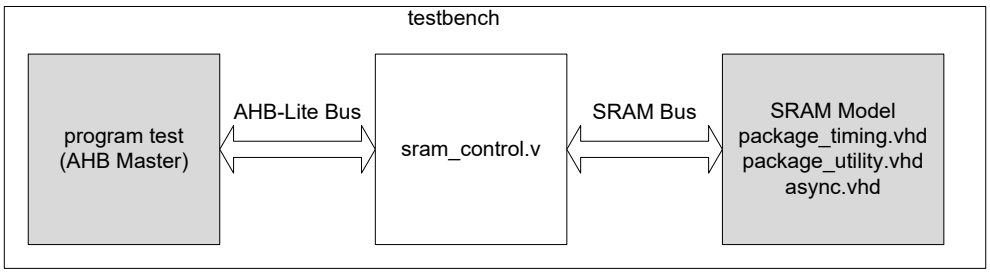
HW9

Chapter 6, Randomization Homework

The purpose of this homework is to use all your knowledge of verification and constraints to begin verifying a controller for an SRAM. Instructions to the controller come from an AHB Master, on an AHB-Lite bus. This is the stimulus. The sram controller is the DUT. An SRAM model and sram ontroller will be provided for you. The testbench will look like:



Requirements:

1. Create a class to encapsulate the AHB transactions. In this class constrain:
   a. The address (HADDR) to be in the lower 5 addresses and upper 5 addresses each with probability 40% and the other addresses with probability 20%.
   b. HTRANS to NONSEQ (HTRANS = 2'b10) and IDLE (HTRANS = 2'b00).
   c. The reset to assert 10% of the time.
   d. All other AHB signals are randomized but unconstrained.
2. Use an interface each for the AHB and SRAM busses
3. Your testbench shall create:
   a. 10 back to back random writes with the following constraints:
      i. HTRANS =NONSEQ
      ii. reset = 0
      iii. HADDR is restricted to be between 0 and 4 inclusive
   b. Display locations 0 to 4 of the memory array after the writes complete.
   c. 10 back to back random reads with the same constraints
   d. 10 random transactions with only the constraints specified in 1. Random variables HWDATA and HWRITE are unconstrained
4. Use a program for the stimulus, a package for the class, and a top module to instantiate the interfaces, and tie the DUT, SRAM, and program together.

Notes:

1. Be sure to drive the AHB bus correctly and on the positive edge of HCLK. Details follow in Appendix: The AHB-Lite Bus.

2. You do not have to make the testbench self checking, however, visually verify that the responses from the 10 back to back reads are correct.
3. The SRAM model is in VHDL. The compile order of VHDL is important. Be sure to compile package_timing.vhd, package_utility.vhd, and then async.vhd.
4. Be sure to set the simulator precision to at least 100ps or else the sram model will not function correctly.

Deliverables:

1. Code for the package, program, top level testbench, and interfaces
2. 3 waveforms on 3 separate pages clearly showing the write(waveform 1), read(waveform 2), random (waveform 3) AHB-Lite transactions and the SRAM bus.

Appendix: The AHB-Lite Bus

The AHB-Lite bus is in the following table. Directions are from the AHB Master block's perspective

Signal	Width	Direction	Description
HCLK	1	output	Clock
HADDR	21	output	Address
HWRITE	1	output	Write flag. 1=write, 0=read
HTRANS	2	output	The transaction type. 2'b00 = IDLE, 2'b10 = NONSEQ
HWDATA	8	output	Data to write
HRDATA	8	input	Data that was read

The AHB-Lite supports write and read transactions. A transaction begins with an address phase which contains the address and control signals, i.e. HWRITE and HTRANS. When HWRITE = 1 the transaction is a write. When HWRITE = 0 the transaction is a read. It is followed by a data phase which contains the data to be written or the data that was read. HTRANS indicates the transfer type. For the AHB-Lite there are 4 transaction types but we will only support two of them. When HTRANS = 2'b00 the bus is idle regardless of the value of HWRITE. When HTRANS=2'b10 the transfer type is NONSEQ, indicating a single transaction. Timing diagrams of a single write, single read, back-to-back write, and back-to-back read are in Figures 1-4 respectively.

