PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100 Feet Ring Road, BSK III Stage, Bengaluru-560 085

Department of Electronics and Communication Engineering

Course Title: ADTDD
COURSE CODE:UE21EC343BB4

Teacher: Dr.Sudeendra Kumar K

Project Title: 8 Input Bitonic Sorter Design

Done By:
Akash Ravi Bhat - PES1UG21EC025
Akash C Katate - PES1UG21EC022
Abhishek A Shetty - PES1UG21EC008

AIM

To write code and test bench for an 8 bit Bitonic Sorter and to test the sorter for 4 8 inputs

INTRODUCTION

- **Definition**: A bitonic sorter is a parallel sorting algorithm designed for hardware implementation.

- **BitonicPattern**: The term "bitonic" refers to the sorting network's pattern, starting with a monotonic sequence (ascending or descending) followed by a bitonic sequence (alternating ascending and descending).

- **8-BitOperation**: Specifically designed for 8-bit data elements.

SortingProcess:

1. Recursively build a bitonic sequence from the input data.
2. Sort the bitonic sequence until the entire sequence is sorted.

- **ParallelProcessing**: The sorting network can be structured to enable parallel processing.

- **HardwareImplementation**: Well-suited for implementation in hardware architectures like parallel processors or digital circuits.

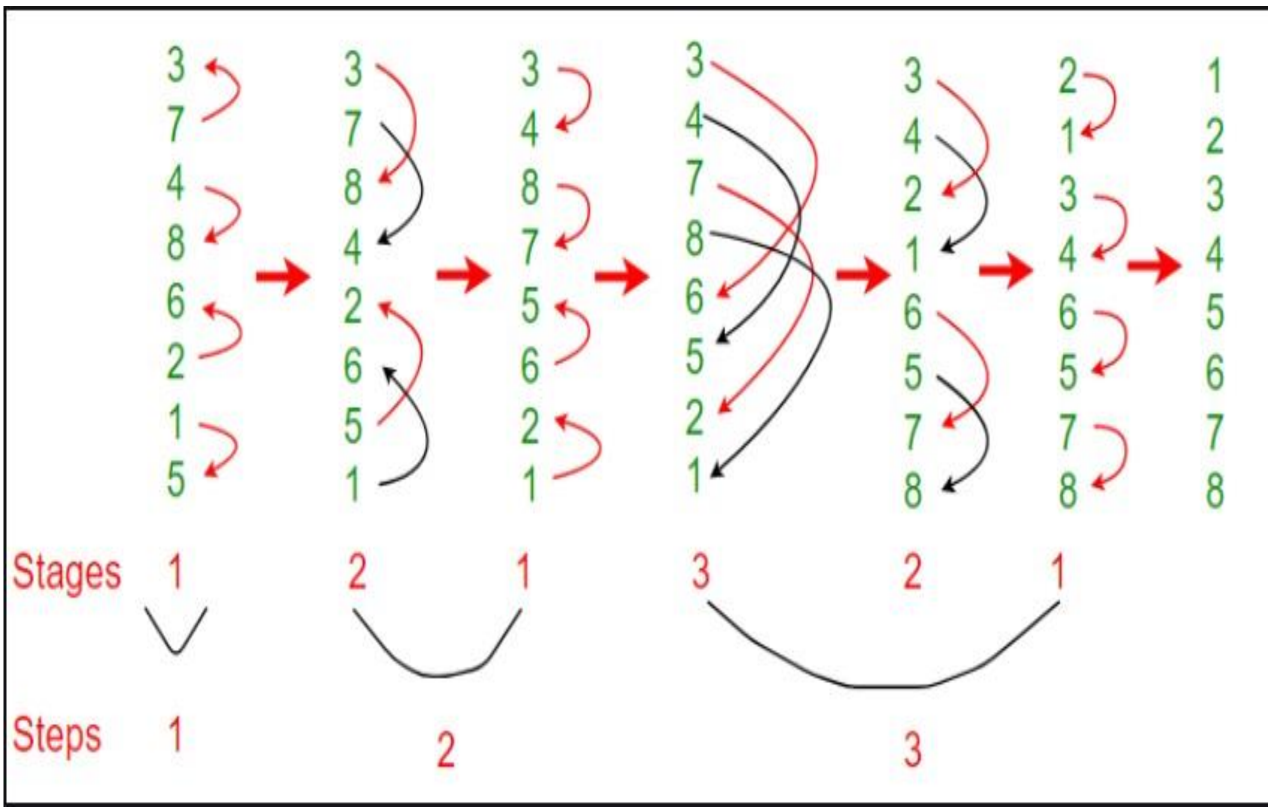
- **Application**: Particularly useful in scenarios where e   cient parallel sorting of 8-bit data is needed, such as in digital signal processing.

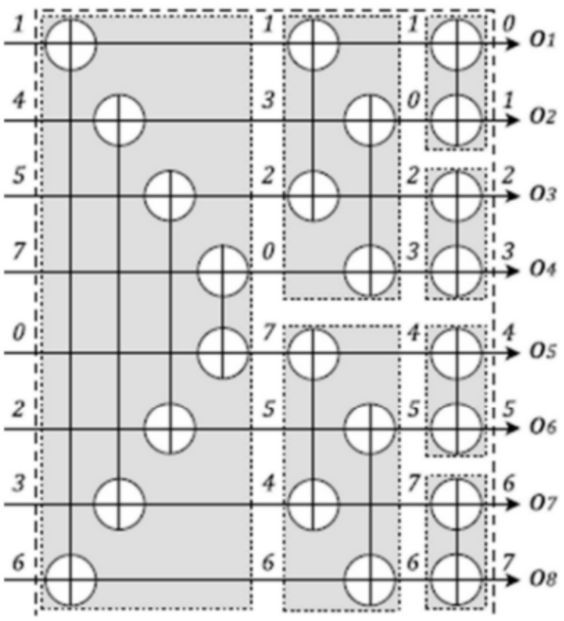
WORKING PRINCIPLE: **Two inputs are compared and exchanged according to the direction(dir) signal.**

If dir = 0, the inputs are sorted in ascending order.

If dir = 1, the inputs are sorted in descending order

.





1.DESIGN FILES :

1. Compare and Exchange block (CAE)

```
`timescale 1ns/1ps

module bitonic_sorter(output wire [31:0] o1, o2, input wire [31:0] A, B, input wire clk, rst, dir,
enable);

 reg sel;

 reg [31:0] out1, out2;


 always @(posedge clk or posedge rst) begin

  if (rst) begin

   out1 <= 32'b0;

   out2 <= 32'b0;

  end

  else if (enable) begin

   if (dir == 1'b0) begin

    sel <= (A > B);

    if (sel==1) begin

              out1<=B;

              out2<=A;

        end

        else if (sel==0) begin

              out1<=A;

              out2<=B;

        end

   end

   if (dir == 1'b1) begin

    sel <= (A < B);

    if (sel==1) begin

              out1<=B;

              out2<=A;
```

```
            end
            else if (sel==0) begin
                    out1<=A;
                    out2<=B;
            end
      end
    end
  end

  assign o1 = out1;
  assign o2 = out2;
endmodule
```

2. 8 Input Bitonic Sorting

```
`timescale 1ns / 1ps

module bitonic8(input [31:0]i0,i1,i2,i3,i4,i5,i6,i7,
output [31:0]o0,o1,o2,o3,o4,o5,o6,o7, input clk,en,rst,dir);
wire [31:0]w,x,i,j,k,l,m,n,o,p,q,r,s,t,u,v;

bitonic_sorter d1(w,x,i0,i7,clk,rst,dir,en);
bitonic_sorter d2(i,j,i1,i6,clk,rst,dir,en);
bitonic_sorter d3(k,l,i2,i5,clk,rst,dir,en);
bitonic_sorter d4(m,n,i3,i4,clk,rst,dir,en);



bitonic_sorter d5(o,p,w,k,clk,rst,dir,en);
bitonic_sorter d6(q,r,i,m,clk,rst,dir,en);
bitonic_sorter d7(s,t,n,j,clk,rst,dir,en);
bitonic_sorter d8(u,v,l,x,clk,rst,dir,en);

bitonic_sorter d9(o0,o1,o,q,clk,rst,dir,en);
bitonic_sorter d10(o2,o3,p,r,clk,rst,dir,en);
bitonic_sorter d11(o4,o5,s,u,clk,rst,dir,en);
bitonic_sorter d12(o6,o7,t,v,clk,rst,dir,en);

endmodule
```

2.TESTBENCH:

1. CAE block

```
module testbench();
  reg [31:0] A, B;
  reg clk, rst, dir, enable;
  wire [31:0] o1, o2;


  bitonic_sorter sorter(
    .o1(o1),
    .o2(o2),
    .A(A),
    .B(B),
    .clk(clk),
    .rst(rst),
    .dir(dir),
    .enable(enable)
  );


  always begin
    #5 clk = ~clk;
```

```
end

initial begin

  A = 0;

  B = 0;

  clk = 0;

  rst = 1;

  dir = 0;

  enable = 0;



  #10 rst = 0;


  // Test Case 1: Sort in ascending order (dir = 0)

  A = 5;

  B = 3;

  dir = 0;

  enable = 1;

  #10;

  // Verify the output o1 and o2


  // Test Case 2: Sort in descending order (dir = 1)

  A = 8;

  B = 12;

  dir = 1;

  enable = 1;

  #10;

  // Verify the output o1 and o2
// Test Case 3: A and B are equal

A = 10;

B = 10;

dir = 0;

enable = 1;

#10;

// Verify the output o1 and o2


// Test Case 4: A and B are equal with descending sort

A = 7;

B = 7;

dir = 1;

enable = 1;

#10;

// Verify the output o1 and o2


// Test Case 5: Sort in ascending order with enable=0

A = 15;

B = 20;

dir = 0;

enable = 0;

#10;

// Verify the output o1 and o2


// Test Case 6: Sort in descending order with reset

A = 3;

B = 1;

dir = 1;

enable = 1;

#10;
```

```
// Verify the output o1 and o2

rst = 1;

#10;

rst = 0;

#10;


// Test Case 7: A greater than B, ascending sort

A = 25;

B = 20;

dir = 0;

enable = 1;

#10;

// Verify the output o1 and o2


// Test Case 8: B greater than A, descending sort

A = 18;

B = 30;

dir = 1;

enable = 1;

#10;

// Verify the output o1 and o2


// Test Case 9: Random values for A and B with enable=1

A = 17;

B = 28;

dir = 0;

enable = 1;

#10;

// Verify the output o1 and o2
```

```
  // Test Case 10: Reset with enable=0
  A = 10;
  B = 15;
  dir = 0;
  enable = 0;
  rst = 1;
  #10;
  rst = 0;
  #10;
//Test Case 11: A greater than B, descending sort
A = 40;
B = 35;
dir = 1;
enable = 1;
#10;
// Verify the output o1 and o2

// Test Case 12: B greater than A, ascending sort
A = 100;
B = 150;
dir = 0;
enable = 1;
#10;
// Verify the output o1 and o2

// Test Case 13: Random values with enable=0
A = 72;
B = 49;
dir = 0;
enable = 0;
```

```
#10;

// Verify the output o1 and o2


// Test Case 14: Random values with enable=1, then reset

A = 31;

B = 55;

dir = 1;

enable = 1;

#10;

// Verify the output o1 and o2

rst = 1;

#10;

rst = 0;

#10;


// Test Case 15: Alternate between ascending and descending sorts

A = 30;

B = 40;

dir = 0;

enable = 1;

#10;

// Verify the output o1 and o2

dir = 1;

#10;

// Verify the output o1 and o2

// Test Case 16: Sort with very large values

A = 2147483647; // Max positive 32-bit integer

//B = -2147483648; // Min negative 32-bit integer

B = -32'h80000000;

dir = 0;
```

```
enable = 1;

#10;

// Verify the output o1 and o2


// Test Case 17: Alternating between enable and disable

A = 42;

B = 18;

dir = 0;

enable = 1;

#10;

// Verify the output o1 and o2

enable = 0;

#10;

enable = 1;

#10;

// Verify the output o1 and o2


// Test Case 18: Large values with descending sort

A = 1000000;

B = 999999;

dir = 1;

enable = 1;

#10;

// Verify the output o1 and o2


// Test Case 19: A and B are swapped in descending sort

A = 60;

B = 75;

dir = 1;

enable = 1;
```

```
#10;
// Verify the output o1 and o2


// Test Case 20: Large negative values with enable=0
A = -2147483647; // Max negative 32-bit integer
B = -1000000;
dir = 0;
enable = 0;
#10;
  $finish;
 end
endmodule
```

2. 8 input Test Bench

```
`timescale 1ns/1ns
module bitonic8_tb();
        // Parameters
        parameter CLK_PERIOD = 10; // Clock period in ns
        reg [31:0]y0, y1, y2, y3, y4, y5, y6, y7; //reg or wire?
        wire [31:0]i0, i1, i2, i3, i4, i5, i6, i7;
        reg clk, rst, en, dir;
        wire [31:0]w,x,i,j,k,l,m,n,o,p,q,r,s,t,u,v;


bitonic8 dut(i0,i1,i2,i3,i4,i5,i6,i7,y1,y2,y3,y4,y5,y6,y7,clk,en,rst,dir);


 // Clock generation
 always begin
  #5 clk = ~clk;
 end


// Initial block for stimulus
 initial begin
```

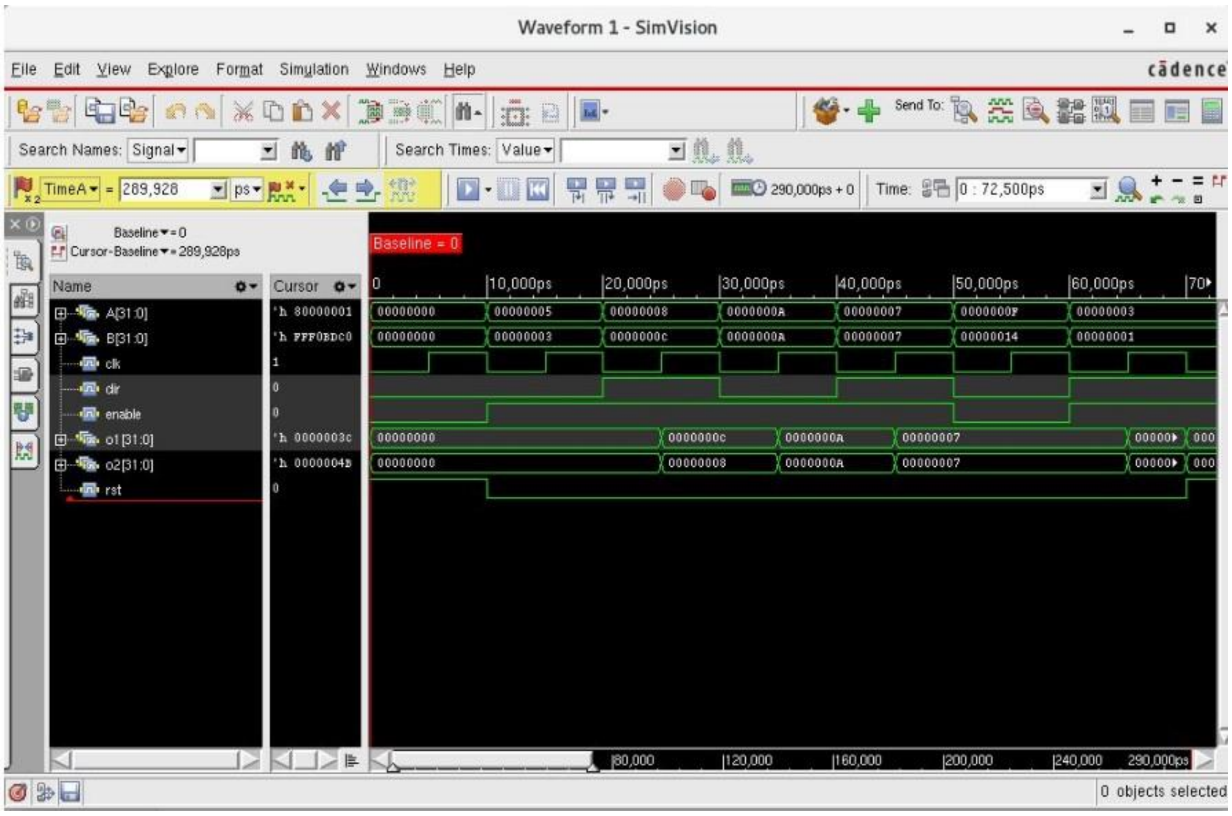
```
    // Initialize signals
    clk = 0;
    rst = 1;
    dir = 0;
    en = 1;

    // Apply reset
    #10 rst = 0;
            //Apply inputs
            y0=8'h01;
            y1=8'h04;
            y2=8'h05;
            y3=8'h07;
            y4=8'h00;
            y5=8'h02;
            y6=8'h03;
            y7=8'h06;

// Test case 1: All elements in ascending order
y0 = 8'h01; y1 = 8'h02; y2 = 8'h03; y3 = 8'h04; y4 = 8'h05; y5 = 8'h06; y6 = 8'h07; y7 = 8'h08;

// Test case 2: All elements in descending order
y0 = 8'h08; y1 = 8'h07; y2 = 8'h06; y3 = 8'h05; y4 = 8'h04; y5 = 8'h03; y6 = 8'h02; y7 = 8'h01;

// Test case 3: Random order
y0 = 8'h04; y1 = 8'h02; y2 = 8'h08; y3 = 8'h01; y4 = 8'h07; y5 = 8'h05; y6 = 8'h03; y7 = 8'h06;

// Test case 4: All elements equal
y0 = 8'h03; y1 = 8'h03; y2 = 8'h03; y3 = 8'h03; y4 = 8'h03; y5 = 8'h03; y6 = 8'h03; y7 = 8'h03;
```

```
// Test case 5: Alternating values

y0 = 8'h01; y1 = 8'h03; y2 = 8'h02; y3 = 8'h04; y4 = 8'h03; y5 = 8'h05; y6 = 8'h04; y7 = 8'h06;


// Test case 6: Random with repeated elements

y0 = 8'h05; y1 = 8'h03; y2 = 8'h06; y3 = 8'h03; y4 = 8'h07; y5 = 8'h06; y6 = 8'h05; y7 = 8'h04;


// Test case 7: All zeros

y0 = 8'h00; y1 = 8'h00; y2 = 8'h00; y3 = 8'h00; y4 = 8'h00; y5 = 8'h00; y6 = 8'h00; y7 = 8'h00;


// Test case 8: All ones

y0 = 8'hFF; y1 = 8'hFF; y2 = 8'hFF; y3 = 8'hFF; y4 = 8'hFF; y5 = 8'hFF; y6 = 8'hFF; y7 = 8'hFF;


// Test case 9: Incrementing sequence

y0 = 8'h01; y1 = 8'h02; y2 = 8'h03; y3 = 8'h04; y4 = 8'h05; y5 = 8'h06; y6 = 8'h07; y7 = 8'h08;


// Test case 10: Decrementing sequence

y0 = 8'h08; y1 = 8'h07; y2 = 8'h06; y3 = 8'h05; y4 = 8'h04; y5 = 8'h03; y6 = 8'h02; y7 = 8'h01;


// Test case 11: Random order with negative values

y0 = 8'hFF; y1 = 8'h02; y2 = 8'h05; y3 = 8'h01; y4 = 8'h07; y5 = 8'h05; y6 = 8'h03; y7 = 8'h06;


// Test case 12: Random order with large positive values

y0 = 8'hFF; y1 = 8'h02; y2 = 8'h7F; y3 = 8'h01; y4 = 8'h7F; y5 = 8'h05; y6 = 8'h03; y7 = 8'h7E;


// Test case 13: Random order with large negative values

y0 = 8'hFF; y1 = 8'h02; y2 = 8'h80; y3 = 8'h01; y4 = 8'h81; y5 = 8'h05; y6 = 8'h03; y7 = 8'h82;


// Test case 14: All elements as 0xFF (maximum value for 8 bits)

y0 = 8'hFF; y1 = 8'hFF; y2 = 8'hFF; y3 = 8'hFF; y4 = 8'hFF; y5 = 8'hFF; y6 = 8'hFF; y7 = 8'hFF;
```

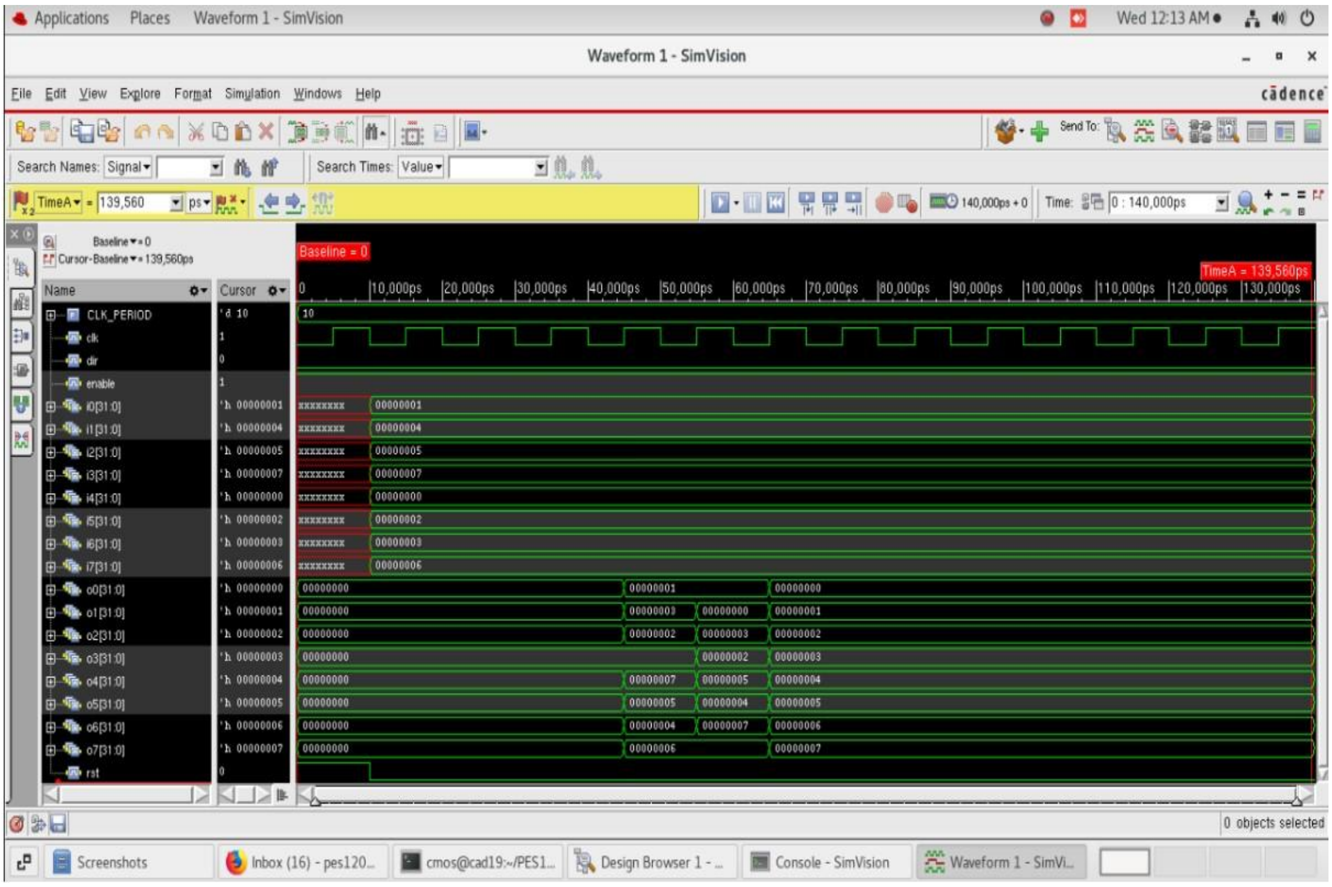
```
// Test case 15: All elements as 0x00 (minimum value for 8 bits)

y0 = 8'h00; y1 = 8'h00; y2 = 8'h00; y3 = 8'h00; y4 = 8'h00; y5 = 8'h00; y6 = 8'h00; y7 = 8'h00;


// Test case 16: Random with alternating sign

y0 = 8'h01; y1 = 8'hFF; y2 = 8'h02; y3 = 8'hFE; y4 = 8'h03; y5 = 8'hFD; y6 = 8'h04; y7 = 8'hFC;


// Test case 17: Random with repeating patterns

y0 = 8'hAA; y1 = 8'hBB; y2 = 8'hAA; y3 = 8'hBB; y4 = 8'hAA; y5 = 8'hBB; y6 = 8'hAA; y7 = 8'hBB;


// Test case 18: Random with mixed patterns

y0 = 8'h12; y1 = 8'h34; y2 = 8'h56; y3 = 8'h78; y4 = 8'h9A; y5 = 8'hBC; y6 = 8'hDE; y7 = 8'hF0;


// Test case 19: Random with alternating high and low bits

y0 = 8'hAA; y1 = 8'h55; y2 = 8'hAA; y3 = 8'h55; y4 = 8'hAA; y5 = 8'h55; y6 = 8'hAA; y7 = 8'h55;




        #30 en=1;

        #100 $finish;

end


//initial begin

//dir=1'b0; rst=1'b1;

//enable=1'b0;

//#40

//rst=1'b0;

//enable=1'b1;

//end

endmodule
```

3.WAVEFORMS:

1.CAE block



2. 8 input bitonic sorter:



4.NETLIST FILE AFTER INSERTING TEST LOGIC:

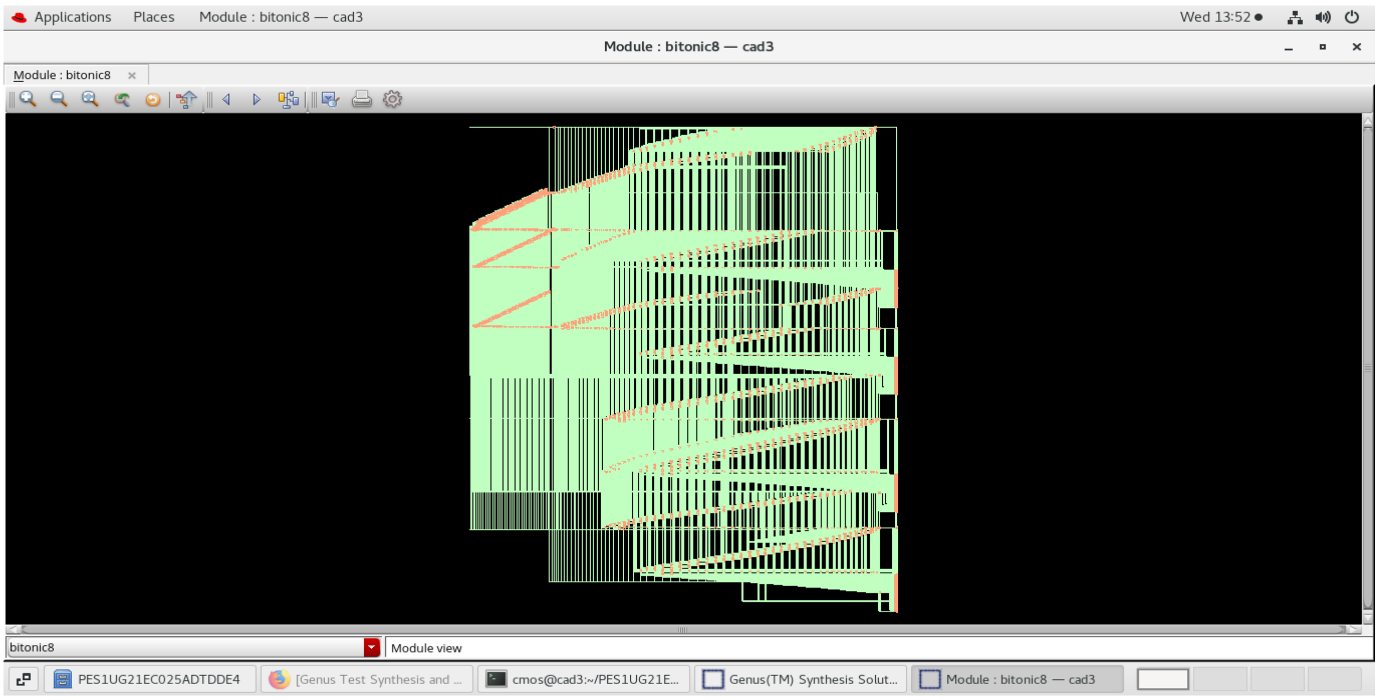
```
// Generated by Cadence Genus(TM) Synthesis Solution 20.11-s111_1

// Generated on: Mar 28 2024 16:56:19 IST (Mar 28 2024 11:26:19 UTC)


// Verification Directory fv/bitonic8


module bitonic8(i0, i1, i2, i3, i4, i5, i6, i7, o0, o1, o2, o3, o4, o5,
    o6, o7, clk, en, rst, dir, SE, scan_in, scan_out);
  input [31:0] i0, i1, i2, i3, i4, i5, i6, i7;
  input clk, en, rst, dir, SE, scan_in;
  output [31:0] o0, o1, o2, o3, o4, o5, o6, o7;
  output scan_out;
  wire [31:0] i0, i1, i2, i3, i4, i5, i6, i7;
  wire clk, en, rst, dir, SE, scan_in;
  wire [31:0] o0, o1, o2, o3, o4, o5, o6, o7;
  wire scan_out;
  wire [31:0] x;
  wire [31:0] l;
  wire [31:0] t;
  wire [31:0] v;
  wire [31:0] m;
  wire [31:0] i;
  wire [31:0] n;
  wire [31:0] j;
  wire [31:0] k;
  wire [31:0] w;
  wire [31:0] q;
  wire [31:0] o;
  wire [31:0] u;
  wire [31:0] s;
  wire [31:0] r;
  wire [31:0] p;
  wire UNCONNECTED, UNCONNECTED0, UNCONNECTED1, UNCONNECTED2,
       UNCONNECTED3, UNCONNECTED4, UNCONNECTED5, UNCONNECTED6;
  wire UNCONNECTED7, UNCONNECTED8, UNCONNECTED9, UNCONNECTED10,
       d1_n_335, d1_sel, d2_sel, d3_n_335;
  wire d3_sel, d4_n_335, d4_sel, d5_sel, d6_n_335, d6_sel, d7_sel,
       d8_sel;
  wire d9_sel, d10_n_335, d10_sel, d11_sel, d12_n_335, d12_sel, n_1,
       n_2;
  wire n_3, n_4, n_5, n_6, n_7, n_8, n_9, n_10;
  wire n_11, n_12, n_13, n_14, n_15, n_16, n_17, n_18;
  wire n_19, n_20, n_21, n_22, n_23, n_24, n_25, n_26;
  wire n_27, n_28, n_29, n_30, n_31, n_32, n_33, n_34;
  wire n_35, n_36, n_37, n_38, n_39, n_40, n_41, n_42;
     (n_3220), .Q (d9_sel), .QN (UNCONNECTED7));
  SDFFX1 d10_sel_reg(.CLK (clk), .D (d10_sel), .SI (d10_n_335), .SE
      (n_3220), .Q (d10_sel), .QN (UNCONNECTED8));
  SDFFX1 d11_sel_reg(.CLK (clk), .D (d11_sel), .SI (n_4199), .SE
      (n_3220), .Q (d11_sel), .QN (UNCONNECTED9));
  SDFFX1 d12_sel_reg(.CLK (clk), .D (d12_sel), .SI (d12_n_335), .SE
      (n_3220), .Q (d12_sel), .QN (UNCONNECTED10));
  INVX4 g123081(.INP (dir), .ZN (n_2959));
  AO222X1 g148495__5107(.IN1 (n_2140), .IN2 (n_2134), .IN3 (n_2957),
      .IN4 (n_2956), .IN5 (n_2144), .IN6 (n_2149), .Q (n_2958));
  OAI222X1 g148496__6260(.IN1 (n_2126), .IN2 (n_2954), .IN3 (i6[29]),
      .IN4 (n_2959), .IN5 (i1[29]), .IN6 (dir), .QN (n_2957));
  OA21X1 g148497__4319(.IN1 (n_2134), .IN2 (n_2140), .IN3 (n_2955), .Q
      (n_2956));
  NAND2X0 g148498__8428(.IN1 (n_2126), .IN2 (n_2954), .QN (n_2955));
  NAND2X0 g148499__5526(.IN1 (n_2952), .IN2 (n_2953), .QN (n_2954));
  OAI21X1 g148501__3680(.IN1 (n_2946), .IN2 (n_2949), .IN3 (n_2124),
      .QN (n_2953));
  AO221X1 g148502__1617(.IN1 (n_2959), .IN2 (i6[28]), .IN3 (dir), .IN4
      (i1[28]), .IN5 (n_2950), .Q (n_2952));
  AO222X1 g148503__2802(.IN1 (n_2148), .IN2 (n_2147), .IN3 (n_2947),
      .IN4 (n_1912), .IN5 (n_2948), .IN6 (n_2143), .Q (n_2951));
  NOR3X0 g148504__1705(.IN1 (n_2949), .IN2 (n_2946), .IN3 (n_2124), .QN
      (n_2950));
  OAI22X1 g148505__5122(.IN1 (n_2294), .IN2 (n_2944), .IN3 (n_1903),
      .IN4 (n_1896), .QN (d6_n_335));
```

