Verilog HW#2: Sorting Network Design

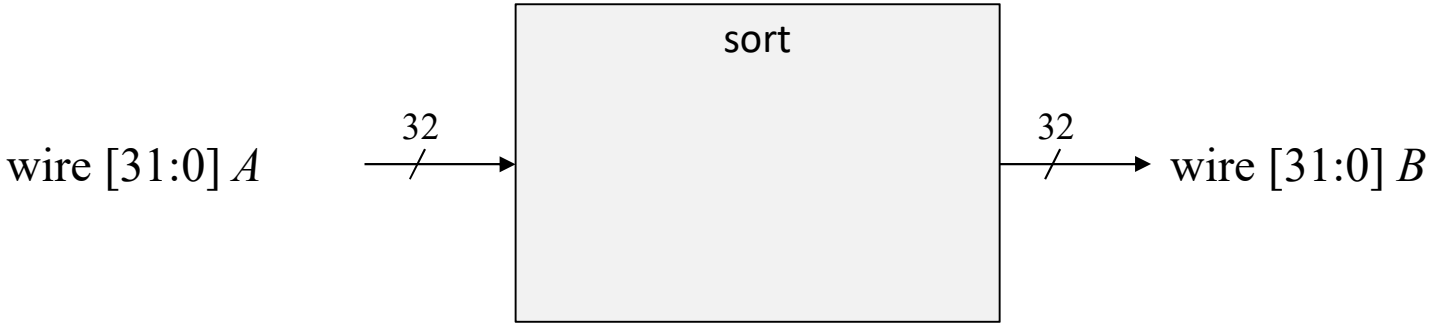
Chun-Jen Tsai
National Chiao Tung University
3/31/2021

Verilog HW#2

- Goal: Design a Verilog module that implements a sorting network that can sort eight 4-bit unsigned numbers in descending order.
  - In your program, you should use array signals and for-loops for repeated circuit generation whenever possible
- Deadline: 4/19, 23:55pm. You must upload your Verilog module to the E3 website by the deadline.

Block Diagram of the Module

- You should call your module `sort`, with the following input/output ports:



  - A is a 32-bit input that contains eight 4-bit unsigned umbers.
  - B is the 32-bit result of the sorted numbers.

Data Signal Format

- The input A[] is packed with eight 4-bit unsigned numbers:

```
{ A[31:28], A[27:24], …, A[ 3: 0] }
```

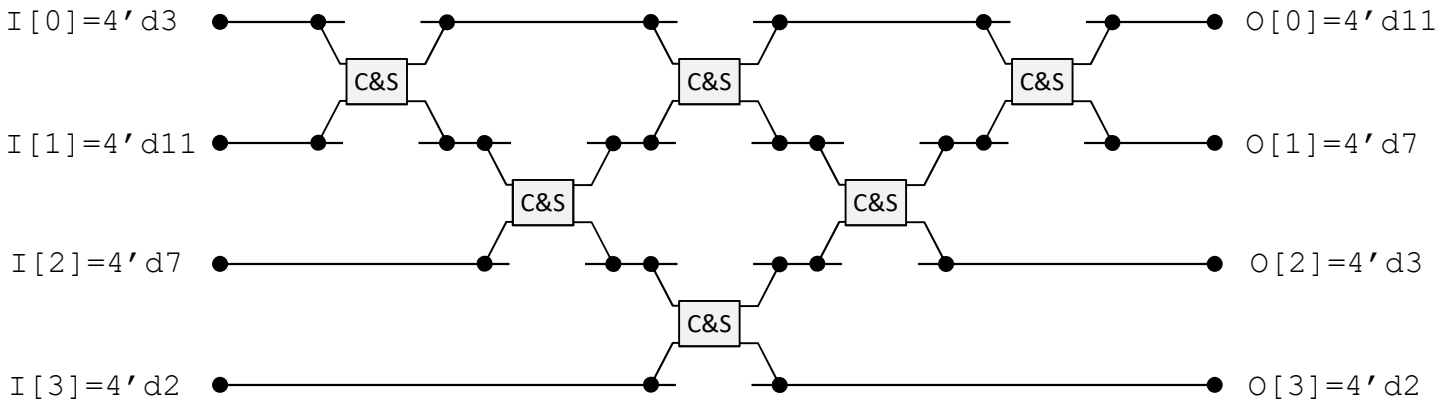
1st number　2nd number　8th number

- You should unpack it into an array signal using bit-field extraction operator `+:` and a for-loop.

- The format of the output B[] is the same, but the numbers should be in descending order.

Logic Diagram of a Sorting Network

- An example of a 4-number sorting network is as follows:
  - The circuit block “C&S” implements the comparison-and-swap operation of two numbers



- This diagram implements the famous bubble-sort
  - Use arrays and for-loops in your code as much as possible

Requirements for Verilog HW#2

- The module you designed must be declared as follows:

```
module sort(input [31:0] A, output [31:0] B);

  /* Implement your design here. */

endmodule
```

- Do not upload your testbench module to E3, just upload the `sort()` module and its supporting modules (if you have more than one modules/files). TA's will use their own testbench to test your module.

Comments on Icarus Verilog (1/2)

- Note that the Icarus Verilog compiler does not dump array signals into a VCD file properly
  - The array signals may not show up in the GTKWave window
  - For HW#2, you can use temporary wires to connect to an array signal for debugging:

```
 wire [3:0] my_array [0:7];
 genvar k;
 generate
   for (k = 0; k < 8; k = k + 1) begin : array_wires
     wire [3:0] my_wire;
     assign my_wire = my_array[k];
   end
endgenerate
```

Comments on Icarus Verilog (2/2)

- Note that in Icarus Verilog simulation, all gates and operators are assumed to have zero delay by default
  - Unless you insert delays manually, you do not see the transitions of sorted signals in the simulated waveforms
  - You can insert a time delay “#1” in the comparison-and-swap circuit block to see the transitions, for example:

```
#1 // delay the following op by 1 time unit.
if (array[j] > array[j+1]) begin
    ...
end
```

  - Make sure to remove all the delays before submitting code to E3, TA’s testbench will assume zero delay in your modules