Fall ’19 CIS 314 Assignment 5 – 100/100 points – Due Friday, 11/15, 11:59 PM

Please submit individual source files for coding exercises (see naming conventions below) and a single solution document for non-coding exercises (.txt or .pdf only). Your code and answers need to be documented to the point that the graders can understand your thought process. Full credit will not be awarded if sufficient work is not shown.

1. [60] Write a Y86-64 program that sorts an array of longs.

- (10) Allocate a hardcoded input array similar to that used by asum.ys (linked on course examples page) with at least 10 entries.
- (20) Implement a *swap* procedure equivalent to the following C code:

```
void swap(long *xp, long *yp) {
    long t0 = *xp;
    long t1 = *yp;
    *xp = t1;
    *yp = t0;
}
```

- (20) Implement a *sort* procedure that sorts the input array using Bubble Sort. Your *sort* procedure should call your *swap* procedure to perform the swaps and it should sort the array in place – there’s no need to allocate additional memory for an output array.
- (10) Implement a *main* procedure to call your *sort* procedure, passing the input array and array length as arguments.

I recommend using the “Y86-64 simulator” (linked on the course Links page) as a programming environment. Use the Y86-64 examples from class and the textbook as a guide.

Hint: Y86-64 doesn’t have leaq or mult (or shifts, for that matter), so calculating a pointer from a base address and index is kinda a pain… Instead, consider using pointer arithmetic like we did back in assignment 4!

Hint: Y86-64 doesn’t have cmpq or testq, either; subq will do the trick, but be careful with the side effects!

Name your source file 5-1.ys.

2. [20] Draw a circuit (using AND, OR, and/or NOT gates) with inputs *A*, *B*, *C*, and *D* and one output such that the output is on only if *A* is on and *B* is off or *C* is on and *D* is off (e.g., (*A* && !*B*) || (*C* && !*D*)). See Figure 4.10 for an example.

Include your answer in your solutions document (a picture of a drawing is sufficient).

3. [20] In our example Y86-64 programs, such as the Sum function shown in Figure 4.6, we encounter many cases (e.g., lines 2 and 10) in which we want to add a constant value to a

register. This requires first using an irmovq instruction to set a register to the constant, and then an addq instruction to add this value to the destination register. Using Figure 4.18 as a guide, draw a diagram expressing how the existing Y86-64 architecture could implement a new iaddq instruction to accomplish this functionality.

Write your answer in your solutions document.

Zip the source files and solution document (if applicable), name the .zip file <Your Full Name>Assignment5.zip (e.g., EricWillsAssignment5.zip), and upload the .zip file to Canvas (see Assignments section for submission link).