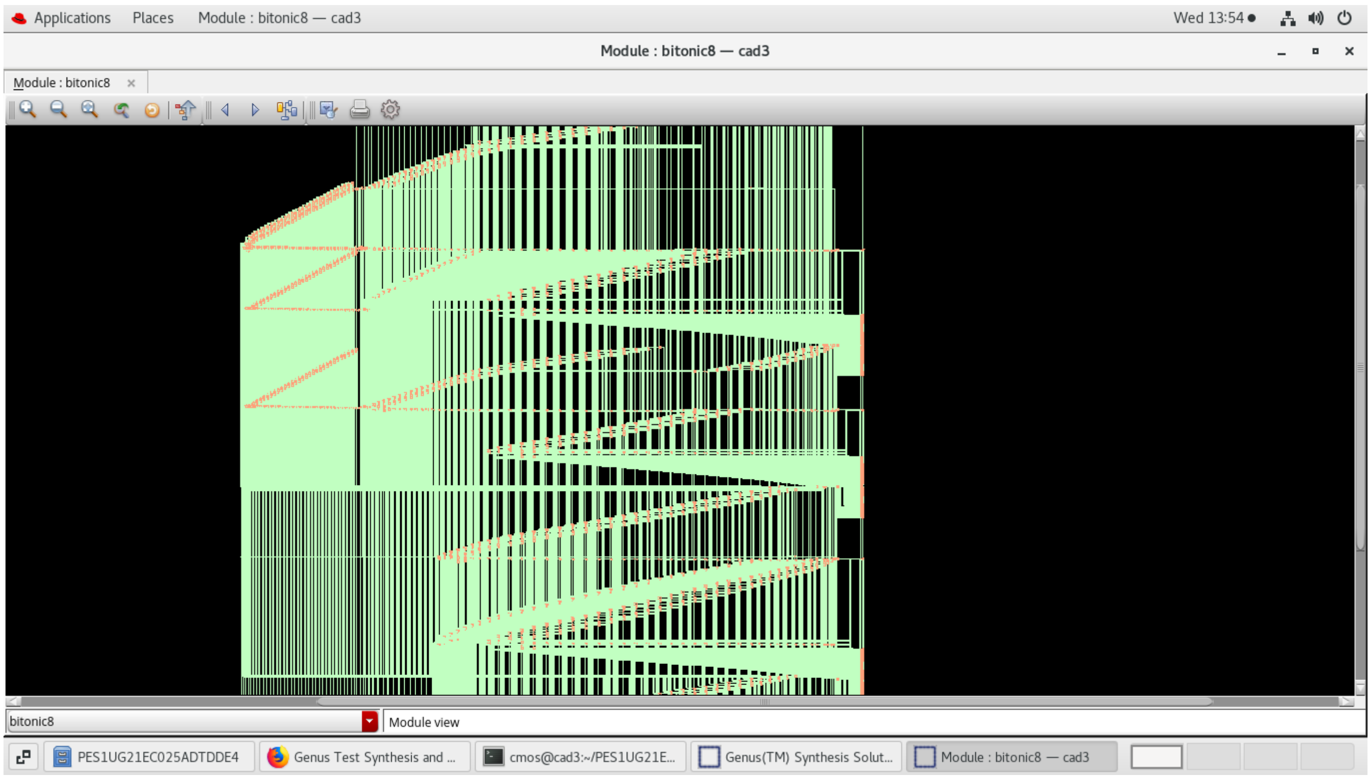
```
OA222X1 g148506__8246(.IN1 (n_2118), .IN2 (n_2943), .IN3 (i1[27]),
    .IN4 (dir), .IN5 (i6[27]), .IN6 (n_2959), .Q (n_2949));
NAND2X0 g148507__7098(.IN1 (n_2383), .IN2 (n_2945), .QN (n_2948));
AO22X1 g148508__6131(.IN1 (n_2942), .IN2 (n_2393), .IN3 (n_1672),
    .IN4 (n_1671), .Q (d4_n_335));
NAND3X0 g148509__1881(.IN1 (n_2945), .IN2 (n_2383), .IN3 (n_2142),
    .QN (n_2947));
AND2X1 g148510__5115(.IN1 (n_2943), .IN2 (n_2118), .Q (n_2946));
OA22X1 g148511__7482(.IN1 (n_2291), .IN2 (n_2941), .IN3 (n_1879),
    .IN4 (n_1887), .Q (n_2944));
AO222X1 g148512__4733(.IN1 (n_2133), .IN2 (n_2136), .IN3 (n_2936),
    .IN4 (n_2414), .IN5 (n_2141), .IN6 (n_2139), .Q (n_2945));
AO21X1 g148513__6161(.IN1 (n_2113), .IN2 (n_2110), .IN3 (n_2940), .Q
    (n_2943));
AO221X1 g148514__9315(.IN1 (n_2930), .IN2 (n_1663), .IN3 (n_1666),
    .IN4 (n_1668), .IN5 (n_2939), .Q (n_2942));
AO21X1 g148515__9945(.IN1 (n_2928), .IN2 (n_1897), .IN3 (n_2937), .Q
    (d1_n_335));
OA22X1 g148517__2346(.IN1 (n_2290), .IN2 (n_2934), .IN3 (n_1864),
    .IN4 (n_1870), .Q (n_2941));
OAI22X1 g148518__1666(.IN1 (n_2260), .IN2 (n_2933), .IN3 (n_1670),
    .IN4 (n_1667), .QN (d3_n_335));
OA22X1 g148519__7410(.IN1 (n_2110), .IN2 (n_2113), .IN3 (n_2343),
    .IN4 (n_2932), .Q (n_2940));
AOI221X1 g148520__6417(.IN1 (n_2959), .IN2 (i4[29]), .IN3 (dir), .IN4
    (i3[29]), .IN5 (n_2935), .QN (n_2939));
AO222X1 g148522__2398(.IN1 (n_2927), .IN2 (n_2920), .IN3 (n_2013),
    .IN4 (n_2155), .IN5 (n_1783), .IN6 (n_2228), .Q (n_2938));
AOI222X1 g148523__5107(.IN1 (n_2929), .IN2 (n_2315), .IN3 (n_2959),
    .IN4 (i7[31]), .IN5 (dir), .IN6 (i0[31]), .QN (n_2937));
```

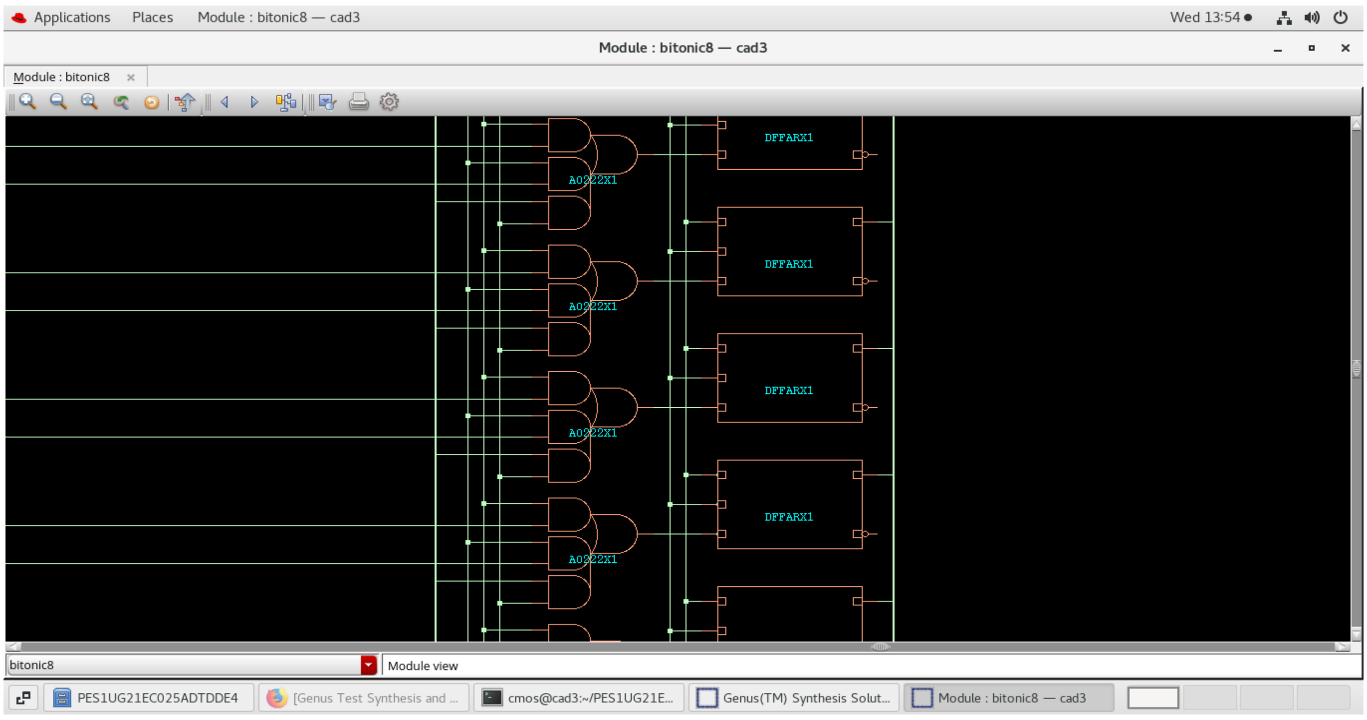
```
 AO222X1 g148524__6260(.IN1 (n_2125), .IN2 (n_2127), .IN3 (n_2917),
      .IN4 (n_2413), .IN5 (n_2130), .IN6 (n_2129), .Q (n_2936));
 NOR2X0 g148525__4319(.IN1 (n_2930), .IN2 (n_1663), .QN (n_2935));
 OA22X1 g148526__8428(.IN1 (n_2288), .IN2 (n_2925), .IN3 (n_1842),
      .IN4 (n_1853), .Q (n_2934));
 OA22X1 g148527__5526(.IN1 (n_2259), .IN2 (n_2926), .IN3 (n_1662),
      .IN4 (n_1660), .Q (n_2933));
 OA22X1 g148528__6783(.IN1 (n_2923), .IN2 (n_2924), .IN3 (n_2102),
      .IN4 (n_2105), .Q (n_2932));
 AO222X1 g148529__3680(.IN1 (n_2234), .IN2 (n_1798), .IN3 (n_2913),
      .IN4 (n_1922), .IN5 (n_2914), .IN6 (n_2229), .Q (n_2931));
 NOR2X0 g148530__1617(.IN1 (n_2921), .IN2 (n_1897), .QN (n_2929));
 AO21X1 g148531__2802(.IN1 (n_2911), .IN2 (n_1661), .IN3 (n_2919), .Q
      (n_2930));
 NAND2X0 g148532__1705(.IN1 (n_2315), .IN2 (n_2922), .QN (n_2928));
 AO21X1 g148533__5122(.IN1 (n_1906), .IN2 (n_1907), .IN3 (n_2918), .Q
      (d12_n_335));
 OAI222X1 g148534__8246(.IN1 (n_2236), .IN2 (n_2912), .IN3 (n_2959),
      .IN4 (x[29]), .IN5 (dir), .IN6 (l[29]), .QN (n_2927));
 OA22X1 g148535__7098(.IN1 (n_2257), .IN2 (n_2909), .IN3 (n_1655),
      .IN4 (n_1652), .Q (n_2926));
 OA22X1 g148536__6131(.IN1 (n_2283), .IN2 (n_2910), .IN3 (n_1831),
      .IN4 (n_1838), .Q (n_2925));
 OA21X1 g148537__1881(.IN1 (n_2895), .IN2 (n_2905), .IN3 (n_2098), .Q
      (n_2924));
INVX8 g150696(.INP (en), .ZN (n_769));
 INVX0 g150697(.INP (rst), .ZN (n_768));
 OAI21X1 g2(.IN1 (n_2144), .IN2 (n_2149), .IN3 (n_2958), .QN (n_4196));
 OAI21X1 g150726(.IN1 (n_2148), .IN2 (n_2147), .IN3 (n_2951), .QN
      (n_4197));
```

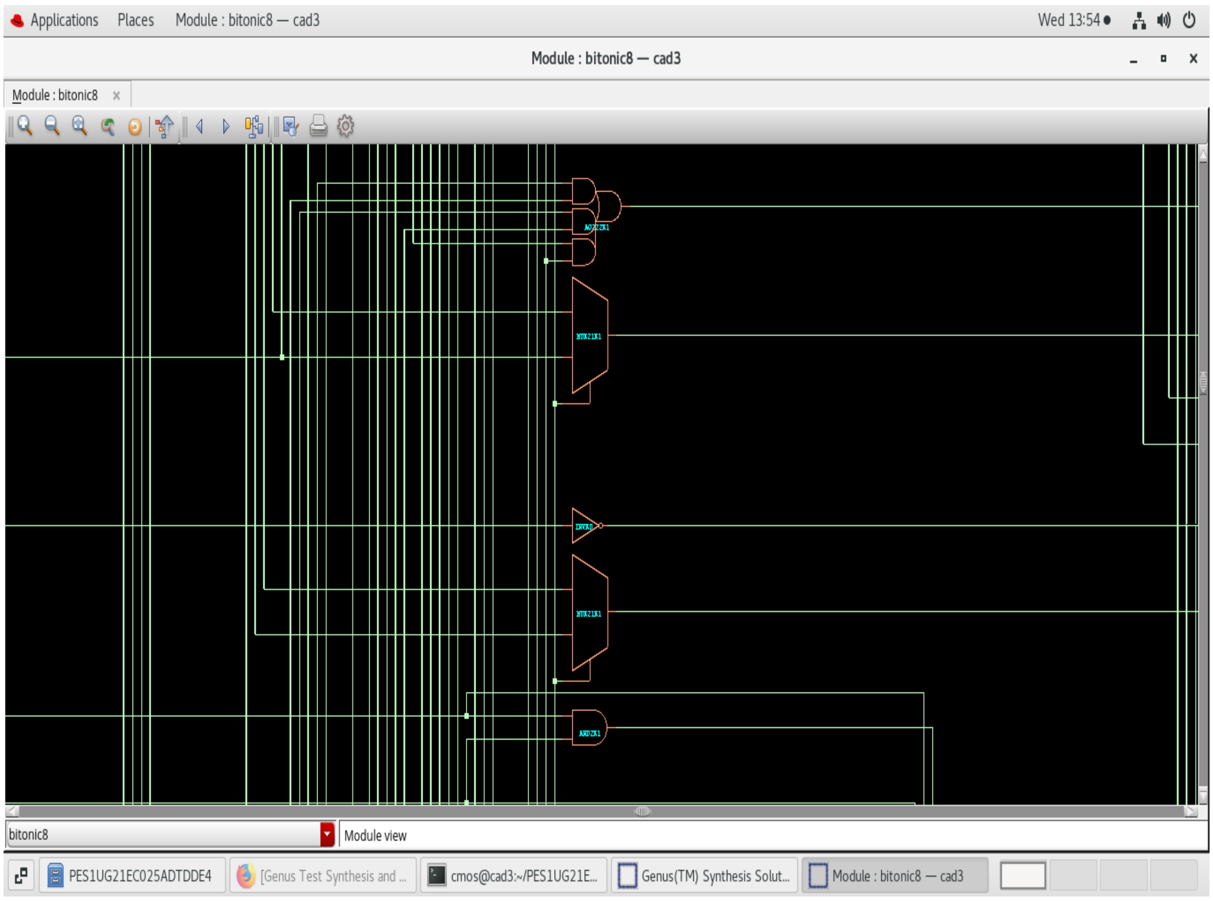
```
  OAI21X1 g150727(.IN1 (n_1783), .IN2 (n_2228), .IN3 (n_2938), .QN
      (n_4198));
  OAI21X1 g150728(.IN1 (n_2234), .IN2 (n_1798), .IN3 (n_2931), .QN
      (n_4199));
  OAI21X1 g150729(.IN1 (n_1904), .IN2 (n_1901), .IN3 (n_2875), .QN
      (n_4200));
  AOI21X1 g150730(.IN1 (n_2754), .IN2 (n_2310), .IN3 (n_1935), .QN
      (n_4201));
  OAI21X1 g150731(.IN1 (n_2730), .IN2 (n_2307), .IN3 (n_1746), .QN
      (n_4202));
  OAI21X1 g150732(.IN1 (n_1880), .IN2 (n_1823), .IN3 (n_2714), .QN
      (n_4203));
  AO22X1 g150733(.IN1 (n_1579), .IN2 (n_1577), .IN3 (n_2385), .IN4
      (n_2630), .Q (n_4204));
  AO21X1 g150734(.IN1 (n_2840), .IN2 (n_1595), .IN3 (n_4205), .Q
      (n_4206));
  AOI21X1 g3(.IN1 (n_2314), .IN2 (n_2833), .IN3 (n_2151), .QN (n_4205));
  AO21X1 g150735(.IN1 (n_2658), .IN2 (n_1582), .IN3 (n_4207), .Q
      (n_4208));
  AOI21X1 g150736(.IN1 (n_2384), .IN2 (n_2652), .IN3 (n_2145), .QN
      (n_4207));
  AO21X1 g150737(.IN1 (n_2574), .IN2 (n_1583), .IN3 (n_4209), .Q
      (n_4210));
  AOI21X1 g150738(.IN1 (n_2313), .IN2 (n_2566), .IN3 (n_1699), .QN
      (n_4209));
endmodule
```

5.NETLIST WITHOUT TESTLOGIC (GUI)

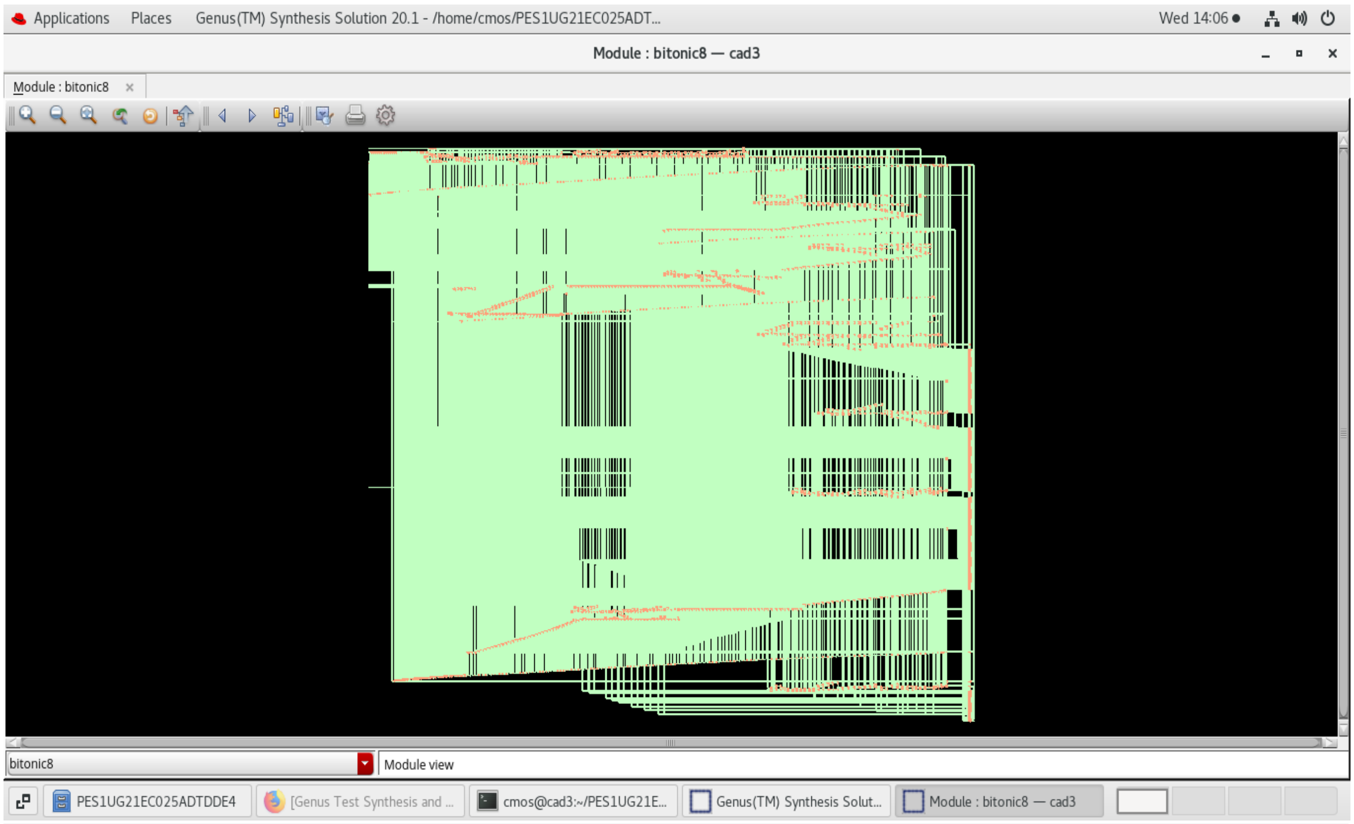


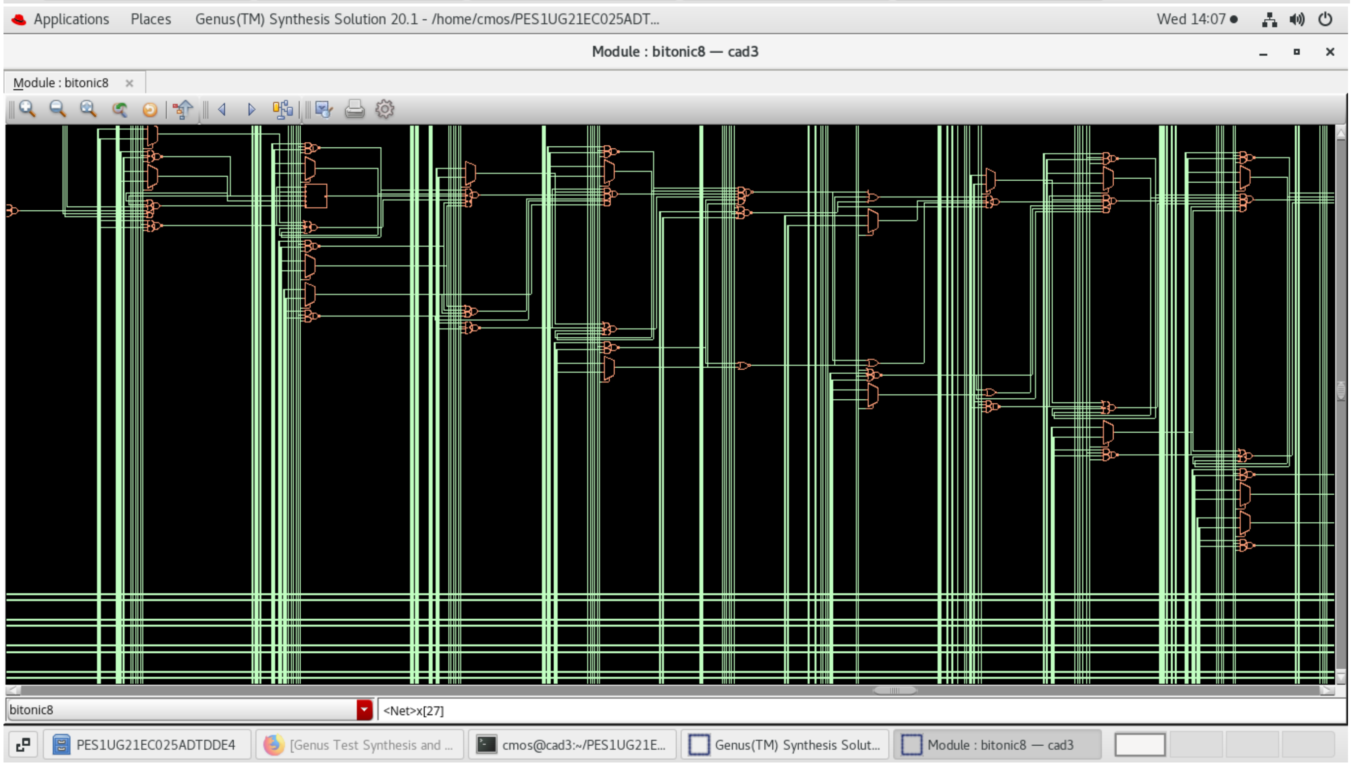


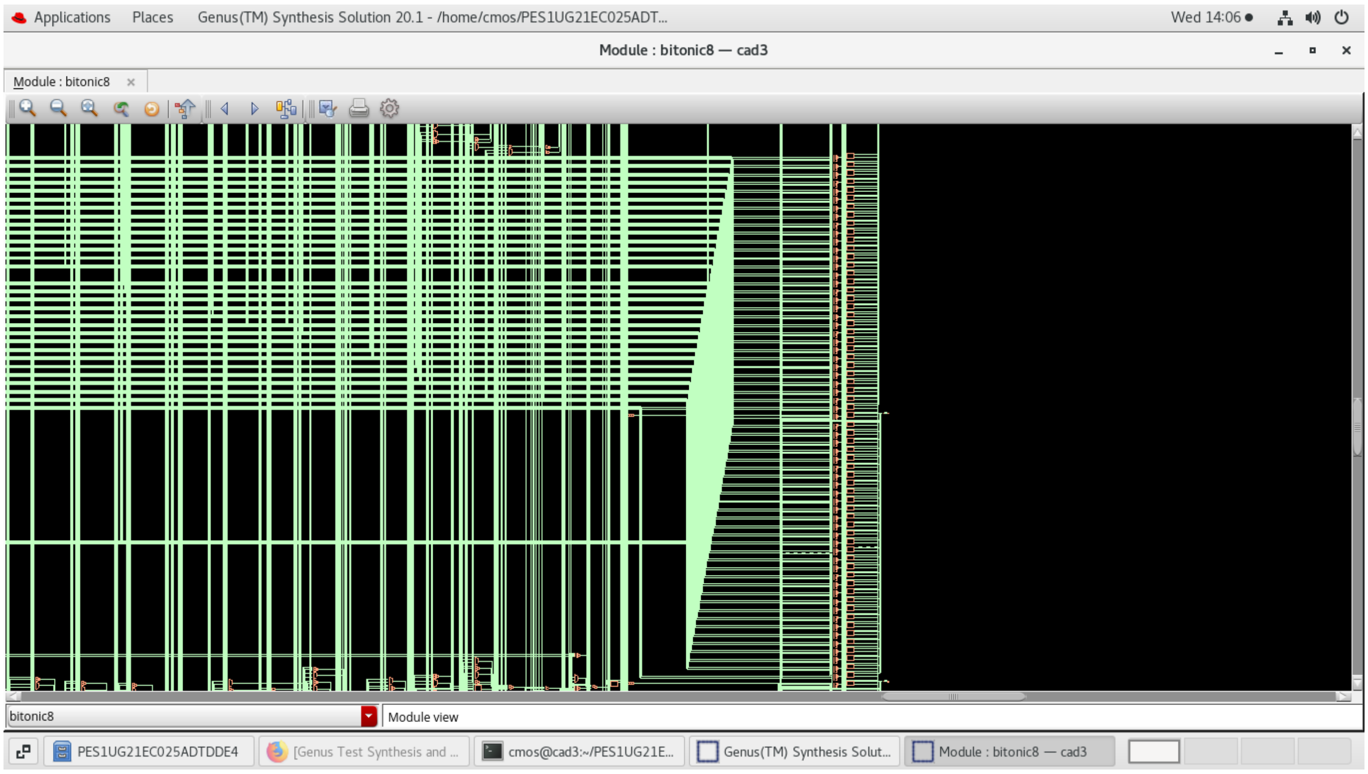

Applications
Places
Module : bitonic8 — cad3
Wed 13:54 ●
Module : bitonic8 — cad3
Module : bitonic8
DFFARX1
DFFARX1
DFFARX1
DFFARX1
AO22X1
AO22X1
AO22X1
AO22X1
bitonic8
Module view
PES1UG21EC025ADTDDE4
[Genus Test Synthesis and ...
cmos@cad3:~/PES1UG21E...
Genus(TM) Synthesis Solut...
Module : bitonic8 — cad3

