Serial Peripheral Interface

Description

Refer to the attached document “Introduction to SPI Interface” to learn about the details of the SPI.

Requirements:

Design and implement the following components of the SPI using verilog:

- Master
- Slave
- Integration of a Master with 3 Slaves in a Regular Multi-slave mode.
- Self-Checking Testbenches for the Master, Slave and Integration (see notes).

Write a report that discusses the following:

- The design process of the Master, Slave and Integration.
- The test cases and the reasons behind your choices.
- The simulation results (Screenshots from the waveform and the output).
- A comparison between the SPI, I^2C and UART.

Frequently Asked Questions:

- What data should be communicated between the master and the slave?
    - Both the master and the slave modules must have an interface to receive command from controllers (which will be the testbenches in our case). These commands could be “Hey Master, send this data to Slave 1” or “Hey Master, read data from Slave 2 and give it to me” or “Hey Slave 1, what data did the Master send you”, etc.
    - The controller should also be able to request the specific mode used to transmit the data over the SPI.
- What is the size of the data to be transmitted?
    - 8-bits.

Deliverables

- All the verilog files in the project.
- A text file with the team number and member names. It also must include the work distribution over the team members.
- An **individual** report by each team member containing all the content mentioned in the requirements. Note that each student has to write the design process of every module, even the ones they did not build themselves.

Grade Distribution

The total grade is calculated from 40 points. It will be distributed as follows:

- Simulation: 15 points
    - Compilation: 6 points (1 point for each module).
    - Simulation: 9 points (3 points for each test bench).
- Design: 10 points
    - Master: 3 points (2 for Master + 1 for Master Testbench).
    - Slave: 3 points (2 for Slave + 1 for Slave Testbench).
    - Integration: 4 points (1 for Integration + 3 for full test bench).
- Report: 15 points
    - Module Design Report: 6 points.
    - Test cases: 3 points.
    - Simulation Results: 1 point.
    - Search and Comparison with I2C and UART: 5 points.

Each team can include up to 4 students. Each member will be evaluated for their contributions and understanding individually. The delivery date is Tuesday June 2nd 2020 11:59 PM.

Notes

- A self checking test bench is a test-bench that automatically verifies the output of the module. So it should have some test vectors consisting of: test inputs and the corresponding expected outputs. At the end of the simulation, the testbench should automatically display whether the test was a success (all outputs match their corresponding expected outputs) or a failure (any output mismatched its corresponding expected output). If the test fails, it should report the failed test cases (test input, expected output and actual output) and the number of failed test cases. Self-checking testbenches are more common in hardware design since real applications tend to have lots of modules, so it is impractical to manually check the outputs of each test case in the project.