AMBA AHB-Lite Slave Protocol Verification Plan

Submitted By:
Yazan Hussnain

CHAPTER 1

INTRODUCTION

AMBA AHB-Lite is a bus interface designed for high-performance computer designs. It allows one device to control the communication on the bus and enables fast data transfer.

AHB-Lite incorporates the necessary elements for advanced and fast

1. Bursts Transfers
2. Operations synchronized with a single clock edge
3. Implementation without the use of tristate logic
4. Support for wide data bus configurations, 64, 128, 256, 512, and 1024 bits

Internal memory devices, external memory interfaces, and high-speed peripherals are the primary types of AHB-Lite slaves commonly used. While low-speed peripherals can also be connected as AHB-Lite slaves, they are typically connected to the AMBA Advanced Peripheral Bus (APB) for better overall system performance. To facilitate communication between the higher-level AHB-Lite bus and the APB, an AHB-Lite slave called an APB bridge is employed.

In Figure 1-1, a system design is depicted using a single AHB-Lite master and three AHB-Lite slaves. The bus interconnect logic, comprised of an address decoder and a slave-to-master multiplexer, plays a crucial role. The address decoder monitors the master's address to select the appropriate slave, while the multiplexer routes the corresponding slave's output data back to the master.

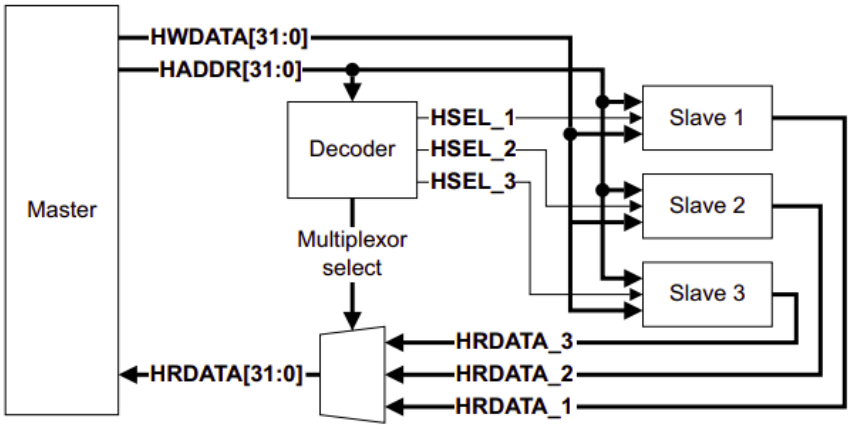

Figure 1-1 AHB-Lite block diagram

The key components of an AHB-Lite system are explained as follows:

1. **Master** - This component serves as the controlling entity in the system.
2. **Slave** - These components are the devices or peripherals that are controlled by the master.
3. **Decoder** - This component is responsible for interpreting the address signals from the master and selecting the appropriate slave.
4. **Multiplexor** - This component routes the data from the selected slave back to the master.

1.1 Master

An AHB-Lite master is responsible for sending address and control signals to initiate read and write operations. Figure 1-2 illustrates the interface design of an AHB-Lite master.

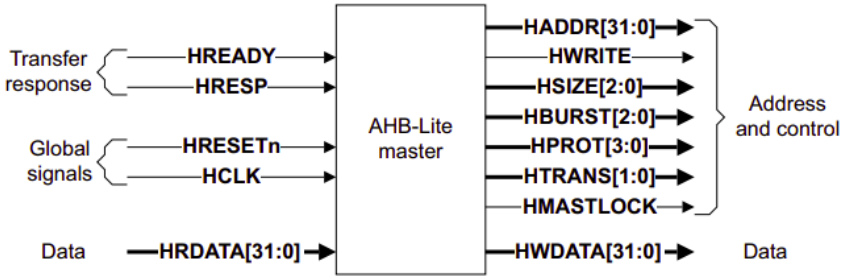

Figure 1-2 Master interface

1.2 Slave

An AHB-Lite slave is responsible for handling transfers initiated by masters within the system. The slave utilizes the HSELx select signal from the decoder to determine when it should respond to a bus transfer. It communicates back

