# Design-of-Dadda-Multiplier

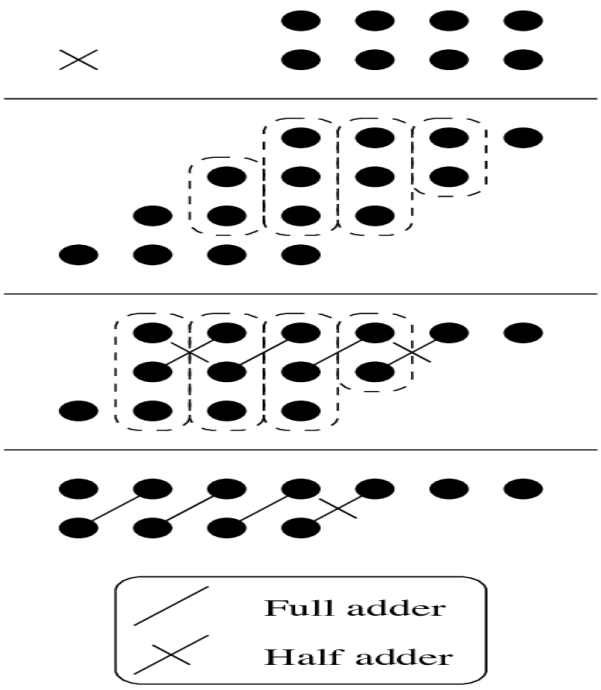
**Designed by Omkar Chavare, IIT Bombay.**

Designed a 16 x 16 Dadda Multiplier in Verilog with a Brent Kung adder for the final addition.
− Verified the design operation through simulation in ModelSim with appropriate test vectors.
− Identified the critical path and computed the worst-case delay using the specified component delays.

## Introduction

The speed of multiplier circuits affects the performance of digital systems and so it is very important to develop better algorithms for faster, more efficient processing and Dadda multiplier is one of the fastest ways of implementing a multiplier.Dadda multipliers are generally more gate efficient than their Wallace counterparts

Reference Circuit



This is the reduction methodology as used in the multiplier circuit. In this the Dadda reduction technique reduces from 4 partial product to 3 and from 3 to 2 from where we can use the normal addition. The dadda reduction series is consulted for reduction of partial products

Code for dadda multiplier: -

```
module adder(a,b,cin,s,cout);

input[31:0] a,b;

input cin;

output[31:0] s;

output cout;

wire [32:0] c;

wire[31:0] p1,g1;

wire[15:0] p2,g2;

wire[7:0] p3,g3;

wire[3:0] p4,g4;

wire[1:0] p5,g5;

wire [0:0]p6,g6;

assign c[0]=cin;

genvar i;
```

```
generate
for(i=0;i<=31;i=i+1)
begin: a1
assign p1[i]=a[i] ^ b[i];
assign g1[i]=a[i] & b[i];

end
endgenerate

generate
for(i=0;i<=15;i=i+1)
begin : a2
assign p2[i]= p1[2*i+1] & p1[2*i];
assign g2[i]= g1[2*i+1]| (p1[2*i+1] & g1[2*i]);
end
endgenerate

generate
for(i=0;i<=7;i=i+1)
begin: a3
assign p3[i]= p2[2*i+1] & p2[2*i];
```

```
assign g3[i]=g2[2*i+1]| (p2[2*i+1] & g2[2*i]);

end

endgenerate



generate

for(i=0;i<=3;i=i+1)

begin: a4

assign p4[i]=p3[2*i+1] & p3[2*i];

assign g4[i]=g3[2*i+1] |  (p3[2*i+1] & g3[2*i]);

end

endgenerate



generate

for(i=0;i<=1;i=i+1)

begin: a5

assign p5[i]=p4[2*i+1] & p4[2*i];

assign g5[i]=g4[2*i+1] |  (p4[2*i+1] & g4[2*i]);

end

endgenerate



generate
```

```
for(i=0;i<=0;i=i+1)

begin: a6

assign p6[i]=p5[2*i+1] & p5[2*i];

assign g6[i]=g5[2*i+1] |  (p5[2*i+1] & g5[2*i]);

end

endgenerate



assign c[1]= g1[0] | ( p1[0] & c[0]);

assign c[2]= g2[0] | ( p2[0] & c[0]);

assign c[4]= g3[0] | ( p3[0] & c[0]);

assign c[8]= g4[0] | ( p4[0] & c[0]);

assign c[16]= g5[0] | ( p5[0] & c[0]);

assign c[32]= g6[0] | ( p6[0] & c[0]);

assign c[3]=g1[2] | (p1[2] & c[2]);

assign c[5]= g1[4] | ( p1[4] & c[4]);

assign c[6]= g2[2] | ( p2[2] & c[4]);

assign c[9]= g1[8] | ( p1[8] & c[8]);

assign c[10]= g2[4] | ( p2[4] & c[8]);

assign c[12]= g3[2] | ( p3[2] & c[8]);
```

```
assign c[13]= g1[12] | ( p1[12] & c[12]);

assign c[7]= g1[6] | ( p1[6] & c[6]);

assign c[11]= g1[10] | ( p1[10] & c[10]);

assign c[14]= g2[6] | ( p2[6] & c[12]);

assign c[15]= g1[14] | ( p1[14] & c[14]);

assign c[17]= g1[16] | ( p1[16] & c[16]);

assign c[18]= g2[8] | ( p2[8] & c[16]);

assign c[20]= g3[4] | ( p3[4] & c[16]);

assign c[19]= g1[18] | ( p1[18] & c[18]);

assign c[21]= g1[20] | ( p1[20] & c[20]);

assign c[22]= g2[10] | ( p2[10] & c[20]);

assign c[24]= g3[5] | ( p3[5] & c[20]);

assign c[23]= g1[22] | ( p1[22] & c[22]);

assign c[25]= g1[24] | ( p1[24] & c[24]);

assign c[26]= g2[12] | ( p2[12] & c[24]);

assign c[28]= g3[6] | ( p3[6] & c[24]);

assign c[27]= g1[26] | ( p1[26] & c[26]);

assign c[29]= g1[28] | ( p1[28] & c[28]);

assign c[30]= g2[14] | ( p2[14] & c[28]);

assign c[31]= g1[30] | ( p1[30] & c[30]);
```

```
generate

for(i=0;i<=31;i=i+1)

begin: a9

assign s[i]= p1[i] ^ c[i];

end

endgenerate



assign  cout=c[32];

endmodule



module multiplier(a,b,mul_out);

input [15:0]a;

input [15:0]b;

output [31:0] mul_out;

wire cout,cin;

wire [15:0] pp1;



wire [16:1]pp2;

wire [17:2]pp3;

wire [18:3]pp4;

wire [19:4]pp5;
```

```
wire [20:5]pp6;

wire [21:6]pp7;

wire [22:7]pp8;

wire [23:8]pp9;

wire [24:9]pp10;

wire [25:10]pp11;

wire [26:11]pp12;

wire [27:12]pp13;

wire [28:13]pp14;

wire [29:14]pp15;

wire [30:15]pp16;

genvar i;


wire [18:13]q0;

wire [19:14]q1;

wire [17:14]q2;

wire [18:15]q3;

wire [16:15]q4;

wire [17:16]q5;
```

```
wire [22:9]r0;

wire [23:10]r1;

wire [21:10]r2;

wire [22:11]r3;

wire [20:11]r4;

wire [21:12]r5;

wire [19:12] r6;

wire [20:13] r7;

//wire [23:8] r8;




wire [25:6] s0;

wire [26:7] s1;

wire [24:7] s2;

wire [25:8] s3;

wire [23:8] s4;

wire [24:9] s5;



wire [27:4]t0;
```

```
wire [28:5]t1;

wire [26:5]t2;

wire [27:6]t3;



wire [28:3]u0;

wire [29:4]u1;

wire [31:0]v0;

wire [31:0]v1;




//partial products :



generate

for(i=0;i<=15;i=i+1)begin : dhanu

