LSU EE 4755 Homework 2 Due: 12 September 2018

Problem 1: The Verilog code below is the `sort3` module from Homework 1. Draw a diagram of the hardware as described by `sort3`, showing the `sort2` modules as boxes. Be sure to label the input and output ports with the same symbols used in the module.

```
module sort3
  #( int w = 8 )
   ( output uwire [w-1:0] x0, x1, x2,
     input uwire [w-1:0] a0, a1, a2 );

   uwire [w-1:0] i10, i11, i21;

   sort2 #(w) s0_01( i10, i11,   a0, a1 );
   sort2 #(w) s1_12( i21, x2,  i11, a2  );
   sort2 #(w) s2_01( x0, x1,  i10, i21  );

endmodule
```

Problem 2: It is possible to build an n-element sorting network using $\frac{n}{2}\lg^2 n$ two-element sorting networks in such a way that the n-element sorting network has a critical path of $\lg^2 n$. (Note: $\lg n \equiv \log_2 n$.) But this assignment is concerned with n-element sorting networks using $n(n-1)/2$ two-element sorting networks, which we will call n-element *bad sorting networks* or *bad sorters* for short.

An n-element bad sorter has inputs $a_0, a_1, \ldots, a_{n-1}$ and outputs $x_0, x_1, \ldots, x_{n-1}$. The largest value is routed to x_{n-1}.

A 2-element bad sorter is a single `sort2` module. An n-element bad sorter, $n > 2$, can be constructed using an $(n-1)$-element bad sorter and $n-1$ `sort2` modules as follows. The $n-1$ `sort2` modules are connected to the n inputs and to each other in such a way that the largest value is routed to a specific output of one of the `sort2` modules. That specific `sort2` output is connected to output x_{n-1} of the n-element sorter. The other values connect to the $(n-1)$-element bad sorter, and the $(n-1)$-element bad sorter outputs connect to outputs $x_0, x_1, \ldots, x_{n-2}$ of the n-element bad sorter that we are constructing. Note that this generalizes the solution to Homework 1 Problem 2.

The description above is recursive. At level i (the same as n above) another $i-1$ `sort2` modules are used. For a 4-element sorter we need $(4-1)+(3-1)+1 = 6$ `sort2` modules. The cost of an n-element bad sorter is found by solving the summation $\sum_{i=2}^{n} i-1$, which is $n(n-1)/2$. That's $O(n^2)$, which is how the bad sorter got its name.

It gets worse. The critical path through the bad sorter can range from bad to awful. That depends on two things: How the `sort2` modules are used to find the largest value, and how the `sort2` modules connect to the $(n-1)$-element bad sorter.

(*a*) Show the worst way that `sort2` modules can be connected to find the largest value. *Hint: the critical path should be* $n-1$ `sort2` *modules.* Provide a sketch for the general case, and an example for $n=4$.

(*b*) Show the worst way that the `sort2` modules, as connected above, can connect to the $(n-1)$-element sorter. Provide a sketch.

(*c*) Determine the critical path for an n-element bad sorter constructed in the way described in the last two parts. *Hint: The math part should be familiar.*

(d) Show a much better way of connecting the `sort2` modules to find the largest value. It should be easy to show that the critical path is the lowest that is possible. Provide a sketch for $n = 8$.

The problem with the approach to building the bad sorters described in this assignment is that each level in the recursion reduces the size by 1 (that is, from n to $n-1$), and so the critical path must be at least $O(n)$. As some students may have realized, a better approach would be to use recursion in which the n inputs were split between two $\frac{n}{2}$-element networks and then somehow combined. But how? The key insight, described by K. E. Batcher in a landmark 1968 paper, is not to try to recursively describe a sorting network, but to instead recursively describe a network that merges two already sorted sequences. The input to a 2-element merge network would be two 1-element sorted sequences. (Of course, every 1-element sequence is sorted.) Pairs of 2-element merge networks feed a 4-element merge network, and so on. This will be further described later in the semester.