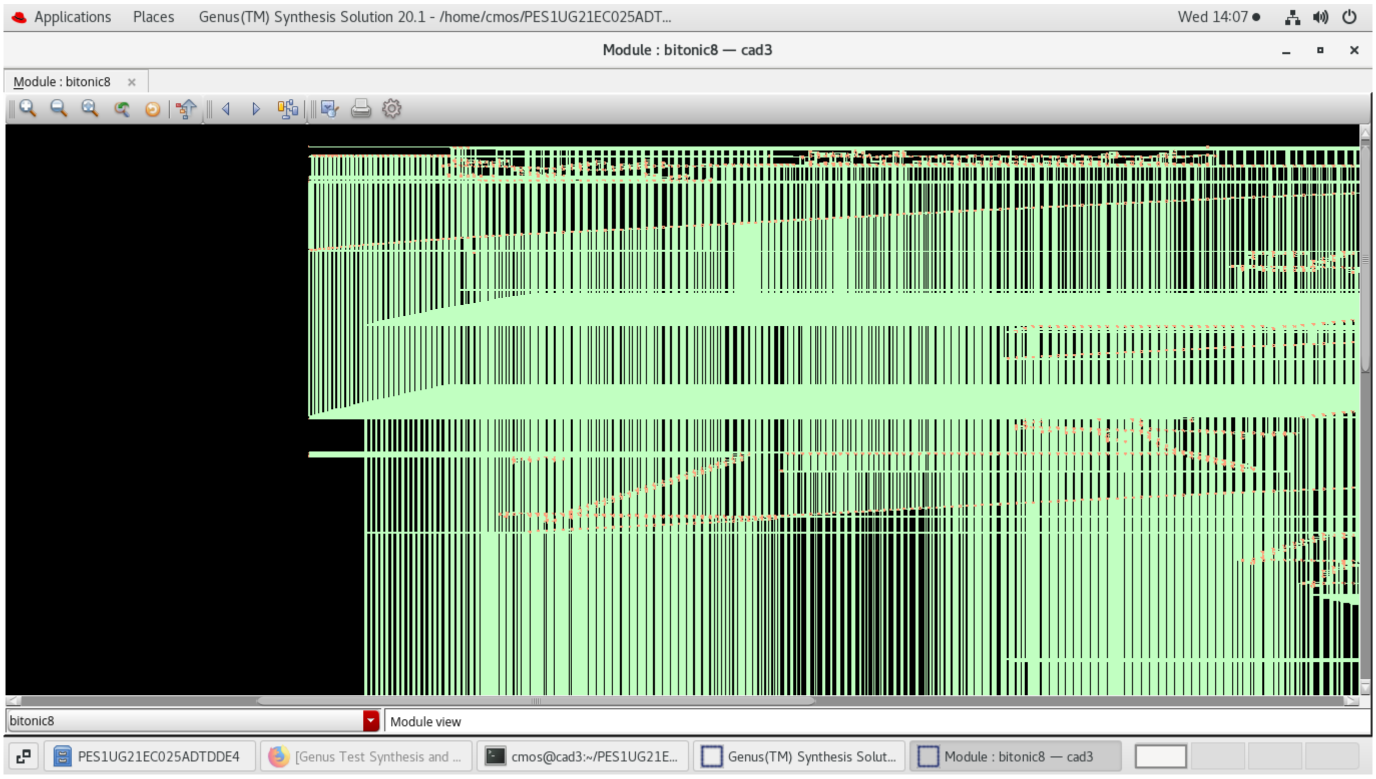

Applications
Places
Module : bitonic8 — cad3
Wed 13:54 ●
Module : bitonic8 — cad3
Module : bitonic8
AO22X1
MUX21X1
INVX0
MUX21X1
AND2X1
bitonic8
Module view
PES1UG21EC025ADTDDE4
[Genus Test Synthesis and ...
cmos@cad3:~/PES1UG21E...
Genus(TM) Synthesis Solut...
Module : bitonic8 — cad3

6.NETLIST AFTER INSERTING TEST LOGIC(GUI):

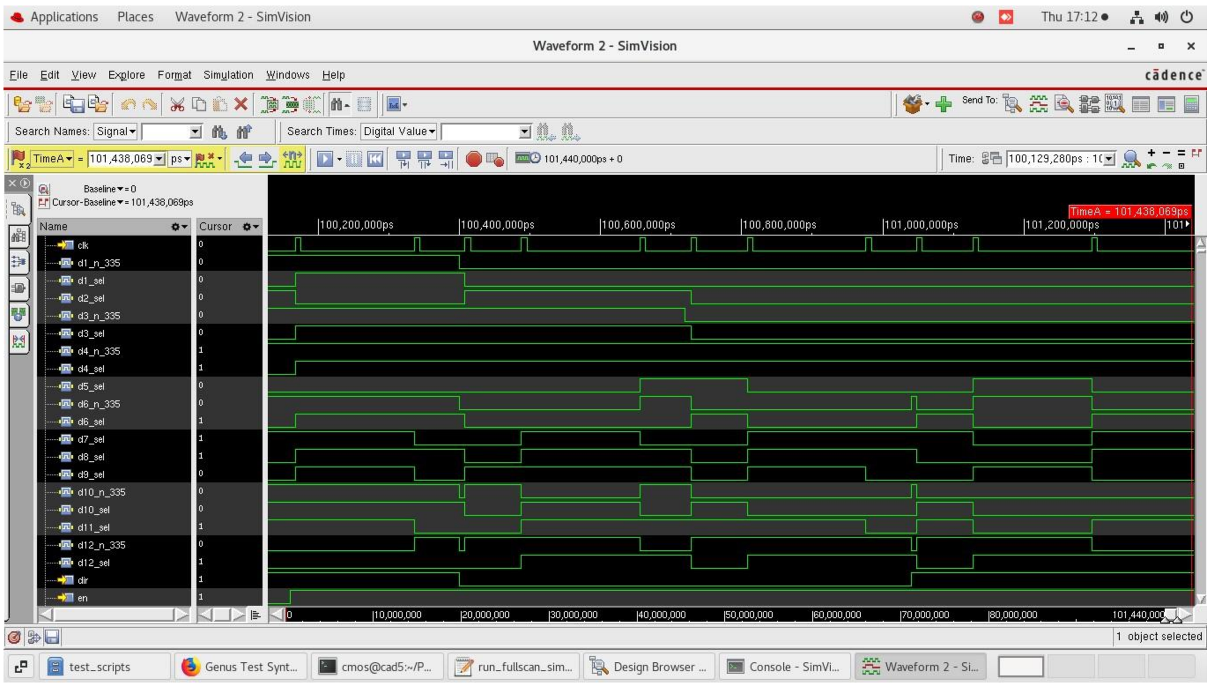




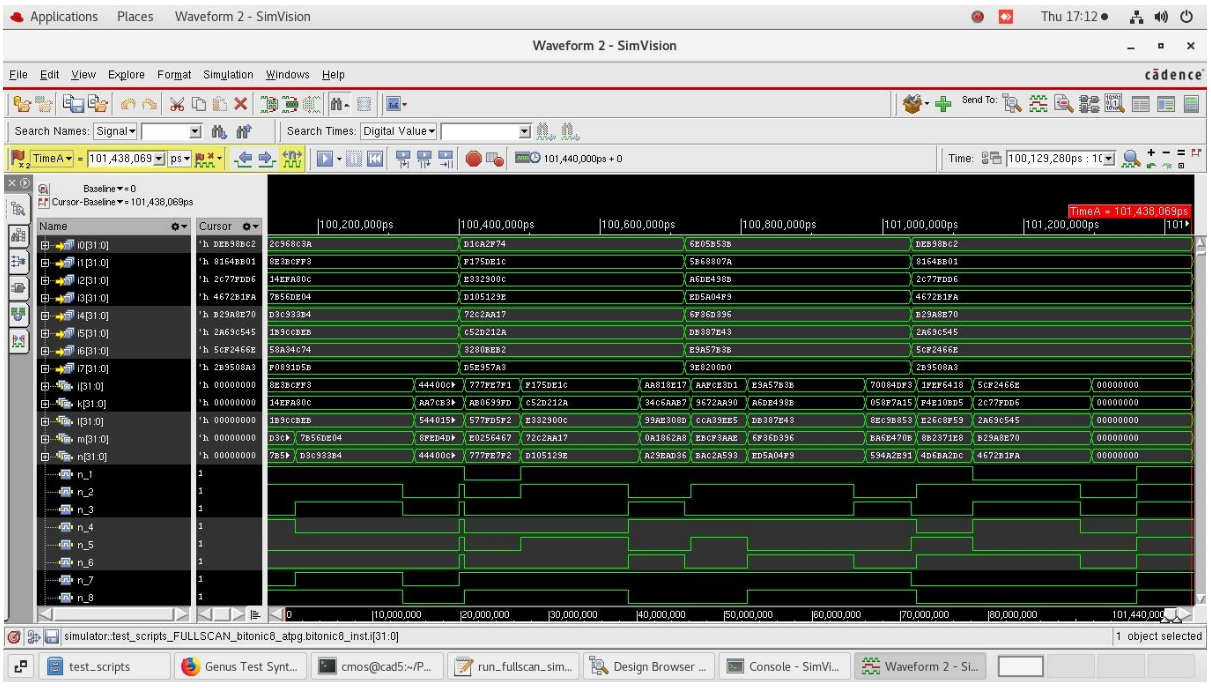

Applications
Places
Genus(TM) Synthesis Solution 20.1 - /home/cmos/PES1UG21EC025ADT...
Wed 14:06
Module : bitonic8 — cad3
Module : bitonic8
bitonic8
Module view
PES1UG21EC025ADTDDE4
[Genus Test Synthesis and ...
cmos@cad3:~/PES1UG21E...
Genus(TM) Synthesis Solut...
Module : bitonic8 — cad3


Applications
Places
Genus(TM) Synthesis Solution 20.1 - /home/cmos/PES1UG21EC025ADT...
Wed 14:07
Module : bitonic8 — cad3
Module : bitonic8
bitonic8
Module view
PES1UG21EC025ADTDDE4
[Genus Test Synthesis and ...
cmos@cad3:~/PES1UG21E...
Genus(TM) Synthesis Solut...
Module : bitonic8 — cad3

7.Xcelium Waveform (GUI):

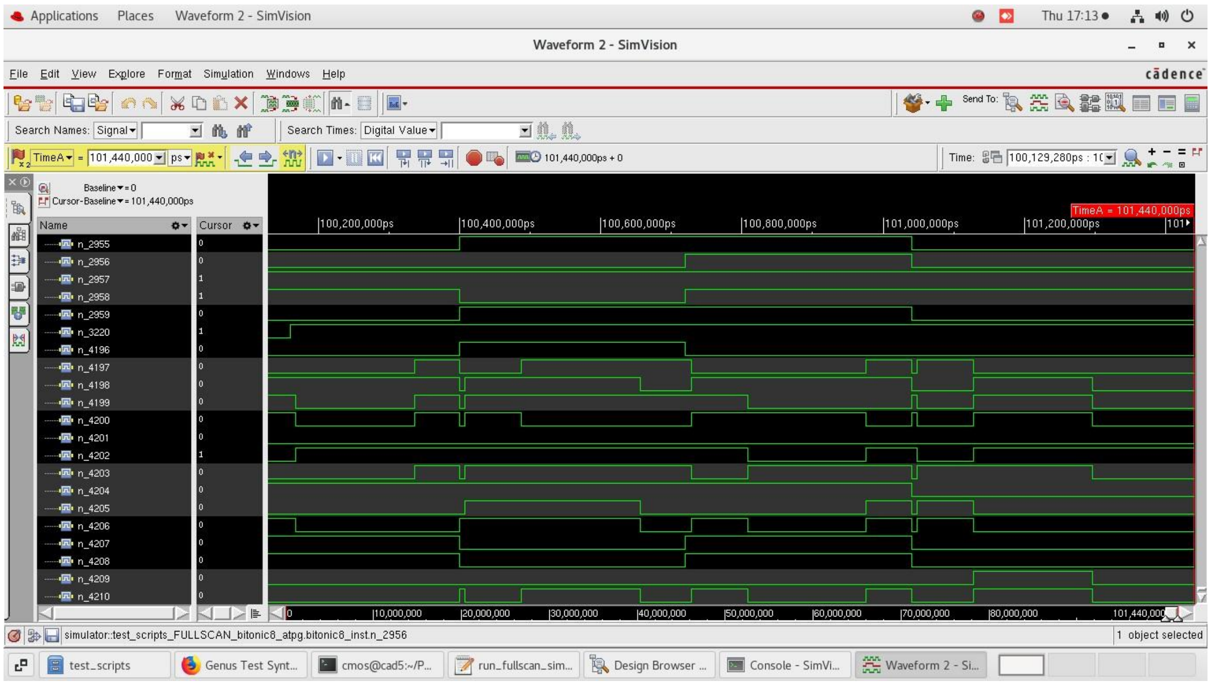


Name	Cursor
clk	0
d1_n_335	0
d1_sel	0
d2_sel	0
d3_n_335	0
d3_sel	0
d4_n_335	1
d4_sel	1
d5_sel	0
d6_n_335	0
d6_sel	1
d7_sel	1
d8_sel	1
d9_sel	0
d10_n_335	0
d10_sel	0
d11_sel	1
d12_n_335	0
d12_sel	1
dir	1
en	1

