Aireen Amir Jalal
010989584

Homework #2 (due beginning of class March 28th)

1. Implement one of the adder topologies discussed capable of adding two 8-bit numbers. This includes the PG logic, the adder (composed of gray and black cells), and the sum logic.

2. Design a Verilog adder_tester block that you can incorporate within Cadence. The adder_tester block should output two 8-bit numbers on buses at the rising edge of a clock input signal. Additionally, the adder_tester should receive as inputs the 8-bit sum from the adder and Cout bit, test them for correct addition, and then output a digital correct signal.

3. Provide printouts of all schematics down to the gate level as well as your testbench and simulation results.

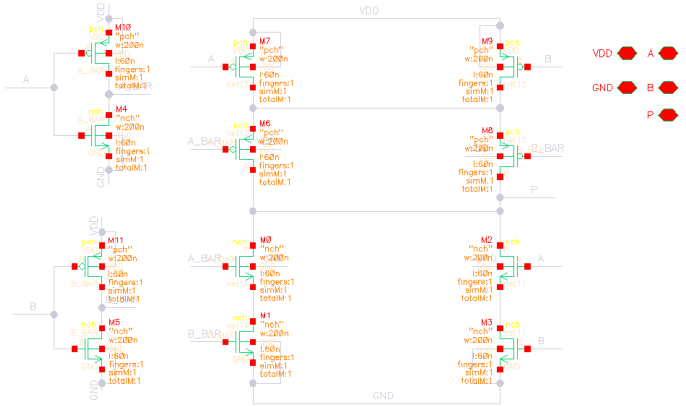
4. On your adder schematic printouts, detail the critical path for the adder you chose.

5. On a separate simulation result, show the critical path delay.

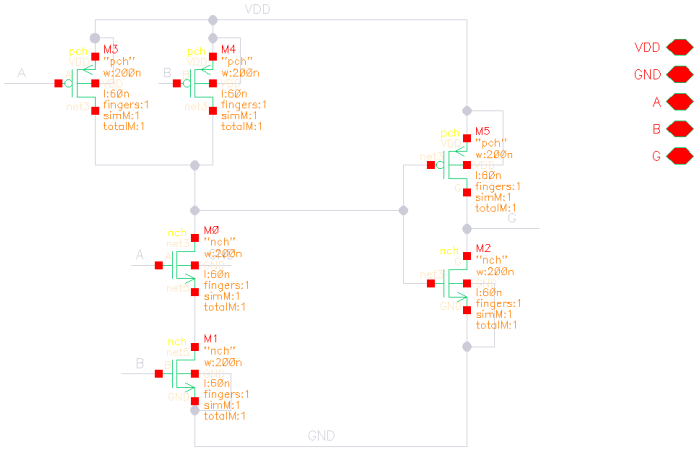
Aireen Amir Jalal
010989584

Schematic:

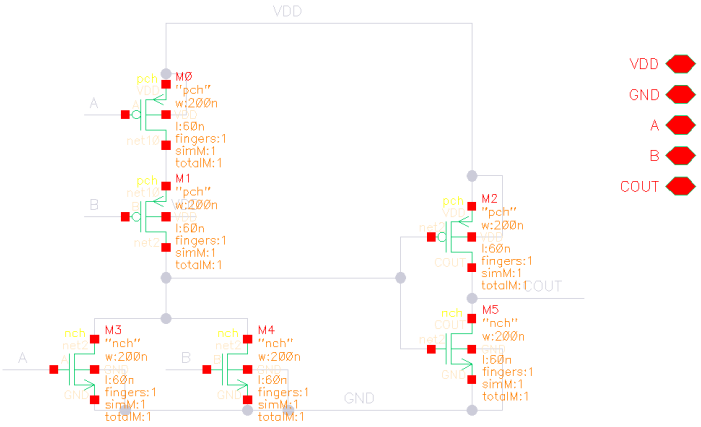
Xor gate:



And gate:

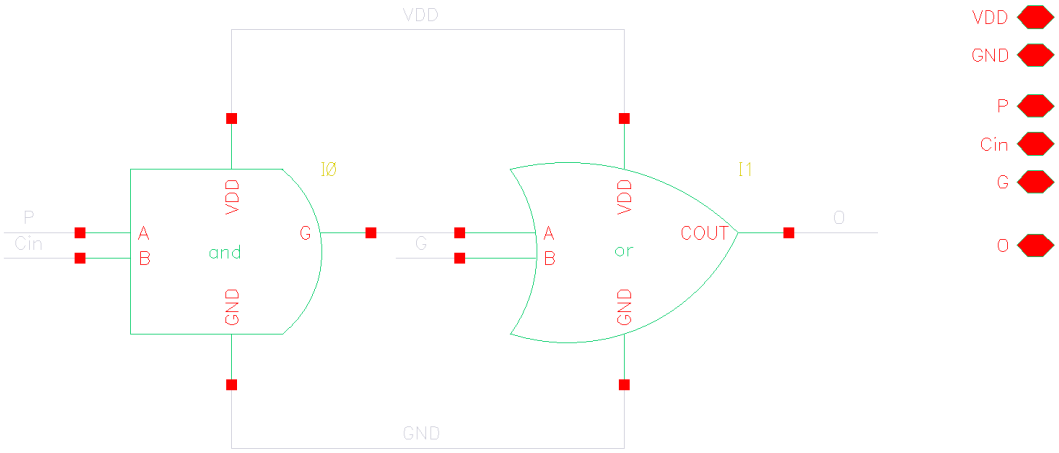


Or gate:

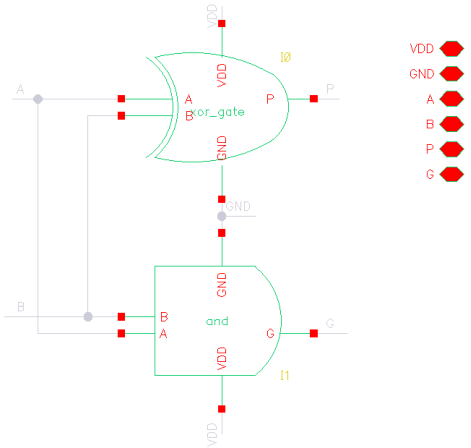


Aireen Amir Jalal
010989584

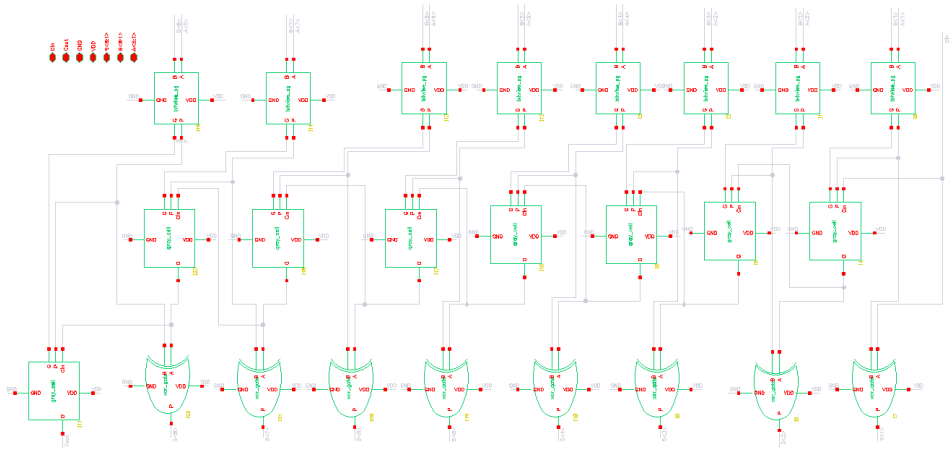
Gray cell:



Bitwise logic:

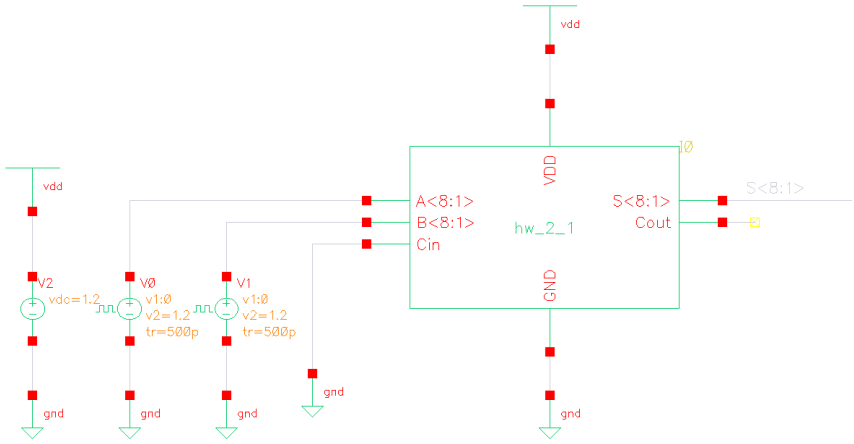


Carry Ripple Adder topology:

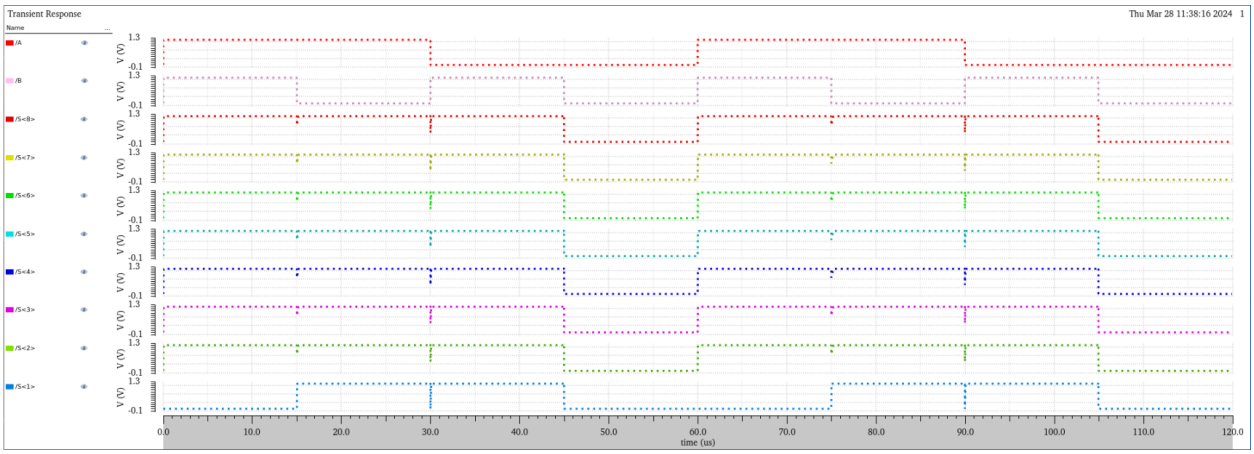


Aireen Amir Jalal
010989584

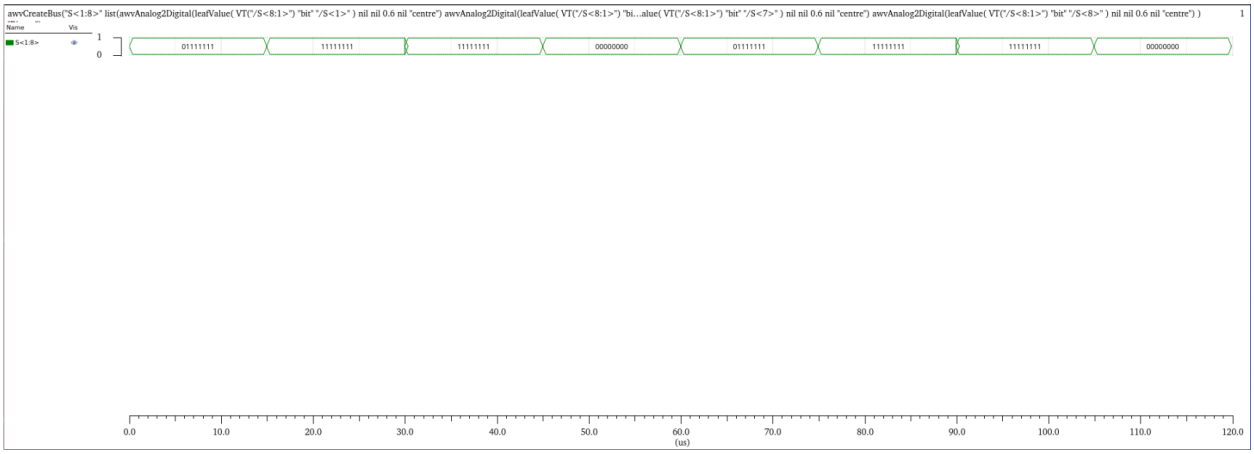
Testbench:



Simulation Result:



Binary values:



Aireen Amir Jalal
010989584

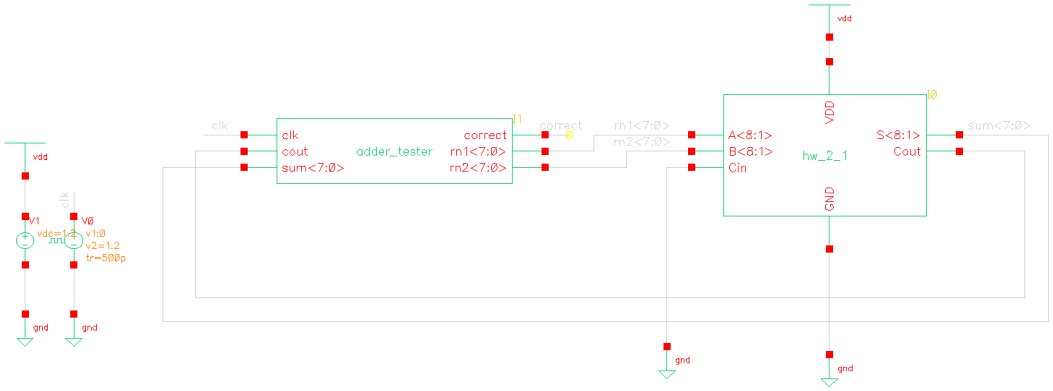
Verilog adder tester block code:

```
//Verilog HDL for "AMS_TEST", "adder_tester" "functional"

module adder_tester (
  input wire clk,
  input wire [7:0] sum,
  input wire cout,
  output reg correct,
  output reg [7:0] rn1,
  output reg [7:0] rn2
);

  reg [7:0] expected_sum;
  reg expected_cout;
 // Internal seeds for the random number generators
  reg [15:0] seed_1 = 0;
  reg [15:0] seed_2 = 12345; // Initial value for the second generator


  // Sequential logic for testing
  always @(posedge clk) begin
  // Generate two 8-bit random numbers independently
    seed_1 <= seed_1 + 1;
    rn1 <= seed_1[7:0];

    seed_2 <= seed_2 + 1;
    rn2 <= seed_2[7:0];

    // Calculate the expected sum and carry-out
    expected_sum <= rn1 + rn2;
    expected_cout <= (rn1 + rn2) > 8'hFF ? 1'b1 : 1'b0;

    // Compare the expected sum and carry-out with the actual sum and cout
    if ((sum == expected_sum) && (cout == expected_cout))
    begin
      correct <= 1'b1; // Set correct signal if the addition is correct
    end
    else
    begin
      correct <= 1'b0; // Reset correct signal if the addition is incorrect
    end
  end

endmodule
```

Aireen Amir Jalal
010989584

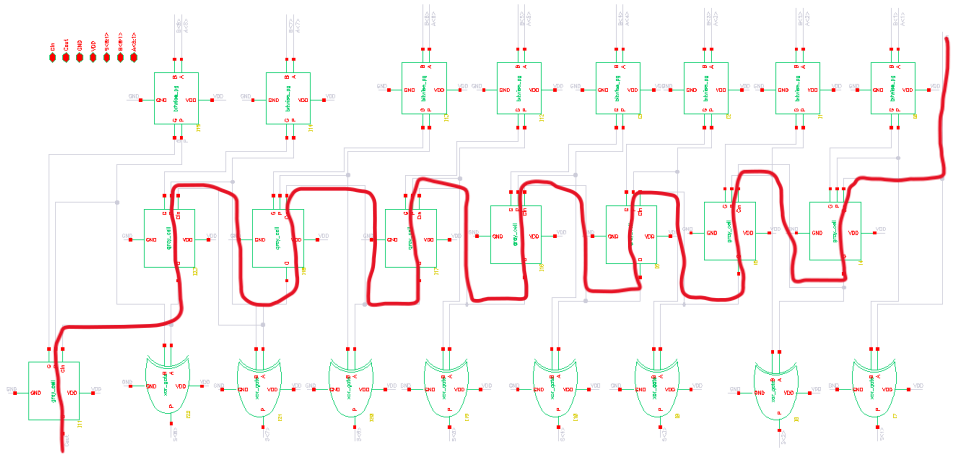
Symbol:

Testbench with the adder_tester block code:

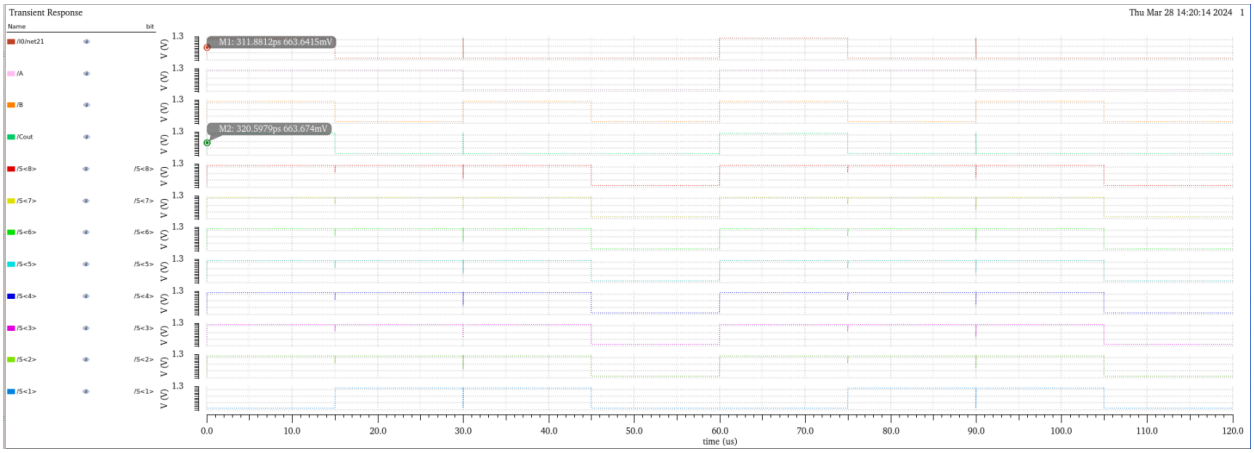


Aireen Amir Jalal
010989584

Critical path:



Critical path delay:



critical path delay= 320.5979ps-311.8812ps=8.7167ps