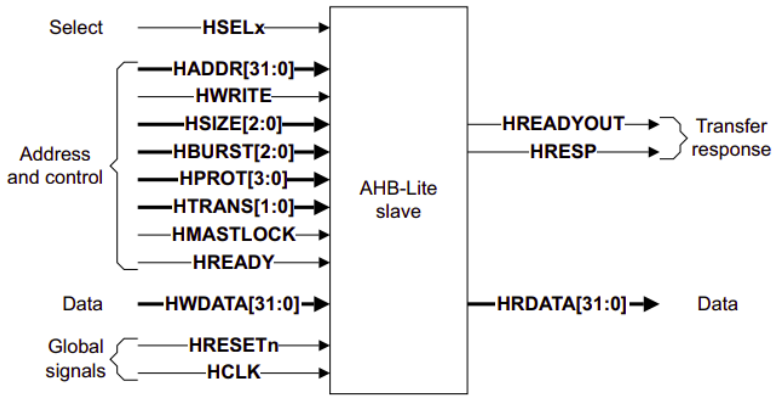

Figure 1-3 Slave interface

to the master to indicate the outcome of the data transfer, whether it was successful, unsuccessful, or if it is waiting for the transfer to complete. Figure 1-3 illustrates the interface design of an AHB-Lite slave.

1.3 Decoder

This particular component interprets the address of every transfer and generates a select signal for the corresponding slave involved in the transfer. Additionally, it supplies a control signal to the multiplexer.

1.4 Multiplexor

To combine the read data bus and response signals from the slaves into a single stream for the master, a slave-to-master multiplexer is necessary. The decoder is responsible for controlling this multiplexer.

1.5 Operation

The initiation of a transfer is carried out by the master through the transmission of address and control signals. These signals convey crucial information such as the address itself, transfer direction, data width, and whether it is part of a burst. Transfers can be categorized as single transfers, incrementing bursts that do not cross address boundaries, or wrapping bursts that wrap around specific address boundaries.

The write data bus facilitates the movement of data from the master to a slave, while the read data bus enables data transfer from a slave to the master.

Each transfer is composed of two phases:

Address phase: In this phase, the address and control information are transmitted in a single cycle.

Data phase: This phase consists of one or more cycles dedicated to transferring the actual data.

It is important to note that a slave cannot request an extension of the address phase; therefore, all slaves must be capable of sampling the address during this time. However, a slave can request the master to extend the data phase using the HREADY signal. When the HREADY signal is set to LOW, wait states are inserted into the transfer, allowing the slave extra time to provide or sample data.

The success or failure of a transfer is indicated by the slave using the HRESP signal.

CHAPTER 2
SIGNALS DESCRIPTIONS

In this chapter, the protocol signals are explained, which are divided into the following sections:

Global signals: This section covers the signals that are applicable globally.

Master signals: Here, the signals specific to the master component are discussed.

Slave signals: This section focuses on the signals associated with the slave component.

Decoder signals: The signals related to the decoder component are detailed in this section.

Multiplexer signals: Here, the signals pertaining to the multiplexer component are outlined.

2.1 Global Signals

Name	**Source**	**Description**
HCLK	Clock Source	The bus clock times all bus transfers. All signal timings are related to the rising edge of HCLK.
HRESETn	Reset Controller	The bus reset signal is active LOW and resets the system and the bus. This is the only active LOW AHB-Lite signal.

2.2 Master Signals

Name	Destination	Description
HADDR[31:0]	Slave	The 32-bit system address bus
HBURST[2:0]	Slave	The burst type indicates if the transfer is a single transfer or forms part of a burst. Fixed length bursts of 4, 8, and 16 beats are supported. The burst can be incrementing or wrapping. Incrementing bursts of undefined length are also supported.
HPROT[3:0]	Slave	The protection control signals provide additional information about a bus access and are primarily intended for use by any module that wants to implement some level of protection. The signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a privileged mode access or user mode access. For masters with a memory management unit these signals also indicate whether the current access is cacheable or bufferable.
HSIZE[2:0]	Slave	Indicates the size of the

		transfer, that is typically byte, halfword, or word. The protocol allows for larger transfer sizes up to a maximum of 1024 bits.
HTRANS[1:0]	Slave	Indicates the transfer type of the current transfer. This can be: -> IDLE -> BUSY -> NON SEQUENTIAL -> SEQUENTIAL.
HWDATA[31:0]	Slave	The write data bus transfers data from the master to the slaves during write operations. A minimum data bus width of 32 bits is recommended. However, this can be extended to enable higher bandwidth operation.
HWRITE	Slave	Indicates the transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer. It has the same timing as the address signals, however, it must remain constant throughout a burst transfer.

