AHB LITE PROJECT

USING VERILOG

Marwan Khaled Mohamed

Contents

1 Design Architecture Overview

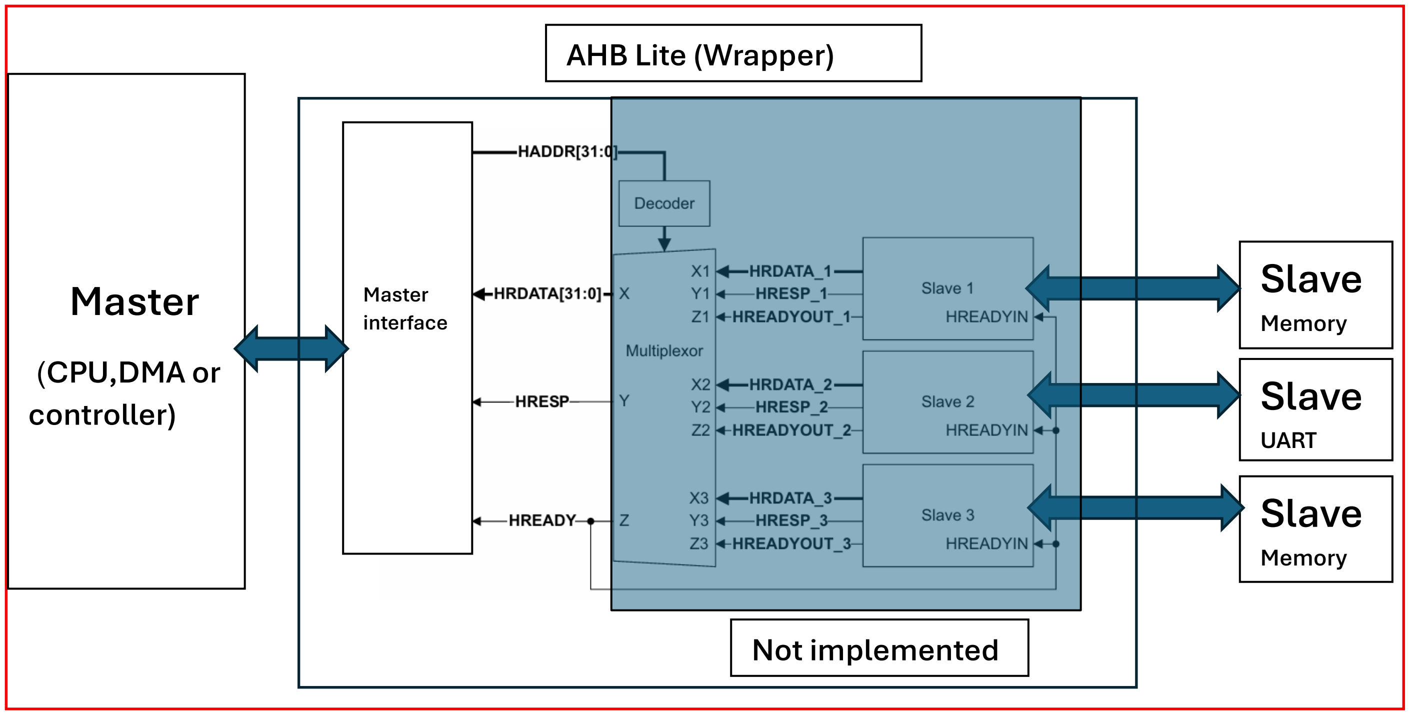
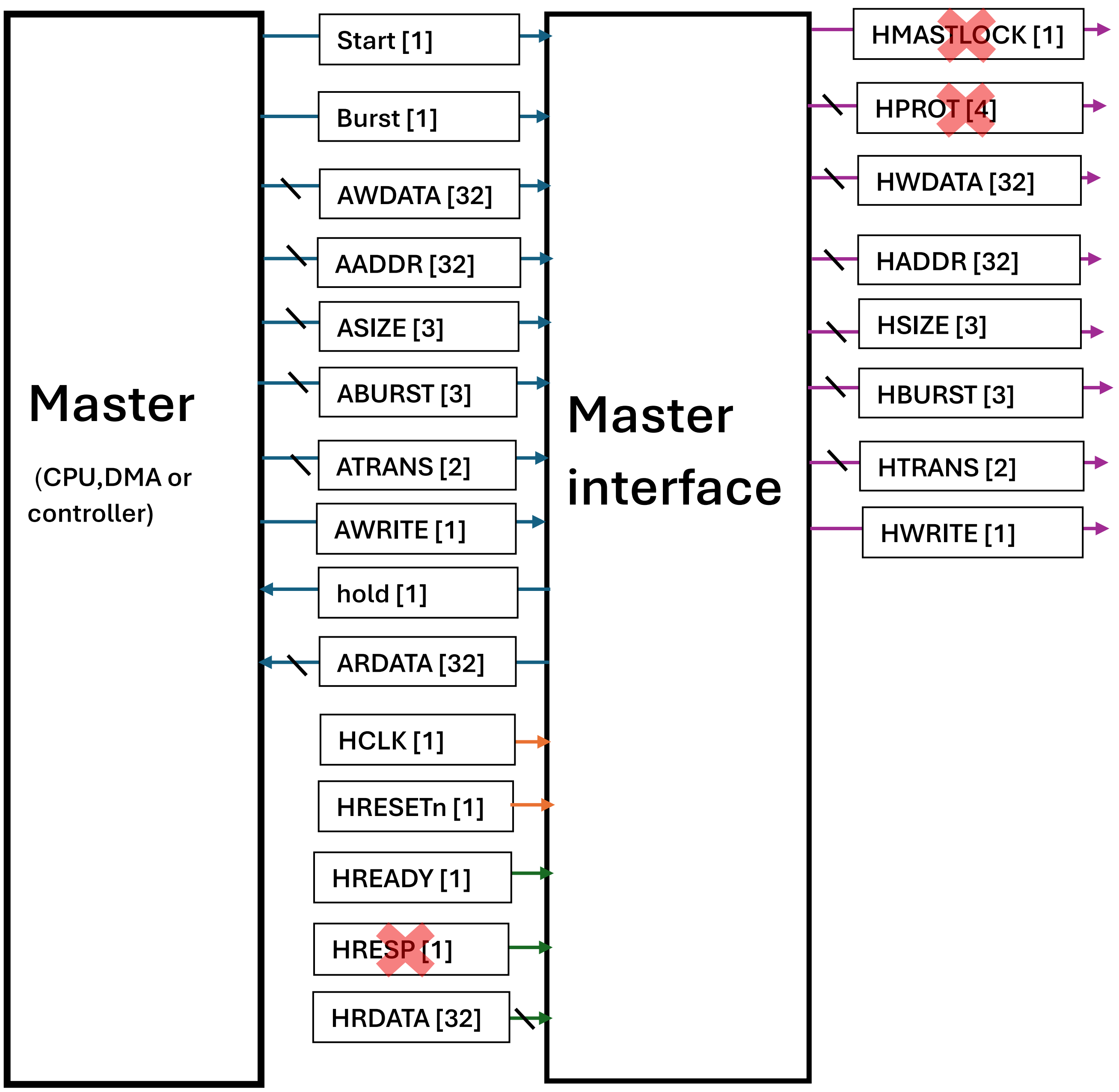


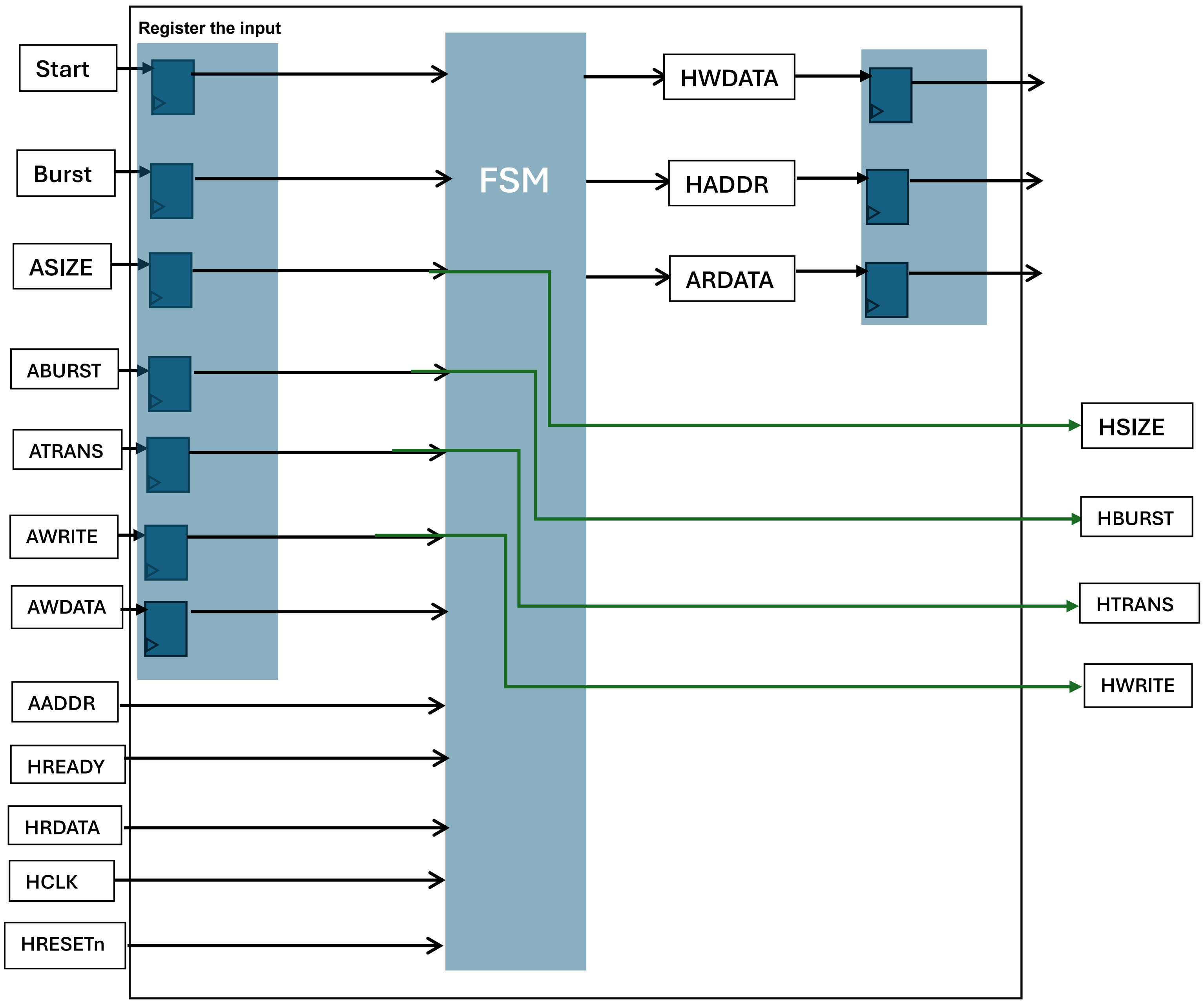
Figure 1:Design Architecture

2 Master interface Architecture Overview



🟦	master-> master interface	🟧	Global input	✖	Not implemented
🟩	interconnect-> master interface	🟪	master interface-> interconnect		

3 Detailed Architecture for Master AHB interface



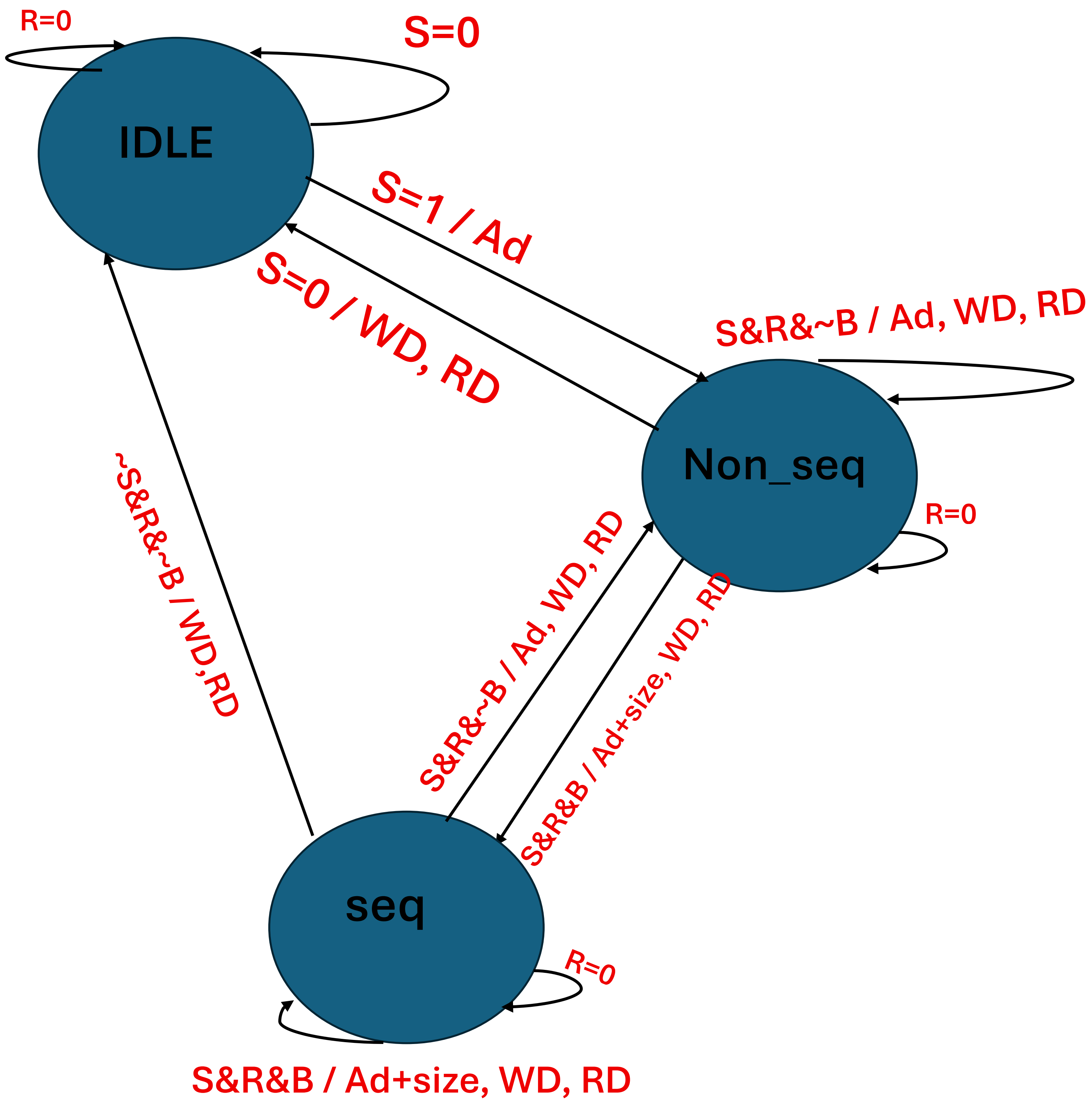
Notes:

1- I registered the input write data only not the address because I assumed that the master send the data and address at the same time

2-the output is registered because I made a mealy FSM

4 Details of the FSM

S :Start	B:Burst	R:Ready	Ad:HADDR	WD:HWDATA	RD:HRDATA



5 FSM explanation

cs	ns	When?	input	output
IDLE	IDLE	Just no transaction	Start =0	No output
IDLE	Non_seq	Send first transaction whether its single or burst transaction	Start =1	address

Non_seq	Non_seq	Another transaction but not burst just multiple independent single transaction	Start=1
			Burst=0
			ready=1
				Address
				write,read data (maybe)

Non_seq	idle	The single transaction has just ended	Start =0	write,read data (maybe

Non_seq	seq	The second cycle at the burst	Start=1
			Burst=1
			ready=1
				Address+size
				write,read data (maybe)

seq	seq	Burst just continues	Start=1
			Burst=1
			ready=1
				Address+size
				write,read data (maybe)

seq	Non_seq	Burst has just ended but another cycle is goanna start	Start=1
			Burst=0
			ready=1
				Address
				write,read data (maybe)

seq	idle	Burst has just ended	Start=0
			Burst=0
			ready=1
				write,read data (maybe)

x	x	State still the same whatever it is	ready=0	Doesn't change

6 Application dependent signals

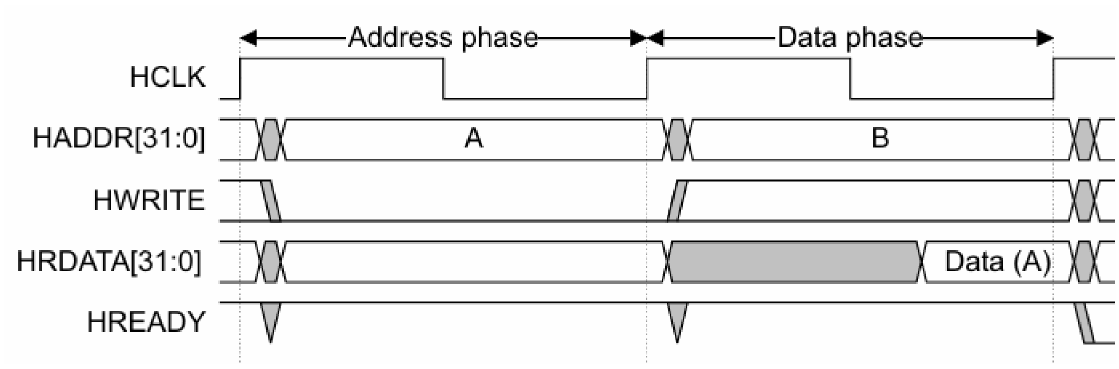
Signal name	direction	How to use it ?
Start	input	Make it high before starting the transaction and make It low back after sending the last data
Burst	input	Make it high before second data and make It low back after sending the last data in the burst
hold	output	The Master interface send it to master to tell it to stop sending data because there is a waiting state

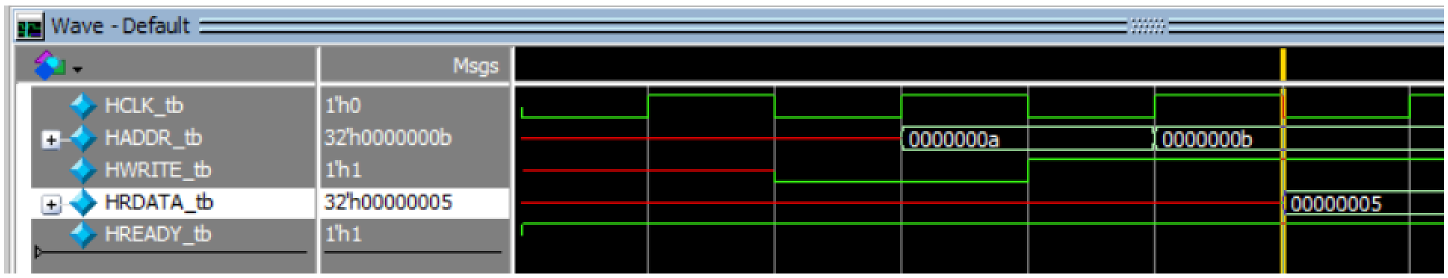
7 Supported specifications

Feature	Level of support	Description
Basic Write/read transfer	Fully supported ✓	Basic write and read
Locked transfers	Unsupported ✗	
Transfer size	Partially supported 📝	Byte Half word Word are supported and in case of other input map it to word
Burst operation	Partially supported 📝	Single INCR are supported and in case of other input map it to INCR
Waited transfer	Fully supported ✓	Wait until the slave be ready
Error handling (HRESP)	Unsupported ✗	
Slave implementation	Unsupported ✗	
Interconnect implementation	Unsupported ✗	

8 Waveform

8.1 Read transfer (Figure 3-1)





	Msgs
HCLK_tb	1'h0
HADDR_tb	32'h0000000b
HWRITE_tb	1'h1
HRDATA_tb	32'h00000005
HREADY_tb	1'h1

Figure 2:Read transfer

8.2 write transfer (Figure 3-2)

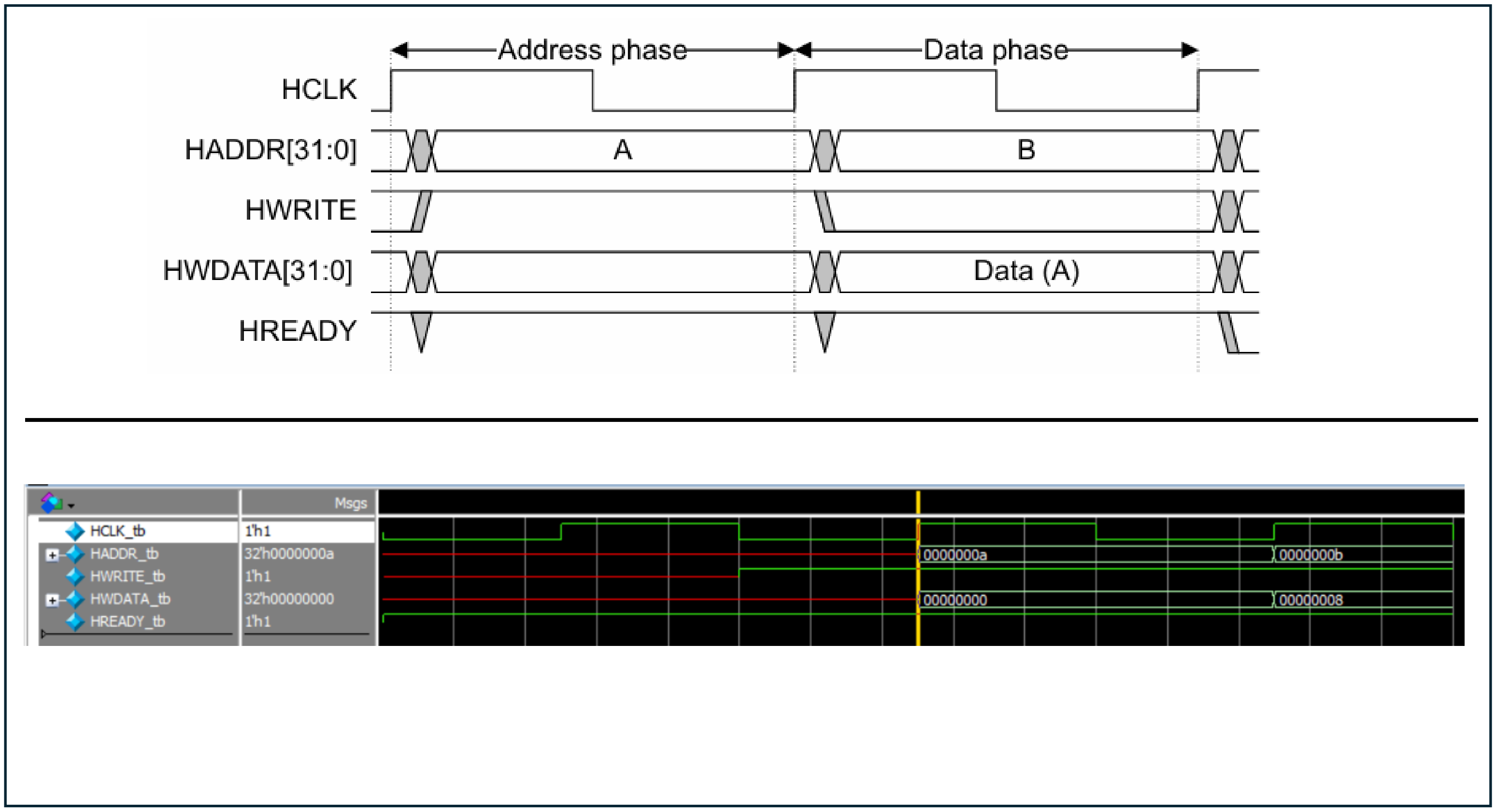
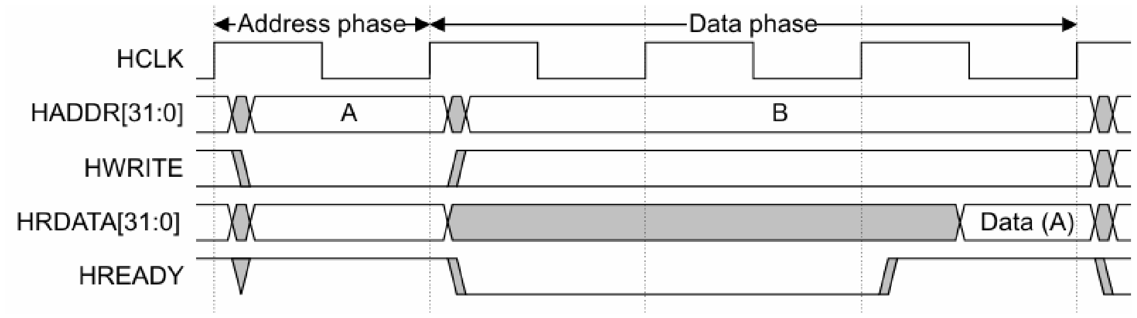
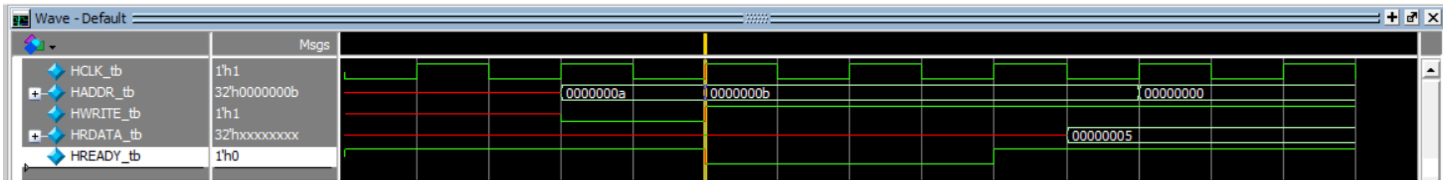


Figure 3:write transfer

8.3 Read transfer with wait states (Figure 3-3)

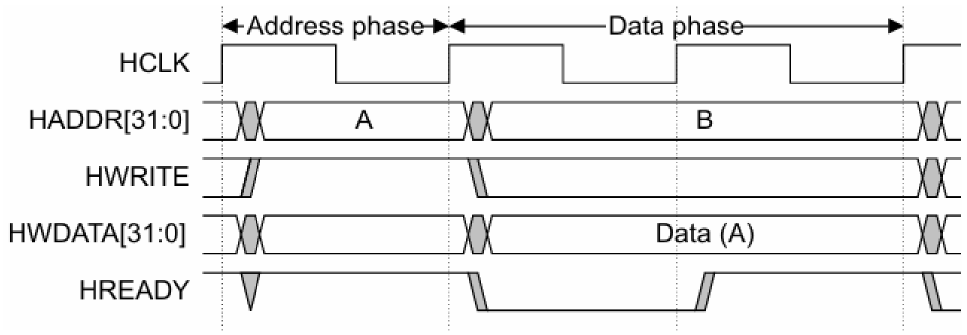


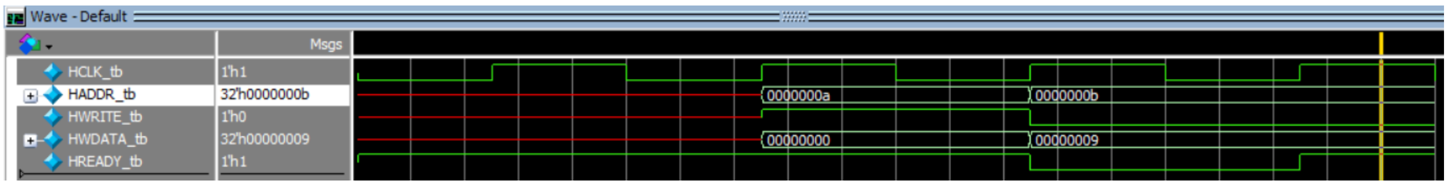


```
HCLK_tb      1'h1
HADDR_tb     32'h0000000b
HWRITE_tb    1'h1
HRDATA_tb    32'hxxxxxxxx
HREADY_tb    1'h0
```

Figure 4:Read transfer with wait states

8.4 Write transfer with wait state (Figure 3-4)

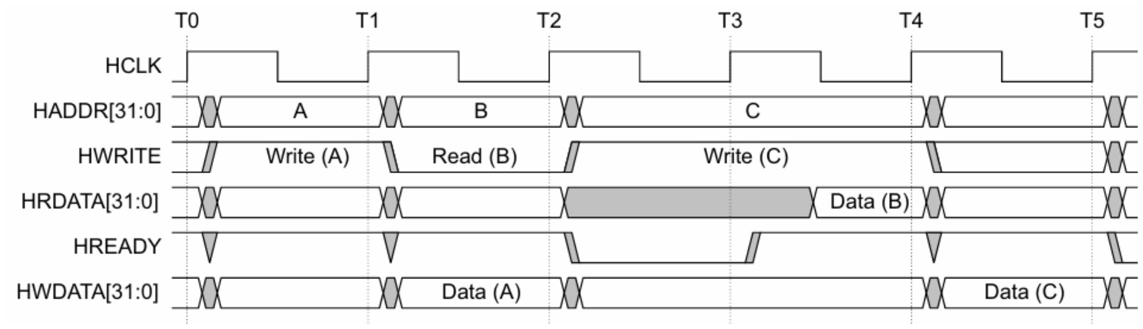




	Msgs
HCLK_tb	1'h1
HADDR_tb	32'h0000000b
HWRITE_tb	1'h0
HWDATA_tb	32'h00000009
HREADY_tb	1'h1

Figure 5:Write transfer with wait state

8.5 Multiple transfers (Figure 3-5)



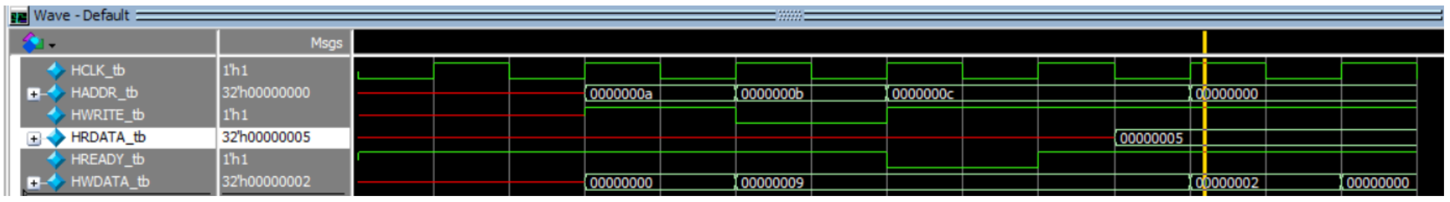
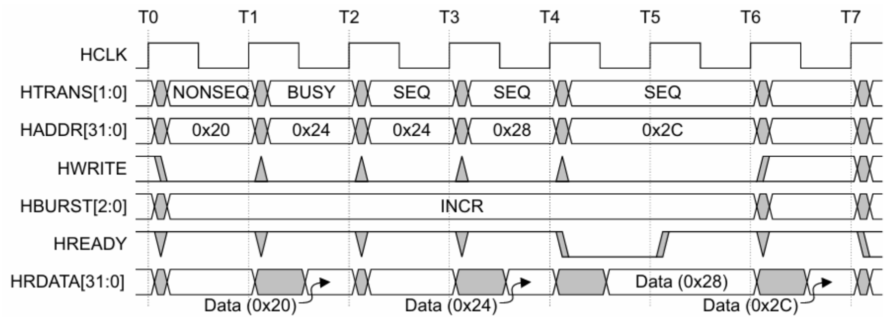


Figure 6:Multiple transfers

8.6 Transfer type examples (Figure 3-6)



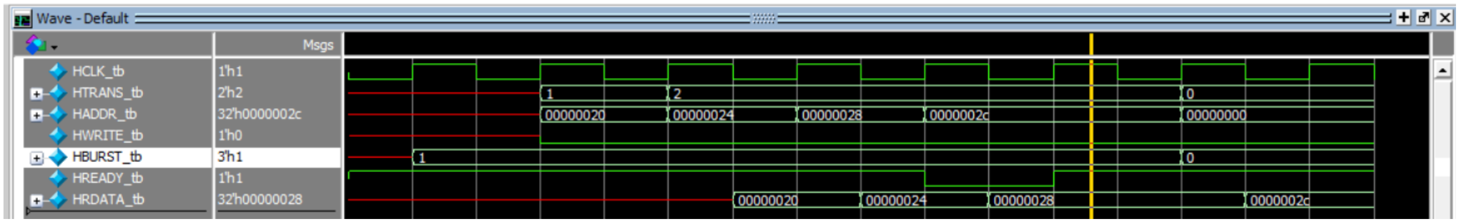
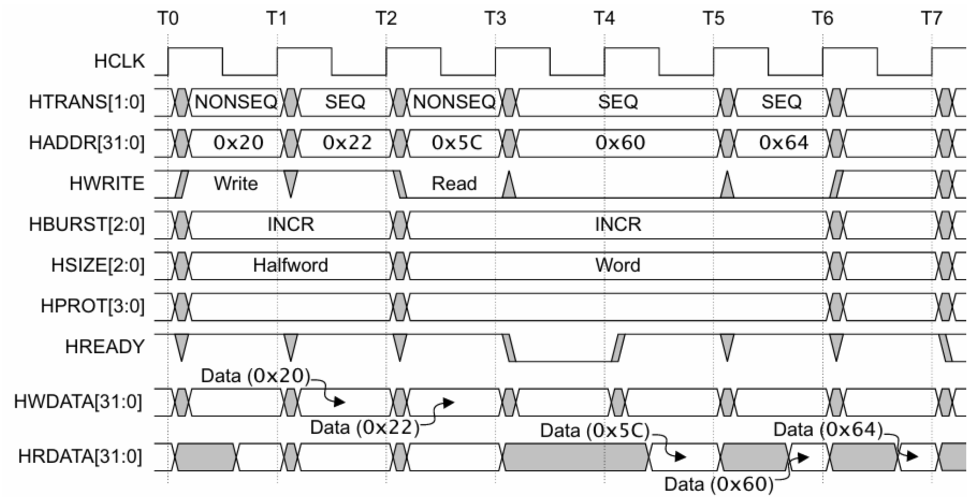


