64-bit Hierarchical Carry Look Ahead Adder Design and Testing

1. Objective

The purpose of this project is to design and test a 64-bit hierarchical Carry Look Ahead Adder (CLA) using Verilog HDL. The design will be simulated, synthesized, and implemented on a FPGA board.

2. RTL Design

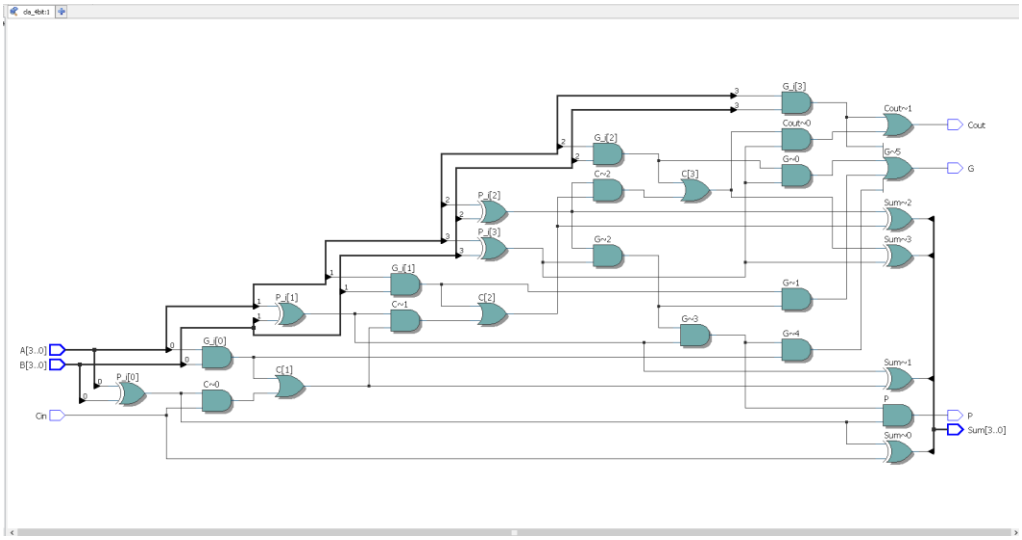
2.1 Verilog Code for CLA

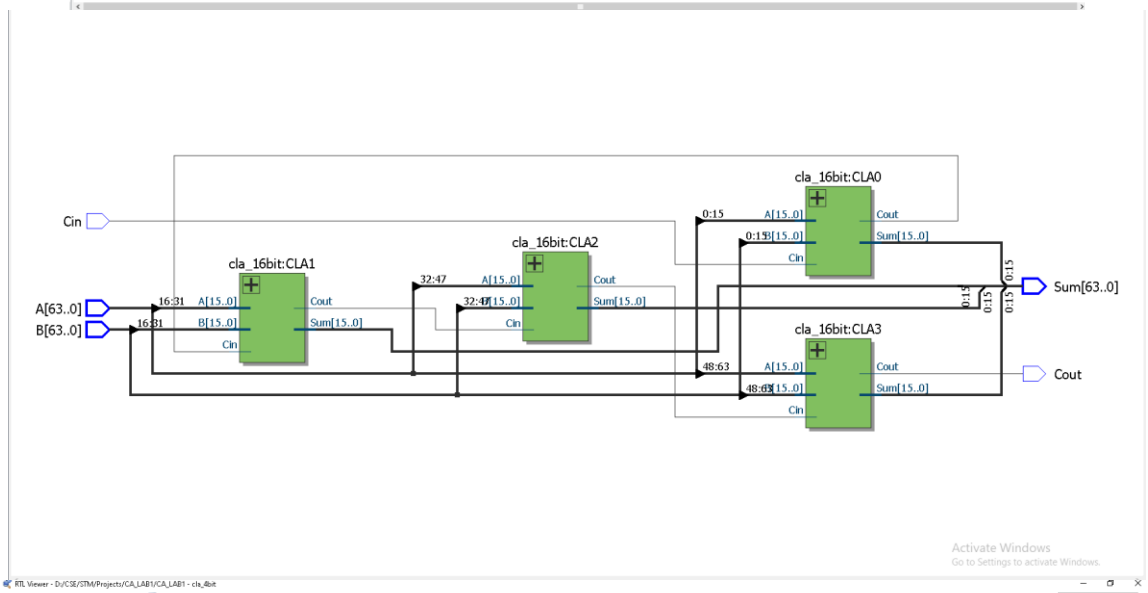
```
module cla_4bit (
    input [3:0] A, B,       // 4-bit inputs
    input Cin,              // Carry-in
    output [3:0] Sum,       // 4-bit sum
    output Cout,            // Carry-out
    output G, P             // Group Generate and Propagate
);
    wire [3:0] G_i, P_i;    // Generate and Propagate signals for each bit
    wire [3:1] C;           // Internal carry signals

    // Generate and Propagate calculations
    assign G_i = A & B;     // G_i = A_i AND B_i
    assign P_i = A ^ B;     // P_i = A_i XOR B_i

    // Carry calculations
    assign C[1] = G_i[0] | (P_i[0] & Cin);
    assign C[2] = G_i[1] | (P_i[1] & C[1]);
    assign C[3] = G_i[2] | (P_i[2] & C[2]);
    assign Cout = G_i[3] | (P_i[3] & C[3]);

    // Sum calculations
    assign Sum[0] = P_i[0] ^ Cin;
    assign Sum[1] = P_i[1] ^ C[1];
    assign Sum[2] = P_i[2] ^ C[2];
    assign Sum[3] = P_i[3] ^ C[3];

    // Group Generate and Propagate
    assign G = G_i[3] | (P_i[3] & G_i[2]) | (P_i[3] & P_i[2] & G_i[1]) | (P_i[3] & P_i[2] & P_i[1] & G_i[0]);
    assign P = P_i[3] & P_i[2] & P_i[1] & P_i[0];
endmodule
```

