APB to AHB-Lite Bridge

Reference Design

FPGA-RD-02264-1.0

February 2023

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults and associated risk the responsibility entirely of the Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AHB	Advanced High-performance Bus
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
FPGA	Field Programmable Gate Array
RTL	Register Transfer Level

1. Introduction

The APB to AHB-Lite Bridge Reference Design provides an interface between the low power APB and the high-speed AHB-Lite. The design is implemented in Verilog HDL and comes in an IPK format that can be installed with Lattice Propel™ Builder Software as an IP. Implementation is done within Lattice Diamond® software as shown in Table 1.1.

Table 1.1. FPGA Software for IP Configuration, Generation, and Implementation

Supported FPGA Family	IP Configuration and Generation	IP Implementation (Synthesis, Map, Place and Route)
MachXO2™	Lattice Propel Builder version 2.2 and above	Lattice Diamond version 3.12 and above
MachXO3™	Lattice Propel Builder version 2.2 and above	Lattice Diamond version 3.12 and above
MachXO3D™	Lattice Propel Builder version 2.2 and above	Lattice Diamond version 3.12 and above

1.1. Features

The key features of the APB to AHB-Lite Bridge include:

- Compliance with AMBA 3 AHB-Lite Protocol v1.0 and AMBA 3 APB Protocol v1.0
- Data Bus width of up to 32 bits [8, 16, 32]
- Address width of up to 32 bits [11,12,...,32]
- Registered output

1.2. Limitations

The following are the limitations of APB to AHB-Lite Bridge:

- This reference design has not undergone UVM verification.
- Specifically designed to access the System Memory Module IP (FPGA-IPUG-02073) from Lattice Propel Builder.

2. Functional Description

2.1. Overview

The APB to AHB-Lite Bridge Reference Design is used for interfacing one APB Master and one AHB-Lite Slave. This bridge has two sections: the APB Slave section, and the AHB-Lite Master section. An FPGA fabric-based APB Master is required to use this bridge. When interfacing to multiple AHB-Lite Slaves, this IP should be used together with an AHB-Lite interconnect. The read and write transfers on the APB side are converted into equivalent transfers on the AHB-Lite side. For read and write access, the bridge may add wait states. This is due to the output register in both the AHB-Lite side and the APB side.

2.2. Interface Description

Figure 2.1 shows the interface diagram of the APB to AHB-Lite Bridge Reference Design. This comes in an IPK file and can be installed within Lattice Propel Builder as an IP. The diagram shows all the available ports for the IP core. The APB side acts as a slave requiring a separate APB-Master to control it, while the AHB-L side acts as a Master that can control an AHB-L Slave device.

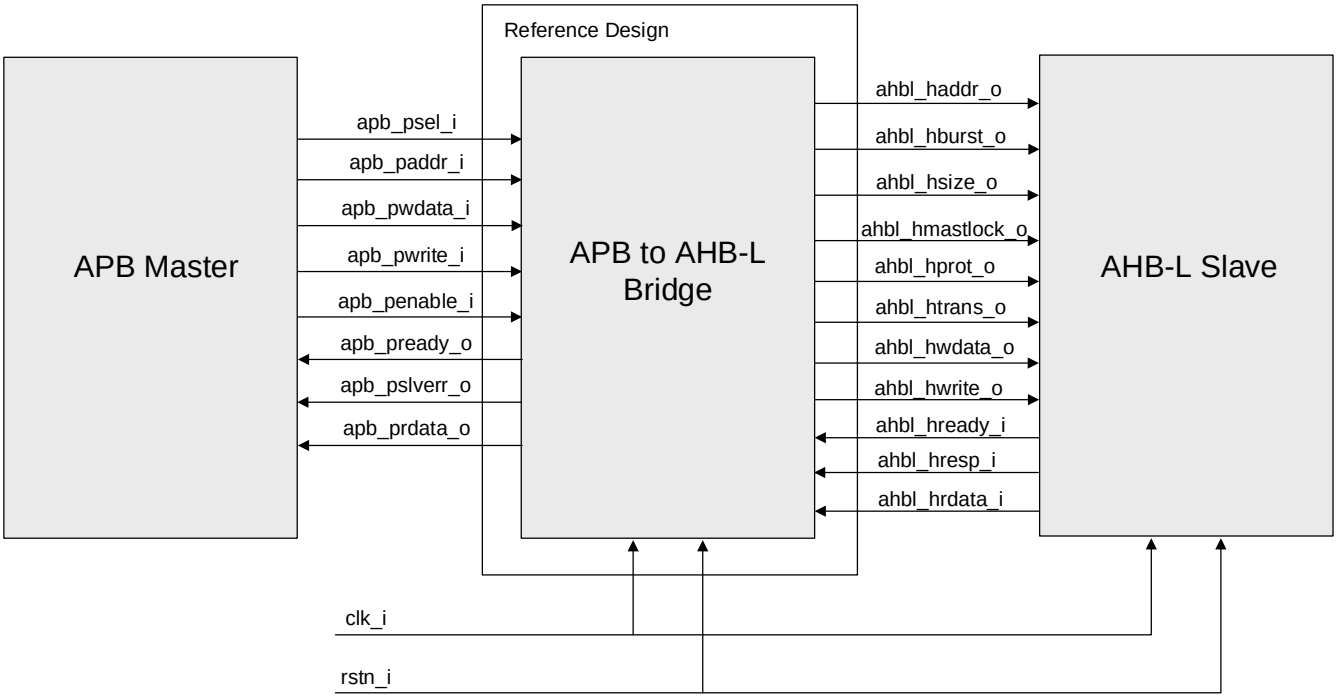

Figure 2.1. Functional Block Diagram

Table 2.1. APB to AHB-Lite Bridge Signal Description

Pin Name	Direction	Width (Bits)	Description
Clock and Reset			
clk_i	In	1	10 MHz system clock or higher.
rst_n_i	In	1	Asynchronous active LOW reset input.
APB Slave Interface (APB_S0)			
apb_psel_i	In	1	Select signal This indicates that the slave device is selected and that a data transfer is required.
apb_paddr_i	In	*ADDR_WIDTH*	Address signal
apb_pwdata_i	In	*DATA_WIDTH*	Write data signal
apb_pwrite_i	In	1	Direction signal Write = 1, Read = 0
apb_penable_i	In	1	Enable signal This indicates that the second and subsequent cycles of an APB transfer.
apb_pready_o	Out	1	Ready signal This indicates transfer completion. Master uses this signal to extend an APB transfer.
apb_pslverr_o	Out	1	Error signal This indicates a transfer failure and propagates from the ahbl_hresp_i signal.
apb_prdata_o	Out	*DATA_WIDTH*	Read data signal
AHB-Lite Master Interface (AHBL_M0)			
ahbl_haddr_o	Out	*ADDR_WIDTH*	Adddress
ahbl_hburst_o	Out	3	Burst type (SINGLE, INCR, INCR4/8/16) This signal is unused; all transactions are treated as SINGLE(0b000).
ahbl_hsize_o	Out	3	Transfer size (8/16/32/64/128/256/512/1024) This signal cannot be modified. The actual transfer size is fixed at 32 bits(0b010).
ahbl_hmastlock_o	Out	1	Current transfer is part of a locked transfer. This signal is unused. The signal is fixed at 0b0.
ahbl_hprot_o	Out	4	Protection control This signal is unused. The signal is fixed at 0b0001.
ahbl_htrans_o	Out	2	Transfer type of the current transfer (IDLE/BUSY/NSEQ/SEQ). This reference design only uses IDLE(0b00) and BUSY(0b10) transfer types.
ahbl_hwdata_o	Out	*DATA_WIDTH*	Write data bus
ahbl_hwrite_o	Out	1	This indicates access direction: Write = 1, Read = 0.
ahbl_hready_i	In	1	This indicates transfer completion.
ahbl_hresp_i	In	1	Slave Response OK(0b0) or ERROR(0b1)
ahbl_hrdata_i	In	*DATA_WIDTH*	Read data bus

2.3. Attributes

Table 2.2 provides the list of user-configurable attributes for the APB to AHB-Lite Bridge. The attribute values are specified using the IP core Configuration user interface in the Propel Builder software as shown in Figure 2.2.

Table 2.2. Attributes Table

Attribute	Selectable Values	Default	Dependency on Other Attributes
Data Width	11–32	32	—
Address Width	8, 16, 32	32	—

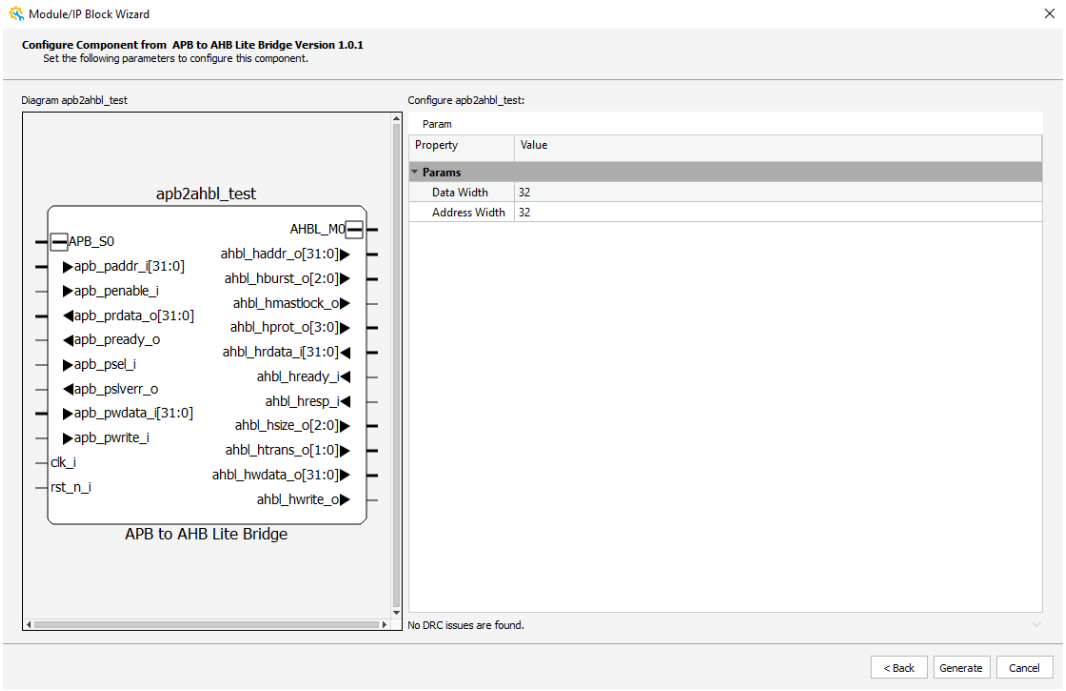

Figure 2.2. APB to AHB-Lite Bridge Configuration User Interface

Table 2.3. Attributes Description

Attribute	Description
Data Width	Specifies the bit with of the apb_pwdata_i, apb_rdata_o, ahbl_hwdata_i and ahbl_hrdata_o data signals.
Address Width	Specifies the bit width of the apb_paddr_i and ahbl_haddr_o address bus signals.

3. Timing Diagram

All signals in this reference design are sampled on the rising edge of clk_i. The following AHBL Master outputs are not shown in the timing diagrams and has fixed values:

- ahbl_hburst_o = 0b000
- ahbl_hsize_o = 0b010
- ahbl_hmastlock_o = 0b0
- ahbl_prot_o = 0b0001

3.1. Write Transactions.

The following steps describes how an APB Master should control this APB Slave reference design when a write transaction is desired.

1. The APB Bus is in idle state. During this period, the APB Master can prepare the data for the *apb_addr_*i and *apb_wdata_i* inputs.
2. The APB Master asserts the *apb_write_i* and *apb_psel_i* signals to HIGH. The APB Slave latches the data presented on the *apb_paddr_i* and *apb_wdata_i* data lines.
3. The APB Master asserts the *apb_penable_i* signal to HIGH. The *ahbl_htrans_o* signal changes to 0b10 for 1 clock cycle and at the same time, *ahbl_hwrite_o* asserts to HIGH. The address and data values from the APB bus now appears on the *ahbl_haddr_o* and *ahbl_hwdata_o* data lines of the AHBL side. Wait states can be inserted by the AHBL slave during this step.
4. The APB Slave asserts the *apb_pready_o* signal to HIGH signifying that a write transaction has completed.
5. The APB bus returns to an idle state. The *apb_psel_i* and *apb_penable_i* signals are deasserted to LOW.
6. Similar to step 2.
7. The APB Master asserts the *apb_penable_i* signal to HIGH. The *ahbl_htrans_o* signal changes to 0b10 for 1 clock cycle and at the same time, *ahbl_hwrite_o* asserts to HIGH. The address and data values now appears on the *ahbl_haddr_o* and *ahbl_hwdata_o* data lines.
8. The AHBL slave can insert wait states during the period by deasserting *ahbl_hready_i* signal to LOW.
9. The AHBL slave asserts *ahbl_hready_i* to HIGH to continue the transaction.
10. The APB Slave asserts the *apb_pready_o* signal to HIGH signifying that a write transaction has completed.
11. The APB bus returns to an idle state. The APB Master can either remain in this state or start a new transaction by repeating the steps.

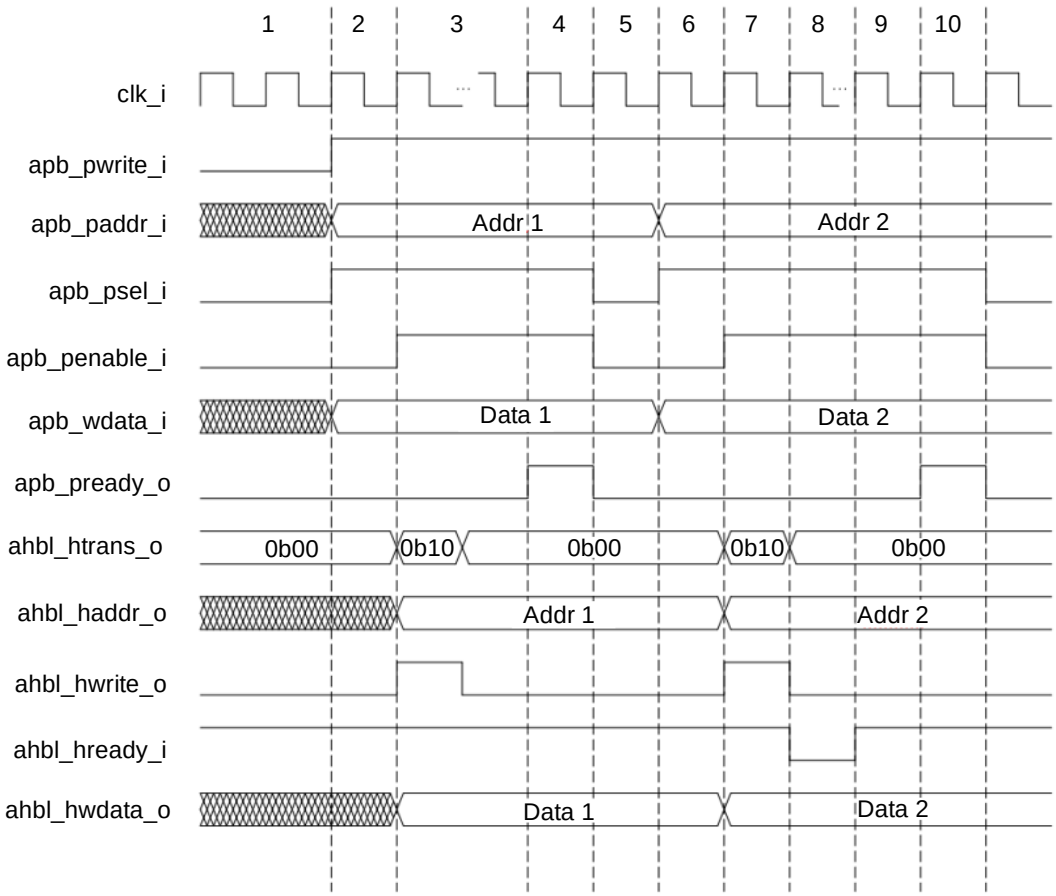

Figure 3.1. APB to AHB-Lite Bridge Write Transactions

3.2. Read Transactions

The following steps describes how an APB Master should control this APB Slave reference design when a write transaction is desired.

1. The APB Bus is in idle state. During this period, the APB Master can prepare the data for the *apb_paddr_i* input. The *apb_pwrite_i* signal remains LOW during read transactions.
2. The APB Master asserts *apb_psel_i* signal to HIGH. The APB Slave latches the data presented on the *apb_paddr_i* data line.
3. The APB Master asserts the *apb_penable_i* signal to HIGH. The *ahbl_htrans_o* signal changes to 0b10 for 1 clock cycle and at the same time, *ahbl_hwrite_o* remains LOW. The address value now appears on the *ahbl_haddr_o* data line. The AHBL slave can start preparing the read data on the *ahbl_hrdata_i* data line after the first clock cycle.
4. The APB Slave asserts the *apb_pready_o* signal to HIGH signifying that a read transaction has completed with the read data from the *ahbl_hrdata_i* data line input appearing on the *apb_prdata_o* data line output.
5. The APB bus returns to an idle state. The *apb_psel_i* and *apb_penable_i* signals are deasserted to LOW.
6. Similar to step 2.
7. The APB Master asserts the *apb_penable_i* signal to HIGH. The *ahbl_htrans_o* signal changes to 0b10 for 1 clock cycle and at the same time, *ahbl_hwrite_o* remains LOW. The address value now appears on the *ahbl_haddr_o* data line of the AHBL side. The AHBL slave can start preparing the read data on the *ahbl_hrdata_i* data line after the first clock cycle.
8. The AHBL slave can insert wait states during the period by deasserting *ahbl_hready_i* signal to LOW.
9. The AHBL slave asserts *ahbl_hready_i* to HIGH to continue the transaction.

10. The APB Slave asserts the *apb_pready_o* signal to HIGH signifying that a write transaction has completed.
11. The APB bus returns to an idle state. The APB Master can either remain in this state or start a new transaction by repeating the steps.

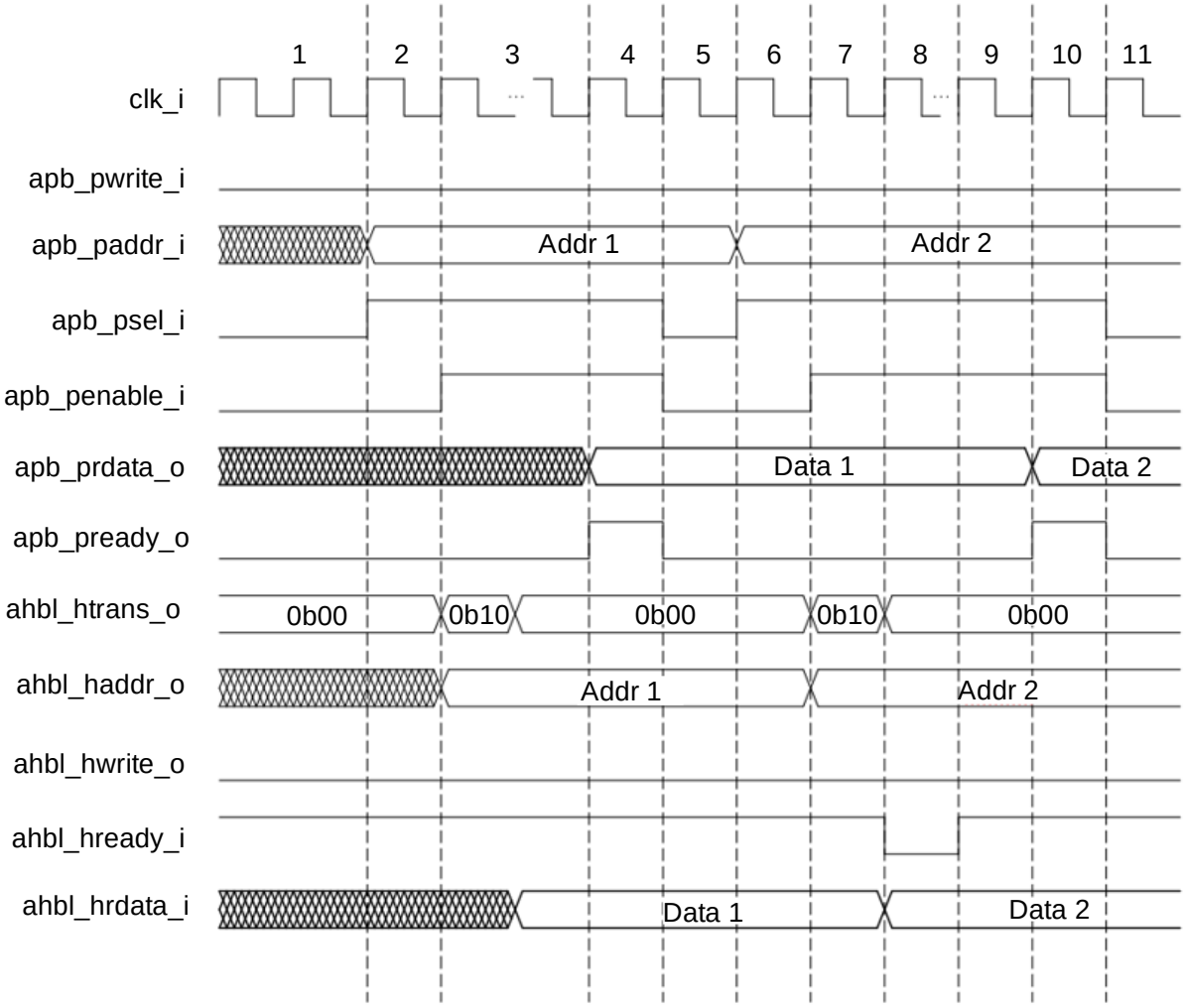

Figure 3.2. APB to AHB-Lite Bridge Read Transactions

4. Packaged Design

The reference design folder (APB_to_AHBL_Bridge) contains two subfolders: IP and Simulation.

- IP – contains the IPK file that can be installed in Lattice Propel Builder.
- Simulation – containt the simulation script file (.DO) and the pregenerated RTL and Testbench files.

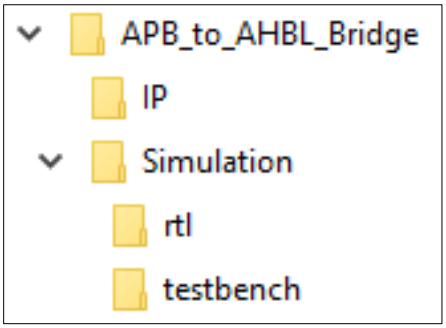

Figure 4.1. Packaged Design Directory Structure

5. IPK Installation

This reference design comes in an IPK format and can be installed within the Lattice Propel Builder by performing the following steps:

1. Go to **File > New Design** and create a new project.

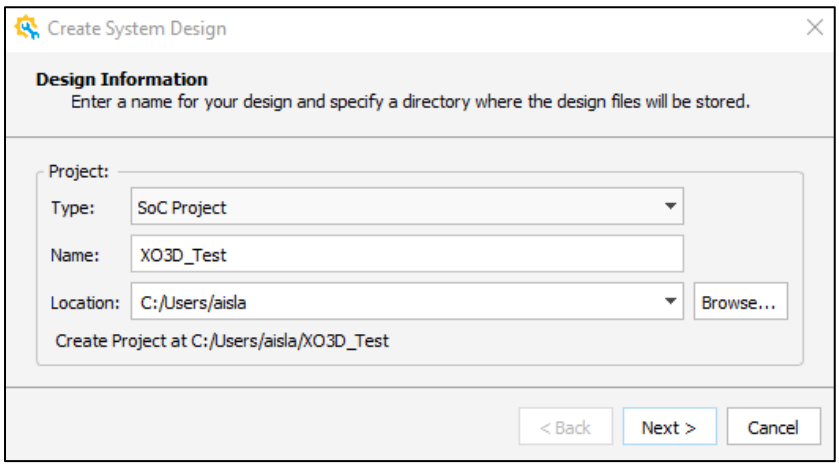

Figure 5.1. New Design Window

2. Select a device family mentioned in Table 1.1.

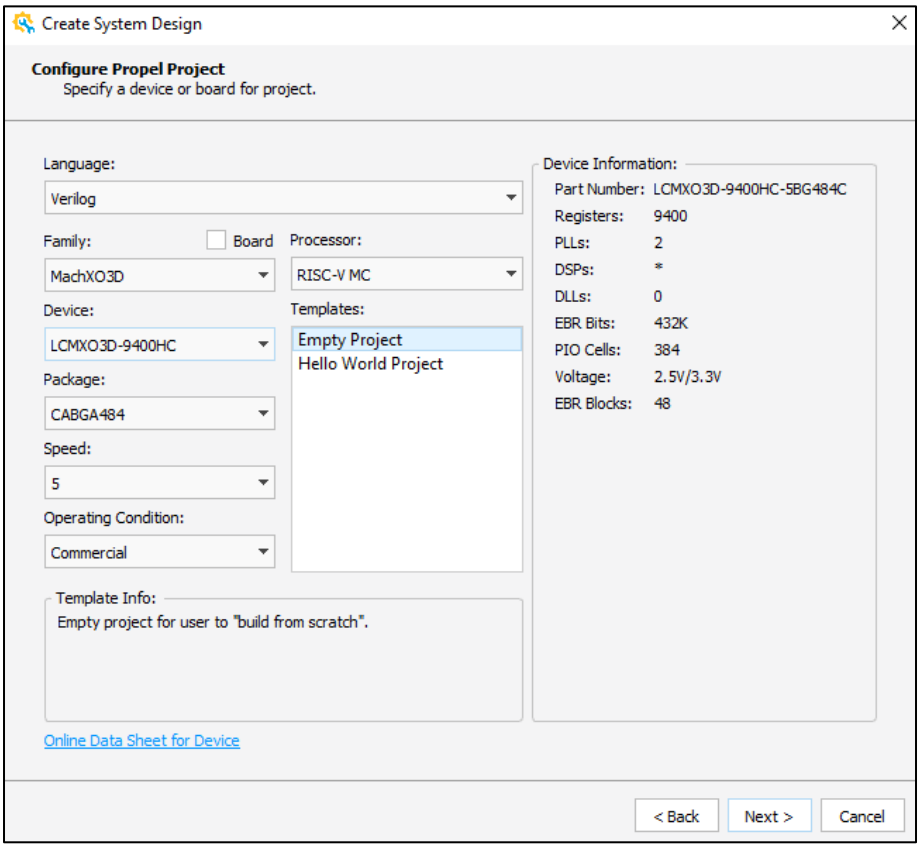

Figure 5.2. Configure Propel Project Window

3. Go to the **IP Catalog** and click the **Install a user IP** icon.

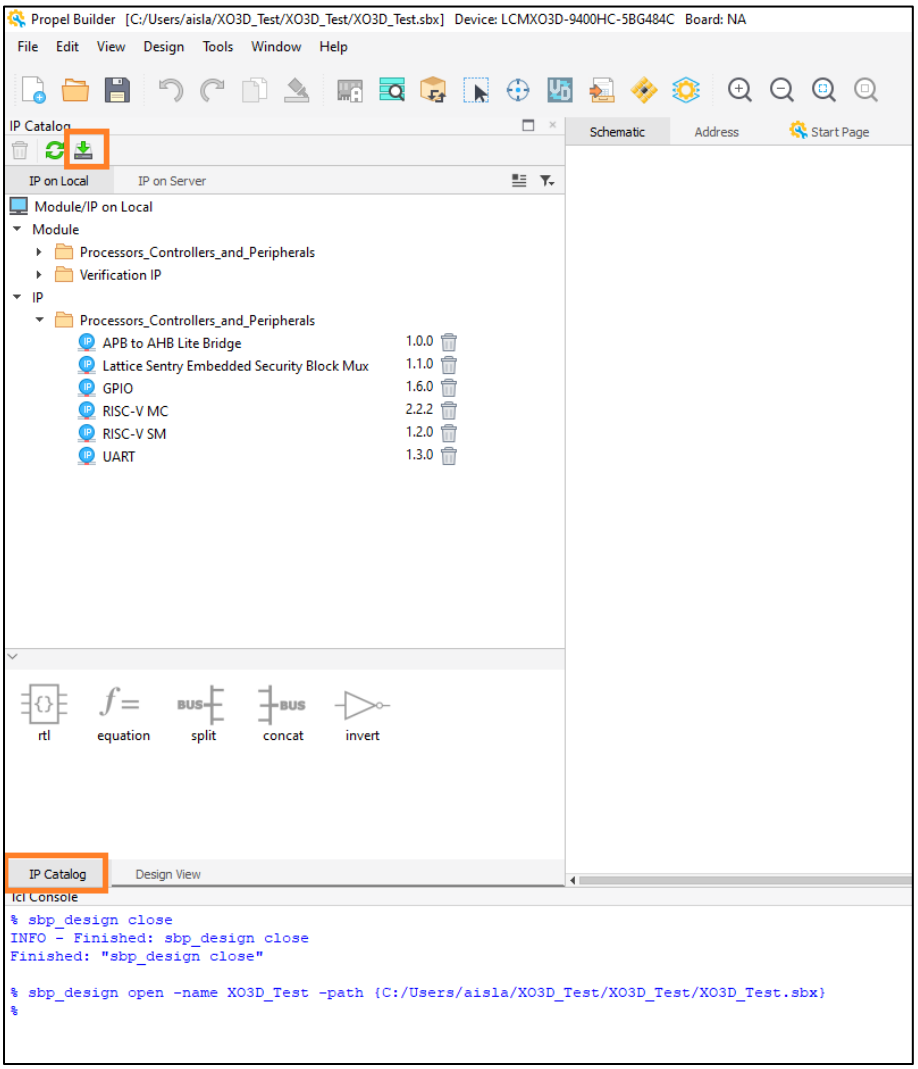

Figure 5.3. IP Catalog Tab

4. Select the IPK file and click **Open**.

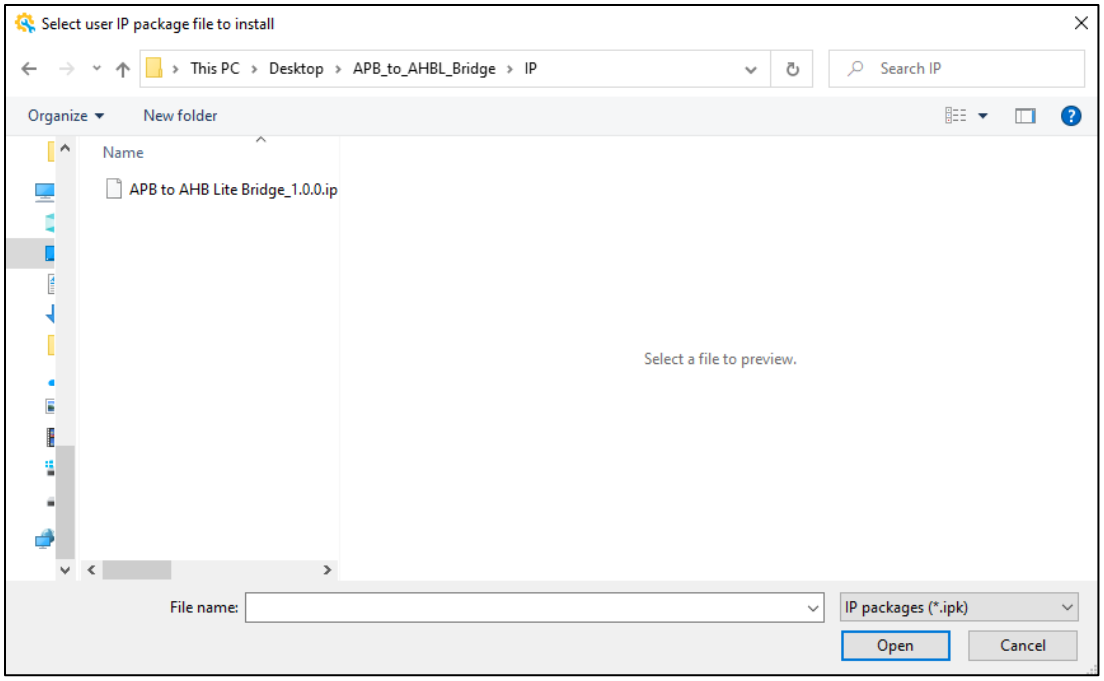

Figure 5.4. Select User IP Window

5. After installation, the IP can be found on the left pane as in Figure 5.5. Double-click and follow the **Module/IP Block Wizard** to add it to the design.

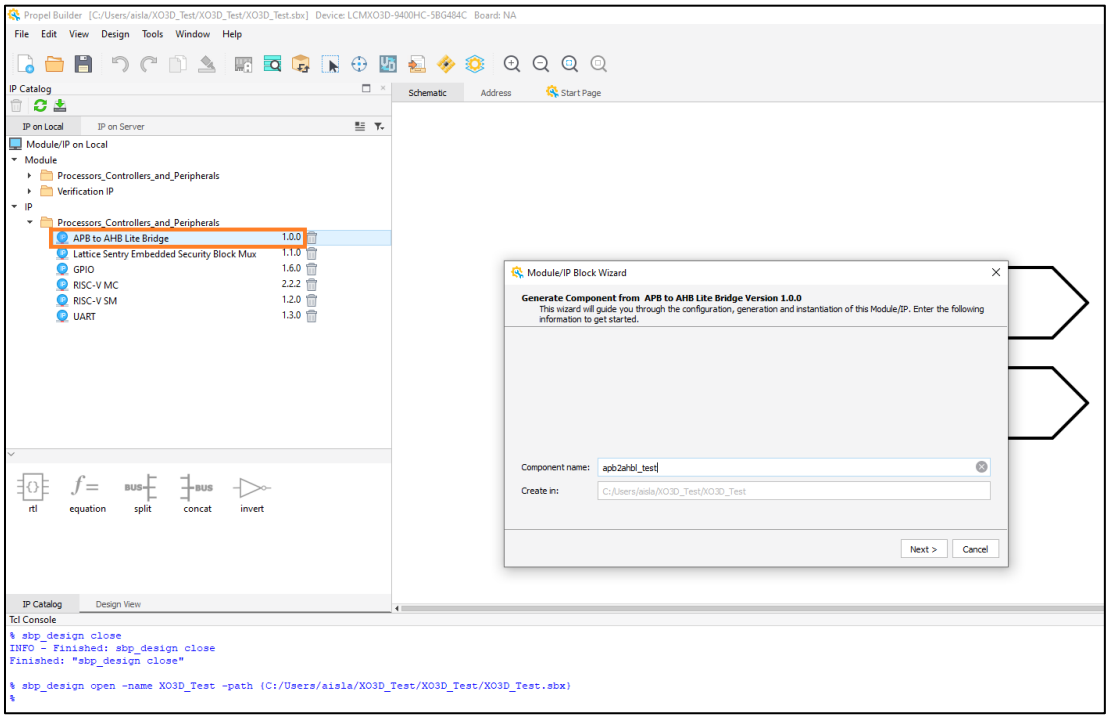

Figure 5.5. Module/IP Block Wizard

6. Using the Simulation Script File (.DO)

This reference design includes pre-generated RTL files that can be directly simulated within ModelSim. To use the simulation file, perform the following steps:

1. Open the DO file on a text editor and replace the text **SET THE SIMULATION PATH HERE** from Line 1 with the directory path of the simulation file. Figure 6.1 shows an example of Line 4 in the file.

```
1  set SIM_DIR "SET THE SIMULATION PATH HERE"
2
3  # Example:
4  # set SIM_DIR "D:/APB_to_AHBL_Bridge/Simulation"
```

Figure 6.1. Changing the Simulation Directory

2. Open ModelSim Lattice FPGA Edition and go to **Tools > Execute Macro**.

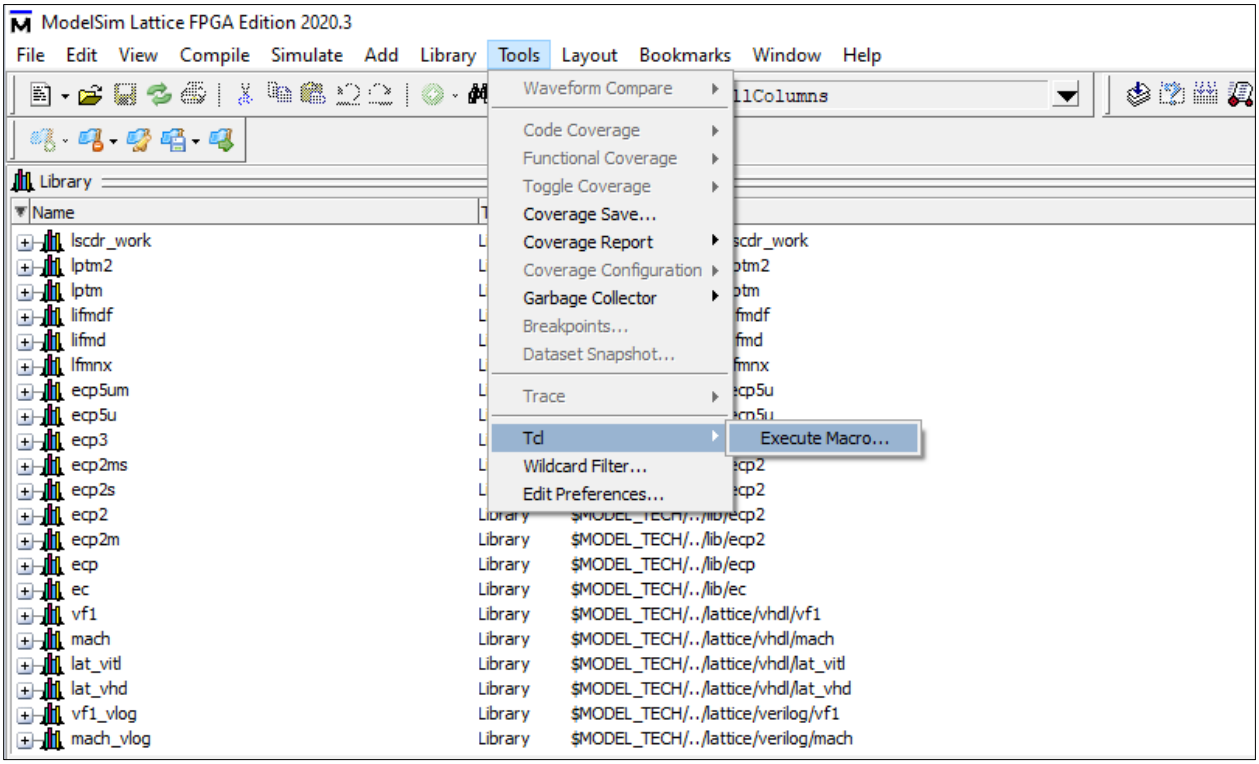

Figure 6.2. Running the Simulation Script File

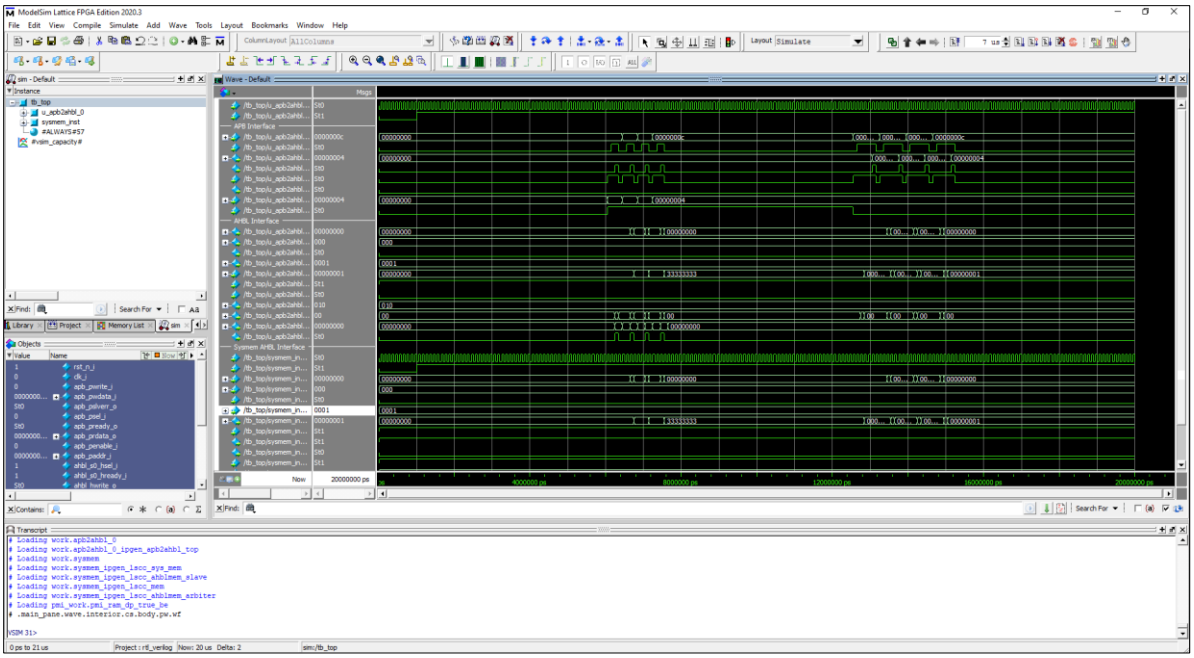

Figure 6.3. Simulation Waveform

7. Implementation

Table 7.1. Resource Utilization

Device Family	Language	LUTs	Register	f_{MAX} (MHz)
MachXO2	Verilog	11	102	>100
MachXO3	Verilog	11	102	>100
MachXO3D	Verilog	11	102	>100

Notes:

1. Performance and utilization characteristics are generated using LCMXO2-7000HE-4TG144C with Lattice Diamond 3.12 design software with either LSE (Lattice Synthesis Engine) or Synplify Pro®.
2. Performance and utilization characteristics are generated using LCMXO3LF-6900C-5BG256C with Lattice Diamond 3.12 design software with either LSE (Lattice Synthesis Engine) or Synplify Pro.
3. Performance and utilization characteristics are generated using LCMXO3D-9400HC-5BG256C with Lattice Diamond 3.12 design software with either LSE (Lattice Synthesis Engine) or Synplify Pro.

References

- MachXO2 FPGA Web Page in latticesemi.com
- MachXO3 FPGA Web Page in latticesemi.com
- MachXO3D FPGA Web Page in latticesemi.com

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.0, February 2023

Section	Change Summary
All	Initial release.

www.latticesemi.com