be chosen here. Only choose bidirectional if you need your bus to be bidirectional: selecting it will cause restrictions later on in the configuration process.

SelectIO Wizard supports Inputs, Outputs, Bidirectional and Separate IO buses.

Separate Inputs and Outputs create independent Inputs and Outputs pins. Other configurable settings such as Serialization factor, data width, and delays are common to both Inputs and Outputs. For example, if Separate Inputs and Outputs is chosen and the serialization factor is set to "5", then this serialization factor of "5" would apply to both Input SERDES as well as Output SERDES.

I/O Signaling

Choose whether your bus is single-ended or differential. Single-ended signals with a serialization factor of 6 or less will occupy half of an I/O pair. Single-ended signals with a serialization factor of 7 or more will occupy an entire I/O pair. Differential signals will be created as I/O pairs.

All I/O signaling standards are shown for the I/O signaling type that has been selected. This value will appear in the generated HDL code.

Data Bus Setup 2

Page 2 of the GUI (Figure 5-2) allows you to specify the configuration for the I/O interconnect datapath.

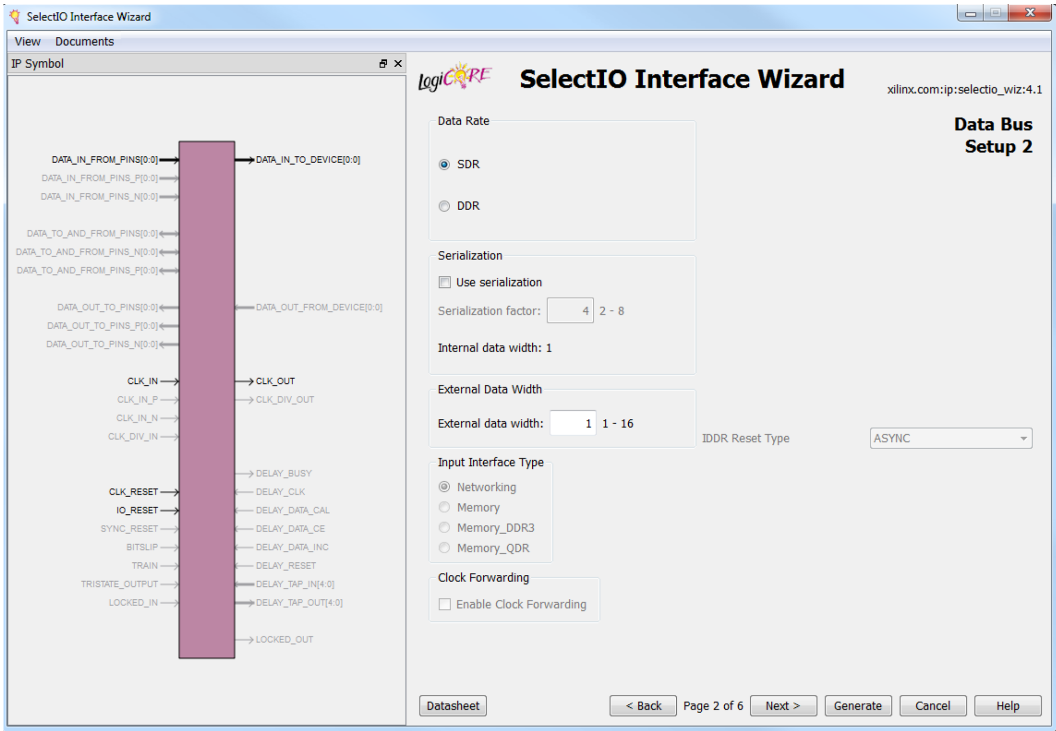

Figure 5-2: **Data Bus Setup - Page 2**

Data Rate

The user selects “SDR” if the data is clocked on rising edge. If the incoming or outgoing data is clocked on both the edges, then the user should select “DDR”.

The selection of Data Rate affects the serialization factor limits. These are displayed dynamically on the page.

Serialization

If Use Serialization is selected, an ISERDESE1/ISERDESE2 and/or OSERDESE1/OSERDESE2 will be instantiated for the user depending on the device selected. The bus on the device side will increase by the serialization factor. All data is collected by timeslice, then concatenated from right to left. For example, assume that the output data bus is 8-bits wide, with a serialization factor of 4. If the data is presented on the pins as: 00, 01, 02, and 03, the data presented to the device will be 03020100.

For Zynq-7000 and 7 Series Devices

If a serialization factor of 10 or 14 is selected, two SERDES blocks per I/O will be instantiated for the user because each SERDES is capable of a maximum serialization of 8:1. Even if a single-ended bus was chosen, the entire I/O pair is now occupied. When the Data Rate is “SDR”, the possible values for the serialization factor are 2-8. When Data Rate is “DDR”, the serialization factor can be set to 4, 6, 8, 10, or 14.

For Virtex-6 Devices

If a serialization factor of 7-10 is selected, two SERDES blocks per I/O will be instantiated for a user because each SERDES is capable of a maximum serialization of 6:1. Even if a single-ended bus was chosen, the entire I/O pair is now occupied. When the Data Rate is “SDR”, the possible values for the serialization factor are 2-8. When Data Rate is “DDR”, the serialization factor can be set to 4, 6, 8, or 10.

IDDR Reset

Input, Bidirectional, and Separate Bus Designs: The IDDR Reset type is a new parameter with two IDDR reset type options: DDR Data and Serialization. When DDR is selected, and serialization is not, you can select the type of reset for the IDDR primitive. The default value is ASYNC. However, if you select SYNC for the reset, it should be synchronized with the clock driving the IDDR primitive.

External Data Width

You can configure the number of bits on the system side, and this will automatically be set up on the device side. Note that differential signals will occupy two pins for each data bit.

Input Interface Type

If serialization is chosen, the interface type can be configured to set to specify the timing of the data on the device side. The SelectIO Interface Wizard only supports NETWORKING type of Input Interface. For other interfaces such as MEMORY, MEMORY_QDR and MEMORY_DDR3 user should refer to the MIG tool. Bitslip functionality is always enabled for NETWORKING mode. User should tie this pin to logic 0 if not required.

Clock Forwarding

Select this option if you want the SelectIO wizard to generate a clock forwarding logic. This option is only available when bus direction is Output, Bidirectional or Separate Inputs and Outputs.

If any delay is set on the output data path, the same delay is assigned to the forwarded clock so that the data and clock remain in sync.

Data Delay

Page 3 of the GUI (Figure 5-3) allows you to specify the type of delay for the data bus.

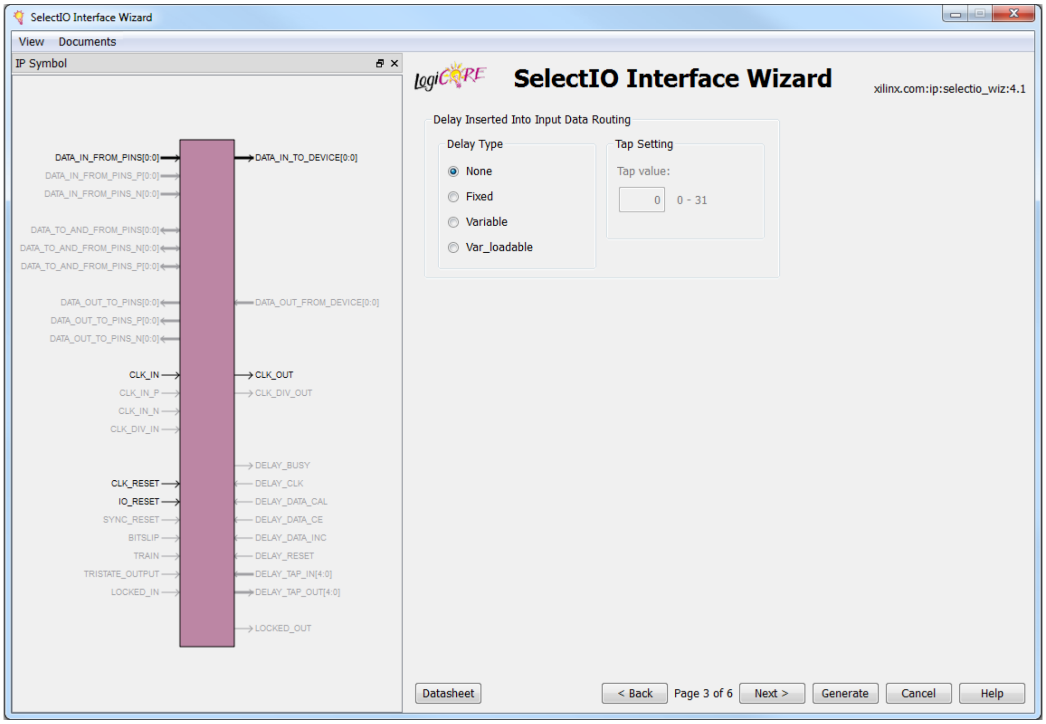

Figure 5-3: **Data Delay - Page 3**

Delay Type

An IODELAYE1 will be instantiated if Default, Fixed, Variable or Var_loadable delay is chosen. Generally, if data is delayed with Fixed or Variable, delay will also be desired for the clock, given the high amount of insertion delay for the IODELAYE1/IODELAYE2 primitive.For a bidirectional bus, only specific combinations of Input and Output delay are possible.

Selecting Variable or Var_loadable option will enable the user to control each IODELAYE1/IODELAYE2 element individually. This means that the control signals of each IODELAYE1/IODELAYE2 element (for example, CE, INC, CNTVALUEIN, CNTVALUEOUT) would be accessible to the user. If the user wishes to control all IODELAYE1s/IODELAYE2s in same way and at same time, then the signals such as CE, INC and CNTVALUEIN can be driven together from a common logic.

Tap Setting

If delay type chosen is FIXED, VARIABLE, the tap value can be specified. The allowed value for tap is 0–31.

Clock Setup

Page 4 of the GUI (Figure 5-4) allows you to configure the behavior of the clock.

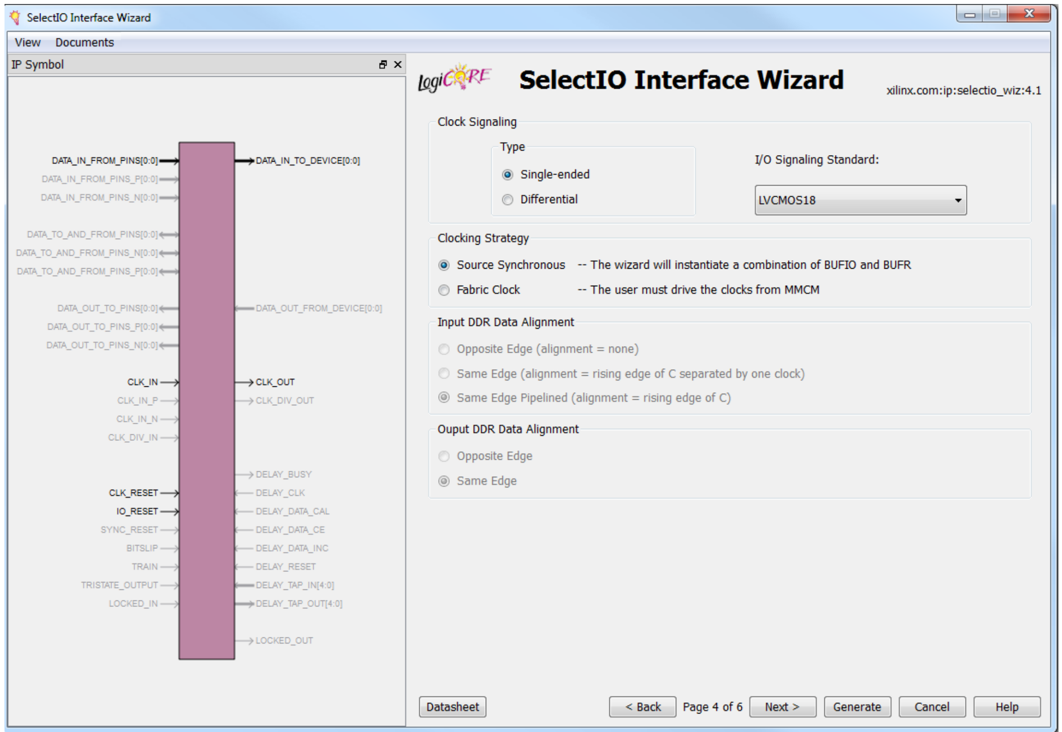

Figure 5-4: **Clock Setup - Page 4**

Clock Signaling

You can specify the signaling type and standard for the input clock. The I/O signaling standard will be embedded in the provided HDL source. For any design that is being configured, the clock I/O signalling standard will be same as that of bus I/O standard.

Clocking Strategy

If your clock comes from a pin, you should leave the input buffer as "Source Synchronous" for the most flexible functionality. Selecting this option will instantiate the necessary circuitry of BUFIO and BUFR and configure the same.

In the event your clock comes from fabric, you will want to choose "Fabric Clock", but you will need to be sure to instantiate a MMCM in the fabric to drive the clocks. Selecting the "Fabric Clock" option will override the "Clock Signaling" section. See the LogiCORE IP Clocking Wizard core and the *LogiCORE IP Clocking Wizard Getting Started Guide* for assistance with MMCM instantiation and configuration.

Input and Output DDR Data Alignment

If serialization is not chosen, but DDR data is chosen, the ODDR and IDDR primitives can be configured to align data to the rising, falling, or both edges of the input clock. Note that the internal data width will double, and that data rate will be grouped by timeslice just as is it for serialization.

The Input DDR Data Alignment option is available when bus direction is input or bidirectional. The Output DDR Data Alignment option is available when bus direction is output or bidirectional.

Clock Delay

Page 5 of the GUI (Figure 5-5, page 29) allows you to specify the type of delay for the clock bus. See Data Delay for the definitions of the clock delay configuration fields.

- If there is no delay in the datapath, there generally should not be any delay in the clock path, choose None.
- If there is delay in the datapath, you'll generally want to match the insertion delay in the clock path, choose Fixed with a tap value of 0.

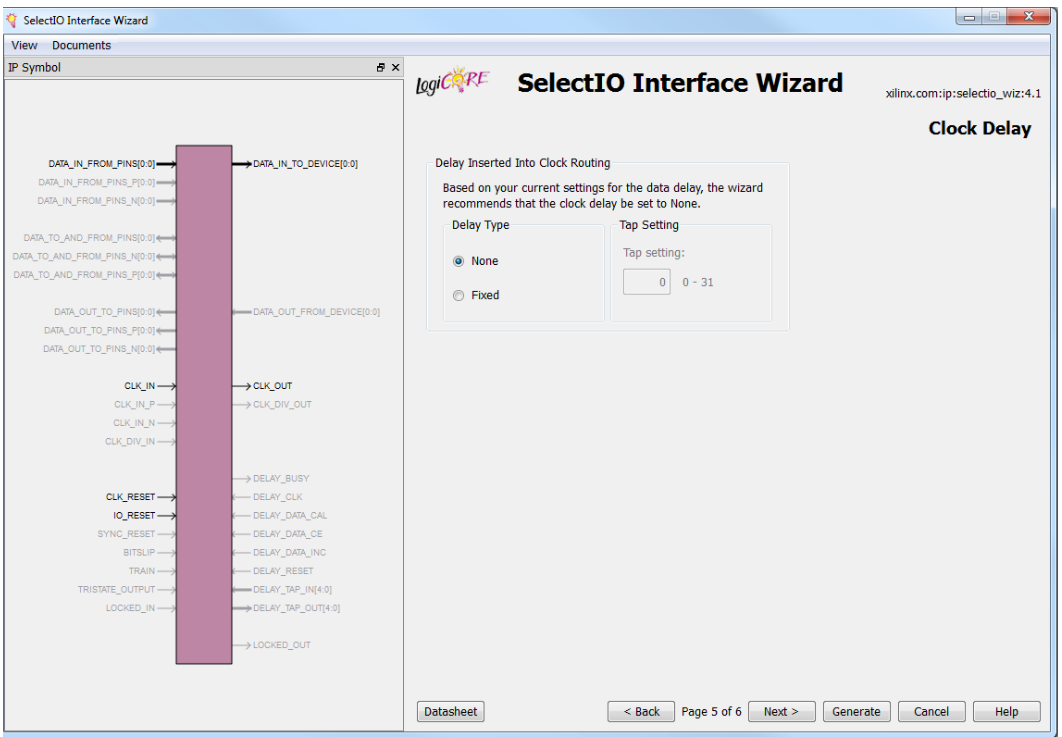

Figure 5-5: **Clock Delay - Page 5**

Summary Page

The summary page lists all the key parameters chosen by the user, such as the number of data I/Os, bus direction, serialization factor, buffers used, and the bus I/O standard.

