CA_Lab3

K. Ram Mohan

COE19B055

Verilog Code:

```
// Module for Full adder
module FA (in0, in1, cin, sum);
   input in0, in1, cin;
   output sum;

   xor(sum, in0, in1, cin);
endmodule

// Module fpr recursive doubling based adder
module RCDA (a, b, c_input, sum, carry);
   input[63:0] a, b;
   input c_input;
   output[63:0] sum;
   output carry;

   // 00-> kill, 01->propagate, 10->generate
   reg[1:0] kgp[0:63];
   reg[63:0] Cin;

   reg[1:0] tmp1, tmp2;
   reg[1:0] kgp2[0:63];
   reg[1:0] kgp_t1[0:63];
   reg[1:0] kgp_t2[0:63];
```

```
reg[6:0] i, j, k, m, n;

always @(*) begin
   // tmp[1] = a[0]&b[0] | a[0]&cin | b[0]&cin;
   // tmp[0] = a[0]&b[0] | a[0]&cin | b[0]&cin;
   // kgp[0] = tmp;

   // Calculating kgp for 0th bit based on carry input given
   if(a[0]==0 & b[0]==0)
      kgp[0] = 2'b00;
   else if(a[0]==1 & b[0]==1)
      kgp[0] = 2'b10;
   else if((a[0]==1 & b[0]==0) | (a[0]==0 & b[0]==1)) begin
      if(c_input==1)
         kgp[0] = 2'b10;
      else
         kgp[0] = 2'b00;
   end

   i = 7'b0000001;
   // While loop to calculate kgp for remaining bits based on value of 'a' and 'b'
   while(i<=63) begin

      if(a[i]==0 & b[i]==0)
         kgp[i] = 2'b00;
      else if(a[i]==1 & b[i]==1)
         kgp[i] = 2'b10;
      else
         kgp[i] = 2'b01;
```

```
   i=i+1;
end

// Making a temporary kgp's for modification during recursive doubling
for (i = 0; i<=63; i=i+1) begin
   kgp2[i] = kgp[i];
   kgp_t1[i] = kgp[i];
end

i=7'b0000001;
while(i<=63) begin
   j = i;
   while(j<=63) begin
      tmp1 = kgp_t1[j-i];
      tmp2 = kgp_t1[j];

      if (tmp2==2'b00) begin
         kgp2[j] = 2'b00;
      end
      else if(tmp2==2'b10) begin
         kgp2[j] = 2'b10;
      end
      else begin
         kgp2[j] = tmp1;
      end

      j=j+1;
   end
```

```
        for(k=0; k<=63; k=k+1) begin
            kgp_t1[k] = kgp2[k];
        end

        i=i<<1;
    end

    i=7'b0000000;
    while(i<=63) begin
        if(kgp2[i]==2'b10) begin
            Cin[i] = 1;
        end
        else begin
            Cin[i] = 0;
        end
        i=i+1;
    end
end


FA fa1(a[0], b[0], c_input, sum[0]);
FA fa2(a[1], b[1], Cin[0], sum[1]);
FA fa3(a[2], b[2], Cin[1], sum[2]);
FA fa4(a[3], b[3], Cin[2], sum[3]);
FA fa5(a[4], b[4], Cin[3], sum[4]);
FA fa6(a[5], b[5], Cin[4], sum[5]);
FA fa7(a[6], b[6], Cin[5], sum[6]);
FA fa8(a[7], b[7], Cin[6], sum[7]);
```

```
FA fa9(a[8], b[8], Cin[7], sum[8]);
FA fa10(a[9], b[9], Cin[8], sum[9]);
FA fa11(a[10], b[10], Cin[9], sum[10]);
FA fa12(a[11], b[11], Cin[10], sum[11]);
FA fa13(a[12], b[12], Cin[11], sum[12]);
FA fa14(a[13], b[13], Cin[12], sum[13]);
FA fa15(a[14], b[14], Cin[13], sum[14]);
FA fa16(a[15], b[15], Cin[14], sum[15]);
FA fa17(a[16], b[16], Cin[15], sum[16]);
FA fa18(a[17], b[17], Cin[16], sum[17]);
FA fa19(a[18], b[18], Cin[17], sum[18]);
FA fa20(a[19], b[19], Cin[18], sum[19]);
FA fa21(a[20], b[20], Cin[19], sum[20]);
FA fa22(a[21], b[21], Cin[20], sum[21]);
FA fa23(a[22], b[22], Cin[21], sum[22]);
FA fa24(a[23], b[23], Cin[22], sum[23]);
FA fa25(a[24], b[24], Cin[23], sum[24]);
FA fa26(a[25], b[25], Cin[24], sum[25]);
FA fa27(a[26], b[26], Cin[25], sum[26]);
FA fa28(a[27], b[27], Cin[26], sum[27]);
FA fa29(a[28], b[28], Cin[27], sum[28]);
FA fa30(a[29], b[29], Cin[28], sum[29]);
FA fa31(a[30], b[30], Cin[29], sum[30]);
FA fa32(a[31], b[31], Cin[30], sum[31]);
FA fa33(a[32], b[32], Cin[31], sum[32]);
FA fa34(a[33], b[33], Cin[32], sum[33]);
FA fa35(a[34], b[34], Cin[33], sum[34]);
FA fa36(a[35], b[35], Cin[34], sum[35]);
FA fa37(a[36], b[36], Cin[35], sum[36]);
```

```
FA fa38(a[37], b[37], Cin[36], sum[37]);
FA fa39(a[38], b[38], Cin[37], sum[38]);
FA fa40(a[39], b[39], Cin[38], sum[39]);
FA fa41(a[40], b[40], Cin[39], sum[40]);
FA fa42(a[41], b[41], Cin[40], sum[41]);
FA fa43(a[42], b[42], Cin[41], sum[42]);
FA fa44(a[43], b[43], Cin[42], sum[43]);
FA fa45(a[44], b[44], Cin[43], sum[44]);
FA fa46(a[45], b[45], Cin[44], sum[45]);
FA fa47(a[46], b[46], Cin[45], sum[46]);
FA fa48(a[47], b[47], Cin[46], sum[47]);
FA fa49(a[48], b[48], Cin[47], sum[48]);
FA fa50(a[49], b[49], Cin[48], sum[49]);
FA fa51(a[50], b[50], Cin[49], sum[50]);
FA fa52(a[51], b[51], Cin[50], sum[51]);
FA fa53(a[52], b[52], Cin[51], sum[52]);
FA fa54(a[53], b[53], Cin[52], sum[53]);
FA fa55(a[54], b[54], Cin[53], sum[54]);
FA fa56(a[55], b[55], Cin[54], sum[55]);
FA fa57(a[56], b[56], Cin[55], sum[56]);
FA fa58(a[57], b[57], Cin[56], sum[57]);
FA fa59(a[58], b[58], Cin[57], sum[58]);
FA fa60(a[59], b[59], Cin[58], sum[59]);
FA fa61(a[60], b[60], Cin[59], sum[60]);
FA fa62(a[61], b[61], Cin[60], sum[61]);
FA fa63(a[62], b[62], Cin[61], sum[62]);
FA fa64(a[63], b[63], Cin[62], sum[63]);
```

```
    // assign sum = kgp2[5];
    // assign sum=Cin;
    assign carry = Cin[63];
endmodule




module RCDA_TB;
    reg[63:0] a, b;

    wire[63:0] sum;
    wire carry;

    RCDA rcda1(a, b, 1'b0, sum, carry);



    initial begin
        // a=2690579312231400633 b=8180393319283454732  sum=10870972631514855445
        #0
a=64'b0010010101010110110110111010000100010001100100010100010001011010;
b=64'b0111000110000110100110000110000111011110110111100111001110111011;
        //sum=1001011011011101011101000000001011110000011011111011100000010101
carry=0

        // a=15703793010140570380 b=8400695157779202120
        #10
a=64'b1101100111101111000101001000101001011011101100001110001100001100;
b=64'b0111010010010101010000111011110010111001110001100000000001001000;
        //sum=0100111010000100010110000100011100010101011101101110001101010100
carry=1
```

```
    end

    initial begin
        $monitor(" in1=%b \n in2=%b \n sum=%b \n carry=%b", a, b, sum, carry);
        $dumpfile("rcda.vcd");
        $dumpvars(0, rcda1);
    end
endmodule
```

Output:

```
C:\Users\rammo\OneDrive\Documents\CA\Lab\Lab3>iverilog -o rcda rcda.v

C:\Users\rammo\OneDrive\Documents\CA\Lab\Lab3>vvp rcda
VCD info: dumpfile rcda.vcd opened for output.
 in1=00100101010101101101101110100001000100011001000101000100011011010
 in2=01110001100001101001100001100001110111011011110011100111011101 1
 sum=10010110110111010111010000000101111000001101111101110000001010 1
 carry=0
 in1=11011001111011110001010010001010010110111011000011100011000011 00
 in2=01110100100101010100001110111100101110011100011000000000010010 00
 sum=01001110100001000101100001000111000101010111011011100011010101 00
 carry=1
 in1=11111111111111111111111111111111111111111111111111111111111111 11
 in2=11111111111111111111111111111111111111111111111111111111111111 11
 sum=11111111111111111111111111111111111111111111111111111111111111 10
 carry=1

C:\Users\rammo\OneDrive\Documents\CA\Lab\Lab3>_
```

Wave form: