CS1220 – C++ Programming

Homework Assignment: HW#6: Term Project
Due: See course web site for due date
Points: 150 (30 + 120)

Name: _____

I. Requirements: Restate the problem specification, and any detailed requirements

II. Design: How did you attack the problem? What choices did you make in your design, and why? Show class diagrams for more complex designs.

III. Implementation: Outline any interesting implementation details.

IV. Testing: Explain how you tested your program. Explain why your test set was sufficient to believe that the software is working properly, i.e., what were the range of possibilities of errors that you were testing for.

V. Summary/Conclusion: Present your results. Did it work properly? Are there any limitations? If it is an analysis-type project, this section may be significantly longer than for a simple implementation-type project.

VI. Lessons Learned: List any lessons learned. What might you have done differently if you were going to attack this again.

CS1220 – C++ Programming

Homework Assignment: Term Project

Objective

Create a circuit simulator which reads a circuit definition (as described below) and an input vector definition (also described below) and simulates the operation of the circuit over time (up to 60 time steps of simulation time or until the circuit output no longer changes).

Description

The purpose of this assignment is to pull together in a single object-oriented and *optionally* GUI-based assignment all of the characteristics of the C++ language which you've learned in this class: file I/O, pointers, classes, containers, inheritance, and polymorphism.

Be aware, for most students this project is significantly more difficult than previous assignments. It is in your best interest to begin the assignment early; otherwise, you may not complete it on time—if at all.

Detailed Problem Statement: Develop a digital simulator which reads a circuit description and input vector from files (formats described below) and performs the digital simulation based on these definitions. The simulation is to be visualized using a console window or *optionally* a GUI via wxWidgets.

- *Circuit File Format*: In general, a circuit definition has the following format:

  CIRCUIT HEADER
  INPUT PAD DEFINITIONS (as many as necessary)
  OUTPUT PAD DEFINITIONS (as many as necessary)
  GATE DEFINITIONS (as many as necessary)

  Where:

  a. The CIRCUIT HEADER consists of the keyword "CIRCUIT" and a circuit name. You may use this name to label the circuit, or simple ignore the line. For example: *CIRCUIT Circuit1*
  b. The CIRCUIT HEADER will be followed by an unspecified number of INPUT PAD DEFINITIONS. An INPUT PAD DEFINITION will consist of the keyword "INPUT" followed by a name label and a wire number. For example, the following line denotes that input pad "A" is associated with wire number two: *INPUT A 2*
     NOTE: although the example circuit files use single-letter names, your simulator should accommodate names of more than one letter.
  c. The INPUT PAD DEFINITIONS will be followed by an unspecified number of OUTPUT PAD DEFINITIONS. An OUTPUT PAD DEFINITION will have the same format as an INPUT PAD DEFINITION, except the keyword will be "OUTPUT" rather than

INPUT". For example, the following line denotes that output pad "E" is associated with wire number six: *OUTPUT E 6*

d. The OUTPUT PAD DEFINITIONS will be followed by an unspecified number of GATE DEFINITIONS. A GATE DEFINITION consists of the gate type (one of "NOT", "AND", "OR", "XOR", "NAND", "NOR", and "XNOR") followed by an integer delay value with its nanosecond units, followed by the input wire numbers (two input wires for all gates, except a NOT gate which uses only one), followed finally by an output wire number: For example, the following line defines an AND gate with a 5 nanosecond delay and having wire 1 and 2 for input and wire 4 for output: *AND 5ns 1 2 4*

Result: Your program should read the circuit file and produce an in-memory representation of the circuit which can be simulated using the information from the vector file (see below) as the initial starting conditions.

- *Vector File Format*: An input vector definition has the following format:

  VECTOR HEADER
  INPUT PAD VALUE DEFINITIONS (as many as necessary)

  Where:

  a. The VECTOR HEADER consists of the keyword "VECTOR" and a vector name. You may use this name to label the simulation output, or you may simply ignore the line. For example: *VECTOR vector1*
  b. The VECTOR HEADER will be followed by an unspecified number of INPUT PAD VALUE DEFINITIONS. An INPUT PAD VALUE DEFINITION consists of the keyword "INPUT" followed by a name label, followed by a time stamp at which the wire associated with the name value changes its value, and then by the value to which the wire changes. For example, the line below indicates that input A changes value at time 0 to a value of 1: *INPUT A 0 1*

  Result: Your program should read the vector file and initialize a priority queue of events (one event for each INPUT PAD VALUE DEFINITION).

Three-valued Digital Logic: The wires in our digital circuit can take on 3 values: 0, 1, and X (undefined). The outputs pads and all gate outputs should be initialized to X at time zero. The truth values for three-valued AND and OR operations appears below, from these tables you should be able to determine the remainder of the gate operations.

Three-Value Truth Table

X	Y	X AND Y	X OR Y
0	0	0	0
0	1	0	1
0	x	0	X
1	0	0	1
1	1	1	1
1	x	x	1
X	0	0	X
X	1	x	1
X	x	x	X

Interim Deliverables: To encourage an early start on this assignment, you are required to turn in (as a minimum) uploaded class specifications AND implementations for the **Gate** and **Wire** objects by the date shown on the Canvas web site. The interim deliverable is worth 20% of the overall assignment. Please note, these classes are likely not all of the classes you will implement for your solution. Additional classes which you may use are Event, Queue, Circuit, etc.

Other Requirements: Your program must perform basic error handling on the input files (however, exception handling is **not** required). For example, you should gracefully handle "File not found" and detect format errors in input files. Any ill-formed input lines can simply be ignored. You do not have to check that the circuit description makes sense; e.g., you don't have to test that circuit inputs can affect the outputs, etc.

Your program should provide a record of the simulation by showing the input and output pad histories of the simulated circuit in a console window or by drawing in a wxWidgets TextField. If developing the *optional* GUI, please use a wxFileDialog to choose the circuit and vector files. NOTE: *you do **not** need to show the event queue, change the simulation speed, single step, etc. as demonstrated in the example program.*

Sample Programs: To get a better feel for the program, you can download a working examples executables for either the console and GUI versions by ctrl-clicking on the following links: text-based version or GUI version (ctrl-click to download).

If you copy the GUI version of the app to your own computer, please be sure to also copy associated button bitmaps (open.bmp, etc.)

Sample Circuits: Sample circuit and vector definitions are also provided in the aforementioned folders or on our course web here.

Teamwork: You may (and are encouraged to) work in teams of two persons. However, if you choose to work alone, you may.

Output: Your simulator should have the capability to display the input and output waveforms. The display can just use text characters in the console window to show the waveform. For example:

```
Input A     XXXXX00000000000111111111111111111
Time        0     5          10

      (or)

Input A     xxxxxx_________-------------------------
Time        0     5          10
```

Optionally use wxWidgets to display the waveforms. Please select this option **only after** you have completed the basic requirements. You will be given 10% (15 points) extra credit for using wxWidgets for file selection and displaying the output (assuming the basic requirements are also met).

Final Details/Turn In

Please see the instructions posted on Canvas for HW8 (or Term Project) for the files you need to turn in for this project.