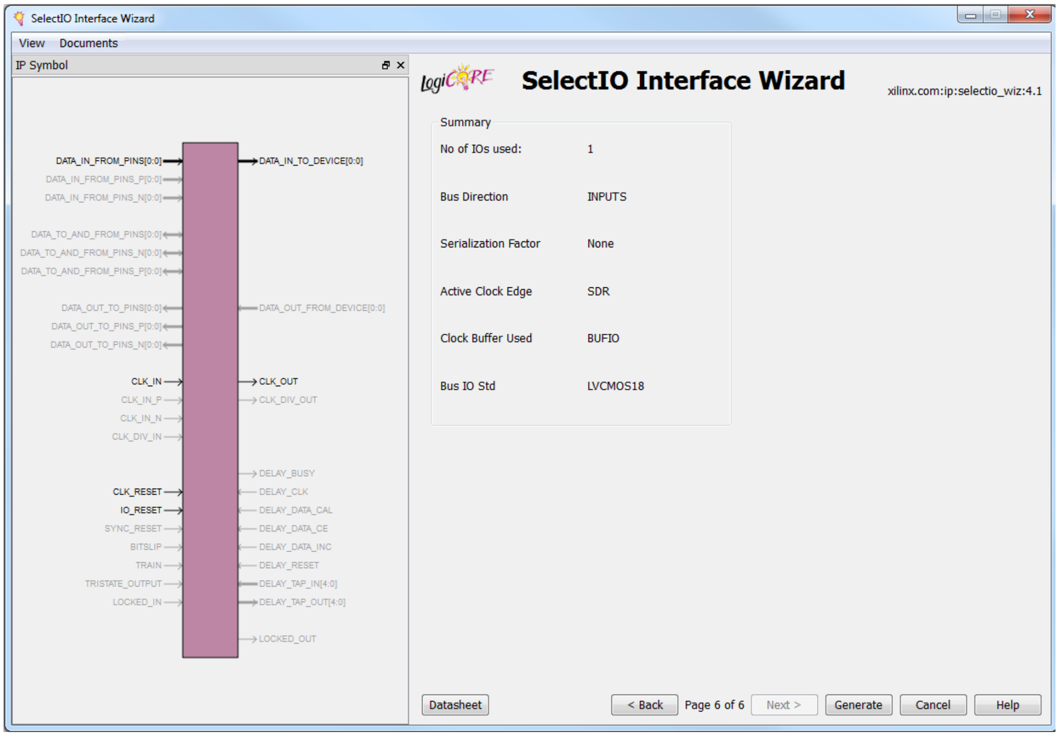

Figure 5-6: **Summary - Page 6**

XILINX

Chapter 6

Detailed Example Design

This chapter provides detailed information about the example design, including a description of files and the directory structure generated by the Xilinx CORE Generator™ tool, the purpose and contents of the provided scripts, the contents of the example HDL wrappers, and the operation of the demonstration test bench.

Directory and File Structure

- **<project directory>**
  Top-level project directory; name is user-defined
  - <project directory>/<component name>
    Core release notes file
    - <component name>/doc
      Product documentation
    - <component name>/example design
      Verilog and VHDL design files
    - <component name>/implement
      Implementation script files
      - implement/results
        Results directory, created after implementation scripts are run, and contains implement script results
    - <component name>/simulation
      Simulation scripts
      - simulation/functional
        Functional simulation files
      - simulation/timing
        Timing simulation files

Directory and File Contents

The SelectIO Interface Wizard core directories and their associated files are defined below.

<project directory>

The <project directory> contains all the CORE Generator tool project files.

Table 6-1: **Project Directory**

Name	Description
<project_dir>	
<component_name>.v[hd]	Verilog or VHDL source code.
phase_detector.v[hd]	Verilog or VHDL source code of phase detector implementation logic.
<component_name>.xco	CORE Generator tool project-specific option file; can be used as an input to the CORE Generator tool.
<component_name>_flist.txt	List of files delivered with the core.
<component_name>.{veo|vho}	VHDL or Verilog instantiation template.
<component_name>.ise	Files used to incorporate the core into an ISE® software project.
<component_name>.ucf	Constraint file for core.

Back to Top

<project directory>/<component name>

The <component name> directory contains the readme file provided with the core, which may include last-minute changes and updates.

Table 6-2: **Component Name Directory**

Name	Description
<project_dir>/<component_name>	
selectio_wiz_v4_1_readme.txt	SelectIO Interface Wizard readme file.

Back to Top

<component name>/example design

The example design directory contains the example design files provided with the core.

Table 6-3: **Example Design Directory**

Name	Description
<project_dir>/<component_name>/example_design	
<component_name>_exdes.v[hd]	Implementable Verilog or VHDL example design.

Table 6-3: **Example Design Directory *(Cont'd)***

Name	Description
<component_name>_exdes.ucf	Constraint file for example design.

Back to Top

<component name>/doc

The doc directory contains the PDF documentation provided with the core.

Table 6-4: **Doc Directory**

Name	Description
<project_dir>/<component_name>/doc	
`selectio_wiz_ds746.pdf`	*LogiCORE IP SelectIO Interface Wizard Data Sheet*
`selectio_wiz_gsg700.pdf`	*LogiCORE IP SelectIO Interface Wizard Getting Started Guide*

Back to Top

<component name>/implement

The implement directory contains the core implementation script files for ISE as well as PlanAhead.

Table 6-5: **Implement Directory**

Name	Description
<project_dir>/<component_name>/implement	
Scripts and projects to implement the example design	

Back to Top

implement/results

The results directory is created by the implement script, after which the implement script results are placed in the results directory.

Table 6-6: **Results Directory**

Name	Description
<project_dir>/<component_name>/implement/results	
Implement script result files.	

Back to Top

<component name>/simulation

The simulation directory contains the simulation test bench for the example design.

Table 6-7: **Simulation Directory**

Name	Description
<project_dir>/<component_name>/simulation	
`<component_name>_tb.v[hd]`	Demonstration test bench.

Back to Top

simulation/functional

The functional directory contains functional simulation scripts provided with the core.

Table 6-8: **Functional Directory**

Name	Description
<project_dir>/<component_name>/simulation/functional	
Contains simulation scripts and waveform formats.	

Back to Top

simulation/timing

The timing directory contains the timing simulation scripts provided with the core.

Table 6-9: **Timing Directory**

Name	Description
<project_dir>/<component_name>/simulation/timing	
Contains the timing scripts, waveform format and the test bench.	

Back to Top

Implementation Scripts

The implementation script is either a shell script or batch file that processes the example design through the Xilinx tool flow. It is located at:

UNIX

```
<project_dir>/<component_name>/implement/implement.sh
<project_dir>/<component_name>/implement/planAhead_rdn.sh
```

Windows

```
<project_dir>/<component_name>/implement/implement.bat
```

The implement script performs the following steps:

- Synthesizes the HDL example design files using XST
- Runs Ngdbuild to consolidate the core netlist and the example design netlist into the NGD file containing the entire design
- Maps the design to the target technology

- Place-and-routes the design on the target device
- Performs static timing analysis on the routed design using Timing Analyzer (TRCE)
- Generates a bitstream
- Enables Netgen to run on the routed design to generate a VHDL or Verilog netlist (as appropriate for the Design Entry project setting) and timing information in the form of SDF files

The Xilinx tool flow generates several output and report files. These are saved in the following directory which is created by the implement script:

```
<project_dir>/<component_name>/implement/results
```

Simulation Scripts

Functional Simulation and Timing Simulation

The test scripts are a ModelSim, IUS, VCS, or ISIM macro that automate the simulation of the test bench. They are available from the following location:

```
<project_dir>/<component_name>/simulation/functional/
<project_dir>/<component_name>/simulation/timing/
```

The test script performs the following tasks:

- Compiles the structural UniSim/SimPrim simulation model
- Compiles Example Design source code or netlist
- Compiles the demonstration test bench
- Starts a simulation of the test bench
- Runs the simulation to completion

Example Design

Top Level Example Design

The following files describe the top-level example design for the SelectIO Interface Wizard core.

VHDL

```
project_dir>/<component_name>/example_design/<component_name>_exdes.vh
d
```

Verilog

```
project_dir>/<component_name>/example_design/<component_name>_exdes.v
```

The top-level example design implements a loop-back strategy to verify the I/O logic implementation. The example design generates data that is loop-backed to itself through the DUT. The data received is verified, and a status signal is generated accordingly. The example design instantiates a PLL/MMCM to generate various clocks. The entire design is synthesized and implemented in a target device.

Demonstration Test Bench

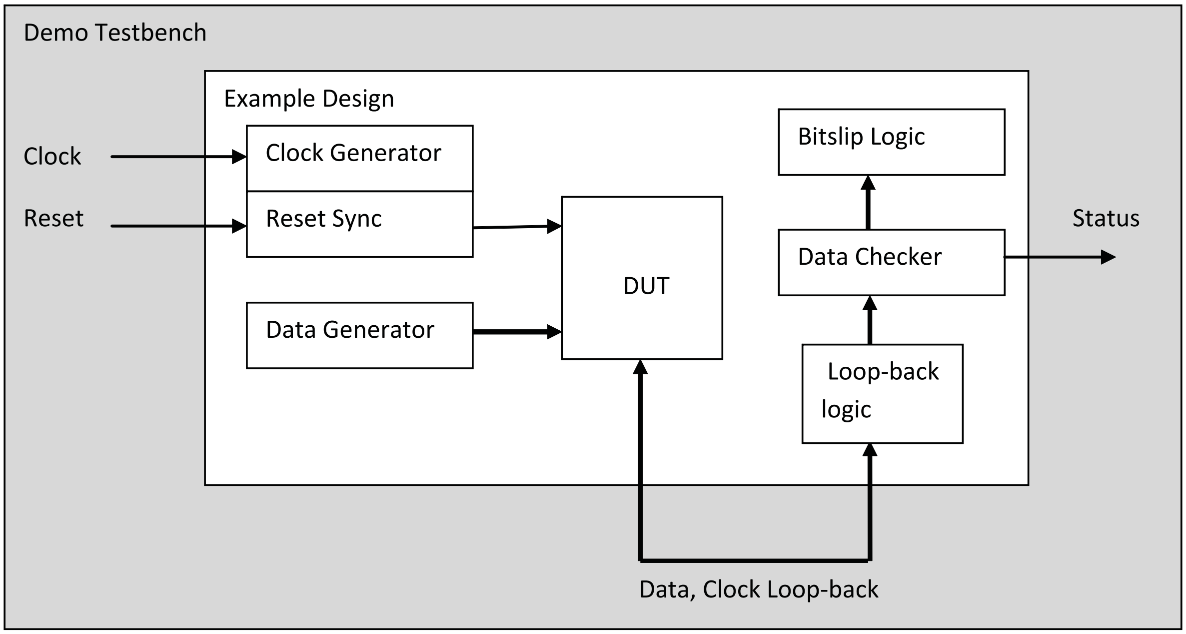

Figure 6-1: **Demonstration Test Bench for the SelectIO Interface Wizard Core and Example Design**

The following files describe the demonstration test bench.

VHDL

```
project_dir>/<component_name>/simulation/<component_name>_tb.vhd
```

Verilog

```
project_dir>/<component_name>/simulation/<component_name>_tb.v
```

The demonstration test bench is a simple VHDL or Verilog program to exercise the example design and the core.

The demonstration test bench performs the following tasks:

- Generates input clock signals.
- Applies a reset to the example design.
- For any type of Bus I/O direction, the example design uses a loop-back architecture. If the design generated is for input direction, then the example design will have an output logic to drive the data and vice-versa. The loop-back connection is done in the test bench.
- The example design has a bitslip logic that generates the required amount of bitslip pulses for ISERDES to get the right data. Once the ISERDESs are locked, the design then starts checking for the output of the ISERDES.