SPI Master–Slave RTL Design (4-Mode Configurable)

Objective

Design Verilog RTL modules for a **configurable SPI Master and SPI Slave**, supporting all **four SPI modes** determined by **CPOL** (clock polarity) and **CPHA** (clock phase).

Each module must follow the step-by-step flow-chart behavior given below and implement it as a clean **Finite State Machine (FSM)**.

SPI MASTER

Functional Overview

Implements full-duplex serial communication with one or more slaves over:

- `MOSI` (Master Out Slave In)
- `MISO` (Master In Slave Out)
- `SCLK` (Serial Clock)
- `CS_N` (Chip Select, active-low)

Supports:

- 4 SPI modes:

Mode	CPOL	CPHA	Description
0	0	0	Sample on rising edge, shift on falling edge
1	0	1	Shift on rising, sample on falling
2	1	0	Sample on falling, shift on rising
3	1	1	Shift on falling, sample on rising

- Configurable bit length (default: 8 bits)

- MSB-first or LSB-first shifting

FSM Design (based on flow chart)

State	Description / Actions
IDLE	CS_N = 1 (inactive); `busy=0`. Wait for `start` = 1.
ASSERT_CS	Pull CS_N = 0, optionally delay 1 cycle. Initialize bit counter.
LOAD	Load `tx_reg` with transmit data; reset bit counter = `N-1`.
SHIFT	Generate SCLK edges as per CPHA/CPOL. On each active edge:– Drive MOSI (MSB/LSB)– Sample MISO– Shift register– Decrement counter
CHECK_DONE	If bit_counter ≥ 0 → go back to SHIFT; else → next.
DEASSERT_CS	Pull CS_N = 1 after optional delay.
DONE	Latch received data into `rx_reg`, clear busy, pulse `done=1`. Return to IDLE.

Expected Inputs

Signal	Description
`clk_sys`	System clock
`rst`	Reset
`start`	Start transfer pulse
`data_in[N-1:0]`	Data to transmit
`cpol`	Clock polarity select
`cpha`	Clock phase select
`msb_first`	Bit-order select
`bit_len`	Configurable number of bits (default = 8)
`miso`	Serial data from slave

Expected Outputs

Signal	Description
`mosi`	Master data out

Signal	Description
`sclk`	Serial clock generated by master
`cs_n`	Active-low chip select
`data_out[N-1:0]`	Received data
`busy`	High during transmission
`done`	Single-cycle pulse after completion

Timing Notes

- For CPHA = 0, the **first data bit** must be ready before the first SCLK edge.
- For CPHA = 1, the first data bit is launched on the first edge.
- SCLK toggles according to CPOL:
  - CPOL = 0 → idle low, active high.
  - CPOL = 1 → idle high, active low.

SPI SLAVE

Functional Overview

Responds to a master transaction when its **CS_N is low**.

- Shifts data on SCLK edges based on CPOL/CPHA.
- Receives MOSI data and transmits MISO data simultaneously.
- Uses same bit length and bit order as master.

FSM Design

State	Description / Actions
IDLE	Wait for CS_N = 1 → 0 (slave selected).
LOAD	Load `tx_reg` with outgoing data; clear `bit_cnt`.
SHIFT_RXTX	For each SCLK edge:– Sample MOSI at correct edge.– Shift out next MISO bit.– Decrement counter.

State	Description / Actions
CHECK_DONE	If counter ≥ 0 → continue shifting; else → next.
LATCH_DATA	Store received data, raise `rx_valid=1`.
WAIT_DEASSERT	Wait for CS_N = 1 (deselected). Go to IDLE.

Expected Inputs

Signal	Description
`sclk`	Clock from master
`cs_n`	Chip select from master
`mosi`	Serial data from master
`rst`	Reset
`cpol`	Clock polarity (matches master)
`cpha`	Clock phase (matches master)

Expected Outputs

Signal	Description
`miso`	Data back to master
`rx_data[N-1:0]`	Received byte
`rx_valid`	Pulse high when new byte received

Integration Requirements

- Connect **Master ↔ Slave** signals:
  - `MOSI` → `MOSI`, `MISO` ← `MISO`, `SCLK` → `SCLK`, `CS_N` → `CS_N`.
- Verify all 4 SPI modes by sweeping CPOL/CPHA.
- Simulate single and multiple transfers.
- Waveforms must show correct bit alignment and edge relationships.

Deliverables

1. `spi_master.v` — Master RTL (FSM per flow chart)
2. `spi_slave.v` — Slave RTL (FSM per flow chart)
3. `spi_master_slave_tb.v` — Testbench with connected master–slave pair
    - Test all four (CPOL, CPHA) combinations
    - Show MOSI/MISO/SCLK timing
    - Validate full-duplex exchange