


# Computation II: embedded system design (5EIB0)


[Dashboard](#) / [My courses](#) / [5EIB0](#) / [Resit Exam 2024-07-03](#) / [Part 2f - 2024-07-03 SPI Subordinate Module \(6pts\)](#)

Description

[Submission view](#)

 **Available from:** Wednesday, 3 July 2024, 6:00 PM

 **Due date:** Wednesday, 3 July 2024, 9:00 PM

 **Requested files:** spi\_subordinate.v ( [Download](#))

**Type of work:**  Individual work

Serial Peripheral Interface (SPI) is a commonly used communication protocol in embedded systems. The communication interface of a subordinate SPI module typically requires four signals:

1. Chip Select (cs) input
2. Serial Clock (sclk) input
3. Manager Out, Subordinate In (MOSI) input
4. Manager In, Subordinate Out (MISO) output

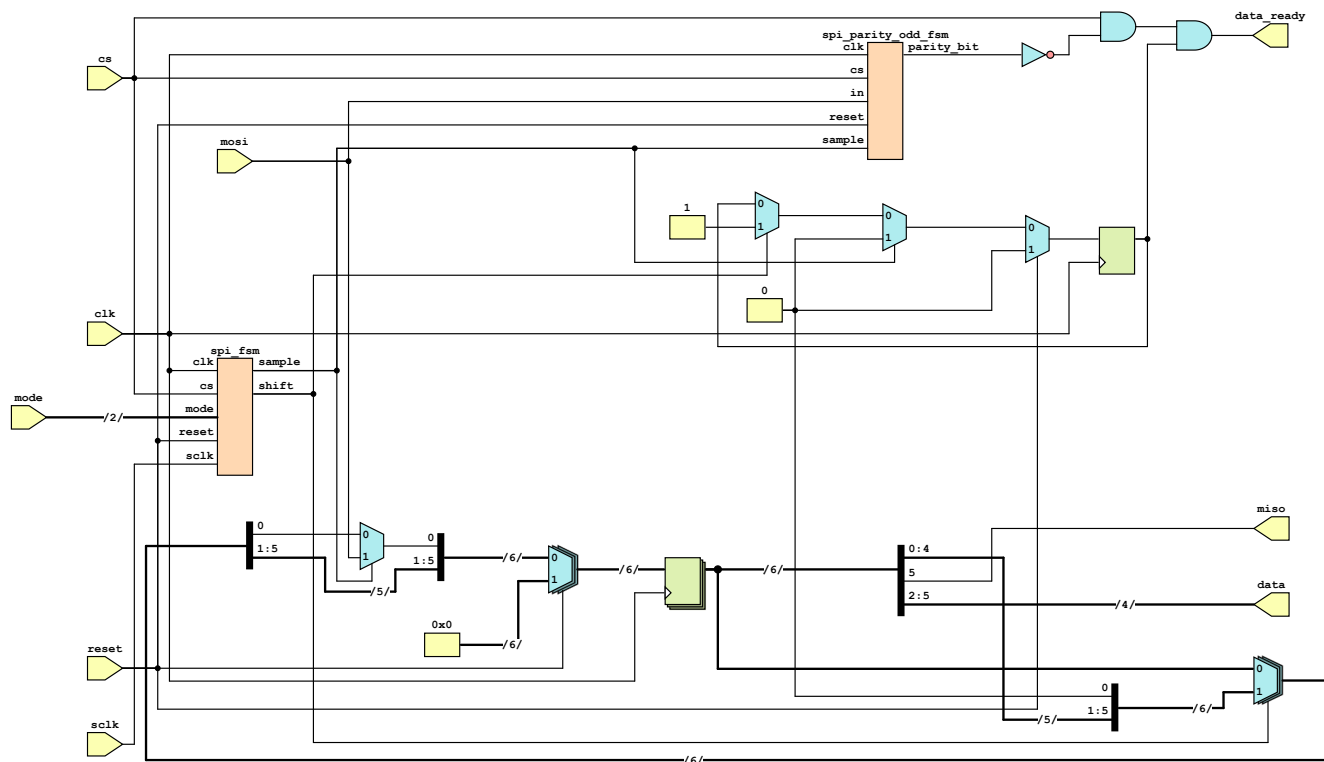
Serial data is communicated to the subordinate SPI module using the MOSI signal and from the subordinate SPI module using the MISO signal. The data is stored in a shift register with most significant bit (MSB) shifted onto the MOSI signal before the received bit on the MISO line is sampled into the least significant bit (LSB) of the shift register. This is regulated using the cs and sclk signals. An SPI transaction is initiated by the master by pulling the cs signal low. The cs signal remains low throughout the transaction. When the MSB is shifted onto the MOSI and the MISO sampled onto the LSB depends on the selected SPI mode and the sclk.

SPI has four modes relating to the sclk polarity and phase for shifting and sampling. These are as follows:

Mode	CPOL	CPHA	MISO Shift On	MOSI Sample On
0	0	0	falling sclk	rising sclk
1	0	1	rising sclk, on cs high	falling sclk
2	1	0	rising sclk	falling sclk
3	1	1	falling sclk, on cs high	rising sclk

In this assignment you will create a Verilog implementation of a module that receives the SPI signal on the MOSI line and performs an odd parity check. If the parity check is good the module will signal that the data is ready. In accordance with the SPI protocol, the data in the internal buffer will also be shifted out onto the MISO line. The SPI subordinate module will shift out the data in the buffer, as is, i.e. without correcting the parity bit if it is incorrect. This provides a basic feedback mechanism to the SPI manager of what was received at the subordinate. The module will have the following structure:





Input signals are illustrated towards the left of the diagram with an arrow to the right. Output signals are illustrated towards the right of the diagram with an arrow to the left. Wires are shown as lines. Bold lines represent multiple wires with the bit width specified on the line between forward slashes, e.g. a bit width of 6 is represented as a /6/ on the line. Binary multiplexers are illustrated as a trapezium with two inputs on the long parallel left side, a single output on the short parallel right side, and an input binary selector signal at the bottom. Modules to be instantiated are shown as vertical rectangular boxes with the name of the module above. Module I/O is illustrated with input ports on the left and output ports on the right of the rectangle. The `spi_parity_odd_fsm` and `spi_fsm` modules have been provided for you and can be instantiated.

Your solution will be tested using bounded model checking against an internal reference implementation. The model checking software will stop at the earliest moment that your implementation output diverges from the output behaviour of the internal reference implementation. If a divergence takes place, a waveform will be created showing a trace of the module signals resulting in the divergent state. This waveform also shows the behaviour of the internal reference (REF) so that you can see what your implementation (DUT) should have done differently.

The internal reference implementation is correct by definition. It is what is commonly known as a golden reference. The generated waveforms that include the signals of the internal reference contain sufficient information to implement the requested hardware design. While we endeavour for all descriptions to be as full and consistent as reasonable, it may occur that this is not the case, or that your interpretation of the various descriptions is not as intended. To avoid confusion, only the reference implementation is used when judging the correctness of your solution. Please note that this means that the internal reference implementation supersedes all other implementation descriptions, e.g. text and diagram descriptions. If the output from your solution implementation does not diverge from the internal reference implementation then your solution is marked as correct. Otherwise, your solution is incorrect and you should use the waveform or text output that is produced to assist you to correct your implementation. No other comparison will be considered at any time when judging correctness.

[VPL](#)

You are logged in as Thomas Stirling Valdez (Log out)  
5EIB0

Data retention summary