SPI Telemetry Link with Deterministic Busy/Done Handshaking: Byte-Level Mixed-Signal Verification in eSim

Abstract:

This project demonstrates the design, simulation, and byte-level verification of an SPI Mode 0 (CPOL=0, CPHA=0) telemetry link (spi_telemetry_v1) using eSim. The master is written in Verilog and converted to an Ngspice-compatible netlist via NgVeri, and the design integrates this digital core with analog pulse sources and a gate-level slave model to create a mixed-signal verification environment.

Rather than bridging the 8-bit parallel data bus as a single vector — a known source of an unresolved "zero length vector / cannot be both analog and digital" parsing error in eSim/Ngspice mixed-signal bridging — every serial control and data line (sclk, cs_n, mosi, miso, busy, done) is bridged individually. This allows direct, byte-level confirmation of data transfer in both directions, not just control-line timing. The simulation results confirm the SPI Master correctly transmits the 8-bit value 0xB6 (MSB-first) on MOSI, while an async-presettable D-flip-flop slave register correctly echoes back the known byte 0xA5 on MISO. The project also documents a real debugging episode — a miswired asynchronous Set pin on one flip-flop stage that broke the MISO shift chain — and the waveform-based methodology used to isolate and resolve it.

1. Theory

The Serial Peripheral Interface (SPI) is a synchronous serial communication protocol used for short-distance communication, primarily in embedded systems. It operates in a Master-Slave architecture where the Master controls the clock (SCLK) and initiates data transfer.

In this project, the system is divided into two functional blocks:

SPI Master (spi_telemetry_v1): Handles parallel-to-serial conversion. It accepts an 8-bit parallel input (tx_data) and transmits it serially via the MOSI line, while driving SCLK, active-low chip select (cs_n), and busy/done handshake flags.

SPI Slave (gate-level D-flip-flop shift register): An async-presettable 8-stage shift register that loads a known byte on reset and echoes it back serially on MISO, synchronized to a gated clock derived from SCLK and chip select.

1.1 Protocol Timing Requirements

For successful transmission, the following timing constraints must be met:

1. **Reset Guard Time:** The system must be held in reset for a defined duration at startup to clear internal registers and asynchronously preset the slave to its known byte.
2. **Start Delay:** The Start signal must not be asserted until the reset sequence is fully complete, avoiding an undefined transmission state.
3. **Clock Stability:** SCLK must be stable and continuous throughout the 8-cycle data transmission window, and the gated slave clock (derived from SCLK AND the inverted chip-select) must track it without glitching.

2. Circuit Diagram

The mixed-signal schematic was designed in eSim (KiCad). Digital stimulus (clk, rst_n, start, tx_data) is generated using DC/pulse voltage sources and converted to digital logic levels via adc_bridge components. Every SPI signal produced by the master and the slave register is individually converted back to an analog trace using dac_bridge components for Ngspice transient analysis — avoiding the parallel-bus vector-bridging limitation described in Section 3.