2.3 Slave Signals

Name	Destination	Description
HRDATA[31:0]	Multiplexor	During read operations, the read data bus transfers data from the selected slave to the multiplexor. The multiplexor then transfers the data to the master.
HREADYOUT	Multiplexor	When HIGH, the **HREADYOUT** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESP	Multiplexor	The transfer response, after passing through the multiplexor, provides the master with additional information on the status of a transfer. When LOW, the **HRESP** signal indicates that the transfer status is OKAY. When HIGH, the **HRESP** signal indicates that the transfer status is ERROR.

2.4 Decoder Signals

Name	Destination	Description
HSELx	Slave	Each AHB-Lite slave has its own slave select

		signal HSELx and this signal indicates that the current transfer is intended for the selected slave. When the slave is initially selected, it must also monitor the status of HREADY to ensure that the previous bus transfer has completed, before it responds to the current transfer. The HSELx signal is a combinatorial decode of the address bus.

2.5 Multiplexor Signals

Name	Destination	Description
HRDATA[31:0]	Master	Read data bus, selected by the decoder.
HREADY	Master and Slave	When HIGH, the **HREADY** signal indicates to the master and all slaves, that the the previous transfer is complete.
HRESP	Master	Transfer response, selected by the decoder

Modify the test bench

-> Transaction

Q. Can I run predefined sequences? (e.g. reset sequence, random write sequence, random read/write bursts, a directed test).

A. Yes, using the program block in a separate file you can run your own sequences.

Q. Can I debug easily if my test fails?

A. Yes, the print_trans() method can be used for debugging purposes.

Q. Do I need one or multiple transactions for bursts?

A. Yes, to check wrapping burst 4 and incrementing burst4, we require multiple transactions while for single burst we need only one transaction.

-> Generator

Q. How to control how many transactions get generated?

A. There is a variable in the generator class named **genCount**. This variable controls how many transactions are generated by the generator. So we can control the number of transactions by controlling the genCount variable in the environment class.

Q. How do I generate non-random transactions when required?

A. If you want to generate non-random transactions, first you have to disable randomization of the specific property of the class or of all the properties, and then you can give your own values in the program block in the test file.

-> Driver

Q. Are the interface signals driven according to the spec?

A. Yes, all the interface signals are driven according to the specifications.

Q. Does the transaction have proper address/data Phases?

A. Yes, read and write transfer have proper address and data phases. This can be observed using the check in the scoreboard and simulation

-> Monitor

Q. Are the interface signals sampled according to the spec? Does the transactions have proper address/data Phases?

A. Yes, they are sampled according to the specifications. Read and write transfers have proper address and data phases. This can be observed using the check in the scoreboard and simulation

->Scoreboard

Q. Does the scoreboard implement proper endianness? How to change endianness if required? How to not compare reset values and to compare only those memory locations which have already been written? Should scoreboard memory be static or dynamic?

A. Yes, the scoreboard takes care of the endianness as well. Endianness can only be changed if design has the liberty to change endianness. As for the design, it is given to us. There is no such signal that is used to change endianness. If you want to change endianness you have to change the design to implement this functionality. If you want to access only those memory locations that are not reset, you require some kind of metadata that tells which memory location is written.

Verification Plan

Test ID	Test Name	Test Status	Test Description	Stimulus Generation Procedure	Test Passes When	Test Fails When	Checking Procedure	Comments
1	Basic Write Operation	Pass	Write Data with no wait states	At Same Address: HWRITE is HIGH. HADDR is a constant. At Different Address HWRITE is HIGH	HWDATA is wrriten on the memory at the given address.	HWDATA is not wrriten on the memory at the given address.	Write the same data in the dummy memory then read the original memory and compare with the dummy memory	HRESP should be OKAY and HREADY should be HIGH
2	Basic Read Operation	Pass	Read Data with no wait states	From Same Address: HWRITE is LOW. HADDR is a constant. From Different Address HWRITE is LOW	Correct Data is read from the memory at given address	Incorrect Data is read from the memory at given address	When read transfer is initiated data from dummy memory and output of DUT is compared.	HRESP should be OKAY and HREADY should be HIGH
3	IDLE Transfer	Fail	Change transfer mode to IDLE	HTRANS is H_IDLE	Slave give zero wait OKAY response and transfer is ignored	Slave does not give zero wait OKAY response or do any read and write transfer.	Check the response of slave whenever IDLE transfer is initiated in reference model. Also using simulation	HRESP should be OKAY IDLE read transfer is not ignored but write transfer is ignored
4	BUSY Transfer	Fail	Change transfer mode to BUSY	HTRANS is H_BUSY	Slave give zero wait OKAY response and transfer is ignored	Slave does not give zero wait OKAY response or do any read and write transfer.	using simulation	HRESP should be OKAY BUSY read transfer is not ignored but write transfer is ignored.
5	Data Access Protection	Fail	Check Data Access Protection	HPROT is 4'h1	Value of HPROT should be 1.	Value of HPROT is other than 1.	Check in the reference model and simulation.	Whatever be the value of HPROT, it is not effecting the output of the design.
6	No Slave Select	Fail	Data Transfer initiate when slave is not connected	HSEL is 1'b0	No read write operation is done by slave	If slave do any transfer	Check in the reference model and also using simulation.	Data is read from the memory but not write to it
7	Transfer Pending	Fail	When a slave adds a certain number of wait states before completing the response.	HREADY drive LOW to insert wait states	Wait states should be inserted	Drive HREADY LOW has not effect	using simulation	During a write transfer, wait states may be inserted, whereas read transfers do not take wait states into account.
8	Waited Read Transfer	Fail	Wait states are inserted during read transfer	HREADY drive LOW to insert wait states	For read transfers the slave does not have to provide valid data until the transfer is about to complete.	For read transfers the slave does not provide valid data when the transfer is about to complete.	using simulation	HRESP is OKAY
9	Waited Write Transfer	Fail	Wait states are inserted during write transfer	HREADY drive LOW to insert wait states	Data is written on the memory at given address when HREADY is HIGH after inserting wait state	Data written on memory at different address which is not corresponding to given HWDATA or data is written when HREADY is LOW	using simulation	HRESP is OKAY
10	Alternate Read and Write at same address	Pass	Perform read transfer at fixed address and then write to that address.	HWRITE drive LOW and HIGH alternatively	Read and Write transfer follow the same timing as in the basic read and write transfer and data is read and write correctly	Read and Write transfer do not follow the same timing as in the basic read and write transfer and data is read and write correctly	using reference model and simulation	
11	Correct Byte Lane	Pass	Byte is write and read to/from memory	Byte alligned addresses are generated	Correct byte is write/read to/from memory.	Incorrect byte is write/read to/from memory	Using reference model and simulation	HRESP is OKAY and HREADYOUT is HIGH
12	Correct Half Lane	Pass	Halfword is write/read to/from memory	Halfword aligned addresses are generated	Correct Halfword is write/read to/from memory	Incorrect halfword is write/read to/from memory	Using reference model and simulation	HRESP is OKAY and HREADYOUT is HIGH
13	Correct Word	Pass	Word is write/read to/from memory	Word aligned addresses are generated	Correct word is write/read to/from memory	Incorrect word is write/read to/from memory	Using reference model and simulation	HRESP is OKAY and HREADYOUT is HIGH
14	Slave Error Response	Fail	No information is given, how slave generate the ERROR response except one case: attempt to write to read-only memory. But HPROT also has no effect is terms of data access protection in design	No information is given which states of given signals leads to ERROR response	ERROR must retain two HCLK cycles.	ERROR does not retain for two clock cycles	Using reference model and simulation	Unable to reproduce slave ERROR response
15	Slave Response	Pass	A slave is required to provide a response that indicates the status of the transfer.	Write or read transfer is initiate to check the slave response	Slave must give response	Slave gives no response on read/write transfer	Using reference model and simulation	HRESP should responed
16	Transfer Done	Pass	A successful completed transfer is signalled		A completed transfer is signalled when HREADY is HIGH and HRESP is OKAY	HREADY is LOW or HRESP is ERROR	using reference model and simulation	HREADY should be HIGH and HRESP should be OKAY
17	Reset Memory	Fail	In reset mode, transfer mode changes to IDLE and HREADYOUT goes HIGH	HRESETn = LOW	Slave ensure that HREADYOUT is HIGH and memory reseted	HREADYOUT is not HIGH and memory is not reseted	using simulation	HREAFYOUT is always one. No effect of HRESETn on memory.

18	Waited transfer, BUSY to SEQ	Fail	During waited busy transfer no data should be write	HREADY is LOW HTRANS is BUSY	No write transfer occur during wait states	Write transfer occur during wait states	using simulation	
19	Waited transfer, IDLE to NONSEQ	Fail	During waited idle transfer no data should be write	HREADY is LOW HTRANS is IDLEY	No write transfer occur during wait state	Write transfer occur during wait states	using simulation	