schematic verification:-

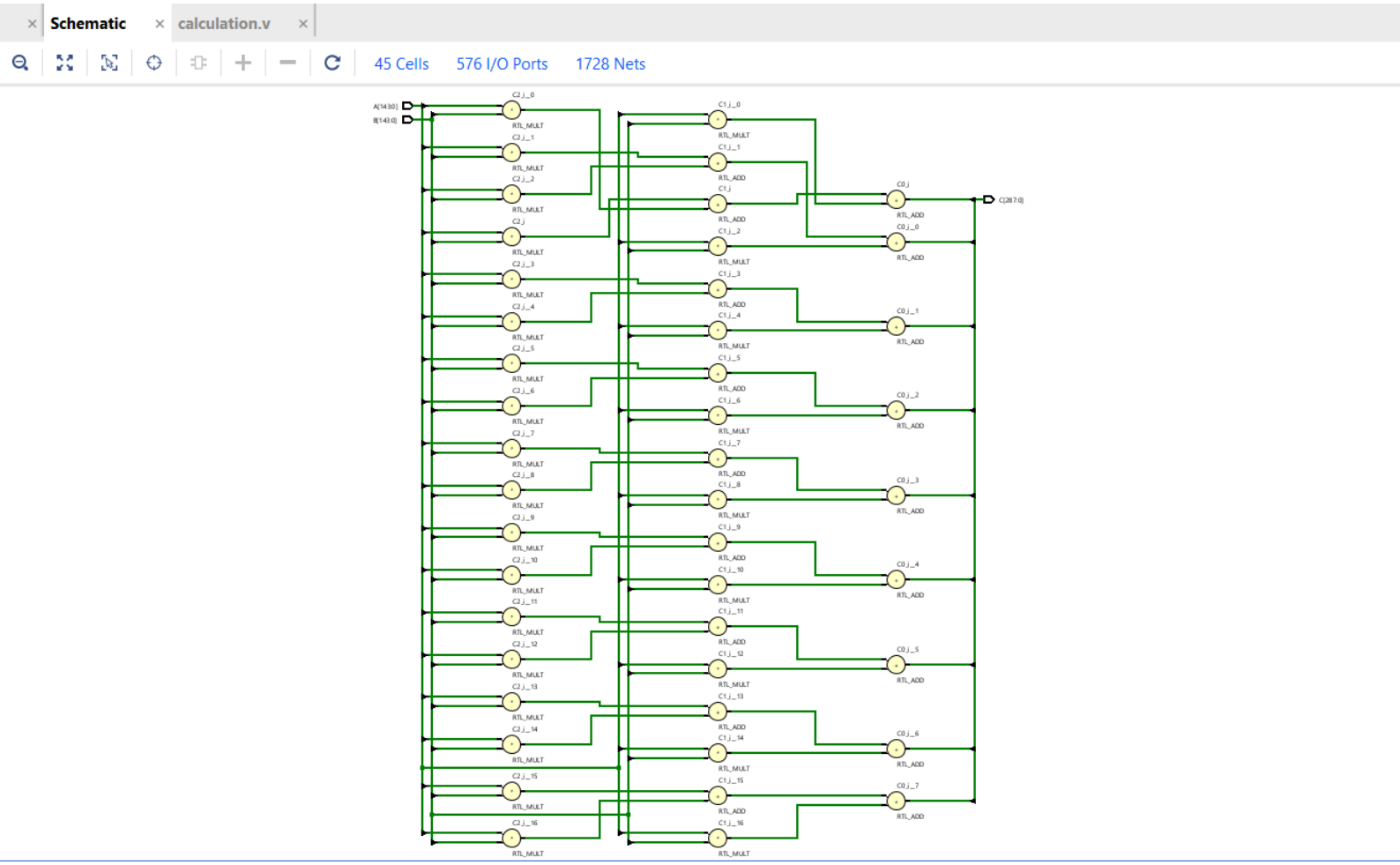
```
`timescale 1ns / 1ps

module multiplier(input wire [15:0] a, input wire [15:0] b, output wire [31:0] product);

   assign product = a * b;

endmodule

module adder(input wire [31:0] a, input wire [31:0] b, output wire [31:0] sum);

   assign sum = a + b;

endmodule

module calculation( input wire [143:0] A,input wire [143:0] B,output wire [287:0] C );

   wire [15:0] A1 [0:2][0:2];wire [15:0] B1 [0:2][0:2];wire [31:0] C1 [0:2][0:2];

   wire [31:0] P[0:2][0:2][0:2]; wire [31:0] S1[0:2][0:2];wire [31:0] S2[0:2][0:2];

   assign {A1[0][0], A1[0][1], A1[0][2], A1[1][0], A1[1][1], A1[1][2], A1[2][0], A1[2][1],
A1[2][2]} = A;

   assign {B1[0][0], B1[0][1], B1[0][2], B1[1][0], B1[1][1], B1[1][2], B1[2][0], B1[2][1],
B1[2][2]} = B;

   genvar i, j, k;

   generate

      for (i = 0; i < 3; i = i + 1) begin : loop_i

         for (j = 0; j < 3; j = j + 1) begin : loop_j

            for (k = 0; k < 3; k = k + 1) begin : loop_k