assign pp1[i]= a[i] & b[0];

assign pp2[i+1]= a[i] & b[1];

assign pp3[i+2]= a[i] & b[2];

assign pp4[i+3]= a[i] & b[3];

assign pp5[i+4]= a[i] & b[4];

assign pp6[i+5]= a[i] & b[5];

assign pp7[i+6]= a[i] & b[6];
```

```
assign pp8[i+7]= a[i] & b[7];

assign pp9[i+8]= a[i] & b[8];

assign pp10[i+9]= a[i] & b[9];

assign pp11[i+10]= a[i] & b[10];

assign pp12[i+11]= a[i] & b[11];

assign pp13[i+12]= a[i] & b[12];

assign pp14[i+13]= a[i] & b[13];

assign pp15[i+14]= a[i] & b[14];

assign pp16[i+15]= a[i] & b[15];

end

endgenerate



//Intializing values:

 assign v0[0] = pp1[0];

 assign v1[0]= 1'b0;

 assign v0[1]=pp1[1];

 assign v1[1]=pp2[1];

 assign v1[2]=pp3[2];

 assign v0[30]=pp16[30];
```

```
 assign v0[31]=1'b0;

 assign v1[31]=1'b0;

//1st layer addition


ha ha13(pp1[13],pp2[13],q0[13],q1[14]);

fa fa14(pp1[14],pp2[14],pp3[14],q0[14],q1[15]);

ha ha14(pp4[14],pp5[14],q2[14],q3[15]);

fa fa15_1(pp1[15],pp2[15],pp3[15],q0[15],q1[16]);

fa fa15_2(pp4[15],pp5[15],pp6[15],q2[15],q3[16]);

ha ha15(pp7[15],pp8[15],q4[15],q5[16]);

fa fa16_1(pp2[16],pp3[16],pp4[16],q0[16],q1[17]);

fa fa16_2(pp5[16],pp6[16],pp7[16],q2[16],q3[17]);

ha ha16(pp8[16],pp9[16],q4[16],q5[17]);

fa fa17_1(pp3[17],pp4[17],pp5[17],q0[17],q1[18]);

fa fa17_2(pp6[17],pp7[17],pp8[17],q2[17],q3[18]);

fa fa18(pp4[18],pp5[18],pp6[18],q0[18],q1[19]);

//2nd layer addition(ha9_2 implies 9th clmn and its is for layer2)

ha ha9_2(pp1[9],pp2[9],r0[9],r1[10]);

fa fa10_2(pp1[10],pp2[10],pp3[10],r0[10],r1[11]);

ha ha10_2(pp4[10],pp5[10],r2[10],r3[11]);
```

```
fa fa11a_2(pp1[11],pp2[11],pp3[11],r0[11],r1[12]);

fa fa11b_2(pp4[11],pp5[11],pp6[11],r2[11],r3[12]);

ha ha11_2(pp7[11],pp8[11],r4[11],r5[12]);

fa fa12a_2(pp1[12],pp2[12],pp3[12],r0[12],r1[13]);

fa fa12b_2(pp4[12],pp5[12],pp6[12],r2[12],r3[13]);

fa fa12c_2(pp7[12],pp8[12],pp9[12],r4[12],r5[13]);

ha ha12_2(pp10[12],pp11[12],r6[12],r7[13]);


fa fa13a_2(pp3[13],pp4[13],pp5[13],r0[13],r1[14]);

fa fa13b_2(pp6[13],pp7[13],pp8[13],r2[13],r3[14]);

fa fa13c_2(pp9[13],pp10[13],pp11[13],r4[13],r5[14]);

fa fa13d_2(pp12[13],pp13[13],pp14[13],r6[13],r7[14]);

fa fa14a_2(q0[14],q2[14],pp6[14],r0[14],r1[15]);

fa fa14b_2(pp7[14],pp8[14],pp9[14],r2[14],r3[15]);

fa fa14c_2(pp10[14],pp11[14],pp12[14],r4[14],r5[15]);

fa fa14d_2(pp13[14],pp14[14],pp15[14],r6[14],r7[15]);

fa fa15a_2(q0[15],q1[15],q2[15],r0[15],r1[16]);

fa fa15b_2(pp10[15],pp9[15],q4[15],r2[15],r3[16]);

fa fa15c_2(pp12[15],pp13[15],pp11[15],r4[15],r5[16]);

fa fa15d_2(pp14[15],pp15[15],pp16[15],r6[15],r7[16]);
```

```
fa fa16a_2(q0[16],q1[16],q2[16],r0[16],r1[17]);

fa fa16b_2(pp10[16],q4[16],q5[16],r2[16],r3[17]);

fa fa16c_2(pp12[16],pp13[16],pp11[16],r4[16],r5[17]);

fa fa16d_2(pp14[16],pp15[16],pp16[16],r6[16],r7[17]);

fa fa17a_2(q0[17],q1[17],q2[17],r0[17],r1[18]);

fa fa17b_2(pp9[17],pp10[17],q5[17],r2[17],r3[18]);

fa fa17c_2(pp11[17],pp12[17],pp13[17],r4[17],r5[18]);

fa fa17d_2(pp14[17],pp15[17],pp16[17],r6[17],r7[18]);

fa fa18a_2(q0[18],q1[18],pp7[18],r0[18],r1[19]);

fa fa18b_2(pp8[18],pp9[18],pp10[18],r2[18],r3[19]);

fa fa18c_2(pp11[18],pp12[18],pp13[18],r4[18],r5[19]);

fa fa18d_2(pp14[18],pp15[18],pp16[18],r6[18],r7[19]);

fa fa19a_2(pp5[19],pp6[19],pp7[19],r0[19],r1[20]);

fa fa19b_2(pp8[19],pp9[19],pp10[19],r2[19],r3[20]);

fa fa19c_2(pp11[19],pp12[19],pp13[19],r4[19],r5[20]);

fa fa19d_2(pp14[19],pp15[19],pp16[19],r6[19],r7[20]);

fa fa20a_2(pp6[20],pp7[20],pp8[20],r0[20],r1[21]);

fa fa20b_2(pp9[20],pp10[20],pp11[20],r2[20],r3[21]);

fa fa20c_2(pp12[20],pp13[20],pp14[20],r4[20],r5[21]);

fa fa21a_2(pp9[21],pp7[21],pp8[21],r0[21],r1[22]);

fa fa21b_2(pp12[21],pp10[21],pp11[21],r2[21],r3[22]);
```

```
fa fa22a_2(pp10[22],pp8[22],pp9[22],r0[22],r1[23]);

//3rd addition

ha ha6_3(pp1[6],pp2[6],s0[6],s1[7]);

fa fa7a_3(pp1[7],pp2[7],pp3[7],s0[7],s1[8]);

ha ha7b_3(pp4[7],pp5[7],s2[7],s3[8]);

fa fa8a_3(pp1[8],pp2[8],pp3[8],s0[8],s1[9]);

fa fa8b_3(pp4[8],pp5[8],pp6[8],s2[8],s3[9]);

ha ha8c_3(pp7[8],pp8[8],s4[8],s5[9]);

fa fa9a_3(pp3[9],pp4[9],r0[9],s0[9],s1[10]);

fa fa9b_3(pp5[9],pp6[9],pp7[9],s2[9],s3[10]);

fa fa9c_3(pp8[9],pp9[9],pp10[9],s4[9],s5[10]);

fa fa10a_3(r0[10],r1[10],r2[10],s0[10],s1[11]);

fa fa10b_3(pp8[10],pp6[10],pp7[10],s2[10],s3[11]);

fa fa10c_3(pp11[10],pp9[10],pp10[10],s4[10],s5[11]);

fa fa11a_3(r0[11],r1[11],r2[11],s0[11],s1[12]);

fa fa11b_3(r3[11],r4[11],pp9[11],s2[11],s3[12]);

fa fa11c_3(pp10[11],pp11[11],pp12[11],s4[11],s5[12]);

fa fa12a_3(r0[12],r1[12],r2[12],s0[12],s1[13]);

fa fa12b_3(r3[12],r4[12],r5[12],s2[12],s3[13]);

fa fa12c_3(r6[12],pp12[12],pp13[12],s4[12],s5[13]);
```

```
fa fa13a_3(r0[13],r1[13],r2[13],s0[13],s1[14]);

fa fa13b_3(r3[13],r4[13],r5[13],s2[13],s3[14]);

fa fa13c_3(r6[13],r7[13],q0[13],s4[13],s5[14]);


fa fa14a_3(r0[14],r1[14],r2[14],s0[14],s1[15]);

fa fa14b_3(r3[14],r4[14],r5[14],s2[14],s3[15]);

fa fa14c_3(r6[14],r7[14],q1[14],s4[14],s5[15]);


fa fa15a_3(r0[15],r1[15],r2[15],s0[15],s1[16]);

fa fa15b_3(r3[15],r4[15],r5[15],s2[15],s3[16]);

fa fa15c_3(r6[15],r7[15],q3[15],s4[15],s5[16]);



fa fa16a_3(r0[16],r1[16],r2[16],s0[16],s1[17]);

fa fa16b_3(r3[16],r4[16],r5[16],s2[16],s3[17]);

fa fa16c_3(r6[16],r7[16],q3[16],s4[16],s5[17]);


fa fa17a_3(r0[17],r1[17],r2[17],s0[17],s1[18]);

fa fa17b_3(r3[17],r4[17],r5[17],s2[17],s3[18]);

fa fa17c_3(r6[17],r7[17],q3[17],s4[17],s5[18]);
```

```
fa fa18a_3(r0[18],r1[18],r2[18],s0[18],s1[19]);

fa fa18b_3(r3[18],r4[18],r5[18],s2[18],s3[19]);

fa fa18c_3(r6[18],r7[18],q3[18],s4[18],s5[19]);


fa fa19a_3(r0[19],r1[19],r2[19],s0[19],s1[20]);

fa fa19b_3(r3[19],r4[19],r5[19],s2[19],s3[20]);

fa fa19c_3(r6[19],r7[19],q1[19],s4[19],s5[20]);



fa fa20a_3(r0[20],r1[20],r2[20],s0[20],s1[21]);

fa fa20b_3(r3[20],r4[20],r5[20],s2[20],s3[21]);

fa fa20c_3(r7[20],pp15[20],pp16[20],s4[20],s5[21]);


fa fa21a_3(r0[21],r1[21],r2[21],s0[21],s1[22]);

fa fa21b_3(r3[21],r5[21],pp13[21],s2[21],s3[22]);

fa fa21c_3(pp14[21],pp15[21],pp16[21],s4[21],s5[22]);



fa fa22a_3(r0[22],r1[22],pp11[22],s0[22],s1[23]);
```

```
fa fa22b_3(r3[22],pp12[22],pp13[22],s2[22],s3[23]);

fa fa22c_3(pp14[22],pp15[22],pp16[22],s4[22],s5[23]);

fa fa23a_3(r1[23],pp9[23],pp10[23],s0[23],s1[24]);

fa fa23b_3(pp11[23],pp12[23],pp13[23],s2[23],s3[24]);

fa fa23c_3(pp14[23],pp15[23],pp16[23],s4[23],s5[24]);

fa fa24a_3(pp10[24],pp11[24],pp12[24],s0[24],s1[25]);

//doubt 2 times pp14 used

fa fa24b_3(pp13[24],pp14[24],pp15[24],s2[24],s3[25]);

fa fa25_3(pp11[25],pp12[25],pp13[25],s0[25],s1[26]);

//4th stage addition

ha ha4a_4(pp1[4],pp2[4],t0[4],t1[5]);

fa fa5a_4(pp1[5],pp2[5],pp3[5],t0[5],t1[6]);

ha ha5b_4(pp4[5],pp5[5],t2[5],t3[6]);

fa fa6a_4(s0[6],pp3[6],pp4[6],t0[6],t1[7]);

fa fa6b_4(pp5[6],pp6[6],pp7[6],t2[6],t3[7]);

fa fa7a_4(s0[7],s1[7],s2[7],t0[7],t1[8]);

fa fa7b_4(pp6[7],pp7[7],pp8[7],t2[7],t3[8]);

fa fa8a_4(s0[8],s1[8],s2[8],t0[8],t1[9]);

fa fa8b_4(s3[8],s4[8],pp9[8],t2[8],t3[9]);

fa fa9a_4(s0[9],s1[9],s2[9],t0[9],t1[10]);

fa fa9b_4(s3[9],s4[9],s5[9],t2[9],t3[10]);
```

```
fa fa10a_4(s0[10],s1[10],s2[10],t0[10],t1[11]);

fa fa10b_4(s3[10],s4[10],s5[10],t2[10],t3[11]);

fa fa11a_4(s0[11],s1[11],s2[11],t0[11],t1[12]);

fa fa11b_4(s3[11],s4[11],s5[11],t2[11],t3[12]);

fa fa12a_4(s0[12],s1[12],s2[12],t0[12],t1[13]);

fa fa12b_4(s3[12],s4[12],s5[12],t2[12],t3[13]);

fa fa13a_4(s0[13],s1[13],s2[13],t0[13],t1[14]);

fa fa13b_4(s3[13],s4[13],s5[13],t2[13],t3[14]);

fa fa14a_4(s0[14],s1[14],s2[14],t0[14],t1[15]);

fa fa14b_4(s3[14],s4[14],s5[14],t2[14],t3[15]);

fa fa15a_4(s0[15],s1[15],s2[15],t0[15],t1[16]);

fa fa15b_4(s3[15],s4[15],s5[15],t2[15],t3[16]);

fa fa16a_4(s0[16],s1[16],s2[16],t0[16],t1[17]);

fa fa16b_4(s3[16],s4[16],s5[16],t2[16],t3[17]);

fa fa17a_4(s0[17],s1[17],s2[17],t0[17],t1[18]);

fa fa17b_4(s3[17],s4[17],s5[17],t2[17],t3[18]);

fa fa18a_4(s0[18],s1[18],s2[18],t0[18],t1[19]);

fa fa18b_4(s3[18],s4[18],s5[18],t2[18],t3[19]);

fa fa19a_4(s0[19],s1[19],s2[19],t0[19],t1[20]);

fa fa19b_4(s3[19],s4[19],s5[19],t2[19],t3[20]);
```

```
fa fa20a_4(s0[20],s1[20],s2[20],t0[20],t1[21]);

fa fa20b_4(s3[20],s4[20],s5[20],t2[20],t3[21]);

fa fa21a_4(s0[21],s1[21],s2[21],t0[21],t1[22]);

fa fa21b_4(s3[21],s4[21],s5[21],t2[21],t3[22]);

fa fa22a_4(s0[22],s1[22],s2[22],t0[22],t1[23]);

fa fa22_4(s3[22],s4[22],s5[22],t2[22],t3[23]);

fa fa23a_4(s0[23],s1[23],s2[23],t0[23],t1[24]);

fa fa23_4(s3[23],s4[23],s5[23],t2[23],t3[24]);

fa fa24a_4(s0[24],s1[24],s2[24],t0[24],t1[25]);

fa fa24_4(s3[24],pp16[24],s5[24],t2[24],t3[25]);

fa fa25a_4(s0[25],s1[25],pp14[25],t0[25],t1[26]);

fa fa25_4(s3[25],pp15[25],pp16[25],t2[25],t3[26]);

fa fa26a_4(s1[26],pp12[26],pp13[26],t0[26],t1[27]);

fa fa26_4(pp14[26],pp15[26],pp16[26],t2[26],t3[27]);

fa fa27_4(pp13[27],pp14[27],pp15[27],t0[27],t1[28]);

//fifth stage addition

ha ha3a_5(pp1[3],pp2[3],u0[3],u1[4]);

fa fa4a_5(pp3[4],pp4[4],pp5[4],u0[4],u1[5]);

fa fa5a_5(t1[5],t2[5],pp6[5],u0[5],u1[6]);

fa fa6a_5(t0[6],t1[6],t2[6],u0[6],u1[7]);

fa fa7_5(t0[7],t1[7],t2[7],u0[7],u1[8]);
```

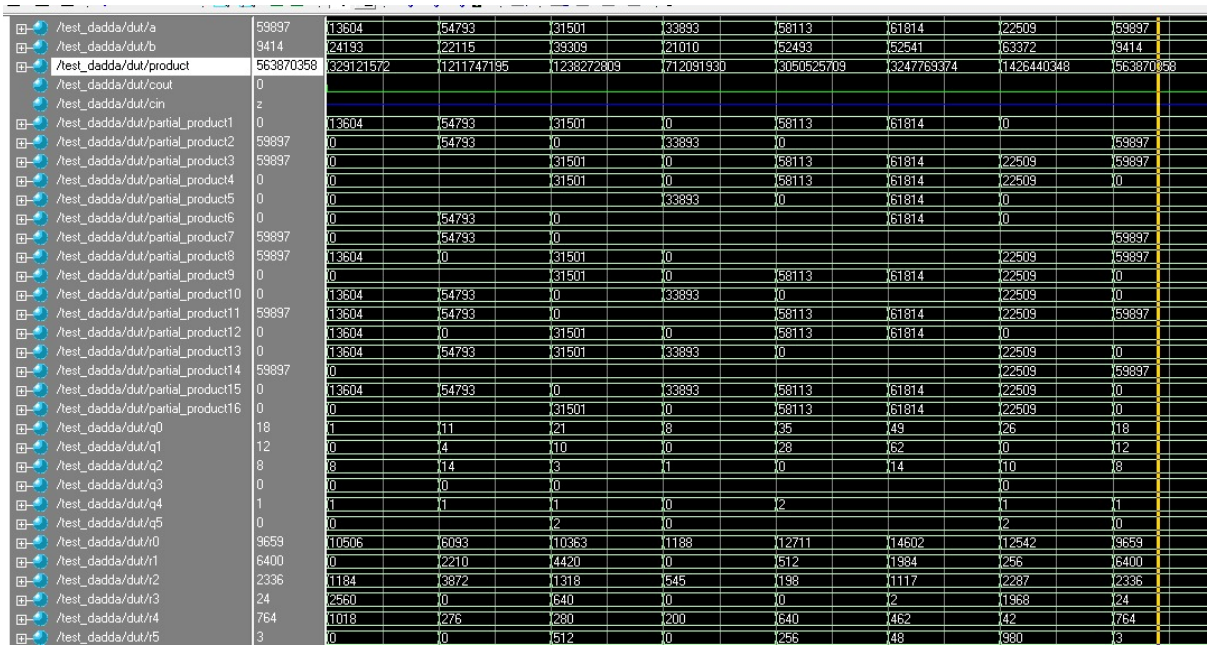
```
fa fa8_5(t0[8],t1[8],t2[8],u0[8],u1[9]);

fa fa9_5(t0[9],t1[9],t2[9],u0[9],u1[10]);

fa fa10_5(t0[10],t1[10],t2[10],u0[10],u1[11]);

fa fa11_5(t0[11],t1[11],t2[11],u0[11],u1[12]);

fa fa12_5(t0[12],t1[12],t2[12],u0[12],u1[13]);

fa fa13_5(t0[13],t1[13],t2[13],u0[13],u1[14]);

fa fa14_5(t0[14],t1[14],t2[14],u0[14],u1[15]);

fa fa15_5(t0[15],t1[15],t2[15],u0[15],u1[16]);

fa fa16_5(t0[16],t1[16],t2[16],u0[16],u1[17]);

fa fa17_5(t0[17],t1[17],t2[17],u0[17],u1[18]);

fa fa18_5(t0[18],t1[18],t2[18],u0[18],u1[19]);

fa fa19_5(t0[19],t1[19],t2[19],u0[19],u1[20]);

fa fa20_5(t0[20],t1[20],t2[20],u0[20],u1[21]);

fa fa21_5(t0[21],t1[21],t2[21],u0[21],u1[22]);

fa fa22_5(t0[22],t1[22],t2[22],u0[22],u1[23]);

fa fa23_5(t0[23],t1[23],t2[23],u0[23],u1[24]);

fa fa24_5(t0[24],t1[24],t2[24],u0[24],u1[25]);

fa fa25_5(t0[25],t1[25],t2[25],u0[25],u1[26]);

fa fa26_5(t0[26],t1[26],t2[26],u0[26],u1[27]);

fa fa27_5(t0[27],t1[27],pp16[27],u0[27],u1[28]);
```

```
fa fa28_5(t1[28],pp14[28],pp15[28],u0[28],u1[29]);

//6th stage addition

ha ha2_6(pp1[2],pp2[2],v0[2],v1[3]);

fa fa3_6(u0[3],pp3[3],pp4[3],v0[3],v1[4]);

fa fa4_6(u0[4],u1[4],t0[4],v0[4],v1[5]);

fa fa5_6(u0[5],u1[5],t0[5],v0[5],v1[6]);

fa fa6_6(u0[6],u1[6],t3[6],v0[6],v1[7]);

fa fa7_6(u0[7],u1[7],t3[7],v0[7],v1[8]);

fa fa8_6(u0[8],u1[8],t3[8],v0[8],v1[9]);

fa fa9_6(u0[9],u1[9],t3[9],v0[9],v1[10]);

fa fa10_6(u0[10],u1[10],t3[10],v0[10],v1[11]);

fa fa11_6(u0[11],u1[11],t3[11],v0[11],v1[12]);

fa fa12_6(u0[12],u1[12],t3[12],v0[12],v1[13]);

fa fa13_6(u0[13],u1[13],t3[13],v0[13],v1[14]);

fa fa14_6(u0[14],u1[14],t3[14],v0[14],v1[15]);

fa fa15_6(u0[15],u1[15],t3[15],v0[15],v1[16]);

fa fa16_6(u0[16],u1[16],t3[16],v0[16],v1[17]);

fa fa17_6(u0[17],u1[17],t3[17],v0[17],v1[18]);

fa fa18_6(u0[18],u1[18],t3[18],v0[18],v1[19]);

fa fa19_6(u0[19],u1[19],t3[19],v0[19],v1[20]);

fa fa20_6(u0[20],u1[20],t3[20],v0[20],v1[21]);
```

```
fa fa21_6(u0[21],u1[21],t3[21],v0[21],v1[22]);

fa fa22_6(u0[22],u1[22],t3[22],v0[22],v1[23]);

fa fa23_6(u0[23],u1[23],t3[23],v0[23],v1[24]);

fa fa24_6(u0[24],u1[24],t3[24],v0[24],v1[25]);

fa fa25_6(u0[25],u1[25],t3[25],v0[25],v1[26]);

fa fa26_6(u0[26],u1[26],t3[26],v0[26],v1[27]);

fa fa27_6(u0[27],u1[27],t3[27],v0[27],v1[28]);

fa fa28_6(u0[28],u1[28],pp16[28],v0[28],v1[29]);

fa fa29_6(pp15[29],u1[29],pp16[29],v0[29],v1[30]);

bentkung adder(v0,v1,1'b0,mul_out,cout);

 endmodule
```

## SIMULATION : - simulation performed using Quartus Prime and Modelsim Altera.

Results: - a,b are inputs and the output product is highlighted.



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| /test_dadda/dut/a | 59897 | 13604 | 54793 | 31501 | 33893 | 58113 | 61814 | 22509 | 59897 |
| /test_dadda/dut/b | 9414 | 24193 | 22115 | 39309 | 21010 | 52493 | 52541 | 63372 | 9414 |
| /test_dadda/dut/product | 563870358 | 329121572 | 1211747195 | 1238272809 | 712091930 | 3050525709 | 3247769374 | 1426440348 | 563870358 |
| /test_dadda/dut/cout | 0 | | | | | | | | |
| /test_dadda/dut/cin | z | | | | | | | | |
| /test_dadda/dut/partial_product1 | 0 | 13604 | 54793 | 31501 | 0 | 58113 | 61814 | 0 | |
| /test_dadda/dut/partial_product2 | 59897 | 0 | 54793 | 0 | 33893 | 0 | | | 59897 |
| /test_dadda/dut/partial_product3 | 59897 | 0 | | 31501 | 0 | 58113 | 61814 | 22509 | 59897 |
| /test_dadda/dut/partial_product4 | 0 | 0 | | 31501 | 0 | 58113 | 61814 | 22509 | 0 |
| /test_dadda/dut/partial_product5 | 0 | 0 | | | 33893 | 0 | 61814 | 0 | |
| /test_dadda/dut/partial_product6 | 0 | 0 | 54793 | 0 | | | 61814 | 0 | |
| /test_dadda/dut/partial_product7 | 59897 | 0 | 54793 | 0 | | | | | 59897 |
| /test_dadda/dut/partial_product8 | 59897 | 13604 | 0 | 31501 | 0 | | | 22509 | 59897 |
| /test_dadda/dut/partial_product9 | 0 | 0 | | 31501 | 0 | 58113 | 61814 | 22509 | 0 |
| /test_dadda/dut/partial_product10 | 0 | 13604 | 54793 | 0 | 33893 | 0 | | 22509 | 0 |
| /test_dadda/dut/partial_product11 | 59897 | 13604 | 54793 | 0 | | 58113 | 61814 | 22509 | 59897 |
| /test_dadda/dut/partial_product12 | 0 | 13604 | 0 | 31501 | 0 | 58113 | 61814 | 0 | |
| /test_dadda/dut/partial_product13 | 0 | 13604 | 54793 | 31501 | 33893 | 0 | | 22509 | 0 |
| /test_dadda/dut/partial_product14 | 59897 | 0 | | | | | | 22509 | 59897 |
| /test_dadda/dut/partial_product15 | 0 | 13604 | 54793 | 0 | 33893 | 58113 | 61814 | 22509 | 0 |
| /test_dadda/dut/partial_product16 | 0 | 0 | | 31501 | 0 | 58113 | 61814 | 22509 | 0 |
| /test_dadda/dut/q0 | 18 | 1 | 11 | 21 | 8 | 35 | 49 | 26 | 18 |
| /test_dadda/dut/q1 | 12 | 0 | 4 | 10 | 0 | 28 | 62 | 0 | 12 |
| /test_dadda/dut/q2 | 8 | 8 | 14 | 3 | 1 | 0 | 14 | 10 | 8 |
| /test_dadda/dut/q3 | 0 | 0 | 0 | 0 | | | | 0 | |
| /test_dadda/dut/q4 | 1 | 1 | 1 | 1 | 0 | 2 | | 1 | 1 |
| /test_dadda/dut/q5 | 0 | 0 | | 2 | 0 | | | 2 | 0 |
| /test_dadda/dut/r0 | 9659 | 10506 | 6093 | 10363 | 1188 | 12711 | 14602 | 12542 | 9659 |
| /test_dadda/dut/r1 | 6400 | 0 | 2210 | 4420 | 0 | 512 | 1984 | 256 | 6400 |
| /test_dadda/dut/r2 | 2336 | 1184 | 3872 | 1318 | 545 | 198 | 1117 | 2287 | 2336 |
| /test_dadda/dut/r3 | 24 | 2560 | 0 | 640 | 0 | 0 | 2 | 1968 | 24 |
| /test_dadda/dut/r4 | 764 | 1018 | 276 | 280 | 200 | 640 | 462 | 42 | 764 |
| /test_dadda/dut/r5 | 3 | 0 | 0 | 512 | 0 | 256 | 48 | 980 | 3 |