# Verification of a SPI Protocol

(Serial Peripheral Interface)

# Description:

The spi_serializer module is responsible for serializing data from a FIFO buffer into a SPI-compatible serial data. The module operates as follows:

State Machine:

- A finite state machine (FSM) controls the serialization process. The FSM progresses through different states to manage the serialization process. FSM states are defined as follows:
- IDLE: Initial state where the module waits for the condition to start serialization (in this case, the FIFO being full).
- LOAD: State where the module loads data from the FIFO.
- SHIFT: State where the module shifts out bits serially.
- COMPLETE: State indicating the completion of the serialization process.

State Transition Logic:

- The always_comb block inside the module determines the next state based on the current state and input conditions. For example:
- When in the IDLE state, if the FIFO is full and not empty, the module transitions to the LOAD state to begin the serialization process.
- In the LOAD state, data is loaded from the FIFO into a shift register.
- In the SHIFT state, the module shifts out the data from the shift register serially, synchronizing with the SPI clock (sclk).
- After shifting out all bits, the module transitions to the COMPLETE state.

Data Serialization:

- During the SHIFT state, the module shifts out the data from the shift register bit by bit, synchronized with the SPI clock (sclk). The most significant bit (MSB) of the data is typically shifted out first. The mosi output provides the serialized data stream compatible with the SPI protocol.

Clock Generation:

- The sclk output generates the SPI clock signal. The clock signal is typically generated based on the system clock (clk) and may be controlled by parameters such as clock frequency, polarity, and phase.

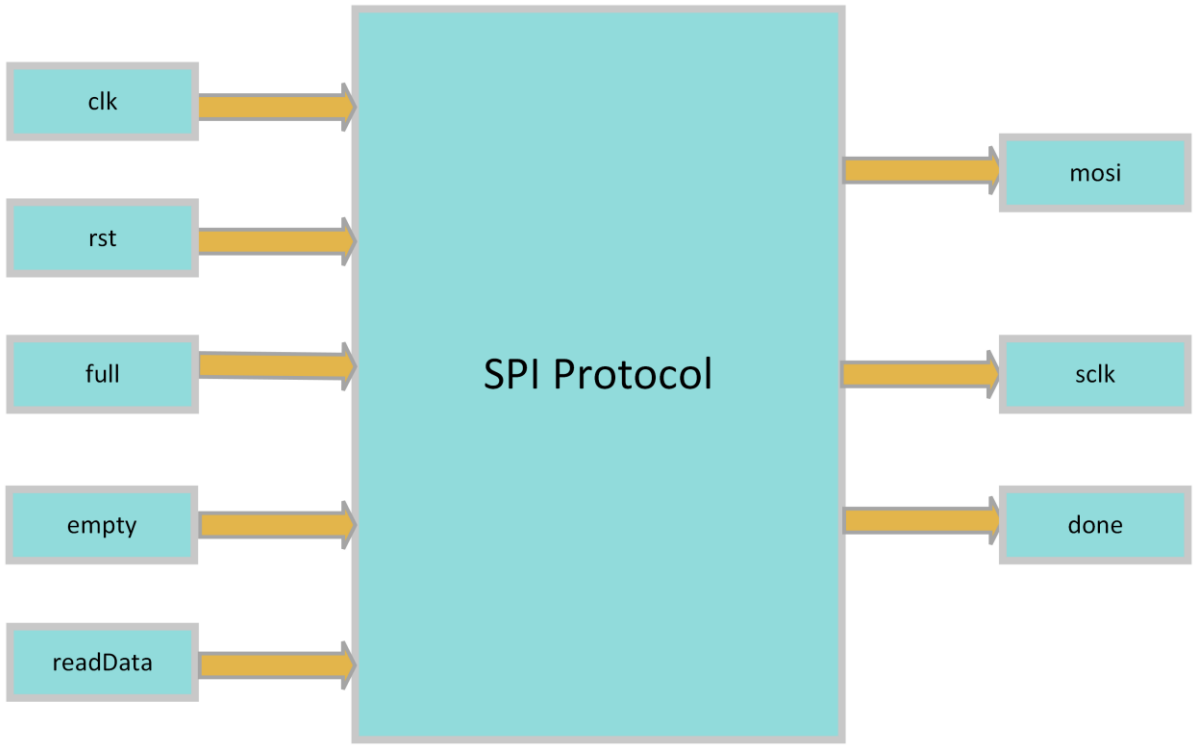
Done Signal:

- The done signal indicates the completion of the serialization process. It is set high when all data bits have been shifted out and the serialization process is complete.

Reset Logic:

- The module includes logic to handle resets (rst) to ensure proper initialization of internal states and signals.

## Block Diagram:




Input Definitions

- clk : Clock signal to synchronize the operations.
- rst : Reset signal to initialize the module.
- full: Signal indicating that the FIFO is full, triggering the start of serialization.
- empty: Signal indicating that the FIFO is empty, preventing serialization when active.
- readData ([DataWidth-1:0]): Data to be serialized.

Output Definitions.

- sclk : SPI clock signal.
- mosi : SPI Master Out Slave In data line (serial data output).
- done: Signal indicating that the serialization process is complete.

Internal Components:

- State Variables: state and next_state used to track the state of the finite state machine.
- Shift Register: shift_reg used to hold the data being serialized before being sent out.

- Bit Counter: bit_counter used to keep track of the number of bits shifted out during serialization.

# Formal Verification:

## Properties

| N# | Name | Category | Status |
|---|---|---|---|
| | | | |

**Verify the correct write operation.**

Properties defined to cover this spec:

Assumptions:

Assertions:

Covers:

**Verify the correct read operation:**

Properties defined to cover this spec:

Assumptions:

Assertions:

Covers

**Verify the correct full signal:**

Assertions:

Covers:

**Verify the correct empty signal:**

Assertions:

Covers:

**Verify the correct simultaneous write and read operation:**

Properties defined to cover this spec:

Covers:

# Results:

# Bug fixing:

Solution: