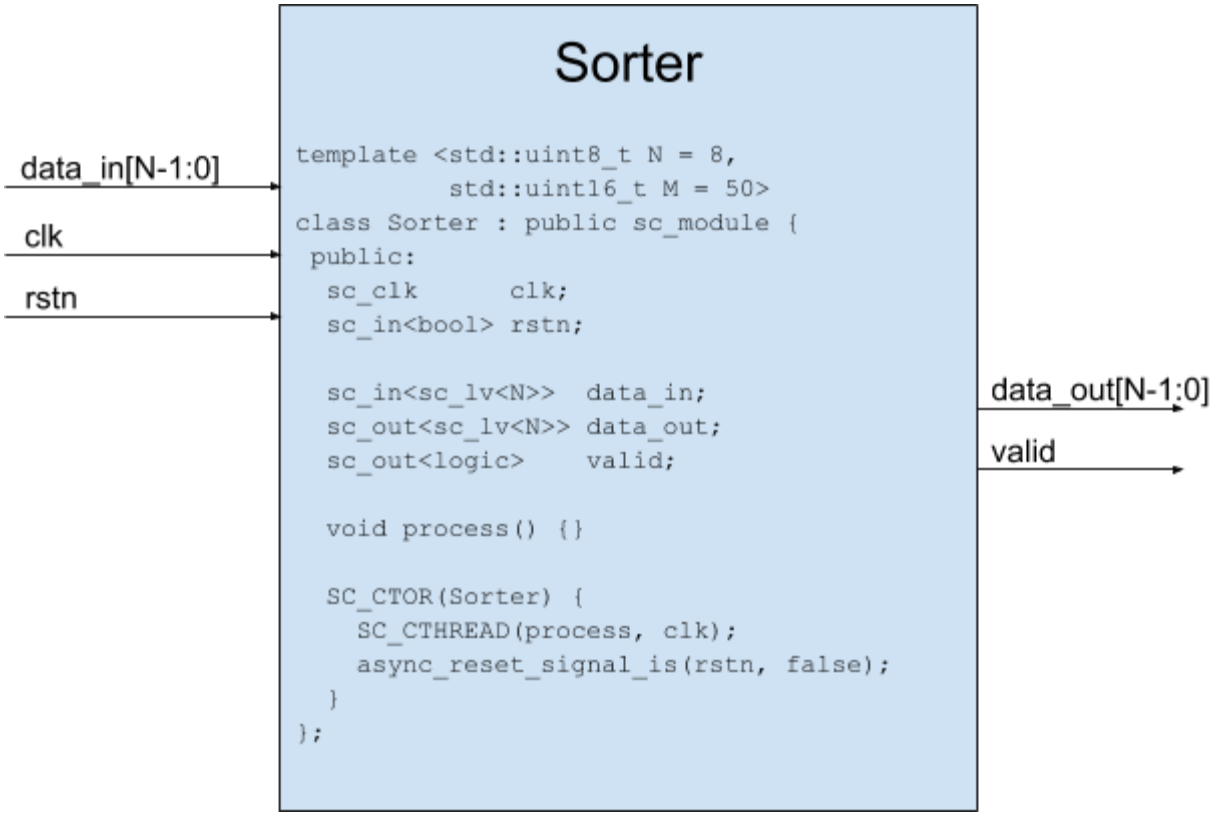

The above module receives an N-bit piece of data per clock. For every M pieces of data the module receives, it will sort thoses M pieces of data and output them in ascending sorted order, asserting `valid` whenever there is valid data on `data_out`. The code snippet inside the block shows the SystemC module interface, but (intentionally) not the implementation, where both N and M are parameterized with default values of N = 8 and M = 50.

Please provide a testbench in SystemC / C++ that will verify this module. This may include directed tests, random tests or a combination of both. For each test provided, please also provide reasons for implementing the test and what functionality or cases it is intended to test.

Additional information:

- If you are not familiar with SystemC, the following equivalences with SystemVerilog may help:

SystemC	SystemVerilog
`sc_lv<N> x`	`logic [N-1 : 0] x`
`sc_in<sc_lv<N>> y`	`input logic [N-1 : 0] y`
`sc_out<sc_lv<N>> z`	`output logic [N-1 : 0] z`
`SC_CTHREAD(process, clk);` `async_reset_signal_is(...);`	`always_ff @(posedge clk or negedge rstn)`

- Each group of M pieces of data is independent. It is *not* a sliding window. Repetition of values is possible.
- You may not make any assumptions about the implementation except that it is synthesizable to gates and/or memory elements (like SRAM cuts)
- You may use any portion of the C++ standard library as long as it conforms to the C++14, or earlier, standard.