University of Tehran

Electrical and Computer Engineering Department

ECE (8101) 342

Object Oriented Modeling of Electronic Circuits – Spring 1402-03

Computer Assignment 4

Computer Assignment 4: C++ & SystemC Gate Modeling
Due Date: Ordibehesht 2024

RTL description
C++ & SystemC RT-level description
C++ procedural RT-level simulation.

Description of the Selection Sorter circuit:

The selection sort algorithm is a simple comparison-based sorting algorithm. It works by dividing the input into two parts: the subarray of sorted elements and the subarray of unsorted elements. In each iteration, the algorithm selects the smallest (or largest, depending on the sorting order) element from the unsorted subarray and swaps it with the first element of the unsorted subarray. This process continues until the entire array becomes sorted. At each step, the smallest (or largest) element is identified and "selected" to be placed in its correct position. This selection process gives the algorithm its name. The selection sort algorithm has an average and worst-case time complexity of O(n^2), where n is the number of elements in the array. It is not suitable for large datasets since its time complexity grows quadratically. However, it has the advantage of being simple to understand and implement.
For more information you can use following sources:
http://www.geeksforgeeks.org/selection-sort/

SSC Design

You are to design a sorter circuit that uses the selection sort algorithm to sort a block of 256 16-bit unsigned words in an ascending order. Figure 1 shows the top-level specification of the SSC. Data collected becomes available in a clocked addressable register file that acts as a buffer between the switches and the selection sorter circuit (SSC). The register file has read and write signals for its read and write operations. A wrapper sitting next to our processing element (PE) is responsible for collecting data and informing our PE that data collection is completed**,** and sorting can start. This is done by the wrapper issuing a complete positive pulse on *start*. When sorting is completed, the sorter issues a complete positive pulse on its *done* output.

The register file contains 256 unsigned words with a length of 16 bits. When the wrapper sends a positive pulse on the *start* signal, two counters and all registers are reset to zero. Once a complete pulse is detected on the *start* signal, the sorting procedure begins and the value 1 is loaded into *counter2*. The *counter1* circuit addresses the last word of the sorted part of the block. At this state it is addressing the first word of the register file. The data and its address are saved in the *minReg* and minAddrReg, respectively. In each cycle, *counter2* is incremented by one, and the data corresponding to the address generated by *counter2* is fetched from the register file.

This data is then compared with the data of *minReg*. If a smaller data value is found, it is saved along with its address. This process continues until the value of *counter2* reaches 255. At this point, the minimum value from the unsorted section has been identified and needs to be moved to the sorted portion. The movement of the minimum value occurs in three update stages: *update0*, *update1*, and *update2*. During these stages, the minimum value is replaced with the first member of the unsorted section, whose address is determined by the value of *counter1* is incremented by one.

This process continues until the value of *counter1* reaches 255, indicating that all the data has been sorted.

Write C++ RTL description of this circuit, using the *bus* library discussed in class.

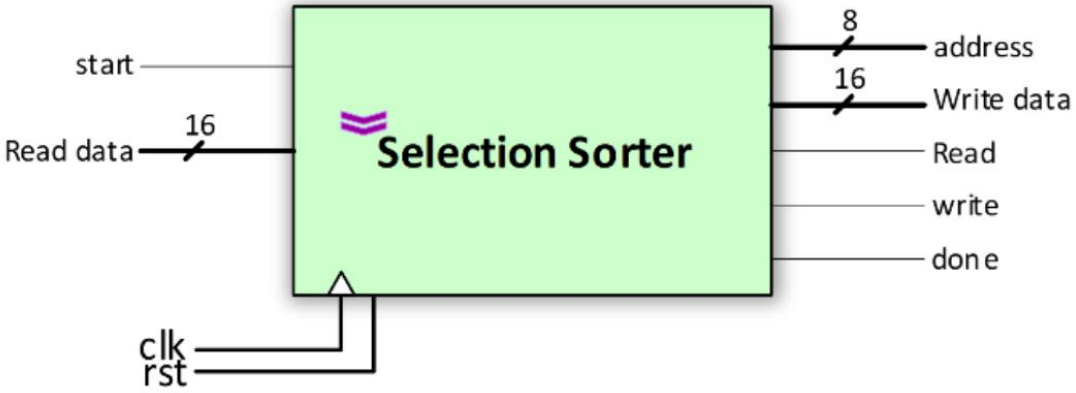

Figure 1: Top view of Selection Sorter Circuit

Extended Part

Write **SystemC** RTL description of this circuit.

Deliverables:

A. All C++ and SystemC codes, with appropriate names.
B. A complete report, containing datapath, controller, and simulation results; your datapath and controller should be explained properly and completely in the report.
C. You should upload a single file named:
YourFisrstNameYourLastName-YourStudentId-CA4.zip
And the file structure should be like this: