CSE490/590 PROJECT 1 REPORT

VERILOG SORTING

Name: VALLABHANENI SUSMITHA CHOWDARY
UB Person #: 50169236
Date: 3/11/2016

I. ABSTRACT

This project is to design a circuit that will input 4 numbers on the FPGA board, sort and display the sorted numbers by decimal. In order to proceed with the project, I performed three modules in order to implement the main objectives of the project: Input the numbers onto the board, sort the numbers, display the simulated output.

In order to input the numbers on to the board, the basic initializations were made for inputs and outputs, partA was used to determine the index of the position where to store the values that are to be sorted. The numbers are being stored in the index provided by partA provided when the button partC is high. Now, when this button is high the numbers are stored and are ready for being sorted.

To perform the sorting, bubble sort was used however the numbers are sorted provided the partD is high. This means when the button is high the numbers get sorted. To perform bubble sort, for loop came into play. These sorted numbers are stored in order to display the sorted numbers at partE.

All of these modules are however being called in the main module. Thus, these modules are invoked and we obtain the results.

II. FLOW CHART

The flow chart is shown in Fig. 1

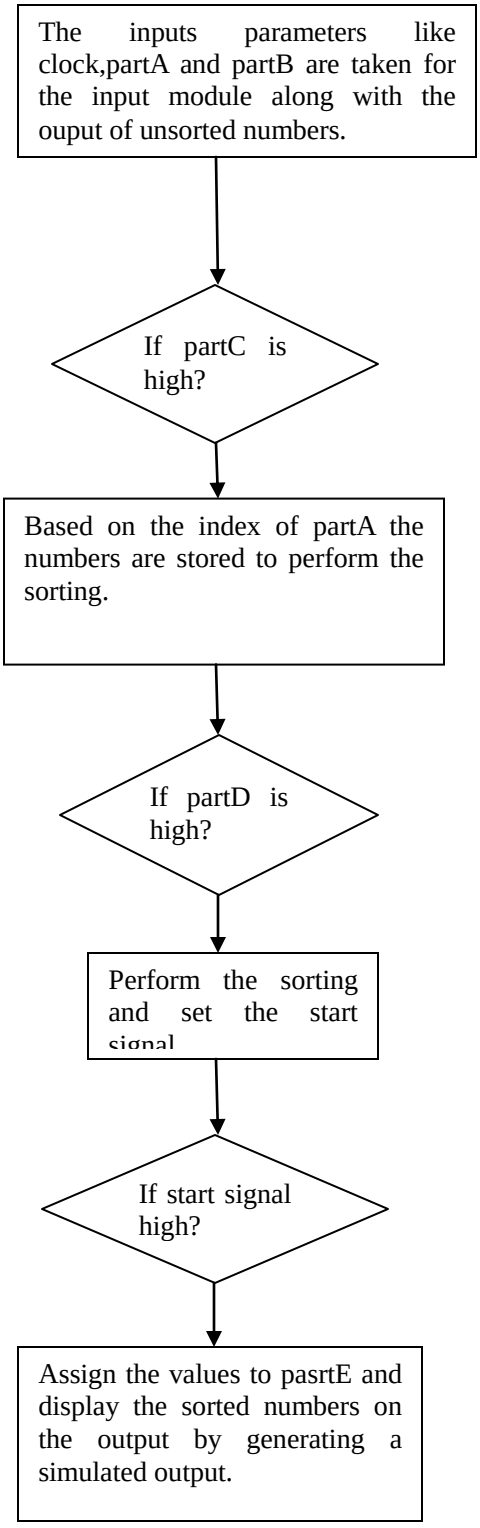


Fig.1. Flow Chart

III. INTERNAL SPECIFIC DESIGN

Code Part 1 - Input

```
module input_part(
    input clk,
    input [3:0] partA,
    input [3:0] partB,
    input partC,

    output reg [3:0] unsorted0,
    output reg [3:0] unsorted1,
    output reg [3:0] unsorted2,
    output reg [3:0] unsorted3
    );

always @(posedge clk)
begin
    if (partC==1)
    begin
        case (partA)
            4'b0001:unsorted0 = partB;
            4'b0010:unsorted1 = partB;
            4'b0100:unsorted2 = partB;
            4'b1000:unsorted3 = partB;
        endcase
    end
end

endmodule
```

Code Part 2 - Sort

```
module sorting_part(
    input clk,
    input partD,
    input [3:0] unsorted0,
    input [3:0] unsorted1,
    input [3:0] unsorted2,
    input [3:0] unsorted3,
    output reg [3:0] sorted0,
    output reg [3:0] sorted1,
    output reg [3:0] sorted2,
    output reg [3:0] sorted3,
    output reg start_display
    );

reg [3:0] in_array[3:0];
```

```
integer i;
integer j;
reg [3:0] temp;


always @(posedge clk)
begin
    if (partD==1'b1)
    begin
        in_array[0] = unsorted0;
        in_array[1] = unsorted1;
        in_array[2] = unsorted2;
        in_array[3] = unsorted3;

         for (i=0; i<3; i=i+1)
         begin
              for (j=0; j<3; j=j+1)
              begin
                  if (in_array[j] > in_array[j+1])
                    begin
                       temp = in_array[j+1];
                          in_array[j+1] = in_array[j];
                             in_array[j] = temp;
                    end
              end
         end

         sorted0 = in_array[0];
         sorted1 = in_array[1];
         sorted2 = in_array[2];
         sorted3 = in_array[3];

         start_display = 1'b1;
    end
end
endmodule
```

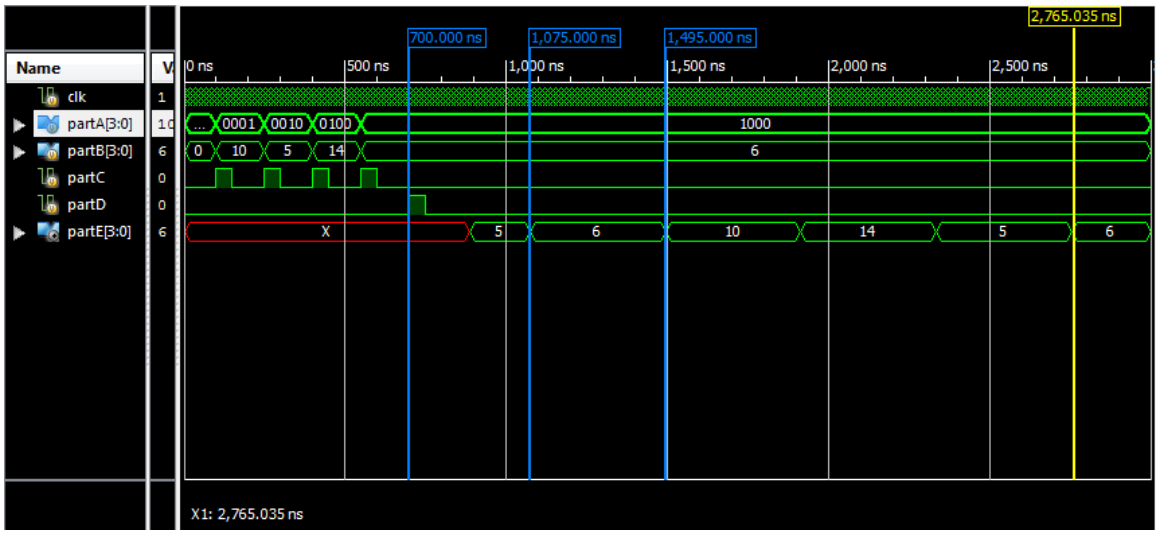
Code Part 3- Output

```
module output_part(
    input clk,
    input [3:0] sorted_num0,
    input [3:0] sorted_num1,
    input [3:0] sorted_num2,
```

```
    input [3:0] sorted_num3,
    input start_display,
    output reg [3:0] partE
    );

always @ (posedge clk)
begin
    if (start_display == 1)
    begin
        partE = sorted_num0;
       #500;
        partE = sorted_num1;
        #500;
        partE = sorted_num2;
        #500;
        partE = sorted_num3;
        #500;
    end
end
endmodule
```

IV. RESULTS


Name
clk
partA[3:0]
partB[3:0]
partC
partD
partE[3:0]
0 ns
500 ns
1,000 ns
1,500 ns
2,000 ns
2,500 ns
700.000 ns
1,075.000 ns
1,495.000 ns
2,765.035 ns
0001
0010
0100
1000
0
10
5
14
6
X
5
6
10
14
5
6
X1: 2,765.035 ns