CSC 306 Fall 2019 Simulation Project Requirements

Simulated Throughput Using Software Registers and Clocking

Final Due Date December 1, 2019 – 100 pts

Project A, Linked List code – Due October 17th – 0 points

Project B, Queue code – Due November 7th – 0 points

Project C, Simulation code – Due December 1st – 100 points

When all parts of your project are put together you will have completed simulated throughput using software registers and clocking. You will have created a library of linked list functions and 4 types of queueing using linked lists. You chose your language, IDE, and target platform. I expect your work to be turned in to Canvas. Submit BOTH your source code and your output file.

Files to turn in:

- LinkedList.C (or other language extension)
- Queues.C
- Main.C
- FilesIO.C
- Any .h files you create
- File of input and output data with clock (tick) counts for Project C.

I do not need any interface and IDE project code. Comment your code so it is well explained.

Using a random number generator, you will need to create (4) data sets of 100 random integers each with values of 0 to 99. Additionally, you will need to create (1) data set of 100 random integers values of 1 to 3. The latter is for priority of the insert priority queue. You may create your data sets once and write them to a file for later reading in or you may create them each run.

Project A:

Insert head (for FIFO)

Insert tail (for LIFO)

Insert sorted (largest to tail)

Insert priority (1 at tail then 2, 3)

Create and destroy/delete will be needed. Use pointer passing in arguments.

Project B:

Build Queues for LIFO, FIFO, Insert sorted, and Insert priority. Initialize the queues with the first 10 numbers of each data set using add functions. Provide functions to add and delete from the queues for remaining 90 nodes. Maintain a queue size of 10 by deleting and then adding a node until all numbers are used. 4 data sets = 4 queues, priority for last queue.

Project C:

Create 8 registers. 4 inputs and 4 outputs. (hint: same size as your data type)

There will be two "runs" your main program. Single register and four registers.

Use looping to load/unload the queues from the data sets and to your output file. Each loop is a clock cycle. Count your clock cycle for each of the two functions. Output these counts to your file. Output your input and output data in 10 by 10 matrix to your file.

- Run one uses a single register to input to queues and a single register output data.
- Run two uses 4 registers to input to queues and 4 registers to output data to files
- **Extra credit (25 points)**: Single register input and 4 registers to output AND 4 registers to input and 1 register to output.
- **More Extra credit (25 points)**: Use a software timer to time the actual speed for each run. Print it to your output data file.

You can make this very elegant if you think about it.

I will go over the logical code structure in class.

Here is the code structure for Project C. Turn in your code and the following printed output in **readable** form: (4) data arrays in plus priority array (10x10), (4) data outputs (10x10) and simulated clock cycle count for BOTH input and out from your linked lists for each run. I expect the output to labeled and neat. I will deduct points if your output is unprofessional. Each loop is a clock cycle.

Create 4 input registers and 4 output registers. Do not run your priority array into or out of a register, just use it.

N = Number of data values

<u>For one input register and one output register:</u>

Clockin = 0

Clockout = 0

Do 10 times

    Do 4 times (once for each queue)

        Load single input register with data value

        Load one queue with data value in single register

        Clockin++

    End

End

Do N times

    Do 4 times (once for each queue) – exit when last head is null

        Load single output register with data value

        Delete single value from the same queue that the above data came from

        Clockout++

    End

    Do 4 times (once for each queue) – skip when N data values loaded

        Load single input register with data value

        Load one queue with data value in single register

        Clockin++

```
        End
End
Print Clockin and Clockout
```

For four input registers and four output registers:

```
Clockin = 0
Clockout = 0
Do 10 times
        Load four input registers with data values from the 4 data arrays
        Load four queues with data values in four registers
        Clockin++
End
Do N times
        Exit when heads are null
                Load four output registers with data values from queues
                Delete four values from the same queues that the above data came from
                Clockout++
        End
        Skip when N data values loaded
                Load four input registers with data value
                Load four queues with data values in four registers
                Clockin++
        End
End
Print Clockin and Clockout
```

```
#include <chrono>
#include <iostream>

unsigned long long fib(unsigned long long n) {
    return (0==n || 1==n) ? 1 : fib(n-1) + fib(n-2);
}

int main() {
    unsigned long long n = 0;
    while (true) {
        auto start = std::chrono::high_resolution_clock::now();
        fib(++n);
        auto finish = std::chrono::high_resolution_clock::now();

        auto microseconds =
std::chrono::duration_cast<std::chrono::microseconds>(finish-start);
        std::cout << microseconds.count() << "µs\n";
        if (microseconds > std::chrono::seconds(1))
            break;
    }
}
```