               multiplier mult(.a(A1[i][k]), .b(B1[k][j]), .product(P[i][j][k]));

            end

         end

      end

   endgenerate
```

```
  generate
    for (i = 0; i < 3; i = i + 1) begin : sum_i
      for (j = 0; j < 3; j = j + 1) begin : sum_j
        adder add1(.a(P[i][j][0]), .b(P[i][j][1]), .sum(S1[i][j]));
        adder add2(.a(S1[i][j]), .b(P[i][j][2]), .sum(C1[i][j]));
      end
    end
  endgenerate
  assign C = {C1[0][0], C1[0][1], C1[0][2], C1[1][0], C1[1][1], C1[1][2], C1[2][0], C1[2][1], C1[2][2]};
endmodule
module tb_calculation;
  reg [143:0] A;reg [143:0] B;wire [287:0] C;
  calculation uut (.A(A),.B(B),.C(C));
  task print_matrix(input [287:0] matrix);
  begin
    $display("Matrix Result:");
    $display("%0d %0d %0d", matrix[287:256], matrix[255:224], matrix[223:192]);
    $display("%0d %0d %0d", matrix[191:160], matrix[159:128], matrix[127:96]);
    $display("%0d %0d %0d", matrix[95:64], matrix[63:32], matrix[31:0]);
    $display("");
   end
  endtask
```

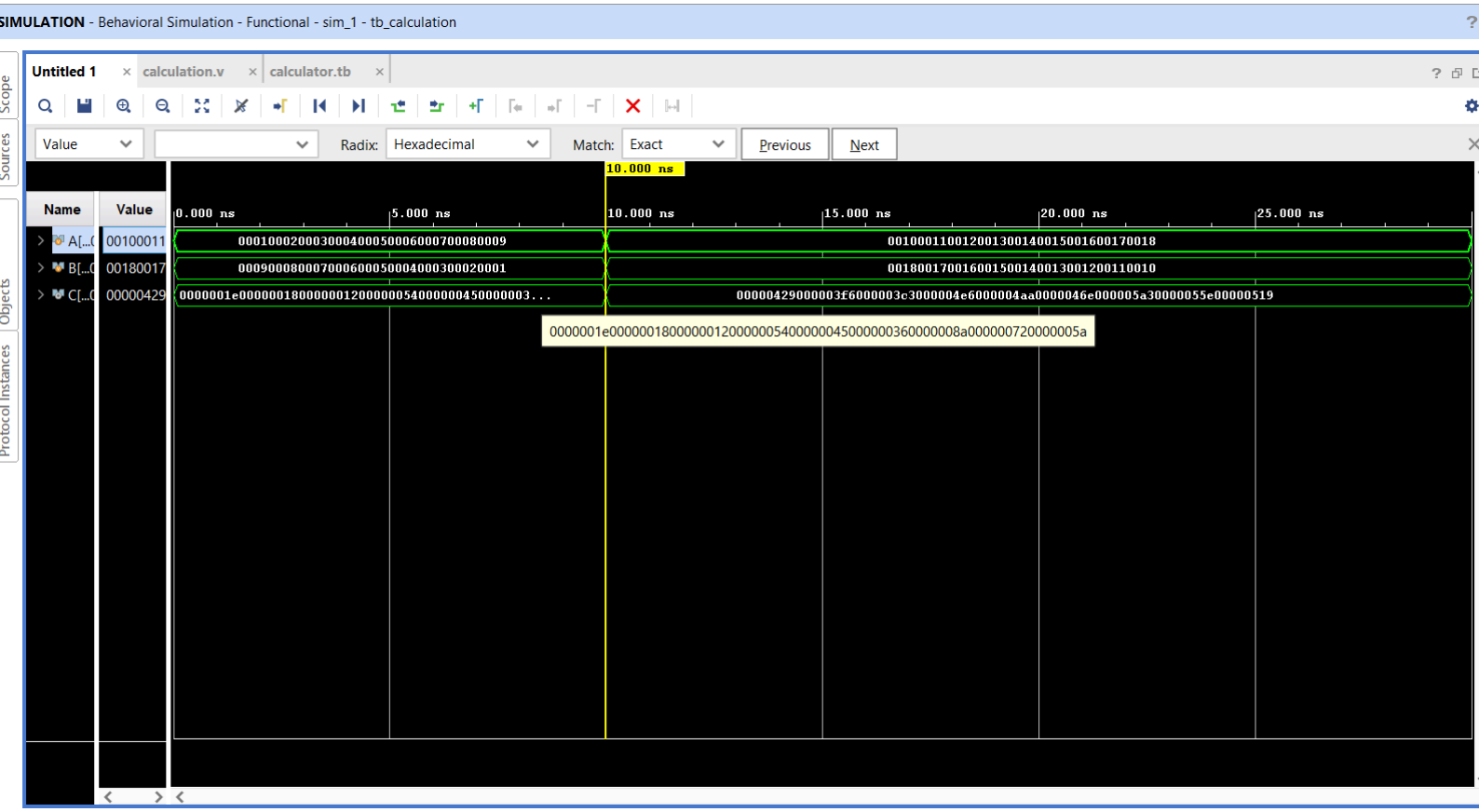
Testbench:-

```
  initial begin

    A = 144'd0;

    B = 144'd0;

    #10;

    A = {16'd1, 16'd2, 16'd3, 16'd4, 16'd5, 16'd6, 16'd7, 16'd8, 16'd9}; // Matrix A: [1 2 3; 4 5 6; 7 8 9]

    B = {16'd9, 16'd8, 16'd7, 16'd6, 16'd5, 16'd4, 16'd3, 16'd2, 16'd1}; // Matrix B: [9 8 7; 6 5 4; 3 2 1]

    #20;

    print_matrix(C);

    #10;

    A = {16'd1, 16'd0, 16'd0, 16'd0, 16'd1, 16'd0, 16'd0, 16'd0, 16'd1};

    B = {16'd9, 16'd8, 16'd7, 16'd6, 16'd5, 16'd4, 16'd3, 16'd2, 16'd1};

    #20;

    print_matrix(C);

    #10;

    A = {16'd0, 16'd0, 16'd0, 16'd0, 16'd0, 16'd0, 16'd0, 16'd0, 16'd0};

    B = {16'd9, 16'd8, 16'd7, 16'd6, 16'd5, 16'd4, 16'd3, 16'd2, 16'd1};

    #20;

    print_matrix(C);

    #10;

    $stop;

  end

endmodule
```

schematic:-



Waveform:-


SIMULATION - Behavioral Simulation - Functional - sim_1 - tb_calculation
Untitled 1
calculation.v
calculator.tb
Value
Radix: Hexadecimal
Match: Exact
Previous
Next
10.000 ns
Name
Value
0.000 ns
5.000 ns
10.000 ns
15.000 ns
20.000 ns
25.000 ns
A[...
00100011
B[...
00180017
C[...
00000429
0000001e000000180000001200000054000000450000003...
00000429000003f6000003c3000004e6000004aa0000046e000005a30000055e00000519
0000001e000000180000001200000054000000450000003600000008a000000720000005a
Scope
Sources
Objects
Protocol Instances