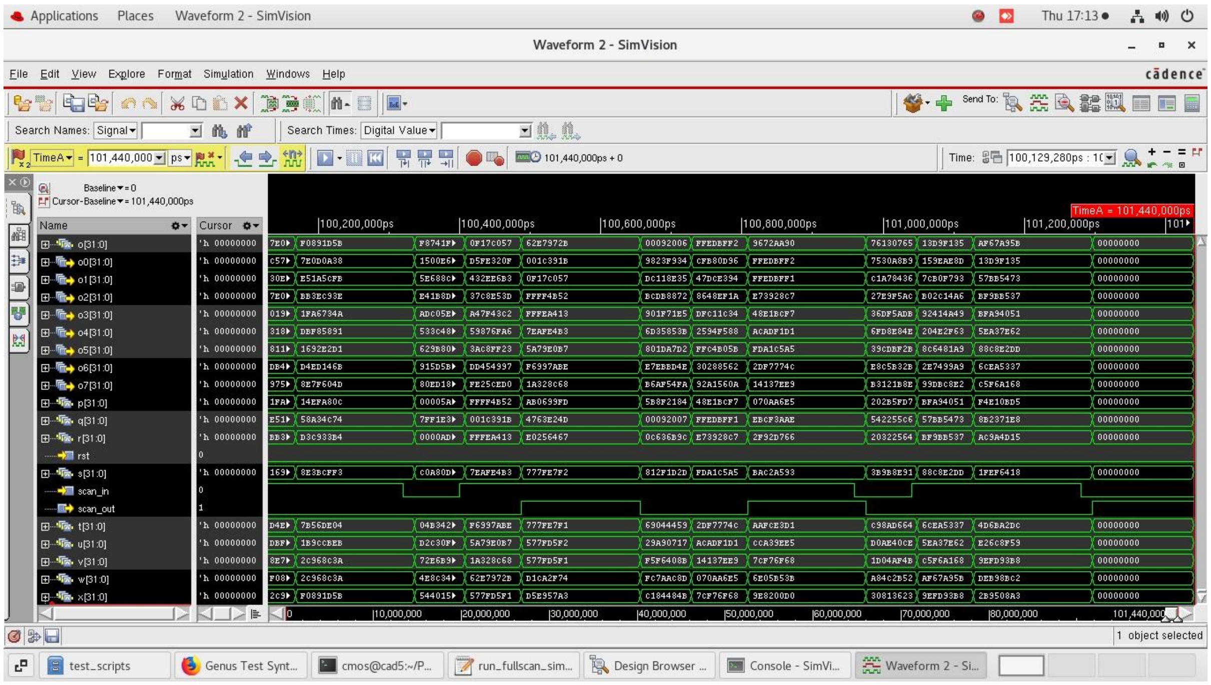


Name	Cursor
i0[31:0]	'h DEB98BC2
i1[31:0]	'h 8164BB01
i2[31:0]	'h 2C77FDD6
i3[31:0]	'h 4672B1FA
i4[31:0]	'h B29A8E70
i5[31:0]	'h 2A69C545
i6[31:0]	'h 5CF2466E
i7[31:0]	'h 2B9508A3
j[31:0]	'h 00000000
k[31:0]	'h 00000000
l[31:0]	'h 00000000
m[31:0]	'h 00000000
n[31:0]	'h 00000000
n_1	1
n_2	1
n_3	1
n_4	1
n_5	1
n_6	1
n_7	1
n_8	1

simulator::test_scripts_FULLSCAN_bitonic8_atpg.bitonic8_inst.i[31:0]



```
Waveform 2 - SimVision
File  Edit  View  Explore  Format  Simulation  Windows  Help
Search Names: Signal    Search Times: Digital Value
TimeA = 101,440,000 ps     101,440,000ps + 0     Time: 100,129,280ps : 1(
Baseline = 0
Cursor-Baseline = 101,440,000ps
TimeA = 101,440,000ps

Name      Cursor
n_2955    0
n_2956    0
n_2957    1
n_2958    1
n_2959    0
n_3220    1
n_4196    0
n_4197    0
n_4198    0
n_4199    0
n_4200    0
n_4201    0
n_4202    1
n_4203    0
n_4204    0
n_4205    0
n_4206    0
n_4207    0
n_4208    0
n_4209    0
n_4210    0

simulator::test_scripts_FULLSCAN_bitonic8_atpg.bitonic8_inst.n_2956      1 object selected
```



```
Waveform 2 - SimVision
TimeA = 101,440,000 ps     101,440,000ps + 0     Time: 100,129,280ps : 1(
Baseline = 0
Cursor-Baseline = 101,440,000ps
TimeA = 101,440,000ps

Name      Cursor
o[31:0]   'h 00000000  7E0▸ F0891D5B F8741F▸ 0F17c057 62E7972B 00092006 FFEDBFF2 9672AA90 76130765 13D9F135 AF67A95B 00000000
o0[31:0]  'h 00000000  c57▸ 7E0D0A38 1500E6▸ D5FE320F 001c391B 9823F934 cFB80D96 FFEDBFF2 7530A8B9 159EAE8D 13D9F135 00000000
o1[31:0]  'h 00000000  30E▸ E51A5cFB 5E688c▸ 432EE6B3 0F17c057 Dc118E35 47DcE394 FFEDBFF1 c1A78436 7cB0F793 57BB5473 00000000
o2[31:0]  'h 00000000  7E0▸ BB3Ec93E E41B8D▸ 37c8E53D FFFF4B52 BcDB8872 8648EF1A E73928c7 27E9F5Ac B02c14A6 BF9BB537 00000000
o3[31:0]  'h 00000000  019▸ 1FA6734A ADc05E▸ A47F43c2 FFFEA413 901F71E5 DFc11c34 48E1BcF7 36DF5ADB 92414A49 BFA94051 00000000
o4[31:0]  'h 00000000  318▸ DBF85891 533c48▸ 59876FA6 7EAFE4B3 6D35853B 2594F588 AcADF1D1 6FD8E84E 204E2F63 5EA37E62 00000000
o5[31:0]  'h 00000000  811▸ 1692E2D1 629B80▸ 3Ac8FF23 5A79E0B7 801DA7D2 FFc4B05B FDA1c5A5 39cDBF2B 8c6481A9 88c8E2DD 00000000
o6[31:0]  'h 00000000  DB4▸ D4ED146B 915D5B▸ DD454997 F6997ABE E7EBBD4E 30288562 2DF7774c E8c5B32B 2E7499A9 6cEA5337 00000000
o7[31:0]  'h 00000000  975▸ 8E7F604D 80ED18▸ FE25cED0 1A328c68 B6AF54FA 92A1560A 14137EE9 B3121B8E 99DBc8E2 c5F6A168 00000000
p[31:0]   'h 00000000  1FA▸ 14EFA80c 00005A▸ FFFF4B52 AB0699FD 5B8F2184 48E1BcF7 070AA6E5 202B5FD7 BFA94051 F4E10BD5 00000000
q[31:0]   'h 00000000  E51▸ 58A34c74 7FF1E3▸ 001c391B 4763E24D 00092007 FFEDBFF1 EBcF3AAE 542255c6 57BB5473 8B2371E8 00000000
r[31:0]   'h 00000000  BB3▸ D3c933B4 0000AD▸ FFFEA413 E0256467 0c636B9c E73928c7 2F92D766 20322564 BF9BB537 Ac9A4D15 00000000
rst       0
s[31:0]   'h 00000000  169▸ 8E3BcFF3 c0A80D▸ 7EAFE4B3 777FE7F2 812F1D2D FDA1c5A5 BAc2A593 3B9B8E91 88c8E2DD 1FEF6418 00000000
scan_in   0
scan_out  1
t[31:0]   'h 00000000  D4E▸ 7B56DE04 04B342▸ F6997ABE 777FE7F1 69044459 2DF7774c AAFcE3D1 c98AD664 6cEA5337 4D6BA2Dc 00000000
u[31:0]   'h 00000000  DBF▸ 1B9ccBEB D2c30F▸ 5A79E0B7 577FD5F2 29A90717 AcADF1D1 ccA39EE5 D0AE40cE 5EA37E62 E26c8F59 00000000
v[31:0]   'h 00000000  8E7▸ 2c968c3A 72E6B9▸ 1A328c68 577FD5F1 F5F6408B 14137EE9 7cF76F68 1D04AF4B c5F6A168 9EFD93B8 00000000
w[31:0]   'h 00000000  F08▸ 2c968c3A 4E8c34▸ 62E7972B D1cA2F74 Fc7AAc8D 070AA6E5 6E05B53B A84c2B52 AF67A95B DEB98Bc2 00000000
x[31:0]   'h 00000000  2c9▸ F0891D5B 544015▸ 577FD5F1 D5E957A3 c184484B 7cF76F68 9E8200D0 30813623 9EFD93B8 2B9508A3 00000000

1 object selected
```

CONCLUSION:

In this project, we have successfully written the CAE block for bitonic sorter and have verified the sorting procedure with 8 inputs.We added test logic to this module and obtained Xcelium Waveform.

REFERENCE:

1. "Digital Design and Computer Architecture" by David Money Harris and Sarah L. Harris.

2. "Verilog HDL: A Guide to Digital Design and Synthesis" by Samir Palnitkar.

3. "Digital System Design with VHDL" by Mark Zwolinski.

4. https://en.wikipedia.org/wiki/Bitonic_sorter