Implementation & Testing of 3D Kogge-Stone Adder

VLSI

HERITAGE INSTITUTE OF TECHNOLOGY, KOLKATA

By

Nishant, Tania, Susmita, Sritama

KOGGE STONE ADDER

1. Abstract

KSA is a parallel prefix form carry look ahead adder. It generates carry in O ($\log n$) time and is widely considered as the fastest adder and is widely used in the industry for high performance arithmetic circuits. In KSA, carries are computed fast by computing them in parallel at the cost of increased area. Our task is to Implement & Test 3D KSA so that we can reduce the cost of silicon chip used mount this complex circuit as well as make it more efficient.

2. Theory

The complete functioning of KSA can be easily comprehended by analyzing it in terms of three distinct parts:

A. Pre processing:

This step involves computation of generate and propagate signals corresponding too each pair of bits in A and B. These signals are given by the logic equations below:

$$pi = Ai \oplus Bi$$
$$gi = Ai \cdot Bi$$

B. Carry look ahead network:

This block differentiates KSA from other adders and is the main force behind its high performance. This step involves computation of carries corresponding to each bit. It uses group propagate and generate as intermediate signals which are given by the logic equations below:

$$P(i,j) = P(i,k+1) \cdot P(k,j)$$

$$G(i,j) = G(i,k+1)\ or\ [P(i,k+1) and\ G(k,j)]$$

C. Post processing:

This is the final step and is common to all adders of this family (carry look ahead). It involves computation of sum bits. Sum bits are computed by the logic given below:

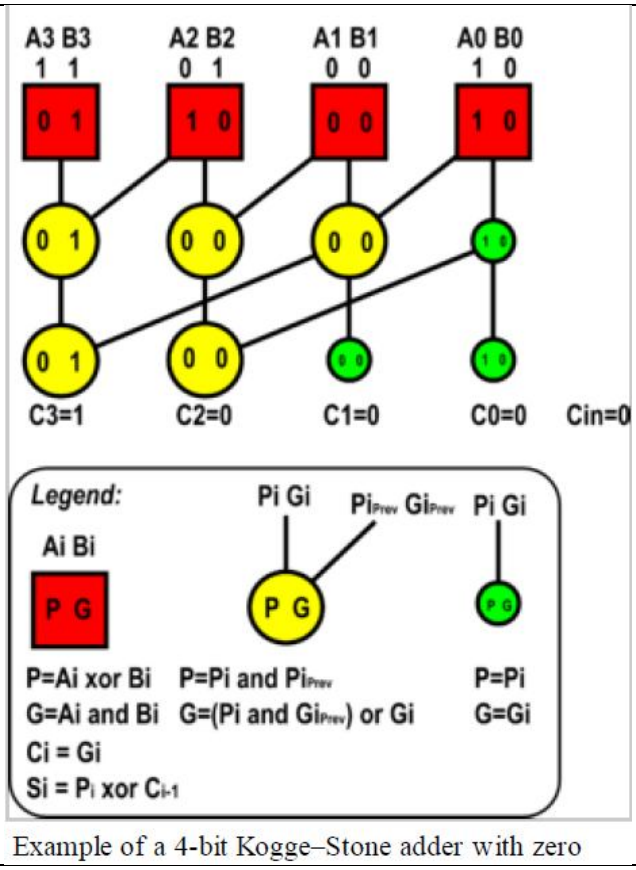
$$S(i) = P(i) \oplus C(i-1)$$

Application of Kogge Stone Adder

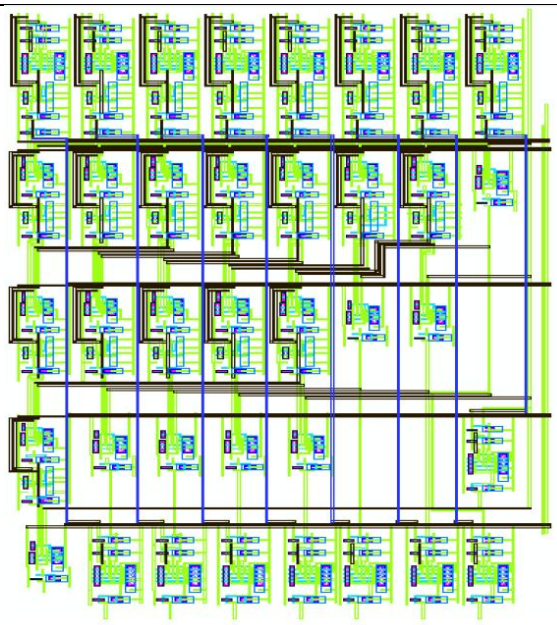
The addition of two binary numbers is one of the most common and important arithmetic functions in modern digital VLSI system. Many research efforts have focused on designing faster and more efficient adder architectures and resulted in a larger number of adder architectures. Some examples of architectures are the Ripple carry adder which, generates carries in series, the carry look-ahead adder which attempts to speed-up the carry propagation by using additional logics, group generate and propagate, and the parallel prefix adder which extends from the idea of carry look-ahead computation. The parallel prefix adders are a more general form where a network is used to pre-calculate the carry signals. Some well know parallel prefix adders are the Kogge-Stone adder [1], Brent-Kung adder [2] and Han-Carlson adder [3].

Since we already discussed that Kogge stone adder performs binary addition in O($\log n$) time. If we will able to reduce the power consumed due to its long wiring then it can be feasible to use KSA in our daily life computation. You can imagine how fast our computers can be if this technology will implement. If we talk about a 64 bit computer, its current time of addition is about O(n) time. But after implementation of KSA it will reduce to O(log n). So in 64 bit PC we will improve performance by ~64/6 = 11 times. Think about the PC of future which wills about 256 bit or more. What efficiency can achieve if we will really be able to reduce the size of wiring to reduce the power use. Other difficulty to implement KSA is that it’s not planner ie we cannot implement this circuit in a plane. We need a lot of overlapping of wiring which cause of capacitance in the circuit.

Following is the block diagram of a simple 4bit Kogge-Stone Adder.



Example of a 4-bit Kogge–Stone adder with zero



What available to us.	What we are going to design.

Equation for a 4-bit kogge-stone adder

S0 = (A0 XOR B0) XOR Cin
S1 = (A1 XOR B1) XOR (A0 AND B0)
S2 = (A2 XOR B2) XOR (((A1 XOR B1) AND (A0 AND B0)) OR (A1 AND B1))
S3 = (A3 XOR B3) XOR ((((A2 XOR B2) AND (A1 XOR B1)) AND (A0 AND B0)) OR (((A2 XOR B2) AND (A1 AND B1))
S4 = ((((A3 XOR B3) AND (A2 XOR B2)) AND (A1 AND B1)) OR (((A3 XOR B3) AND (A2 AND B2)) OR (A3 AND B3)))

Following is the figure of a 16-bit KSA.

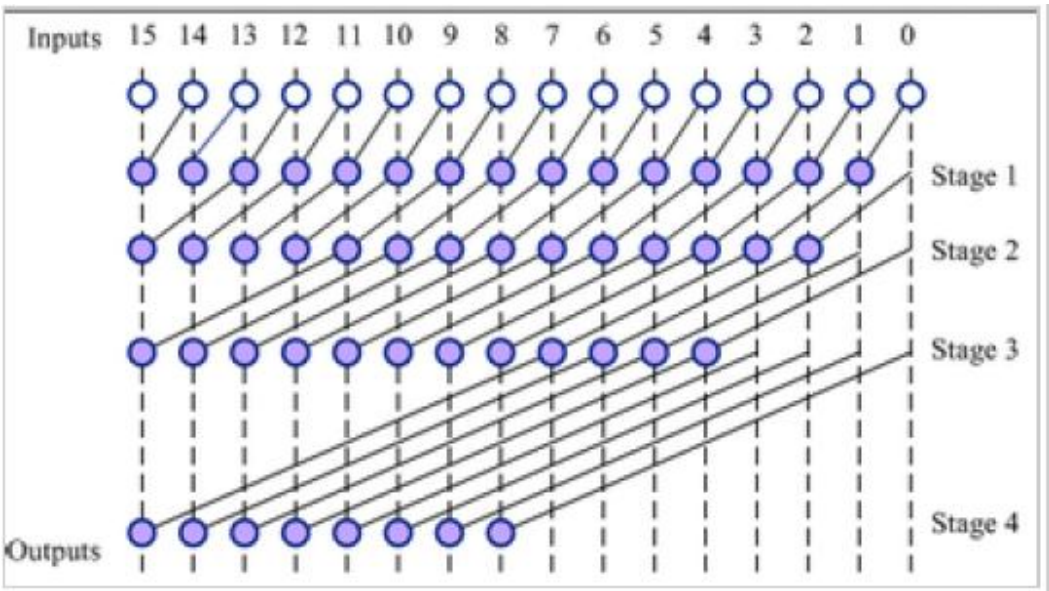
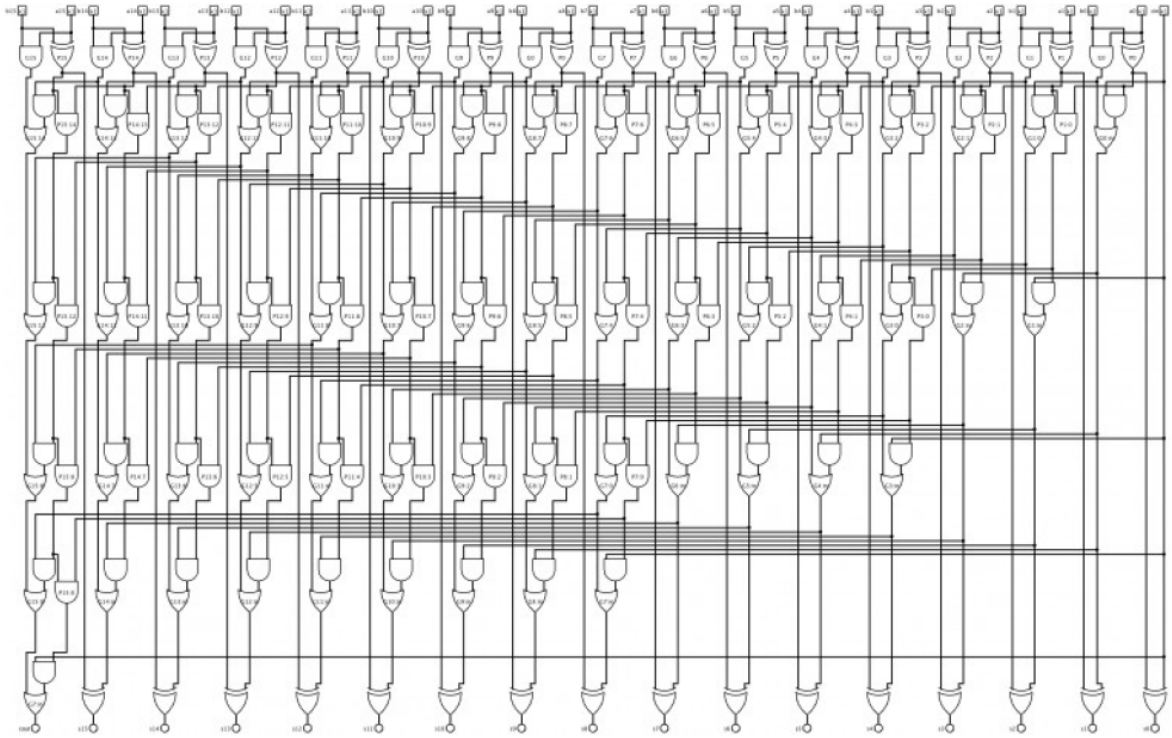


Figure 1 - 16-bit Kogge-Stone tree graph



16 bit Kogge Stone Adder PPA diagram

Why 3D layout implementation of Kogge Stone Adder

As we already discussed that Kogge stone adder is a very fast adder but it is difficult to use because it need huge amount of power. First we try to know what is general benefits of 3D layout implementation, then we'll talk about why 3D layout in Kogge Stone Adder. There are following benefits of using 3D layout.

Footprint

More functionality fits into a small space. This extends Moore's law and enables a new generation of tiny but powerful devices.

Cost

Partitioning a large chip into multiple smaller dies with 3D stacking can improve the yield and reduce the fabrication cost if individual dies are tested separately.

Heterogeneous integration

Circuit layers can be built with different processes, or even on different types of wafers. This means that components can be optimized to a much greater degree than if they were built together on a single wafer. Moreover, components with incompatible manufacturing could be combined in a single 3D IC.

Shorter interconnect

The average wire length is reduced. Common figures reported by researchers are on the order of 10–15%, but this reduction mostly applies to longer interconnect, which may affect circuit delay by a greater amount. Given that 3D wires have much higher capacitance than conventional in-die wires, circuit delay may or may not improve.

Power

Keeping a signal on-chip can reduce its power consumption by 10–100 times. Shorter wires also reduce power consumption by producing less parasitic capacitance. Reducing the power budget leads to less heat generation, extended battery life, and lower cost of operation.

Design

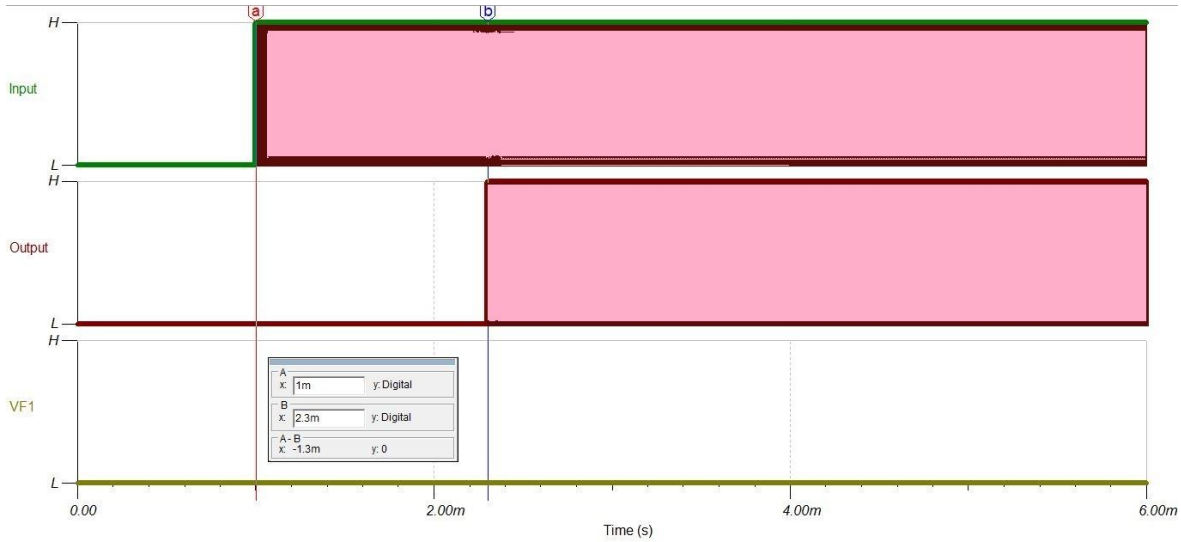
The vertical dimension adds a higher order of connectivity and offers new design possibilities.

Circuit security

The stacked structure complicates attempts to reverse engineer the circuitry. Sensitive circuits may also be divided among the layers in such a way as to obscure the function of each layer.