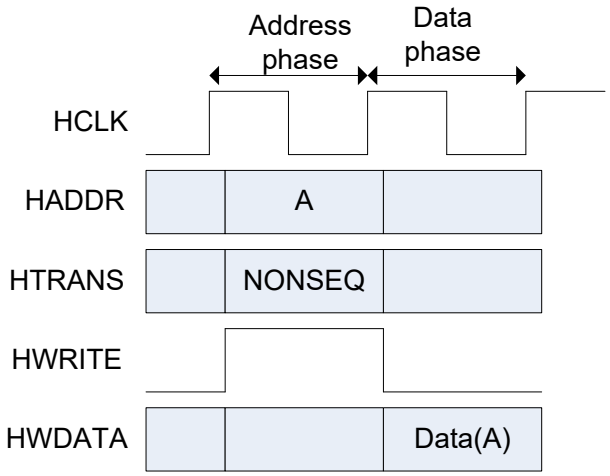


Figure 1: AHB-Lite Write Transaction

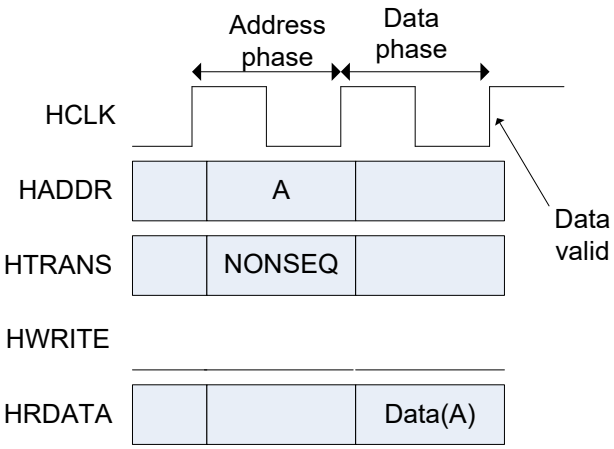


Figure 2: AHB-Lite Read Transaction

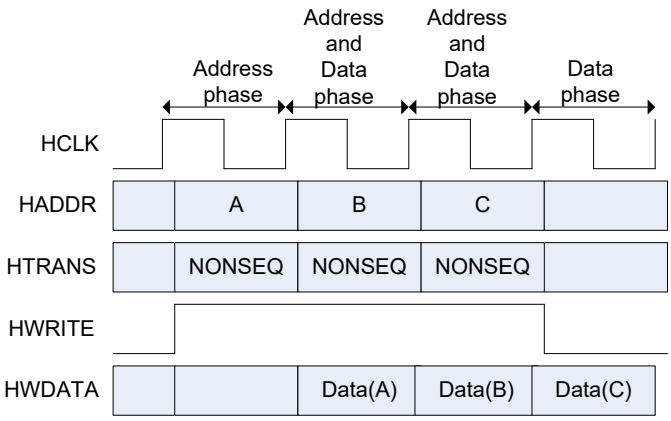


Figure 3: AHB Back-to-Back Writes

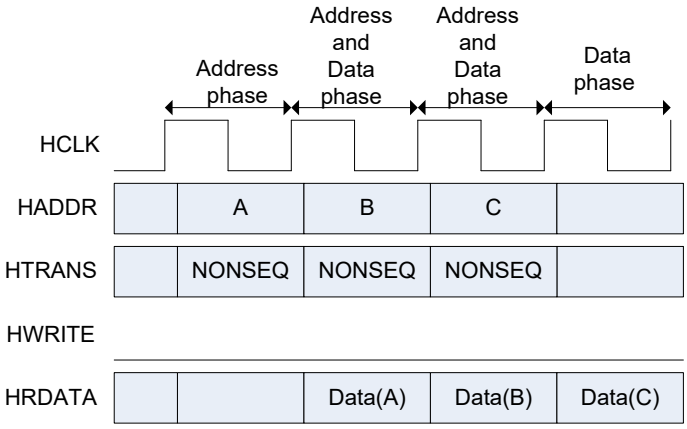


Figure 4: AHB Back-to-Back Reads