ECE 465: Digital System design
Fall 2021

Project 3: Design of a standardized Bus Interface

Introduction:

In large digital systems, various sub-components may communicate via a shared interface called a bus. To allow sub-components from different designers to communicate with each other, various bus architecture standards are developed in the industry. PCI, PCI express, AMBA and Avalon are some examples of bus architectures.
The sub-components or devices are usually divided into two classes based on the Master/Slave model of communication:

- Master: usually initiates and controls the communication process
- Slave: responds to the communication instructions from the master

Example of a typical Avalon system is shown in figure below.

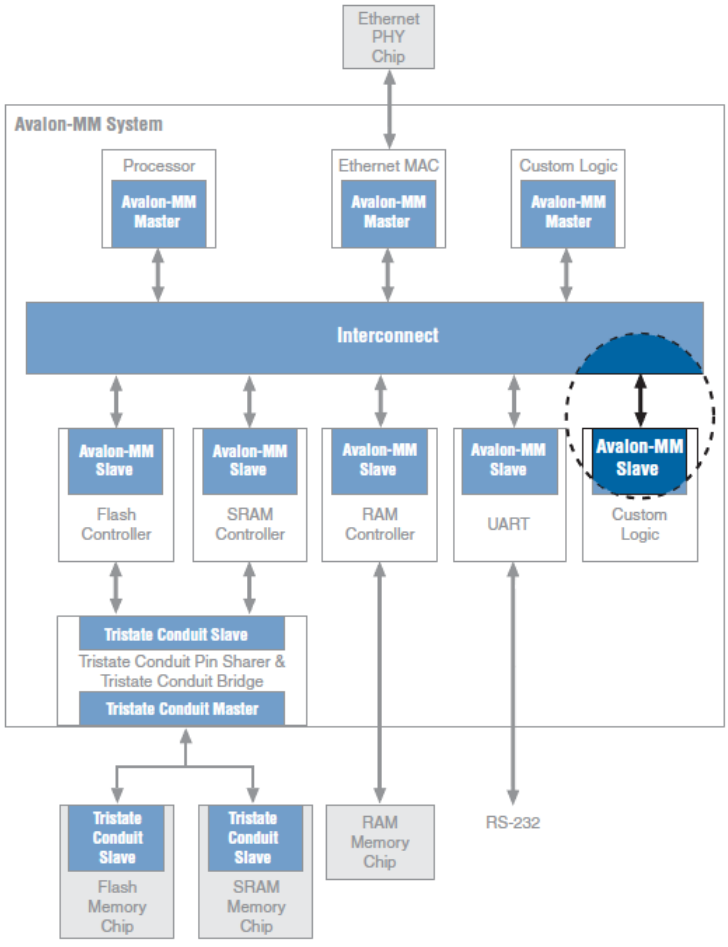


Figure 1: Avalon System

Project Description:

As part of this project, we will design a functional model of a 1 KB Random Access Memory (RAM) that has a standard Avalon slave interface. The block diagram for our design is given below:

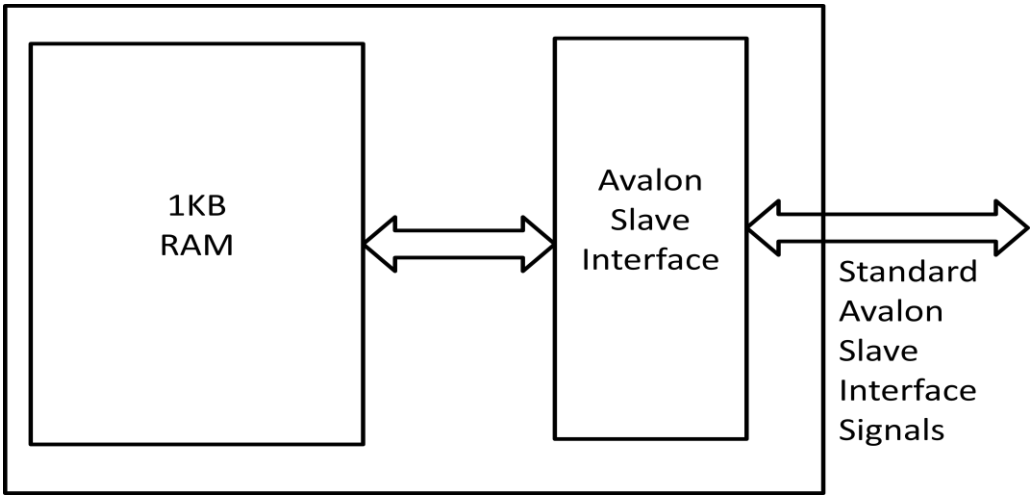


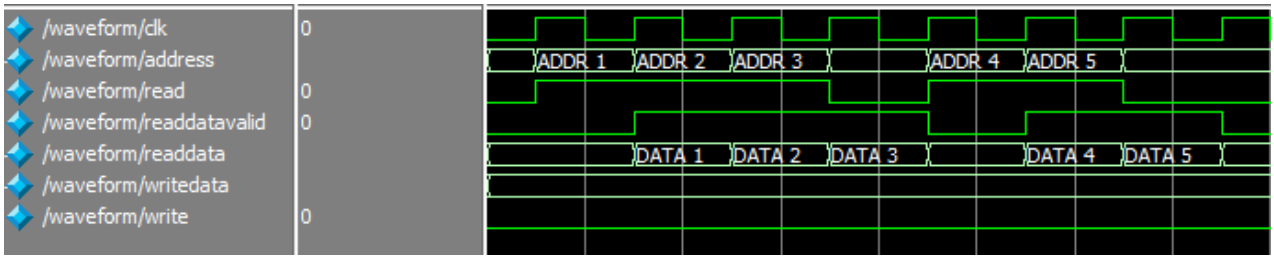
Figure 2: Block Diagram

Avalon Slave Interface Signals Description:

Signal	Width	Direction	Description
avl_clk	1	Input	Clock input
avl_rst_n	1	Input	Active low reset
avl_address	10	Input	Word address in the RAM For example, address= 0 selects the first word of the RAM and address 1 selects the second word of the RAM.
avl_read	1	Input	Asserted to indicate a read transfer.
avl_readdata	8	Output	The readdata driven from the slave to the master in response to a read transfer.
avl_readdatavalid	1	Output	Asserted by the slave to indicate that the readdata signal contains valid data in response to a previous read request
avl_writedata	8	Input	Data for write transfers
avl_write	1	Input	Asserted to indicate a write transfer
avl_burstcount	10	Input	Used by to indicate the number of transfers in each burst. Burstcount will always be a power of 2
avl_beginbursttransfer	1	Input	Asserted for the first cycle of a burst to indicate when a burst transfer is starting. This signal is deasserted after one cycle.

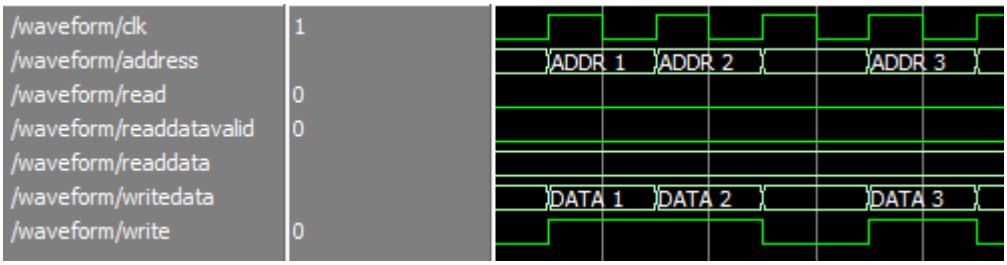
Operation:
Your design should support the following functionalities:

A simple read operation:



1. The master asserts *address* and *read*, initiating a read transfer.
2. The slave captures addr1, provides data1 and asserts *readdatavalid*.
3. The slave captures addr2 and provides the response data2 and asserts *readdatavalid*
4. Same process repeated for addr3, addr4 and addr5.