Figure 7:Transfer type examples

8.7 Undefined length bursts (Figure 3-12)



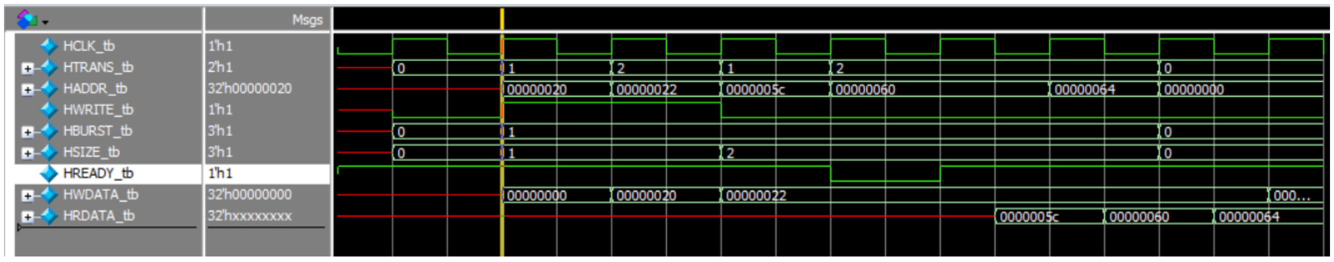


Figure 8:Undefined length bursts

9 Notes

9.1 Burst signal

Master AHB doesn't rely on HBURST value to calculate the seq state it depends only on the Burst signals comes from master to avoid this corner case

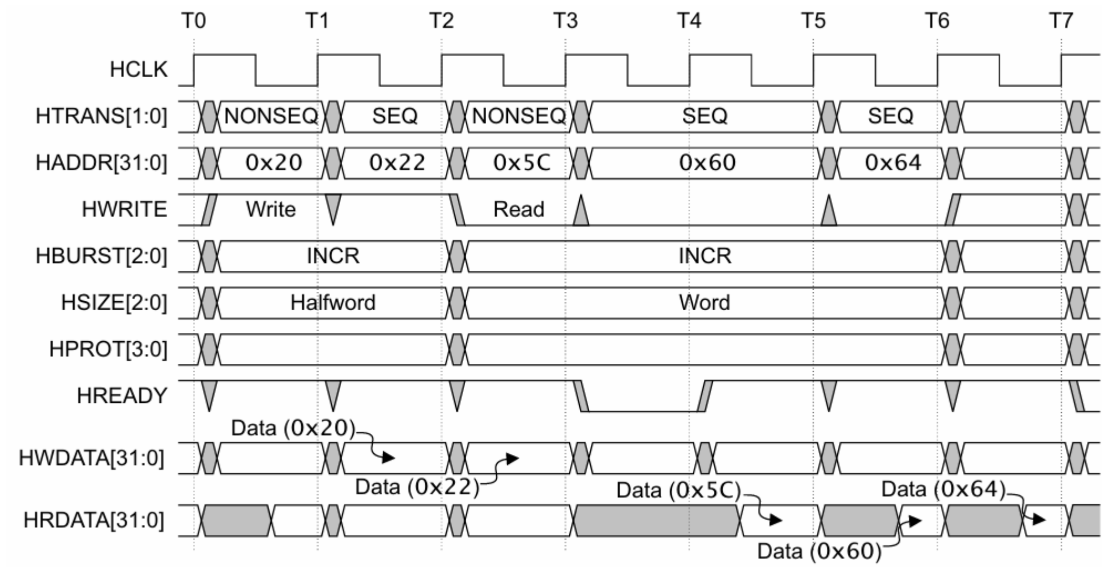


Figure 3-12 Undefined length bursts

- Although there are 2 different INCR but we need the burst signal to show me that and to make me move from seq to non seq

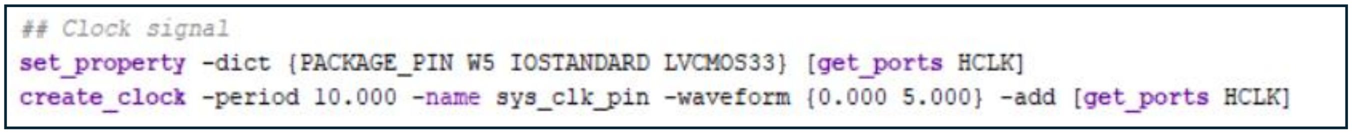
- If there is another burst case like (INCR4 ,..) the master will also make the burst high so they will act like INCR

- regarding the burst signal (the master must make it high only between the non seq and seq cycle (not before that) to make the AHB-Master notices if there's sequential burst with different address like the previous figure

- here I made the master end the transfer incase of INCR by using the signal burst but in the standard the INCR can be ended if the burst size exceed 1 KB but here It's not implemented

10 FPGA Design flow

10.1 Time constraints



```
## Clock signal
set_property -dict {PACKAGE_PIN W5 IOSTANDARD LVCMOS33} [get_ports HCLK]
create_clock -period 10.000 -name sys_clk_pin -waveform {0.000 5.000} -add [get_ports HCLK]
```

Figure 9:Time constraints

10.2 Elaborated Design

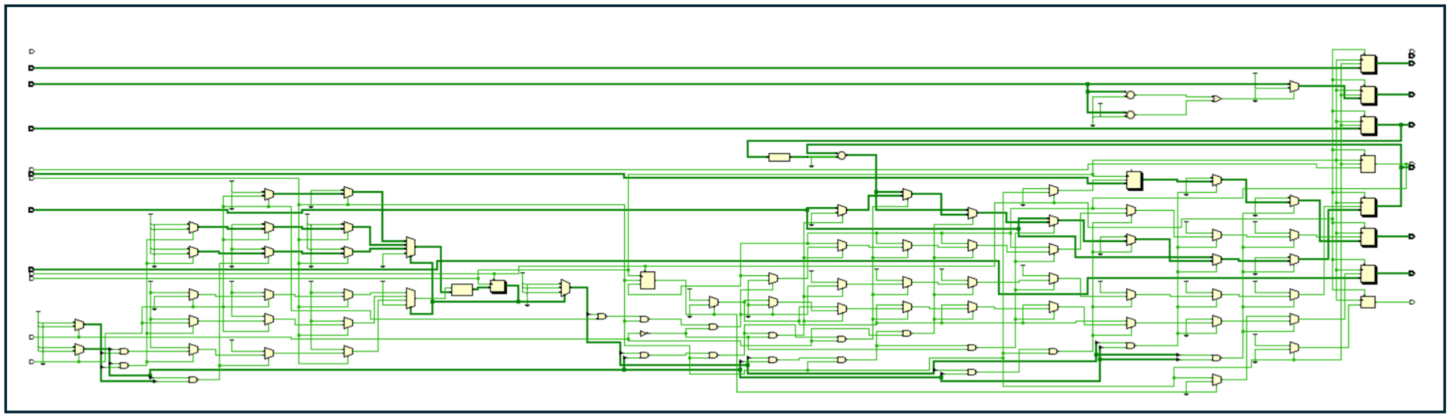


Figure 10:Elaborated Design

10.3 Synthesized Design

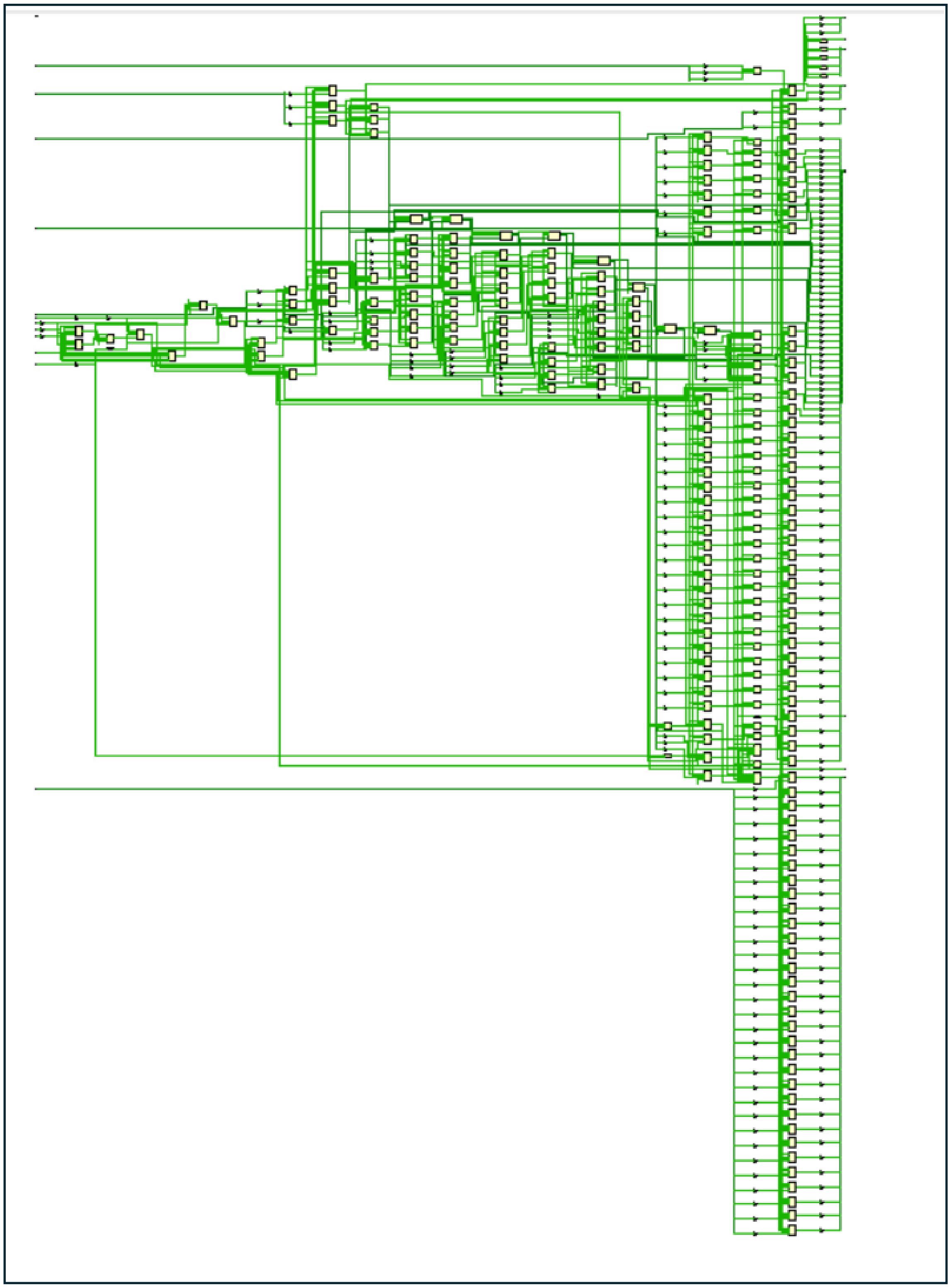


Figure 11:Synthesized Design

10.4 Implemented design

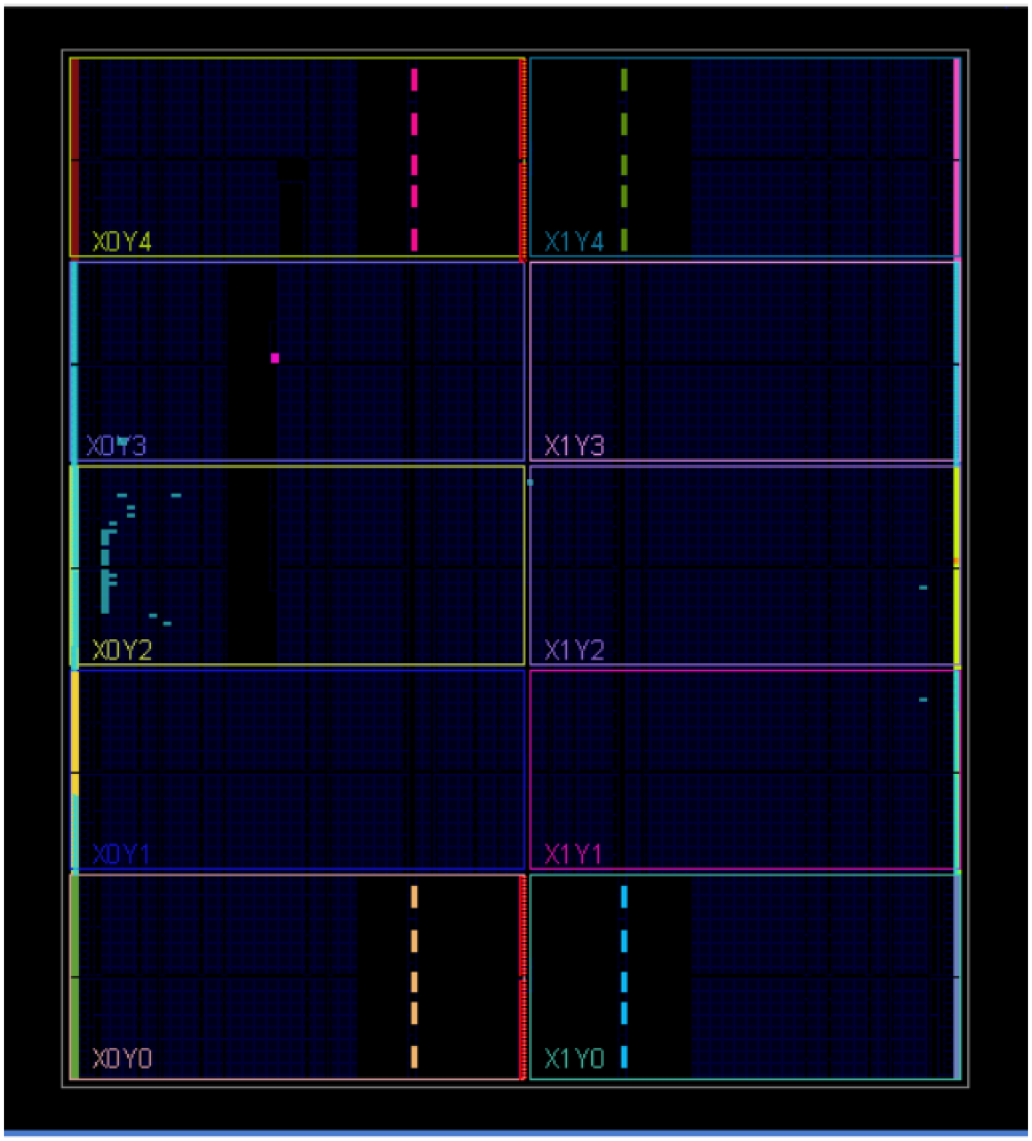
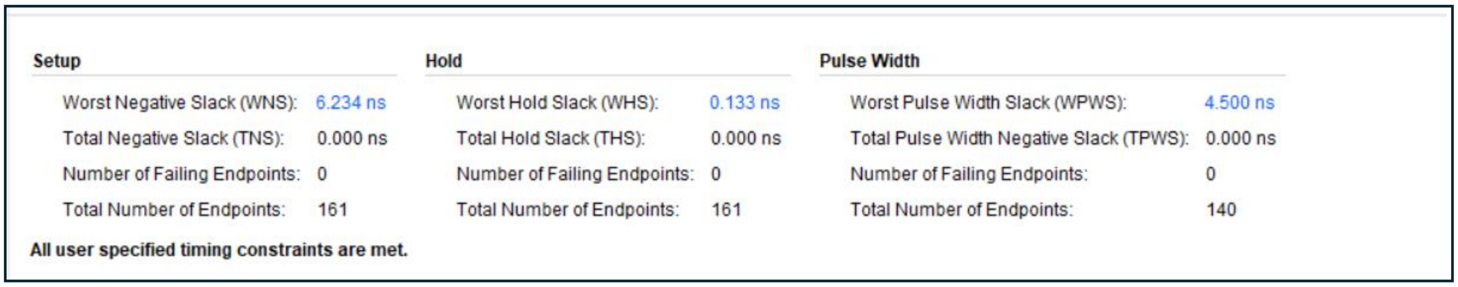


Figure 12:implemented Design

10.5 Timing report at 100MHZ



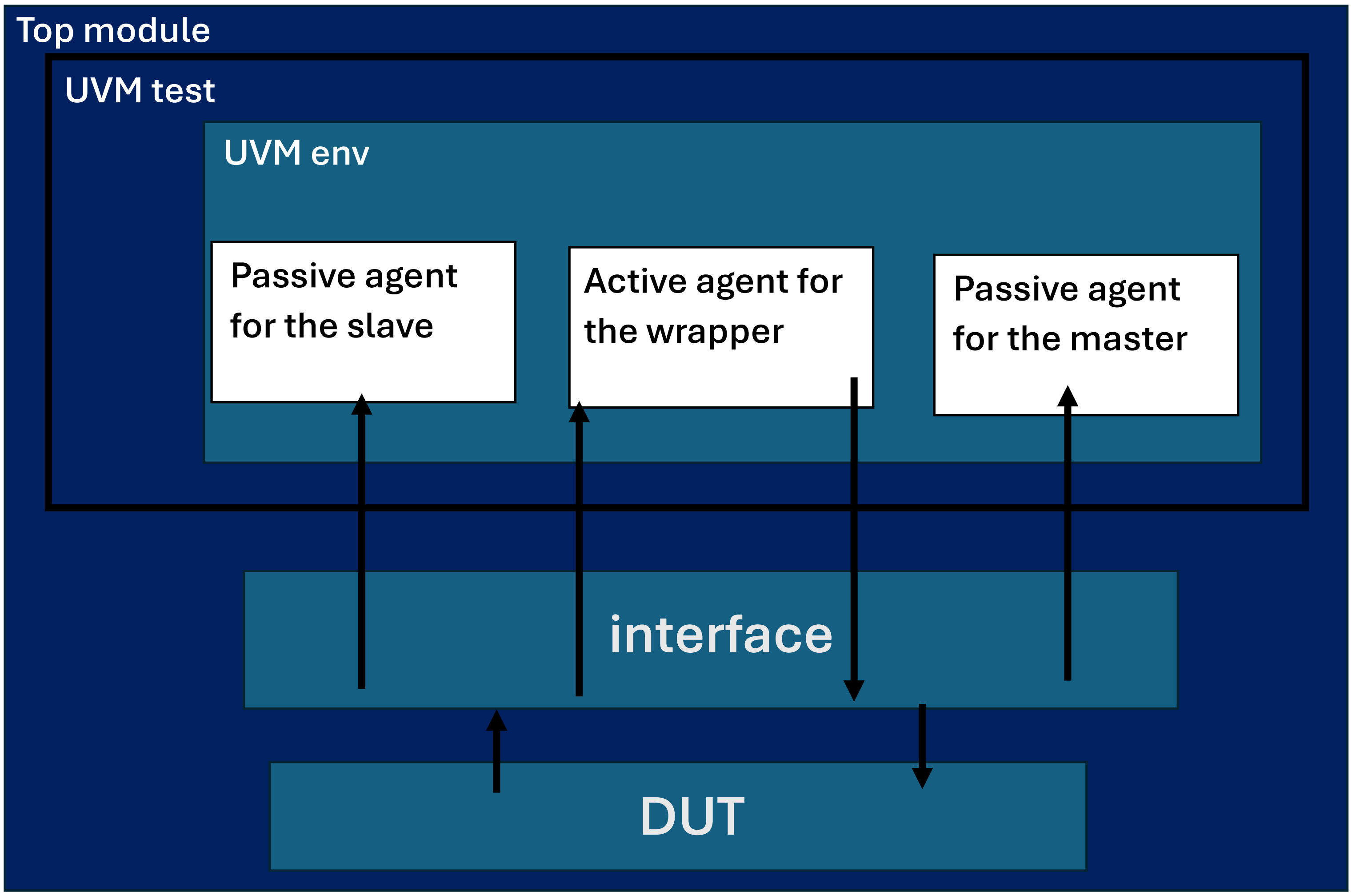
Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	6.234 ns	Worst Hold Slack (WHS):	0.133 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	161	Total Number of Endpoints:	161	Total Number of Endpoints:	140

All user specified timing constraints are met.

Figure 13:timing report at 100MHZ

11 Future Work

- Implementation of remain features (Master lock – Burst – size -error handling)
- Implementation of interconnection – slave interface
- Implementation of a full UVM environment (active – passive) agents to verify it


Top module
UVM test
UVM env
Passive agent for the slave
Active agent for the wrapper
Passive agent for the master
interface
DUT

12 Reference

[1] "university of Michigan," [Online]. Available: https://www.eecs.umich.edu/courses/eecs373/readings/ARM_IHI0033A_AMBA_AHB-Lite_SPEC.pdf.

13 Attachment Code

https://github.com/MarwanKRyad/AHB_LITE_-AMBA-

14 Real world applications

14.1 ARM Cortex-M3 Microcontrollers

Used in embedded systems like automotive control units and smart appliances. AHB is used to connect CPU, memory, and peripherals efficiently

14.2 Xilinx Zynq-7000 SoC

Combines an ARM processor with FPGA fabric. AHB connects the processor system to memory and hardware accelerators.