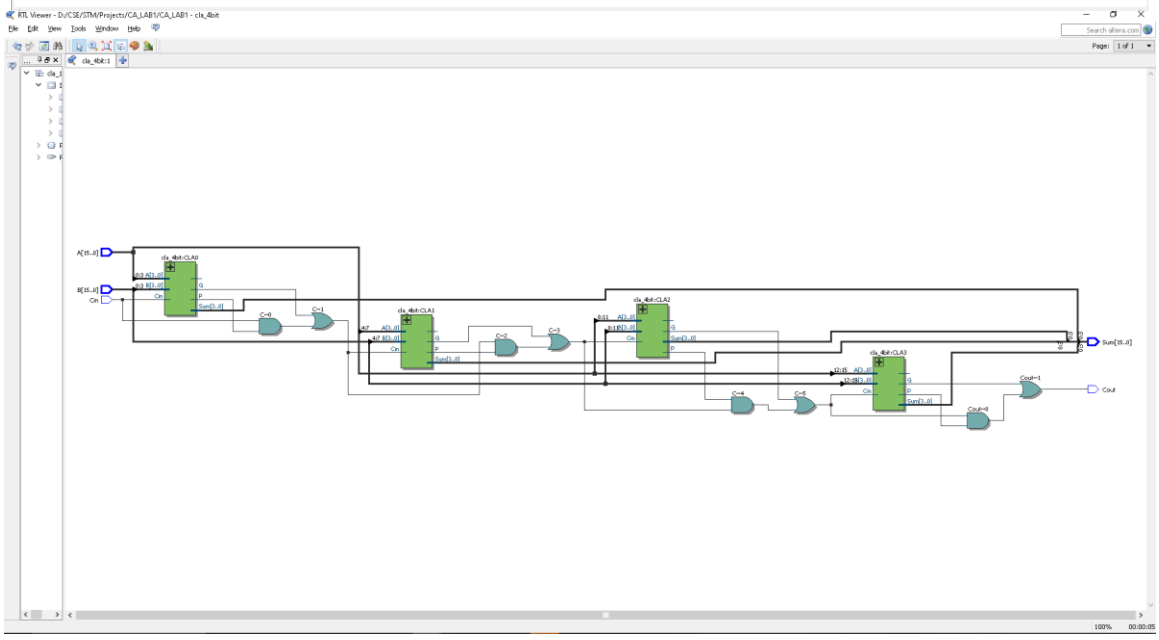
```
module cla_16bit (
    input [15:0] A, B,      // 16-bit inputs
    input Cin,              // Carry-in
    output [15:0] Sum,      // 16-bit sum
    output Cout             // Carry-out
);
    wire [3:0] G, P;        // Group Generate and Propagate for each 4-bit block
    wire [3:0] C;           // Carry signals between the blocks

    // 4-bit CLA instances
    cla_4bit CLA0 (.A(A[3:0]),   .B(B[3:0]),   .Cin(Cin),   .Sum(Sum[3:0]),   .Cout(), .G(G[0]), .P(P[0]));
    cla_4bit CLA1 (.A(A[7:4]),   .B(B[7:4]),   .Cin(C[0]),  .Sum(Sum[7:4]),   .Cout(), .G(G[1]), .P(P[1]));
    cla_4bit CLA2 (.A(A[11:8]),  .B(B[11:8]),  .Cin(C[1]),  .Sum(Sum[11:8]),  .Cout(), .G(G[2]), .P(P[2]));
    cla_4bit CLA3 (.A(A[15:12]), .B(B[15:12]), .Cin(C[2]),  .Sum(Sum[15:12]), .Cout(), .G(G[3]), .P(P[3]));

    // Carry calculations for 4-bit blocks
    assign C[0] = G[0] | (P[0] & Cin);
    assign C[1] = G[1] | (P[1] & C[0]);
    assign C[2] = G[2] | (P[2] & C[1]);
    assign Cout = G[3] | (P[3] & C[2]);
endmodule
```

```
module cla_64bit (
    input [63:0] A, B,      // 64-bit inputs
    input Cin,              // Carry-in
    output [63:0] Sum,      // 64-bit sum
    output Cout             // Final carry-out
);
    wire [3:0] C;           // Carry signals between the 16-bit CLA blocks

    // 16-bit CLA instances
    cla_16bit CLA0 (.A(A[15:0]),   .B(B[15:0]),   .Cin(Cin),   .Sum(Sum[15:0]),   .Cout(C[0]));
    cla_16bit CLA1 (.A(A[31:16]),  .B(B[31:16]),  .Cin(C[0]),  .Sum(Sum[31:16]),  .Cout(C[1]));
    cla_16bit CLA2 (.A(A[47:32]),  .B(B[47:32]),  .Cin(C[1]),  .Sum(Sum[47:32]),  .Cout(C[2]));
    cla_16bit CLA3 (.A(A[63:48]),  .B(B[63:48]),  .Cin(C[2]),  .Sum(Sum[63:48]),  .Cout(Cout));
endmodule
```

2.2 RTL Schematic


A[3..0]
B[3..0]
Cin
G_i[0]
P_i[0]
C~0
C[1]
G_i[1]
P_i[1]
C~1
C[2]
G_i[2]
P_i[2]
C~2
C[3]
G_i[3]
P_i[3]
G~2
G~3
G~4
G~1
G~0
G~5
Cout~1
Cout~0
Cout
G
Sum~2
Sum~3
Sum~1
Sum~0
P
Sum[3..0]


Cin
A[63..0]
B[63..0]
cla_16bit:CLA1
cla_16bit:CLA2
cla_16bit:CLA0
cla_16bit:CLA3
16:31
32:47
0:15
48:63
A[15..0]
B[15..0]
Cout
Sum[15..0]
Sum[63..0]



3. Simulation Results

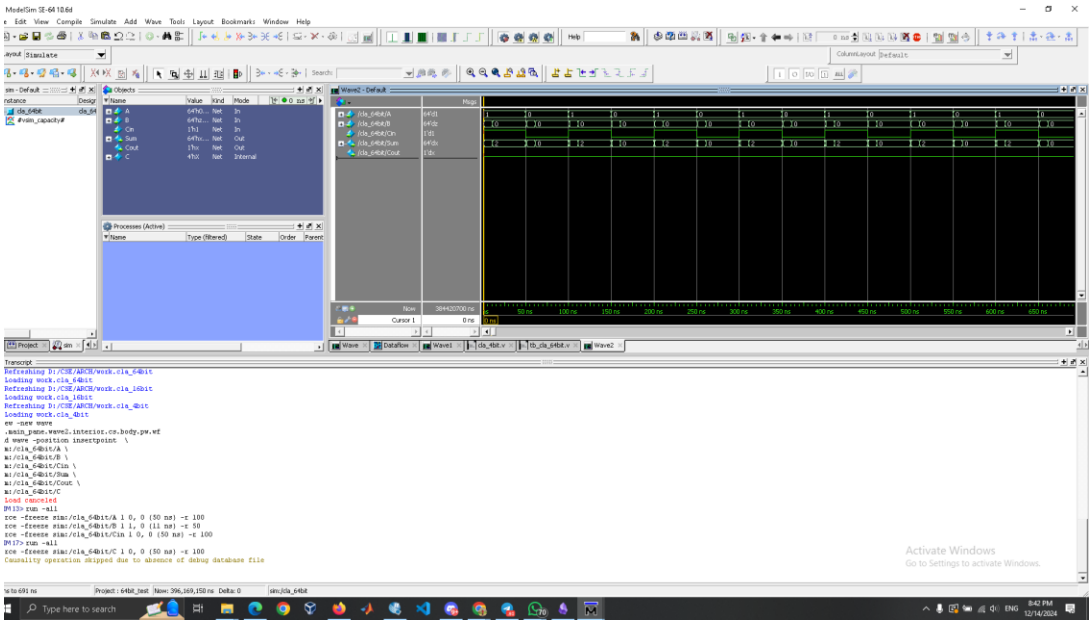
3.1 Testbench

```
VSIM 4> run -all
# At time                10000, a = 0000000000000001, b = 0000000000000001, cin = 0, sum = 0000000000000002, cout = 0
# At time                20000, a = 00000000000000ff, b = 0000000000000001, cin = 0, sum = 0000000000000100, cout = 0
# At time                30000, a = ffffffffffffffff, b = 0000000000000001, cin = 0, sum = 0000000000000000, cout = 1
# At time                40000, a = aaaaaaaaaaaaaaaa, b = 5555555555555555, cin = 0, sum = ffffffffffffffff, cout = 0
# At time                50000, a = fffffffffffffff0, b = 000000000000000f, cin = 1, sum = 0000000000000000, cout = 1
# At time                60000, a = 0000000000000000, b = 0000000000000000, cin = 1, sum = 0000000000000001, cout = 0
# At time                70000, a = 123456789abcdef0, b = 0fedcba987654321, cin = 1, sum = 2222222222222212, cout = 0
# At time                80000, a = 0000ffff0000ffff, b = ffff0000ffff0000, cin = 0, sum = ffffffffffffffff, cout = 0
# ** Note: $stop    : D:/CSE/STM/Projects/CA_LAB1/tb_cla_64bit.v(41)
#    Time: 90 ns  Iteration: 0  Instance: /tb_cla_64bit
# Break at D:/CSE/STM/Projects/CA_LAB1/tb_cla_64bit.v line 41
```

```
Ln#
 4  parameter N = 64; // 64-bit inputs
 5
 6  // Inputs
 7  reg [N-1:0] a;
 8  reg [N-1:0] b;
 9  reg cin;
10
11  // Outputs
12  wire [N-1:0] sum;
13  wire cout;
14
15  // Instantiate the Unit Under Test (UUT)
16  cla_64bit uut (
17      .A(a),
18      .B(b),
19      .Cin(cin),
20      .Sum(sum),
21      .Cout(cout)
22  );
23
24  initial begin
25      // Initialize Inputs
26      a = 0;
27      b = 0;
28      cin = 0;
29
30      // Apply test vectors
31      #10 a = 64'h0000000000000001; b = 64'h0000000000000001; cin = 0;  // 1 + 1 = 2
32      #10 a = 64'h00000000000000FF; b = 64'h0000000000000001; cin = 0; // 255 + 1 = 256
33      #10 a = 64'hFFFFFFFFFFFFFFFF; b = 64'h0000000000000001; cin = 0; // Max 64-bit + 1 = 0 with carry-out
34      #10 a = 64'hAAAAAAAAAAAAAAAA; b = 64'h5555555555555555; cin = 0; // Patterned inputs: 170... + 85... = All ones
35      #10 a = 64'hFFFFFFFFFFFFFFF0; b = 64'h000000000000000F; cin = 1; // Edge case: High carry propagation
36      #10 a = 64'h0000000000000000; b = 64'h0000000000000000; cin = 1; // 0 + 0 + 1 = 1
37      #10 a = 64'h123456789ABCDEF0; b = 64'h0FEDCBA987654321; cin = 1; // Random inputs with carry-in
38      #10 a = 64'h0000FFFF0000FFFF; b = 64'hFFFF0000FFFF0000; cin = 0; // Mixed patterns
39
40      // Add more test cases as needed
41      #10 $stop; // Stop the simulation
42  end
43
44  initial begin
45      $monitor("At time %t, a = %h, b = %h, cin = %b, sum = %h, cout = %b",
46               $time, a, b, cin, sum, cout);
47  end
48
49  endmodule
50
```

Wave | Dataflow | Wave1 | cla_4bit.v | tb_cla_64bit.v | Wave2

3.2 Waveforms

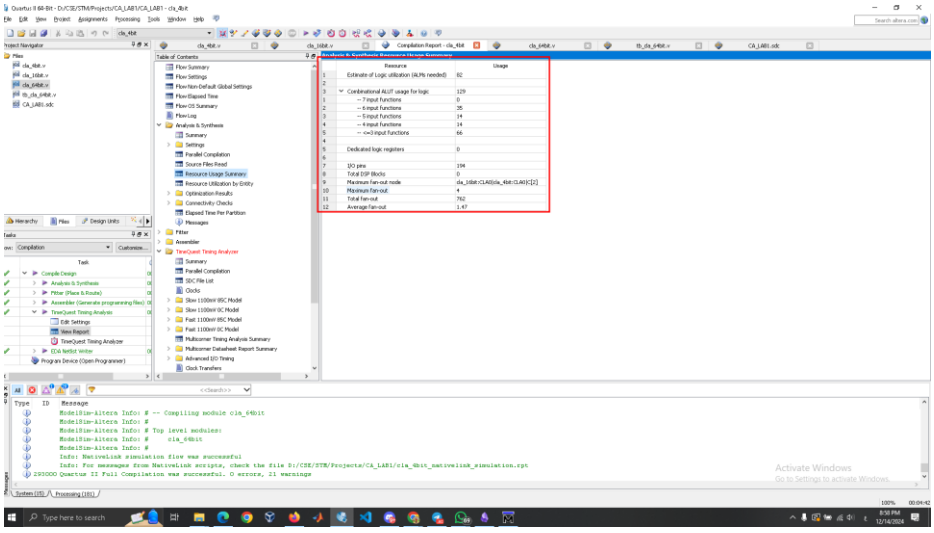


4. FPGA Synthesis and Implementation

4.1 Target Board

- FPGA Board: 5CSXFC5D6F31

4.2 Utilization Results



4.3 Timing Report