A simple write operation:

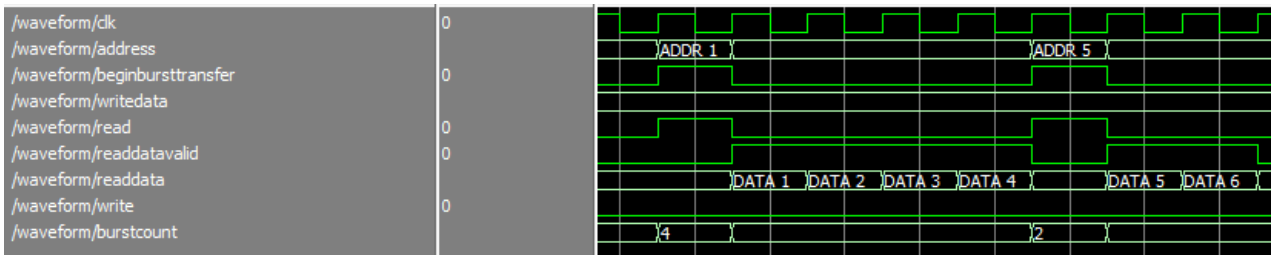


1. The master asserts *address*, *write* and places the data on the *writedata*.
2. The slave captures data1 and stores it at the addr1.
3. The slave captures data2 and stores it at the addr2.

Burst Operations:

A burst executes multiple transfers as a unit, rather than treating every word independently. Bursts may increase throughput for slave device.

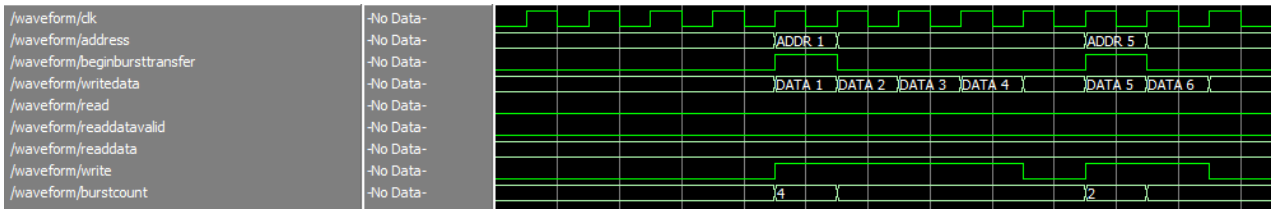
Read Burst operation:



1. Master asserts *beginbursttransfer*, *read* signal, *address* and *burstcount*.
2. The slave captures addr1, provides data1 and asserts *readdatavalid*
3. Slave provides data1 and asserts *readdatavalid*

4. Slave provides data2 and asserts *readdatavalid*
5. Slave provides data3 and asserts *readdatavalid*
6. Slave provides data4 and asserts *readdatavalid*

Write Burst operation:



1. Master asserts *beginbursttransfer*, *write*, *addres,burstcount* and *writedata*.
2. The slave captures data1 and stores it at the addr1.
3. The slave captures data2 and stores it at the addr2.
4. The slave captures data3 and stores it at the addr3.
5. The slave captures data4 and stores it at the addr4.

Submission guidelines:

- Please use exact same port names as defined above (It helps us in grading!)
- Submit the compressed version of Quartus project on blackboard.
  - The Project should also have a waveform file that demonstrates the working of your design
- A lab report (in hard form) should be submitted in class. It should have the following contents:
  - Design description using a block diagram
  - Snapshots of the RTL schematic
  - Snapshot of simulation of each of the operation. (Use timing simulation with clock period of 40 ns)
  - Conclusion
  - Verilog/VHDL code (You can use smaller font size to reduce the number of pages)

Reference:

Avalon Interface Specifications (Avalon® Interface Specifications (intel.com))