Verification Plan of ARM AMBA-3 AHB-lite Protocol

ECE-593 Fundamentals of Pre-Silicon Validation Spring -2020

AAYUSH RAVICHANDRAN [981120898]

KAUSTUBH MHATRE [974819541]

SAURABH WAMAN CHAVAN [911836716]

I. Verification Requirements

A. Verification Levels-

The AMBA 3 AHB is a simple bus interface. In this particular design there are one master and two slaves. Hence, we only need to verify at two levels. One is where the smaller design unit level module will be individually verified and then a top-level verification of the complete interface.

1. Unit level verification

This level of verification was chosen because it would be simple to thoroughly check the working and functionality of each individual design modules before we move to the top-level verification. The AHB lite bus design consist of the following unit level components that could be verified

a) Memory Controller
b) Decoder
c) Multiplexer
d) Memory module/ slave

2. Chip level verification

This level of verification will test the interconnection between the different modules and that the entire interface is working as expected. Since the individual components were verified at the earlier level this makes verification at this level only verify the interworking between them. The top module instantiates all the unit level components and so is suitable for another level of verification. It needs to be verified for proper connectivity and functionality. Deterministic as well as random testcases are tested using class based testbench.

B. Functions-

The following are the functions and the appropriate action that are verified

- Different transfer types / burst operations (Single, INCR, INCR4, INCR8, INCR16, WRAP4, WRAP8, WRAP16) for both Read and Write operations.
- Waited operations such as master inserting Idle or Busy.
- Check if the two slave devices/ memory are being selected properly or not.
- Check if multiplexer is working or not.
- Check if memory module works as expected or not for both read and writes.

There is some functionality that won't be checked or some that have been limited because of the limitation in the design such as

- Protection types won't be verified.
- Locking operations won't be verified.
- Address and Data will be restricted to 8 bits.
- The multi slave functionality will only be tested for top level and after all individual unit levels have been tested.

C. Specific Tests & Methods-

i. Type of Verification

A black box approach will be taken to thoroughly verify the design. The stimulus will be driven into the design and a self-checking scoreboard checks the output.

ii. Verification Strategy

A mix of deterministic and constrained random strategy will be used to verify the design. The memory controller, decoder, multiplexer and the memory module will be verified with deterministic tests. Similarly, for the top level a mix of the two methods will be used along with a self-checking testbench.

In the unit level testing of the memory controller a write operation is followed by a read operation hence verifying both the operations. But for the top level a self-checking scoreboard performs all operations to its local memory array and then compares the output of the design to its own.

iii. Abstraction Level

For the top-level verification, the generator has cycle specific values for all master signals which are then driven into the interface by the driver.

iv. Checking

A golden vector and reference model type of self-checking testbench will be used to verify the design. The golden vector model will be easier to use for unit testing and for deterministic testing of the design modules. As we move to the top-level verification and perform many combinations of the operations it will be better to move to the reference model as we can just monitor the interface pins and check what results should be produced. The memory controller will be thoroughly checked to verify that the protocol is being followed as per the specification. To summarise.

For the top-level verification, a reference model is used where it performs the operation and then compares its results to that of the design.For the unit level verification, a mix of the two methods are used. Like for the memory controller a write operation performed followed by a read operation to the same address and hence its verified. For other modules expected results are compared with that of the design.

Some of the protocol specific checkers are as follows

When reset all signals must be at their default values
When master is in BUSY state the present operation is halted and it remains halted until master remains in the BUSY state
For IDLE transfer type no data transfer should occur and any pending transaction should be terminated
For BUSY transfer type when used the address and control signals should reflect the next transfer
Only burst of undefined length (INCR) can have a BUSY transfer as the last cycle of burst
The control information for a SEQ transfer type is identical to previous transfer (SEQ transfer type happens for burst operations after the first cycle where the control information is given along with address and data and remains SEQ till the operation is complete)
For SEQ transfer type the next address is addition of previous address plus the transfer size in bytes (the controller calculates the next address for the operation by adding the transfer size to the current address)
HSIZE signal should remain constant throughout the transfer (the transfer size should remain constant for the entire operation)
For wrapping burst transfer when address crosses the address boundary, they should wrap around
A master should not attempt to start incrementing burst that crosses a 1KB address boundary
All transfers in a burst must be aligned to the address boundary equal to the size of transfer
Master cannot perform a BUSY transfer immediately after a SINGLE operation
Transfers of fixed length burst types must terminate with a SEQ transfer
When a wait state is introduced the master can only change transfer type and address

D. Coverage-

Code and functional coverage will be used to find interesting points like what all different control signals were used, different operations performed, different types of operations performed and different transfer sizes and burst, etc as well as different combinations of the operations. As per the test plan come coverage components will be limited in some capacity such as address and data to 8 bits for reduced test cases and better results.

E. Scenarios-

Some of the scenarios are as follows

TEST NUMBER	TEST CASE NAME
	Single Read Operation
1	Read operation in Single burst mode when master is idle or busy and slave is ready (Here we perform a Read operation with HBURST = SINGLE and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = IDLE or BUSY for the second cycle)
2	Read operation in Single burst mode when master is ready and slave is ready (Here we perform a Read operation with HBURST = SINGLE and HTRANS = NONSEQ for the first cycle)
	Single Write Operation
3	Write operation in Single burst mode when master is idle or busy and slave is ready (Here we perform a Write operation with HBURST = SINGLE and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = IDLE or BUSY for the second cycle)
4	Write operation in Single burst mode when master is ready and slave is ready (Here we perform a Write operation with HBURST = SINGLE and HTRANS = NONSEQ for the first cycle)

	INCR Burst Read Operation
5	Read operation in INCR burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = INCR and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and then we introduce an HTRANS = IDLE or BUSY in one of the following cycles)
6	Read operation in INCR burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = INCR and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next n cycles)
	INCR4 Burst Read Operation
7	Read operation in INCR4 burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = INCR4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
8	Read operation in INCR4 burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = INCR4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 2 cycles)
	INCR8 Burst Read Operation
9	Read operation in INCR8 burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = INCR8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
10	Read operation in INCR8 burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = INCR8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 6 cycles)

	INCR16 Burst Read Operation
11	Read operation in INCR16 burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = INCR16 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
12	Read operation in INCR16 burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = INCR16 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 14 cycles)
13	Read operation in INCR, INCR4, INCR8 or INCR16 burst mode when both master and slave are ready and transaction happens beyond the range specified by the burst operation
	INCR Burst Write Operation
14	Write operation in INCR burst mode when master is not ready and slave is ready (Here we perform a Write operation with HBURST = INCR and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and then we introduce an HTRANS = IDLE or BUSY in one of the following cycles)
15	Write operation in INCR burst mode when both master and slave are ready (Here we perform a Write operation with HBURST = INCR and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next n cycles)
	INCR4 Burst Write Operation
16	Write operation in INCR4 burst mode when master is not ready and slave is ready (Here we perform a Write operation with HBURST = INCR4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the

	operation resumes till it's completed)
17	Write operation in INCR4 burst mode when both master and slave are ready (Here we perform a Write operation with HBURST = INCR4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 2 cycles)
	INCR8 Burst Write Operation
18	Write operation in INCR8 burst mode when master is not ready and slave is ready (Here we perform a Write operation with HBURST = INCR8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
19	Write operation in INCR8 burst mode when both master and slave are ready (Here we perform a Write operation with HBURST = INCR8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 6 cycles)
	INCR16 Burst Write Operation
20	Write operation in INCR16 burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = INCR16 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
21	Write operation in INCR16 burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = INCR16 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 14 cycles)
22	Write operation in INCR, INCR4, INCR8 or INCR16 burst mode when both master and slave are ready and transaction happens beyond the range specified by the burst operation

	WRAP4 Burst Read Operation
23	Read operation in WRAP4 burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = WRAP4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
24	Read operation in WRAP4 burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = WRAP4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 2 cycles)
	WRAP8 Burst Read Operation
25	Read operation in WRAP8 burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = WRAP8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
26	Read operation in WRAP8 burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = WRAP8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 6 cycles)
	WRAP16 Burst Read Operation
27	Read operation in WRAP16 burst mode when master is not ready and slave is ready (Here we perform a Read operation with HBURST = WRAP16 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
28	Read operation in WRAP16 burst mode when both master and slave are ready (Here we perform a Read operation with HBURST = WRAP16 and HTRANS = NONSEQ

	for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 14 cycles)
29	Read operation in WRAP4, WRAP8 or WRAP16 burst mode when both master and slave are ready and transaction happens beyond the range specified by the burst operation
	WRAP4 Burst Write Operation
30	Write operation in WRAP4 burst mode when master is not ready and slave is ready (Here we perform a Write operation with HBURST = WRAP4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
31	Write operation in WRAP4 burst mode when both master and slave are ready (Here we perform a Write operation with HBURST = WRAP4 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 2 cycles)
	WRAP8 Burst Write Operation
32	Write operation in WRAP8 burst mode when master is not ready and slave is ready (Here we perform a Write operation with HBURST = WRAP8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
33	Write operation in WRAP8 burst mode when both master and slave are ready (Here we perform a Write operation with HBURST = WRAP8 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 6 cycles)
	WRAP16 Burst Write Operation
34	Write operation in WRAP16 burst mode when master is not ready and slave is ready (Here we perform a Write operation with HBURST = WRAP16 and HTRANS =

	NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on but for some cycles in between HTRANS = INDE or BUSY is asserted and then the operation resumes till it's completed)
35	Write operation in WRAP16 burst mode when both master and slave are ready (Here we perform a Write operation with HBURST = WRAP16 and HTRANS = NONSEQ for the first cycle and then introduce an HTRANS = SEQ for the second cycle and so on for the next 14 cycles)
36	Write operation in WRAP4, WRAP8 or WRAP16 burst mode when both master and slave are ready and transaction happens beyond the range specified by the burst operation
	Different Combinations of Above operations
37	Write operations followed by Read operations (of the same Burst transfer type and of different types)
38	Read operations followed by Write operations
39	Read operations followed by other Read operations
40	Write operations followed by other Write operations

II. Project management:

A. Tools-

1. QuestaSim 2019
2. Modelsim PE Students Edition 10.4a

B. Risk/Dependencies-

- This plan does not have any specific risks / dependencies.
- But the short schedule forms a part of risk factor for completion of project.
- The new way of implementing testbench environment can be a potential risk due to lack of familiarity to this implementation.

C. Resources-

The team will consist of the following three members

1. Aayush Ravichandran (981120898)
2. Kaustubh Herambh Mhatre (974819541)
3. Saurabh Waman Chavan (9811836716)

D. Schedule-

1. Week1- Unit level testing of unit level components
2. Week2- Self checking testbench for memory controller
3. Week3- Verify top level module & multiple slaves
4. Week4- Revisit verification plan check for missed test cases