**General Requirements**

**Core Requirements:**

1. The core shall have two sides: AXI4-Lite and SPI.
2. The core shall translate between AXI4-Lite and SPI protocol communication.
3. The core shall contain status and control registers as follows:
   1. Status Register (SPISR).
   2. Control Register (SPICR).
   3. Slave Select Register (SPISSR).
   4. Transmit Register (SPIDTR).
   5. Receive Register (SPIDRR).
   6. Interrupt Controller Register Set.
4. The core may use internal transmit and receive FIFOs as specified in the data sheet.
5. The core shall implement interrupt signaling as per the data sheet.
6. The core shall allow reading and writing to register module (s) according to data sheet specifications.
7. The SPIDRR register may be a double buffered register.
8. ‘S\_AXI\_ARESETN’ may be an asynchronous reset.

**AXI4-Lite Requirements:**

1. AXI interface shall return error handling information for incorrectly fulfilled requests.
2. Module shall implement two state machines; one for read request and one for write request.
3. AXI4-Lite interface side shall act as AXI4-Lite slave only.
4. C\_BASEADDR parameter shall be set to 0.
5. AXI4-Lite side operations shall be synchronized with the rising edge of the ‘S\_AXI\_ACLK’.
6. ‘S\_AXI\_AWREADY’ shall be asserted in order to receive write address.
7. Address on the ‘S\_AXI\_AWADDR’ channel line shall be latched when ‘S\_AXI\_AWREADY’ and ‘S\_AXI\_AWVALID’ are asserted.
8. ‘S\_AXI\_AWREADY’ shall be de-asserted when ‘S\_AXI\_AWVALID’ is asserted.
9. ‘S\_AXI\_WREADY’ shall be asserted in order to receive write data.
10. Data on the ‘S\_AXI\_WDATA’ channel line shall be latched when ‘S\_AXI\_WREADY’ and ‘S\_AXI\_WVALID’ are asserted.
11. ‘S\_AXI\_WREADY’ shall be de-asserted when ‘S\_AXI\_WVALID’ is asserted.
12. Latched data shall be sent to the register module when both address and data have been latched.
13. ‘S\_AXI\_ BVALID’ response shall be asserted after sending data internally to the register module.
14. Error information shall be sent on the 'S\_AXI\_BRESP’ channel line when ‘S\_AXI\_BVALID’ and ‘S\_AXI\_BREADY’ are asserted.
15. ‘S\_AXI\_BVALID’ shall be de-asserted when ‘S\_AXI\_BREADY’ and ‘S\_AXI\_BVALID’ are asserted.
16. ‘S\_AXI\_ARESETN’ shall de-assert ‘S\_AXI\_AWREADY’, ‘S\_AXI\_WREADY’, ‘S\_AXI\_BVALID’, ‘S\_AXI\_BRESP’ and set its 2 bits to “00”.
17. If ‘S\_AXI\_AWVALID’ and ‘S\_AXI\_AWREADY’ are asserted, ‘S\_AXI\_AWREADY’ shall be de-asserted.
18. If ‘S\_AXI\_WVALID’ and ‘S\_AXI\_WREADY’ are asserted, ‘S\_AXI\_WREADY’ shall be de-asserted.
19. S\_AXI\_AWREADY and S\_AXI\_AWVALID being asserted shall mark the beginning of an address write operation.
20. S\_AXI\_WREADY and S\_AXI\_WVALID being asserted shall mark the beginning of data write operation.
21. ‘S\_AXI\_BRESP’ shall convey information about the integrity of the current write transaction it is signaling the end of.
22. Written data shall be passed to the appropriate register according to the received write address.
23. ‘S\_AXI\_ARESETN’ shall de-assert ‘S\_AXI\_ARREADY’, ‘S\_AXI\_RVALID’, set ‘S\_AXI\_RDATA’ 32 bits low, and ‘S\_AXI\_RRESP’ all 2 bits low.
24. If ‘S\_AXI\_ARVALID’ and ‘S\_AXI\_ARREADY’ are asserted then ‘S\_AXI\_ARREADY’ shall be de-asserted.
25. If ‘S\_AXI\_RVALID’ and ‘S\_AXI\_RREADY’ are asserted, ‘S\_AXI\_RVALID’ shall be de-asserted and ‘S\_AXI\_ARREADY’ shall be asserted.
26. ‘S\_AXI\_RDATA’ shall be a 32-bit data bus.
27. ‘S\_AXI\_ARADDR’ shall be a 32-bit address bus.
28. ‘S\_AXI\_ARREADY’ shall remain de-asserted until ‘S\_AXI\_RREADY’ and ‘S\_AXI\_RVALID’ are asserted.
29. ‘S\_AXI\_ARREADY’ and ‘S\_AXI\_ARVALID’ being asserted shall mark the beginning of a read transaction.
30. ‘S\_AXI\_RREADY’ and ‘S\_AXI\_RVALID’ being asserted shall mark the end of the current transaction. Subsequent assertion of ‘S\_AXI\_ARREADY’ per requirement 32marks readiness for the master to initiate a new read transaction.
31. Address on ‘S\_AXI\_ARADDR’ when ‘S\_AXI\_ARREADY’ and ‘S\_AXI\_ARVALID’ are asserted shall be used to put data from Register Module on ‘S\_AXI\_RDATA’ to complete the current transaction.
32. ‘S\_AXI\_RRESP’ shall be considered valid when ‘S\_AXI\_RVALID’ is asserted.
33. ‘S\_AXI\_RRESP’ shall convey information about the integrity of the current transaction it is signaling the end of.

**SPI Requirements:**

1. The module shall function as a master or a slave.
2. The maximum allowable number of slave-select lines shall be 32.
3. The SPI module shall use a C\_NUM\_TRANSFER\_BITS wide shift register to convert between serial and parallel data.
4. The maximum allowable clock rate for SPI shall be one-fourth of the AXI clock rate.
5. SPI module shall operate with CPOL/CPHA set as 0/0.
6. The SPI clock speed shall be a forth of AXI clock in SPI master operation.
7. SPI clock shall drive SPI shift register.
8. SPI interface shall not have write access to any register in the Register Module other than the SPIDRR.
9. Signals ‘SCK\_T’, ‘MOSI\_T’, ‘MISO\_T’ and ‘SS\_T’ shall enable/disable their respective output signals.