Computation II: embedded system design (5EIB0)

Available from: Wednesday, 3 July 2024, 6:00 PM
Due date: Wednesday, 3 July 2024, 9:00 PM
Requested files: spi_fsm.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:

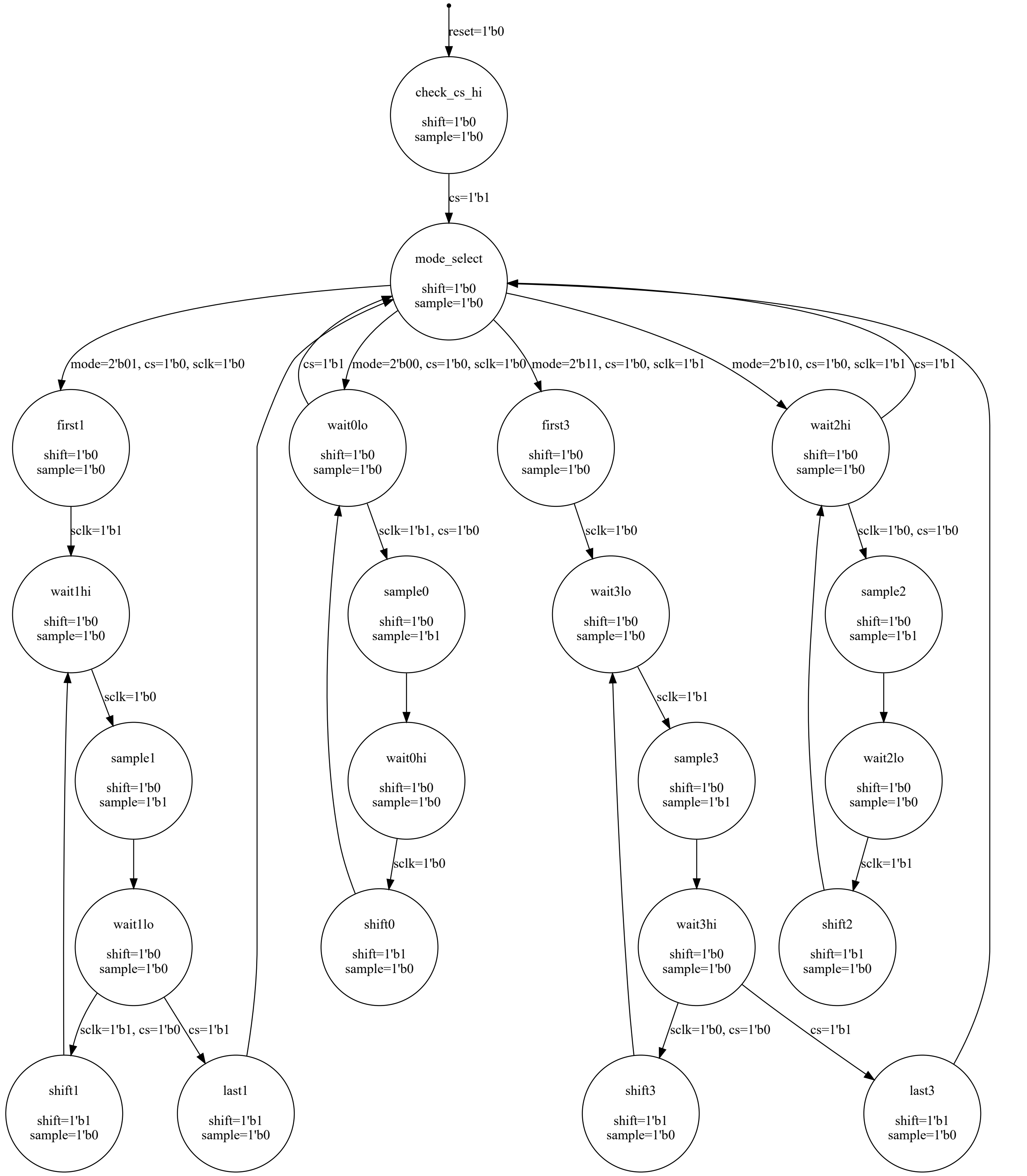
1. Chip Select (cs) input
2. Serial Clock (sclk) input
3. Manager Out, Subordinate In (MOSI) input
4. Manager In, Subordinate Out (MISO) output

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 Finite State Machine (FSM) with the following specifications. Including the clock (clk) and reset signals, the implemented FSM will have 5 input signals and 2 output signals.



The clk input signal is 1-bit wide, the reset input signal is 1-bit wide, the mode input signal is 2-bits wide, the cs input signal is 1-bit wide and the sclk input signal is 1-bit wide. The state elements of the FSM are to capture their inputs on the positive clock (clk) edge and reset their state when the reset signal is high. A reset can be asserted at any time, but should be sampled synchronously with the clock signal, causing all state elements to reset and hence a transition to the initial state.

The shift output signal is 1-bit wide and the sample output signal is 1-bit wide. Initial values of the output signals upon reset are shown in the FSM diagram. If output signal values are not stated at any point in the diagram, it is assumed that they retain their prior state until they reach a point in the diagram where they are stated again.

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.

To assist with debugging, the internal reference implementation uses the following decimal values for the named states in the diagram, with reset implemented as an explicit state:

0: reset
1: check_cs_hi
2: mode_select
3: first1
4: first3
5: last1
6: last3
7: wait1lo
8: wait0lo
9: wait2lo
10: wait3lo
11: wait0hi
12: wait1hi
13: wait2hi
14: wait3hi
15: shift0
16: shift1
17: shift2
18: shift3
19: sample0
20: sample1
21: sample2
22: sample3

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