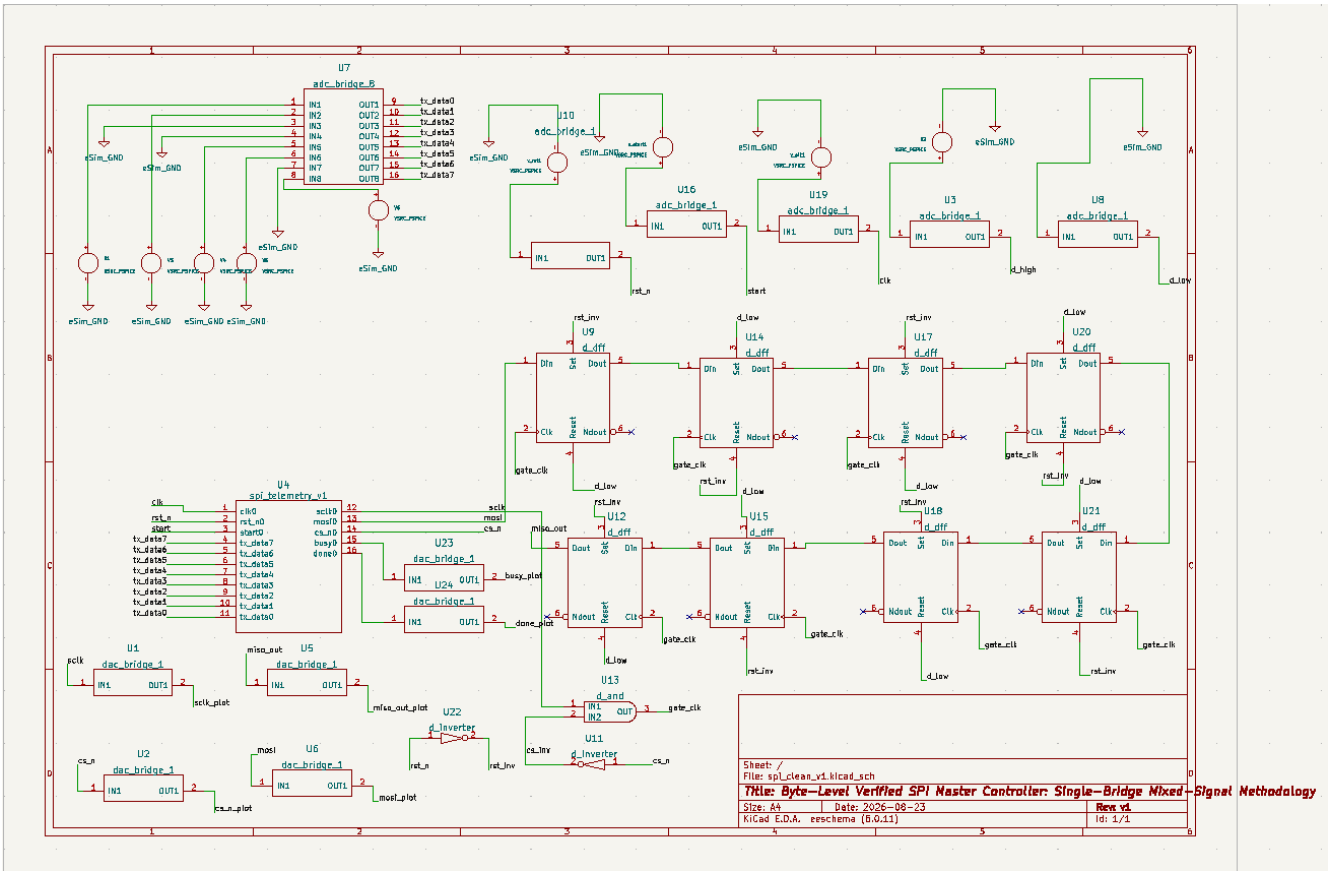

Figure 1: Complete mixed-signal SPI master–slave testbench schematic. U4 (spi_telemetry_v1) is the NgVeri-converted master core; U9–U21 form the async-presettable D-flip-flop MISO slave register; each control/data line is bridged individually rather than as a parallel vector.

3. Simulation Challenges and Resolution

A critical aspect of this project was overcoming known limitations of mixed-signal simulation within the Ngspice environment, and resolving a genuine wiring defect discovered during verification.

3.1 Avoiding the “Zero Length Vector” Bridging Error

Bridging an 8-bit parallel data bus directly to a single analog plotting node is a known failure mode in eSim/Ngspice: the simulator cannot resolve the vector as simultaneously analog and digital, producing a “node cannot be both analog and digital” / “zero length vector” parse error and preventing the circuit from simulating at all. This project’s bridging strategy was designed from the outset to avoid this failure mode entirely: each SPI signal is bridged individually as a single-bit node (via one dac_bridge_1 or adc_bridge_1 per signal, or adc_bridge_8 for the 8-bit tx_data input only, which is never plotted as a bridged vector). This allowed full transient simulation and byte-level waveform verification without ever encountering the parsing error.

3.2 The MISO Shift-Register Set-Pin Defect

During verification, the MISO output was observed to rise once (consistent with the async preset) and then remain flat for the remainder of the transaction, despite SCLK toggling continuously. Systematic waveform-based isolation — confirming cs_n and rst_n behaved correctly, then inspecting the shift-register chain stage by stage — traced the fault to a single D-flip-flop (U18) whose asynchronous Set pin was marked as a no-connect despite a wire being routed to it from the preset signal, leaving that stage unable to shift regardless of its Din or Clk inputs and freezing every downstream stage.

The fix was to remove the erroneous no-connect flag and correctly wire the Set pin to the preset signal, matching the other seven flip-flop stages. Re-running the transient simulation after this correction produced the full, correctly shifting 8-bit MISO waveform documented in Section 4.

4. Simulation Results

The transient analysis was performed with a 10 ns timestep over a 100 µs window. Results were visualized using eSim's Python plotter.

4.1 Full Transient Analysis

Figure 3 displays the full simulation window with all six bridged signals overlaid: busy, cs_n, done, miso_out, mosi, and sclk. SCLK toggles continuously and only during the interval that cs_n is held low and busy is high, confirming the master correctly gates the clock to the active transaction window.

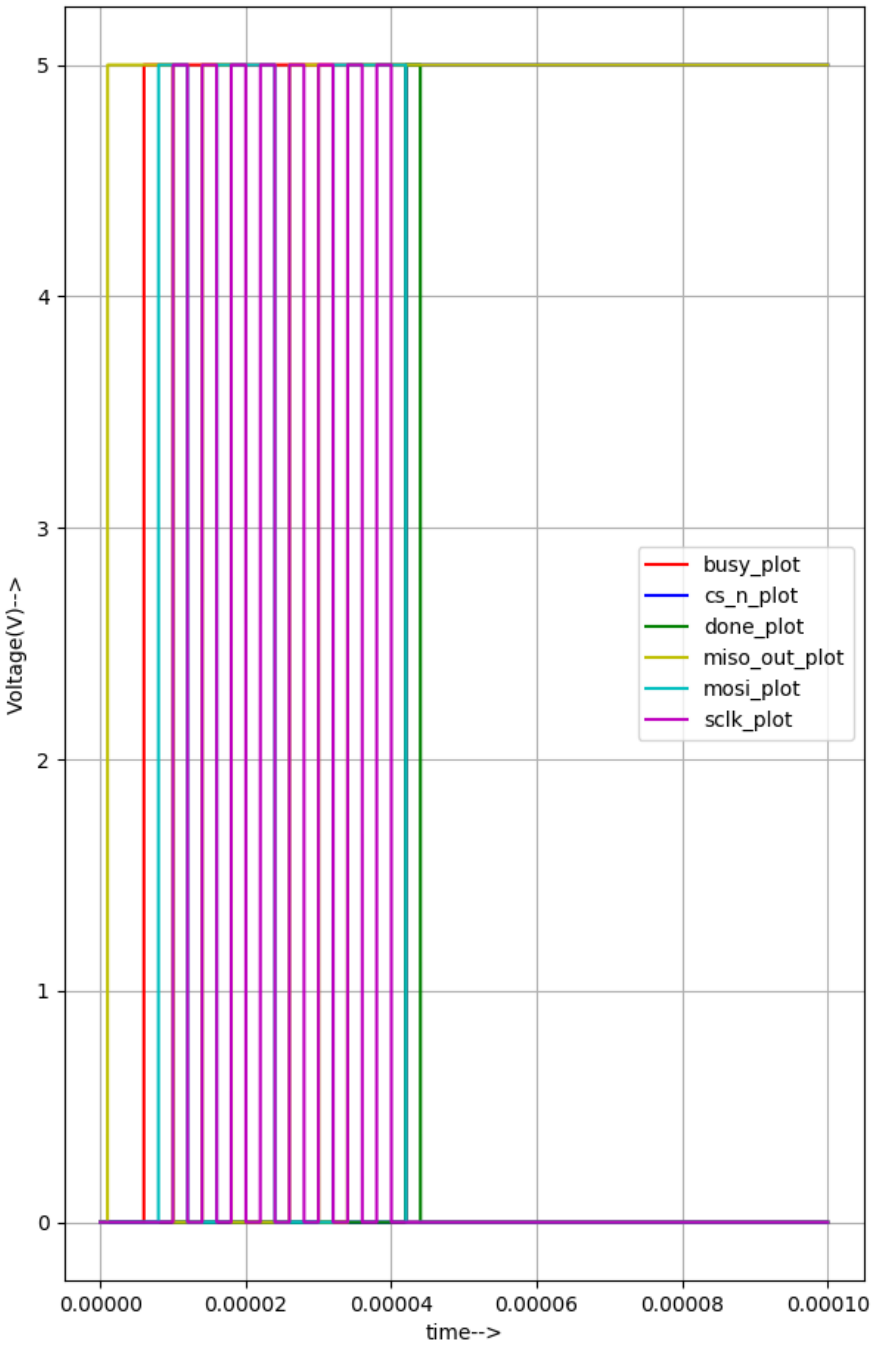

Figure 3: Full transient window — busy, cs_n, done, miso_out, mosi, and sclk.

4.2 Timing Verification (Zoomed Analysis)

To verify the end-of-transaction handshake precisely, the plot was zoomed to the completion instant (Figure 4).

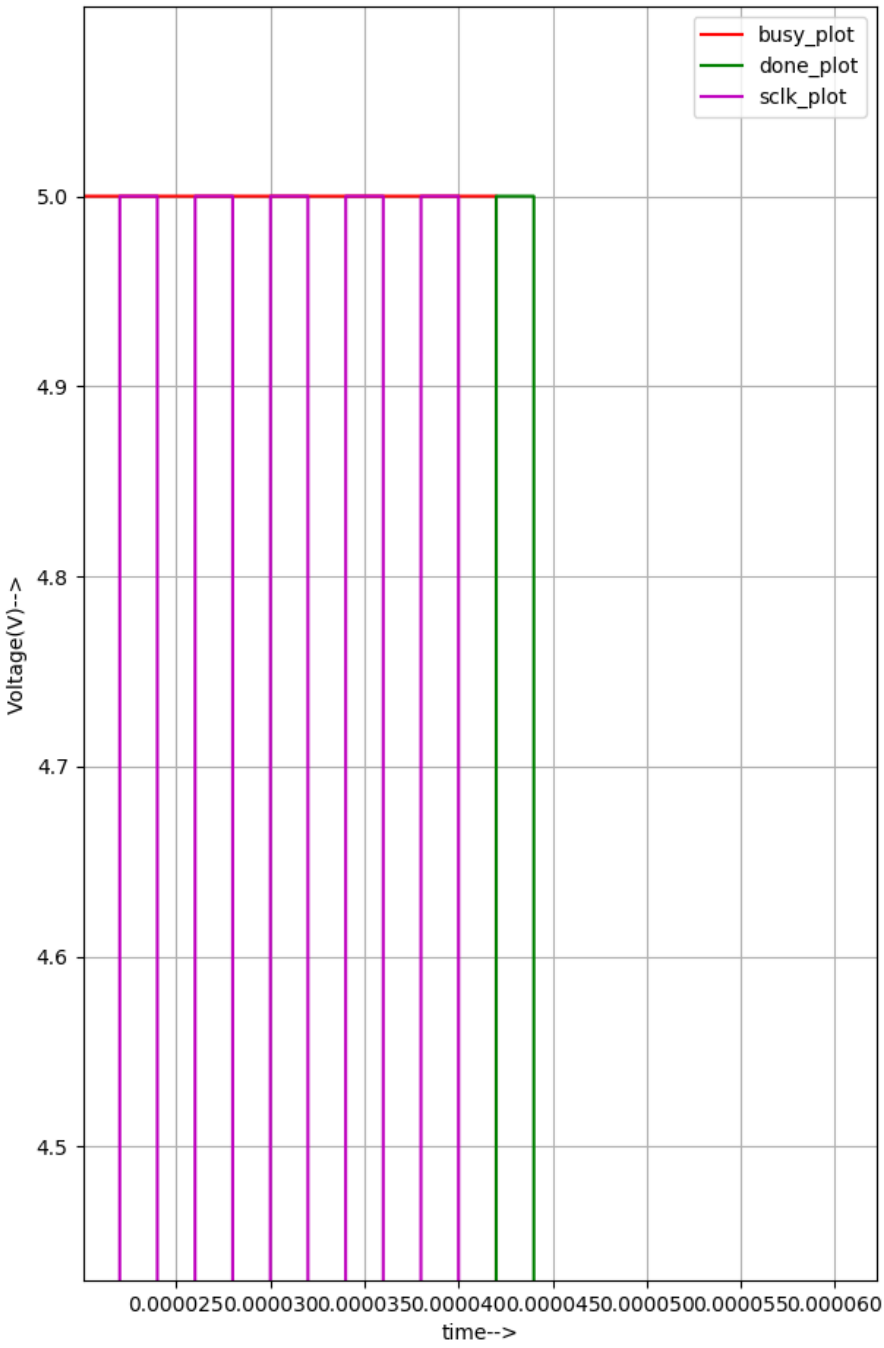

Figure 4: Zoomed timing verification. sclk toggles continuously through the transfer; busy remains high for the full duration and falls at the exact instant the sclk burst ends; done pulses once at that same instant, confirming correct handshake timing with no observable skew between the three signals.

Technical Specifications & Verification Methodology

1. SPI Protocol Configuration

- **SPI Mode:** Mode 0 (CPOL = 0, CPHA = 0)

- **Clock Polarity (CPOL):** 0 — SCLK idles Low
- **Clock Phase (CPHA):** 0 — data sampled on the rising edge, shifted on the falling edge
- **Bit Order:** MSB First
- **Data Frame Size:** 8 bits (1 byte per transaction)

2. Input/Output Signal Description

- **MOSI (Master Out Slave In):** Carries the 8-bit tx_data byte, individually set via 8 discrete pulse sources (pulse(5 5) for a 1-bit, tied to eSim_GND for a 0-bit). The byte transmitted in this simulation is **0xB6** (1011 0110).
- **MISO (Master In Slave Out):** Driven by the 8-stage D-flip-flop slave register, asynchronously presettable via the inverted reset signal to a known byte, **0xA5** (1010 0101), which is shifted out MSB-first synchronized to the gated clock.
- **SCLK (Serial Clock):** Generated by the Master; gated at the slave via gate_clk = sclk AND (NOT cs_n) so the slave register only shifts while selected.
- **CS_N (Chip Select):** Active-Low. The Master drives this Low to select the slave and initiate the frame, returning High once the transaction completes.
- **BUSY / DONE:** busy remains High for the full duration of the transaction; done pulses once, exactly as busy falls, signaling transaction completion.

3. Operational Workflow

1. **Idle State:** SCLK is Low, CS_N is High, busy and done are Low.
2. **Initialization:** The system receives a reset pulse, clearing the master's internal state and asynchronously presetting the slave register to 0xA5.
3. **Start Condition:** The Master receives a Start signal after the reset guard time elapses, and pulls CS_N Low to select the slave.
4. **Data Transfer:** Over 8 SCLK cycles, the Master shifts tx_data (0xB6) out on MOSI while the gated slave register simultaneously shifts its preset byte (0xA5) out on MISO.
5. **End of Transmission:** After the 8th bit, the Master stops the clock, pulls CS_N High, drops busy, and pulses done once to signal completion.

4. Verification Methodology & Waveform Analysis

Unlike a verification approach limited to confirming control-line timing alone, this methodology confirms the transferred data itself. Figure 5 isolates mosi and miso_out on a shared timebase.

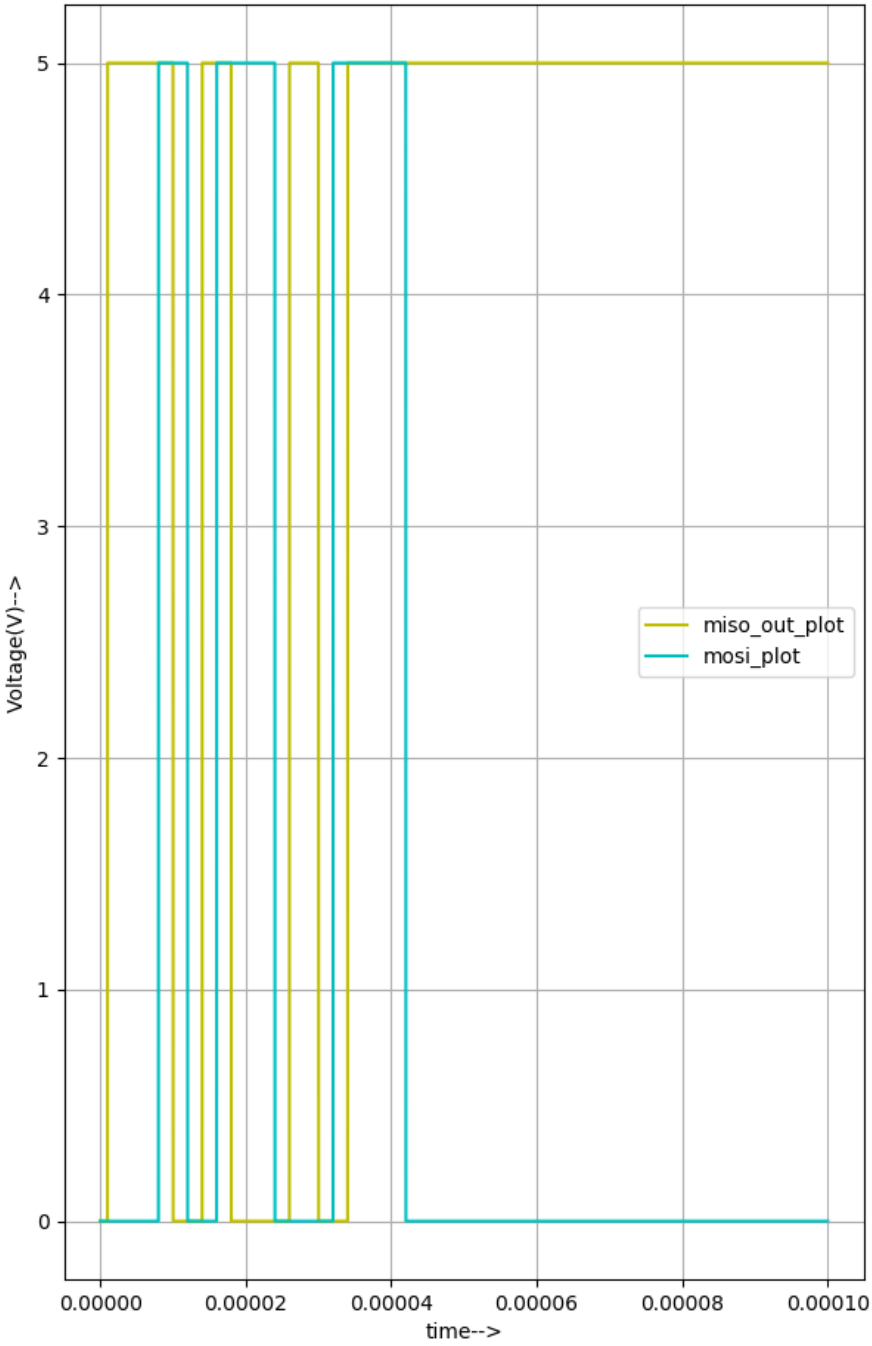

Figure 5: mosi and miso_out isolated on a shared timebase. mosi shifts out 0xB6 master-to-slave; miso_out shifts out 0xA5 slave-to-master, MSB-first, in full duplex over the same 8 sclk cycles. Both signals settle to their final bit value and hold once the transfer completes, consistent with the master halting sclk and de-asserting cs_n.

Observed Results: The synchronization between busy, cs_n, done, and sclk (Figure 4) confirms the handshake completes with no timing skew between signals. The mosi/miso_out comparison (Figure 5) confirms full-duplex, byte-accurate, MSB-first data transfer in both directions — the master's transmitted byte (0xB6) and the slave's known preset byte (0xA5) are both directly recoverable from the captured transient waveforms, rather than inferred from control-signal timing alone.

5. Conclusions

The SPI telemetry link was successfully designed, debugged, and verified in eSim. By bridging each control and data signal individually rather than as a parallel vector, the project avoided a known Ngspice mixed-signal bridging limitation and achieved full byte-level verification of a full-duplex SPI Mode 0 transaction — confirming not only correct control-line timing (busy, done, cs_n, sclk) but also the exact transmitted (0xB6) and received (0xA5) data bytes. The project additionally documents a genuine root-cause hardware debugging episode, in which a single miswired flip-flop Set pin was isolated through systematic waveform analysis and corrected, restoring full MISO functionality.

6. References

1 eSim User Manual, FOSSEE, IIT Bombay.

2 “SPI Protocol Fundamentals,” Analog Devices Technical Articles.

3 Ngspice User Manual, Section 12: Mixed-Mode Simulation.

4 NgVeri: Verilog-to-Ngspice Conversion Documentation, FOSSEE, IIT Bombay.

5 https://spoken-tutorial.org/tutorial-search/?search_foss=eSim&search_language=English