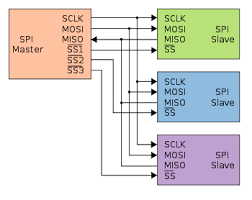
**Serial Peripheral Interface**

Advanced Logic Design



**Team Members:**

**1. Ahmed Alaa El-Sayed Arabi Zidan**

**2. Ahmed Ata**

**3. Mennatallah Hossam**

**4. Sarah El-Zayat**

**Supervised by:**

* **Dr. Ihab Talkhan**
* **Eng. Yahya Zakaria**
* **Eng. Mustafa Mahmoud**

**Part I: Master Module**

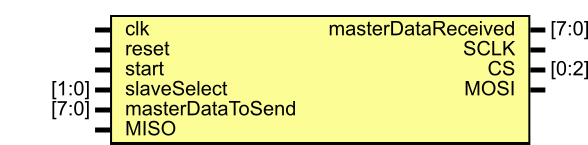


Figure 1: Master Prototype

**A. Ports:**

|  |  |  |
| --- | --- | --- |
| **Port Name** | **Type / Size** | **Description** |
| clk | Input | The synchronous clock generated from the Microcontroller feeding the master of SPI. |
| reset | Input | Resets the Register’s Data to 0  (Asynchronous reset). |
| start | Input | The bit which trigger the transmission and receiving process. |
| slaveSelect | Input [1:0] | The number of the slave to be connected to. |
| masterDataToSend | Input [7:0] | The 8-bits data we put in the Register to be sent in the transmission process. |
| MISO | Input | The bit received by the master from slave at each clock pulse. |
| masterDataRecieved | Output reg [7:0] | The 8-bits data which The Slave Receives at the end of receiving process. |
| SCLK | output | The Synchronous clock of The Master. |
| CS | Output [0:2] | Enable bit of The Transmission (Active-Low). |
| MOSI | Output reg | The bit transmitted to the slave from the master at each clock pulse. |

**B. Local and Unported Signals:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type / Size** | **Description** |
| counter | integer | A counter that counts the number of bets |
| flag | reg | A bit raised to high when the transmission and receiving process are terminated. |
| buffer | Reg [7:0] | - The Register inside the master  - A normal shift Register with MISO as its IL and MOSI as its IR. |

**C. Processes and The Data Flow:**

1) At the raising edge of start:

- The master read the slaveSelect.

- Sent the CS bit for each slave (in the same time)

- Initialize the masterDataRecieved by unknowns.

- Initialize the buffer by masterDataToSend.

- Initialize the counter by 1

- Initialize the flag by zero

1. At the rising edge of SCLK (shifting process):

- shifting the LSB in the buffer into the MISO.

1. At the falling edge of SCLK (Sampling process):

- shifting the MOSI bit into the Register from the left.

- updating the SlaveDataReceived to contain the same bits as the Register.

4) At the raising edge of flag:

- Both transmission and receiving process are terminated.

**PART II: Self-Checking Testbench for the Master Module**

**Part III: Slave Module**

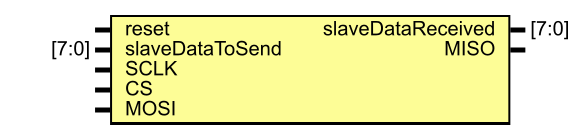


Figure 2 Slave

**A. Ports:**

|  |  |  |
| --- | --- | --- |
| **Port Name** | **Type / Size** | **Description** |
| reset | Input | Resets the Register’s Data to 0  (Asynchronous reset). |
| slaveDataToSend | Input [7:0] | The 8-bits data we put in the Register to send in the transmission process. |
| slaveDataReceived | output reg [7:0] | The 8-bits data which The Slave Receives at the end of transmission. |
| SCLK | Input | The Synchronous clock of The Master. |
| CS | Input | Enable bit of The Transmission (Active-Low). |
| MOSI | Input | The bit received by slave from the master at each clock pulse. |
| MISO | Output reg | The bit transmitted to the master from slave at each clock pulse. |

**B. Local and Unported Signals:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type / Size** | **Description** |
| Reg Data | reg [7:0] | - The Register inside The Slave  - A normal shift Register with MOSI as its IL and MISO as its IR. |

**C. Processes and The Data Flow:**

1) At the falling edge of CS:

- Indicates the start of the transmission.

- The Slave loads the data that will be transmitted into the register.

1. At the rising edge of CS:

- Indicates the end of the transmission

- Setting MISO to high-z.

1. At the rising edge of SCLK (shifting process):

- Shifting the LSB in the register into the MISO.

1. At the falling edge of SCLK (Sampling process):

- Shifting the MOSI bit into the Register from the left.

- Updating the SlaveDataReceived to contain the same bits as the Register.

**Part IV: Self-Checking Testbench for the Slave Module**

**A. Local and Unported Signals:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| reset | reg | Input to Slave |
| SCLK | reg | Input to Slave |
| CS | reg | Input to Slave |
| MOSI | reg | Input to Slave |
| MISO | wire | output from Slave |
| slaveDataToSend | reg [7:0] | Input to Slave |
| slaveSataReceived | wire [7:0] | output from slave module |
| ExpectedMasterDataToReceive | reg [7:0] | The 8-bits data which the Master is expected to receive at the end of transmission |
| testcase\_MasterDataToSend | wire [7:0] array [1:4] | The 4-element array of 8-bit data which is expected to be sent by Master in transmission |
| testcase\_SlaveDataToSend | wire [7:0] array [1:4] | The 4-element array of 8-bit data which is an input to Slave in transmission (slaveDataToSend) |

**B. Processes and The Data Flow:**

1. Initializing Local and unported signals:

- initialize CS with 1 then 0 (triggering falling edge of CS).

- initialize input data loaded to Slave.

1. Set CS =1:

- Triggering rising edge of CS.

- Checking if data received and sent by Slave is as Expected and display an indicative message.

1. At the rising edge of SCLK (mimicking the shifting process of Master):

- Setting MOSI with bit of testcase\_MasterDataToSend.

- Increase i by 1.

1. At the falling edge of SCLK (sampling process):

- Updating the ExpectedMasterDataToReceive to contain all the bits shifted by Slave and sampled by Master.

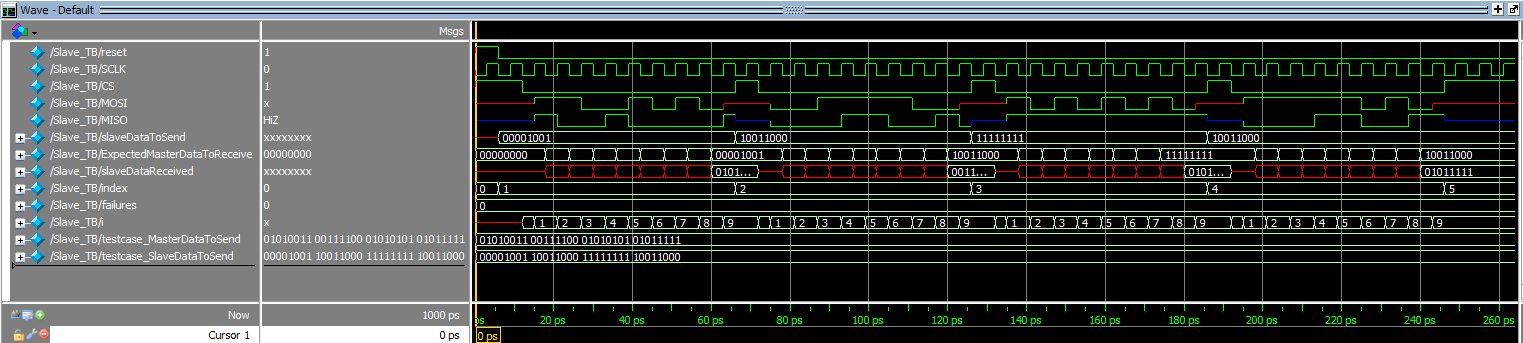
**C. Simulation of Slave:**

Figure 3: Wave form

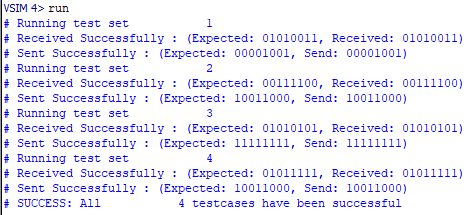


Figure 4: Displayed Messages