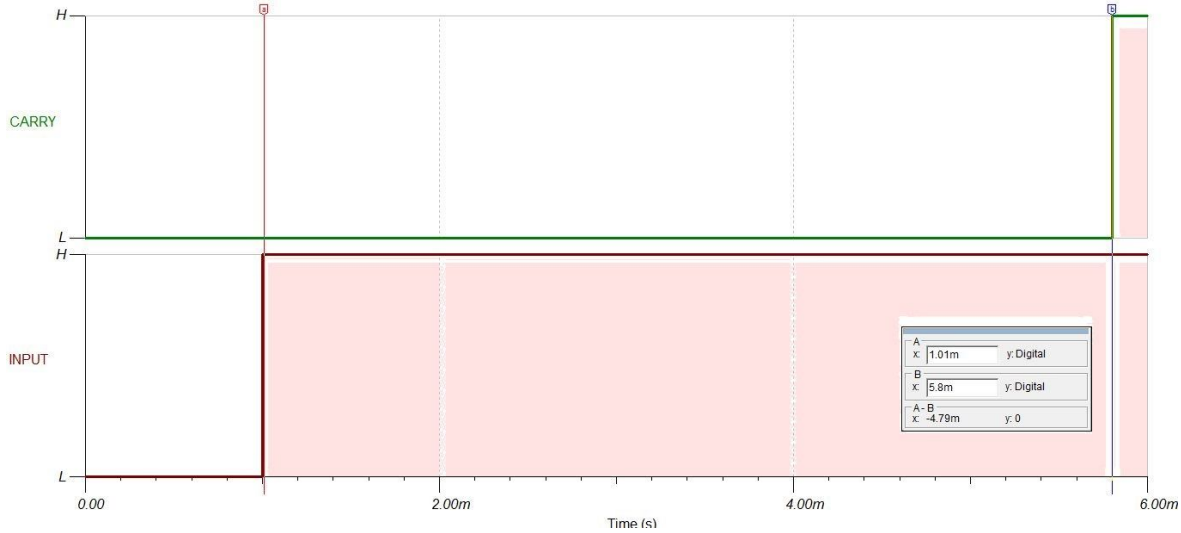
A working model of 8 bit KSA vs 8 bit Ripple Carry Adder

DELAY IN KSA 8 bit IN GENERATING



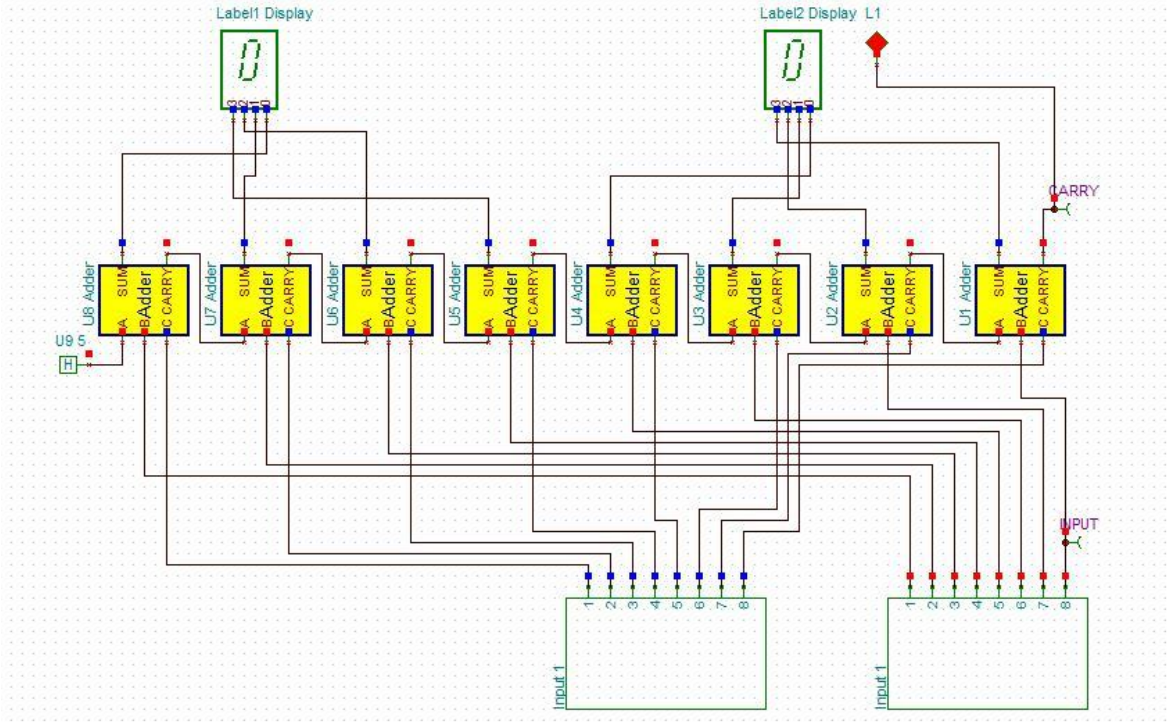
DELAY = 2.3 – 1 = 1.3 time unit.

DELAY IN Ripple Carry Adder 8 bit

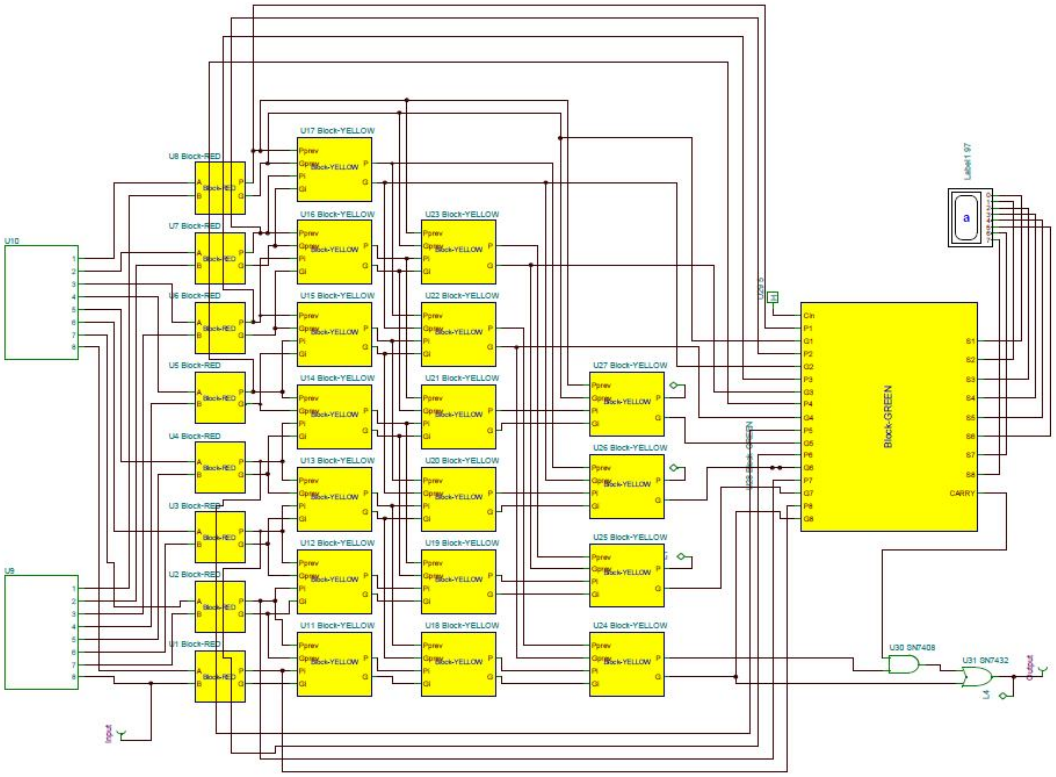


DELAY = 5.8 – 1.01 = 4.79 time unit.

Working Models of KSA vs RCA



Ripple Carry Adder



Kogge Stone Adder

Delay Calculation by Verilog Modules

To calculate delay we can also use Verilog code. Following is the Verilog code and their output for delay calculation.

```
module ksa8bit(sum,carry,a,b,c);

input [7:0] a,b;
input c;
output [7:0] sum;
output carry;

wire p00,p01,p02,p03,p04,p05,p06,p07;
wire p10,p11,p12,p13,p14,p15,p16;
wire p20,p21,p22,p23,p24,p25;
wire p30,p31,p32,p33;

wire g00,g01,g02,g03,g04,g05,g06,g07;
wire g10,g11,g12,g13,g14,g15,g16;
wire g20,g21,g22,g23,g24,g25;
wire g30,g31,g32,g33;

      red r00(p00,g00,a[7],b[7]);
      red r01(p01,g01,a[6],b[6]);
      red r02(p02,g02,a[5],b[5]);
      red r03(p03,g03,a[4],b[4]);
      red r04(p04,g04,a[3],b[3]);
      red r05(p05,g05,a[2],b[2]);
      red r06(p06,g06,a[1],b[1]);
      red r07(p07,g07,a[0],b[0]);
//-------------------------------------------------
      yellow y10(p10,g10,p00,p01,g00,g01);
      yellow y11(p11,g11,p01,p02,g01,g02);
      yellow y12(p12,g12,p02,p03,g02,g03);
      yellow y13(p13,g13,p03,p04,g03,g04);
      yellow y14(p14,g14,p04,p05,g04,g05);
      yellow y15(p15,g15,p05,p06,g05,g06);
      yellow y16(p16,g16,p06,p07,g06,g07);
//-------------------------------------------------
      yellow y20(p20,g20,p10,p12,g10,g12);
      yellow y21(p21,g21,p11,p13,g11,g13);
      yellow y22(p22,g22,p12,p14,g12,g14);
      yellow y23(p23,g23,p13,p15,g13,g15);
      yellow y24(p24,g24,p14,p16,g14,g16);
      yellow y25(p25,g25,p15,p07,g15,g07);
//-------------------------------------------------
      yellow y30(p30,g30,p20,p24,g20,g24);
      yellow y31(p31,g31,p21,p25,g21,g25);
      yellow y32(p32,g32,p22,p16,g22,g16);
      yellow y33(p33,g33,p23,p07,g23,g07);
//-------------------------------------------------
      xor #(3,2)   x0 (sum[0],p07,  c);
```

```
      xor #(3,2)   x1 (sum[1],p06,g07);
      xor #(3,2)   x2 (sum[2],p05,g16);
      xor #(3,2)   x3 (sum[3],p04,g25);
      xor #(3,2)   x4 (sum[4],p03,g24);
      xor #(3,2)   x5 (sum[5],p02,g33);
      xor #(3,2)   x6 (sum[6],p01,g32);
      xor #(3,2)   x7 (sum[7],p00,g31);
//-------------------------------------------------
      and #(2,1)   a0  (carry,g30,1);
endmodule
```

```
module red(p,g,a,b);
      input a,b;
      output p,g;

            xor #(3,2)  x0(p,a,b);
            and #(2,1)  a0(g,a,b);

endmodule
```

```
module yellow(p,g,pi,pp,gi,gp);
      input pi,pp,gi,gp;
      output p,g;
      wire temp;
            and #(2,1)  a0(p,pi,pp);
            and #(2,1)  a1(temp,pi,gp);
            or  #(2,1)  o1(g,temp,gi);

endmodule
```

```
module ksa_tb;
      reg [7:0]a,b;
      reg c;
      wire [7:0]sum;
      wire carry;
      ksa8bit my_ksa(sum,carry,a,b,c);
      initial
      begin
            $dumpfile("ksa_tb.vcd");
            $dumpvars(0, ksa_tb);
            a=255; b=0; c=0;
            #15;
            a=255; b=1; c=0;
            #20;

      end
endmodule
```

Property	Kogge Stone Adder 8bit	Ripple Carry Adder 8bit
Delay(ns)	1.3	4.79
No of Gates	194	32
Footprint		

Planarity Implementation in KSA

As we know that 16 bit KSA is not a planner circuit, hence we need to make 2 or more layer to make it planner. Before dividing it into planes, we need to find those edges which lead this problem. To find those edges we are using a Library called **Boost Lib** [36]. Boost Lib has some C++ header files which have been used by us to check the planarity of our given circuit.

To check planarity concept we have generate all the subset of edges. And continue to removing edges until we do not get the planer graph. This is a broth force technique. It took time. It is ok for small circuits but in case of large circuits we need some more generic approach. However, as our circuit is not so much large so we had implemented this technique for our circuit.

Following is the whole thing that we need to implement this technique.

Planarity Testing

Graph for KSA (edge list)

{1,17}, {1,32}, {1,46}, {1,58}, {2,17}, {2,18}, {3,18}, {3,19}, {4,19}, {4,20}, {5,20}, {5,21}, {6,21}, {6,22}, {7,22}, {7,23}, {8,23}, {8,24}, {9,24}, {9,25}, {10,25}, {10,26}, {11,26}, {11,27}, {12,27}, {12,28}, {13,28}, {13,29}, {14,29}, {14,30}, {15,30}, {15,31}, {16,31}, {17,33}, {17,47}, {17,59}, {18,32}, {18,34}, {19,33}, {19,35}, {20,34}, {20,36}, {21,35}, {21,37}, {22,36}, {22,38}, {23,37}, {23,39}, {24,38}, {24,40}, {25,39}, {25,41}, {26,40}, {26,42}, {27,41}, {27,43}, {28,42}, {28,44}, {29,43}, {29,45}, {30,44}, {31,45}, {32,48}, {32,60}, {33,49}, {33,61}, {34,46}, {34,50}, {35,47}, {35,51}, {36,48}, {36,52}, {37,49}, {37,53}, {38,50}, {38,54}, {39,51}, {39,55}, {40,52}, {40,56}, {41,53}, {41,57}, {42,54}, {43,55}, {44,56}, {45,57}, {46,62}, {47,63}, {48,64}, {49,65}, {50,58}, {51,59}, {52,60}, {53,61}, {54,62}, {55,63}, {56,64}, {57,65}

This will generate all the test cases for checking to find planarity.

```
#include<stdio.h>
void allsubset(int no_of_element){
      int no_of_subset,start,i,j,index;
      for(no_of_subset=1;no_of_subset<=no_of_element;no_of_subset++)
      {
            for(start=0;start<=no_of_element-no_of_subset;start++)
            {
                  if(no_of_subset==1)
                        printf("%d -1\n",start);
                  else
                  {
                        index=start+no_of_subset-1;
                        for(j=index;j<no_of_element;j++)
                        {
                              for(i=start;i<index;i++)
                                    printf("%d ",i);
                              printf("%d -1\n",j);
                        }
                  }
```

```
            }
      }
      printf(" -2 -2 -2 -2 -2");
}
int main(){
      allsubset(98);
      return 0;
}
```

```
This program will find the minimum edge should be removed to make
ksa16 planner.
/**
./test < subset.txt
KSA16 is nonplanner

Found at: 117237 KSA16 - 39edges
{1,17}, {1,32}, {1,46}, {1,58}, {2,17}, {2,18}, {3,18}, {3,19},
{4,19}, {4,20}, {5,20}, {5,21}, {6,21}, {6,22}, {7,22}, {7,23},
{8,23}, {8,24}, {9,24}, {9,25}, {10,25}, {10,26}, {11,26}, {11,27},
{12,27}, {12,28}, {13,28}, {13,29}, {14,29}, {14,30}, {15,30},
{15,31}, {16,31}, {17,33}, {17,47}, {17,59}, {18,32}, {18,34},
{19,33},
*/
#include <iostream>
#include <boost/graph/adjacency_list.hpp>
#include <boost/graph/boyer_myrvold_planar_test.hpp>
#include <vector>
//98 edges
int ksa_edges[][3]={
{1,17,1},
{1,32,1},
{1,46,1},
{1,58,1},
{2,17,1},
{2,18,1},
{3,18,1},
{3,19,1},
{4,19,1},
{4,20,1},
{5,20,1},
{5,21,1},
{6,21,1},
{6,22,1},
{7,22,1},
{7,23,1},
{8,23,1},
{8,24,1},
{9,24,1},
{9,25,1},
{10,25,1},
{10,26,1},
```

```
{11,26,1},
{11,27,1},
{12,27,1},
{12,28,1},
{13,28,1},
{13,29,1},
{14,29,1},
{14,30,1},
{15,30,1},
{15,31,1},
{16,31,1},
{17,33,1},
{17,47,1},
{17,59,1},
{18,32,1},
{18,34,1},
{19,33,1},
{19,35,1},
{20,34,1},
{20,36,1},
{21,35,1},
{21,37,1},
{22,36,1},
{22,38,1},
{23,37,1},
{23,39,1},
{24,38,1},
{24,40,1},
{25,39,1},
{25,41,1},
{26,40,1},
{26,42,1},
{27,41,1},
{27,43,1},
{28,42,1},
{28,44,1},
{29,43,1},
{29,45,1},
{30,44,1},
{31,45,1},
{32,48,1},
{32,60,1},
{33,49,1},
{33,61,1},
{34,46,1},
{34,50,1},
{35,47,1},
{35,51,1},
{36,48,1},
{36,52,1},
{37,49,1},
{37,53,1},
```

```
{38,50,1},
{38,54,1},
{39,51,1},
{39,55,1},
{40,52,1},
{40,56,1},
{41,53,1},
{41,57,1},
{42,54,1},
{43,55,1},
{44,56,1},
{45,57,1},
{46,62,1},
{47,63,1},
{48,64,1},
{49,65,1},
{50,58,1},
{51,59,1},
{52,60,1},
{53,61,1},
{54,62,1},
{55,63,1},
{56,64,1},
{57,65,1}
};



int main(int argc, char** argv)
{

  // This program illustrates a simple use of
boyer_myrvold_planar_embedding
  // as a simple yes/no test for planarity.

  using namespace boost;

  typedef adjacency_list<vecS,
                         vecS,
                         undirectedS,
                         property<vertex_index_t, int>
                         > graph;
     int edg[200],i,j,c=0,min=9999;
  graph KSA16(65);
     for (int i=0;i<98;i++)
          add_edge(ksa_edges[i][0],ksa_edges[i][1], KSA16);

if (boyer_myrvold_planarity_test(KSA16))
    std::cout << "KSA16 is planar." << std::endl;
else
    std::cout << "KSA16 is nonplanner"<< std::endl;
```

```
      while(1){
            //std::cout<<c/100<<"\n";
            c++;
            i=0;
            do{
                  std::cin>>edg[i];
                  //std::cout<<edg[i]<<", ";
            }while(edg[i++]>=0);
            for(j=0;j<i;j++){
                  remove_edge(ksa_edges[j][0],ksa_edges[j][1],KSA16);
                  remove_edge(ksa_edges[j][1],ksa_edges[j][0],KSA16);
            }
            if (boyer_myrvold_planarity_test(KSA16)){

                  if(min>i){
                  std::cout<<"\nFound at:" << c<<" KSA16 - " << i <<
"edges\n";
                  for(j=0;j<i;j++){

      std::cout<<"{"<<ksa_edges[j][0]<<","<<ksa_edges[j][1]<<"}, ";
                  }
                  min=i;
                  break;
                  }

            }
            //else
            //    std::cout<<"Not Found";
            for(j=0;j<i;j++){
                  add_edge(ksa_edges[j][0],ksa_edges[j][1],KSA16);
                  add_edge(ksa_edges[j][1],ksa_edges[j][0],KSA16);
            }
      }


  return 0;
}
```

```
/**
./test < subset.txt
KSA16 is nonplanner

Found at:117237 KSA16 - 39edges
{1,17}, {1,32}, {1,46}, {1,58}, {2,17}, {2,18}, {3,18}, {3,19},
{4,19}, {4,20}, {5,20}, {5,21}, {6,21}, {6,22}, {7,22}, {7,23},
{8,23}, {8,24}, {9,24}, {9,25}, {10,25}, {10,26}, {11,26}, {11,27},
{12,27}, {12,28}, {13,28}, {13,29}, {14,29}, {14,30}, {15,30},
{15,31}, {16,31}, {17,33}, {17,47}, {17,59}, {18,32}, {18,34},
{19,33},
*/
#include <iostream>
```

```
#include <boost/graph/adjacency_list.hpp>
#include <boost/graph/boyer_myrvold_planar_test.hpp>
#include <vector>
//98 edges
int ksa_edges[][3]={
{1,17,1},
{1,32,1},
{1,46,1},
{1,58,1},
{2,17,1},
{2,18,1},
{3,18,1},
{3,19,1},
{4,19,1},
{4,20,1},
{5,20,1},
{5,21,1},
{6,21,1},
{6,22,1},
{7,22,1},
{7,23,1},
{8,23,1},
{8,24,1},
{9,24,1},
{9,25,1},
{10,25,1},
{10,26,1},
{11,26,1},
{11,27,1},
{12,27,1},
{12,28,1},
{13,28,1},
{13,29,1},
{14,29,1},
{14,30,1},
{15,30,1},
{15,31,1},
{16,31,1},
{17,33,1},
{17,47,1},
{17,59,1},
{18,32,1},
{18,34,1},
{19,33,1},
{19,35,1},
{20,34,1},
{20,36,1},
{21,35,1},
{21,37,1},
{22,36,1},
{22,38,1},
{23,37,1},
```

```
{23,39,1},
{24,38,1},
{24,40,1},
{25,39,1},
{25,41,1},
{26,40,1},
{26,42,1},
{27,41,1},
{27,43,1},
{28,42,1},
{28,44,1},
{29,43,1},
{29,45,1},
{30,44,1},
{31,45,1},
{32,48,1},
{32,60,1},
{33,49,1},
{33,61,1},
{34,46,1},
{34,50,1},
{35,47,1},
{35,51,1},
{36,48,1},
{36,52,1},
{37,49,1},
{37,53,1},
{38,50,1},
{38,54,1},
{39,51,1},
{39,55,1},
{40,52,1},
{40,56,1},
{41,53,1},
{41,57,1},
{42,54,1},
{43,55,1},
{44,56,1},
{45,57,1},
{46,62,1},
{47,63,1},
{48,64,1},
{49,65,1},
{50,58,1},
{51,59,1},
{52,60,1},
{53,61,1},
{54,62,1},
{55,63,1},
{56,64,1},
{57,65,1}
};
```

```
int main(int argc, char** argv)
{

  // This program illustrates a simple use of
boyer_myrvold_planar_embedding
  // as a simple yes/no test for planarity.

  using namespace boost;

  typedef adjacency_list<vecS,
                         vecS,
                         undirectedS,
                         property<vertex_index_t, int>
                         > graph;
     int edg[200],i,j,c=0,min=9999;
  graph KSA16(65);
     for (int i=0;i<98;i++)
          add_edge(ksa_edges[i][0],ksa_edges[i][1], KSA16);

if (boyer_myrvold_planarity_test(KSA16))
    std::cout << "KSA16 is planar." << std::endl;
else
    std::cout << "KSA16 is nonplanner"<< std::endl;

std::cout<<"After Removal of Edges:\n";
//=========================================================================
remove_edge(1,46,KSA16);
remove_edge(1,58,KSA16);
remove_edge(2,17,KSA16);
remove_edge(3,18,KSA16);
remove_edge(4,19,KSA16);
remove_edge(5,20,KSA16);
remove_edge(6,21,KSA16);
remove_edge(7,22,KSA16);
remove_edge(8,23,KSA16);
remove_edge(9,24,KSA16);
remove_edge(10,25,KSA16);
remove_edge(11,26,KSA16);
remove_edge(12,27,KSA16);
remove_edge(13,28,KSA16);
remove_edge(14,29,KSA16);
remove_edge(15,30,KSA16);
remove_edge(17,33,KSA16);
remove_edge(18,32,KSA16);
remove_edge(19,33,KSA16);

//=========================================================================
if (boyer_myrvold_planarity_test(KSA16))
```

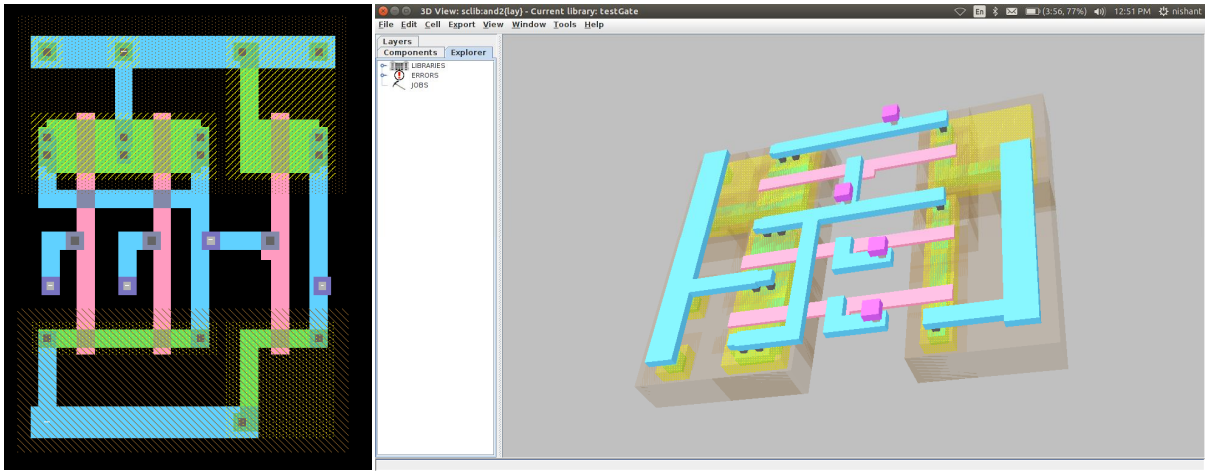
```
    std::cout << "KSA16 is planar." << std::endl;
else
    std::cout << "KSA16 is nonplanner"<< std::endl;

  return 0;
}
```

{1,46}, {1,58}, {2,17}, {3,18}, {4,19}, {5,20}, {6,21}, {7,22}, {8,23}, {9,24}, {10,25}, {11,26}, {12,27}, {13,28}, {14,29}, {15,30}, {17,33}, {18,32}, {19,33}

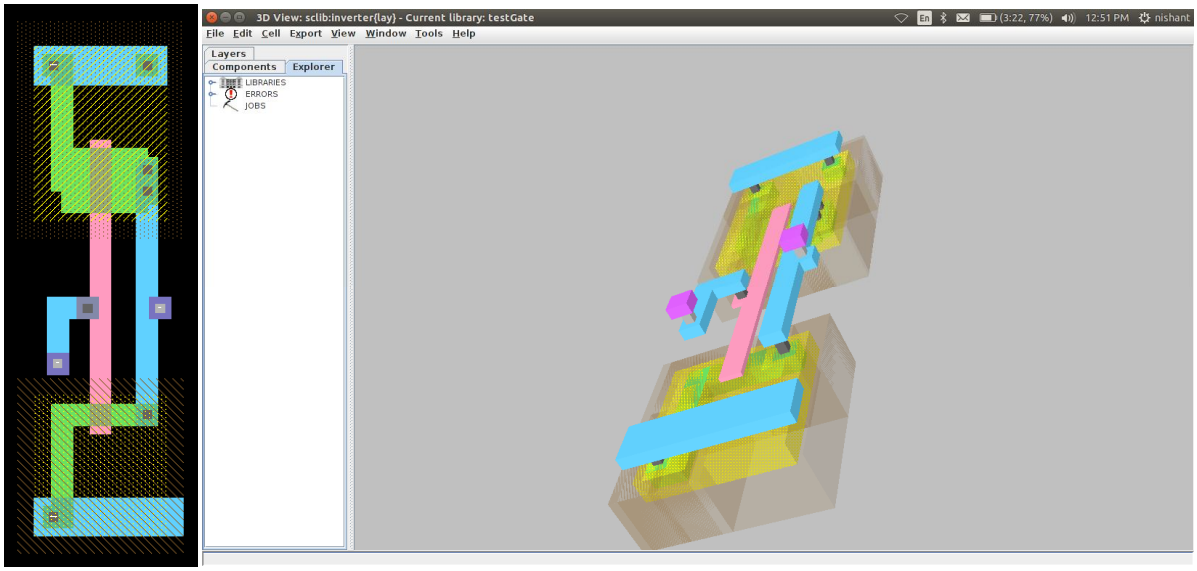
Above technique gives us conclusion that if we remove **above** edges from our design then it will become planner.

So now we can make our circuit in 2 planes, First plane contain all the edges of KSA except the above edges. Second plane contain all the edges above. And we create vertical connection between those nodes called bumps.

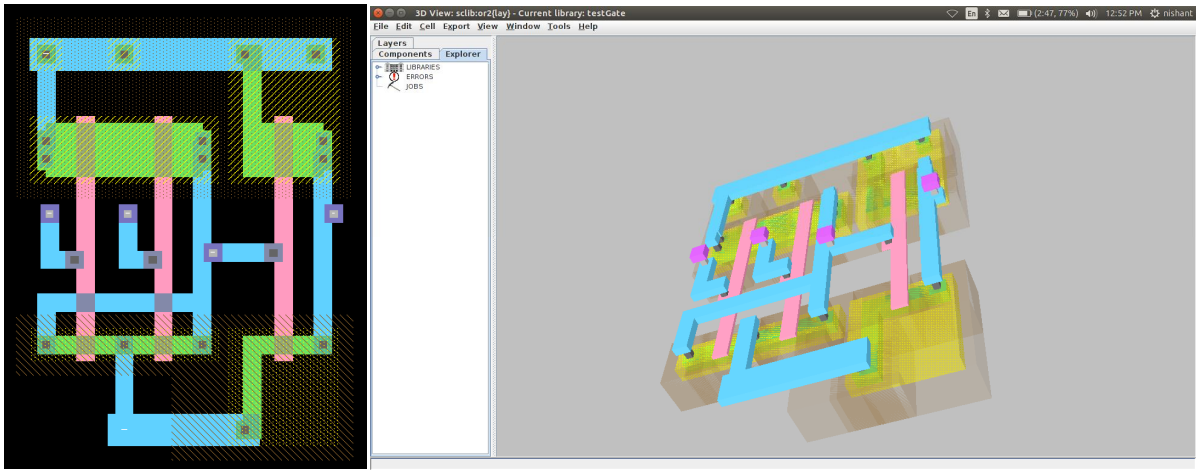
STANDARD CELLS USED



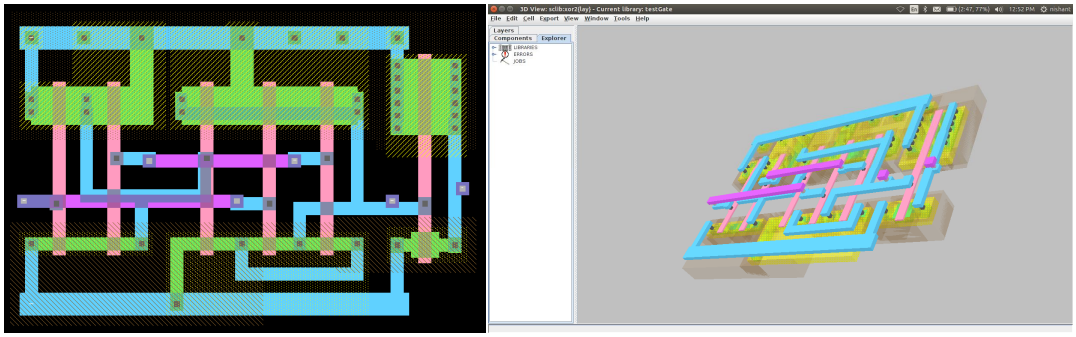
AND2 Gate Layout



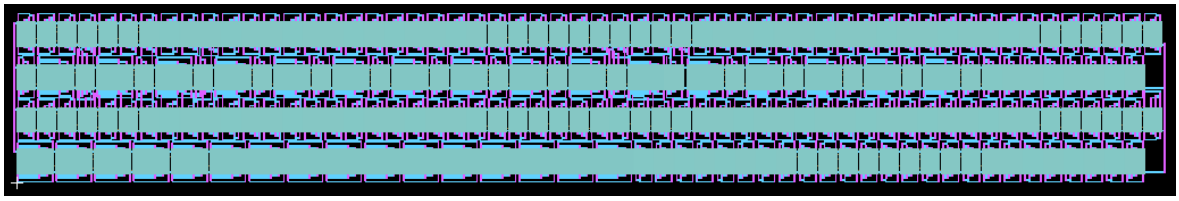
NOT Gate Layout



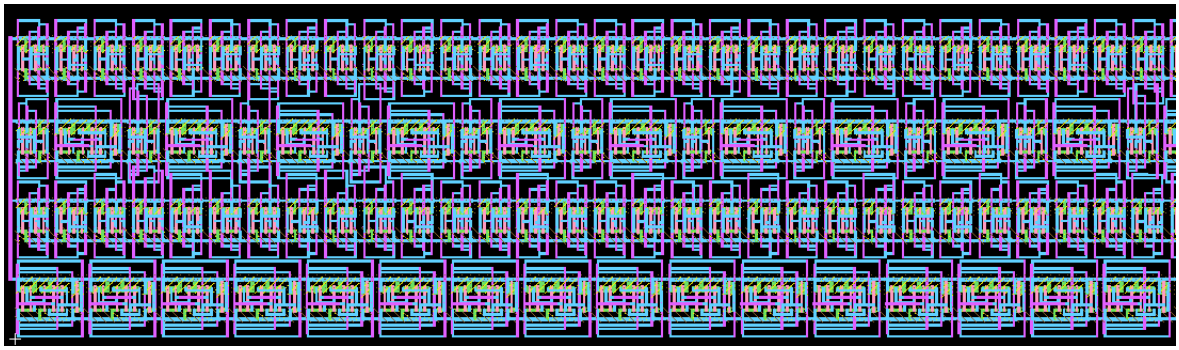
OR2 Gate Layout



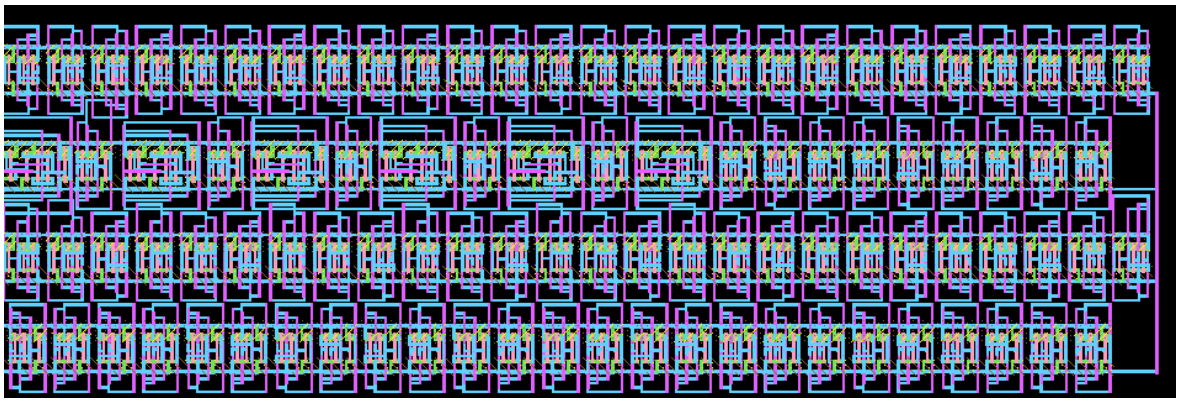
XOR2 Gate Layout



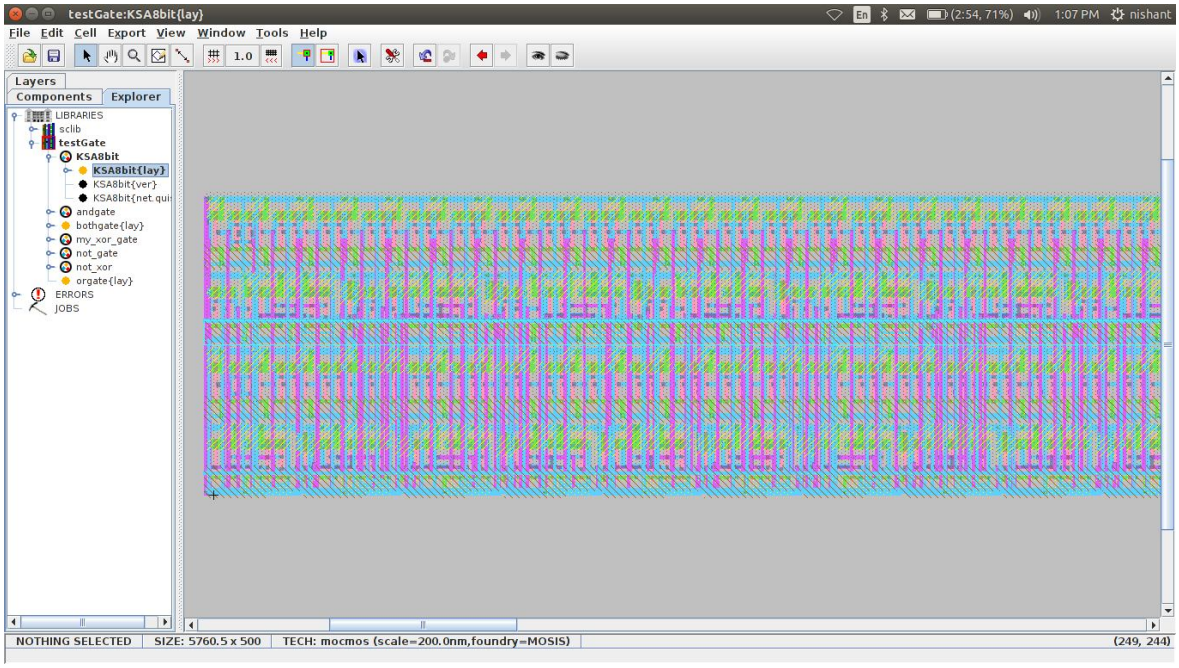
KSA 16 bit Layout



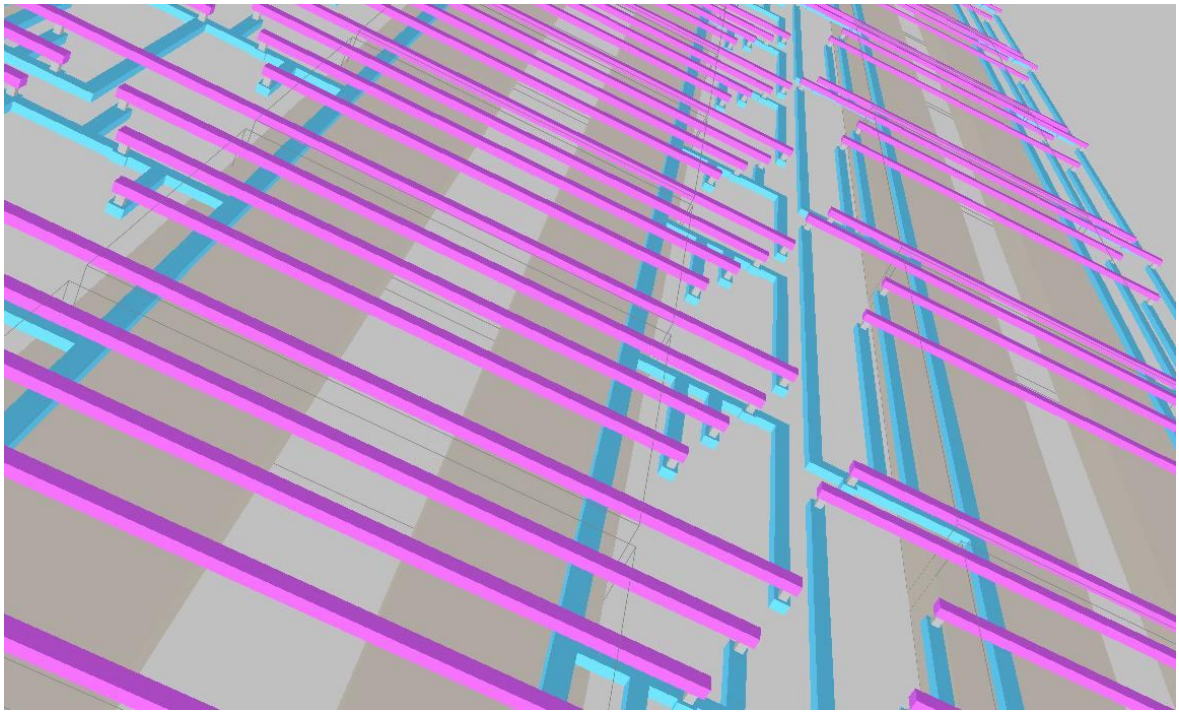
KSA 16 bit Part1 ZOOM 3x



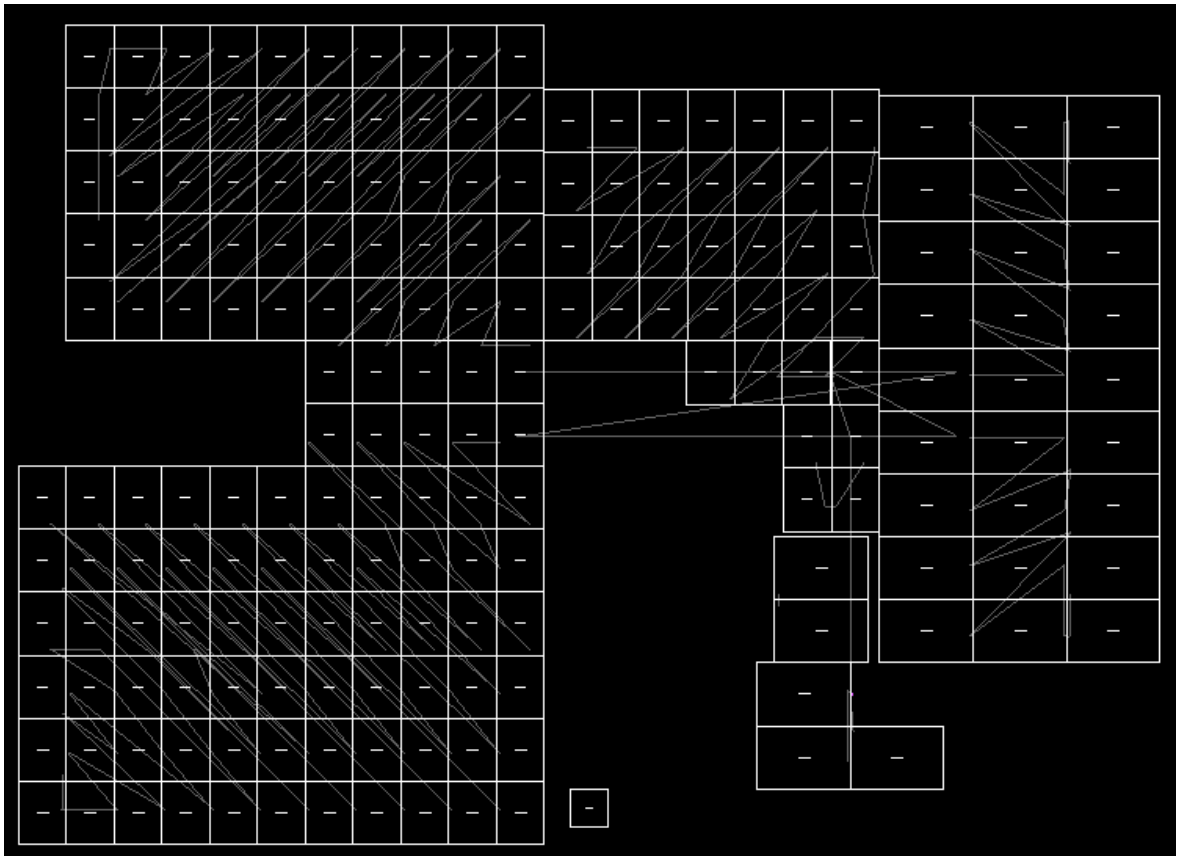
KSA 16 bit Part2 ZOOM 3x



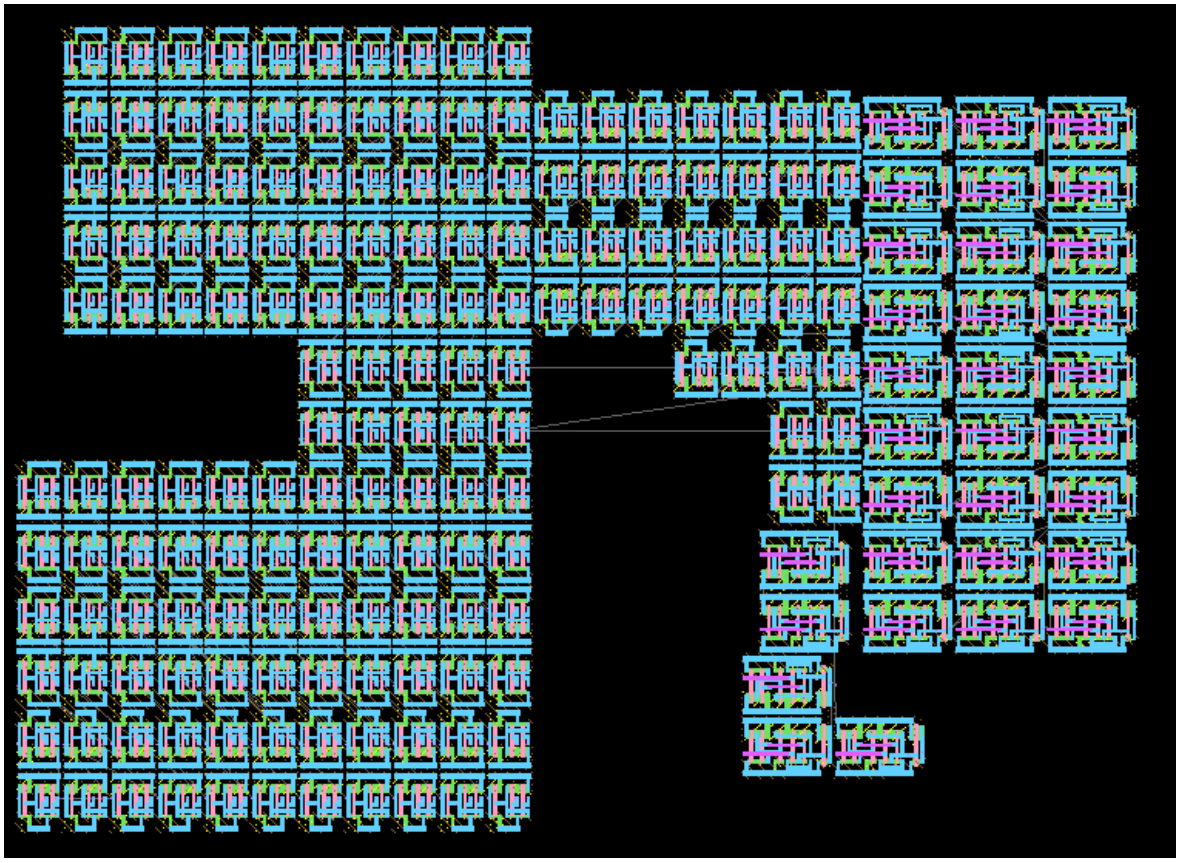
16bit KSA after Compactation



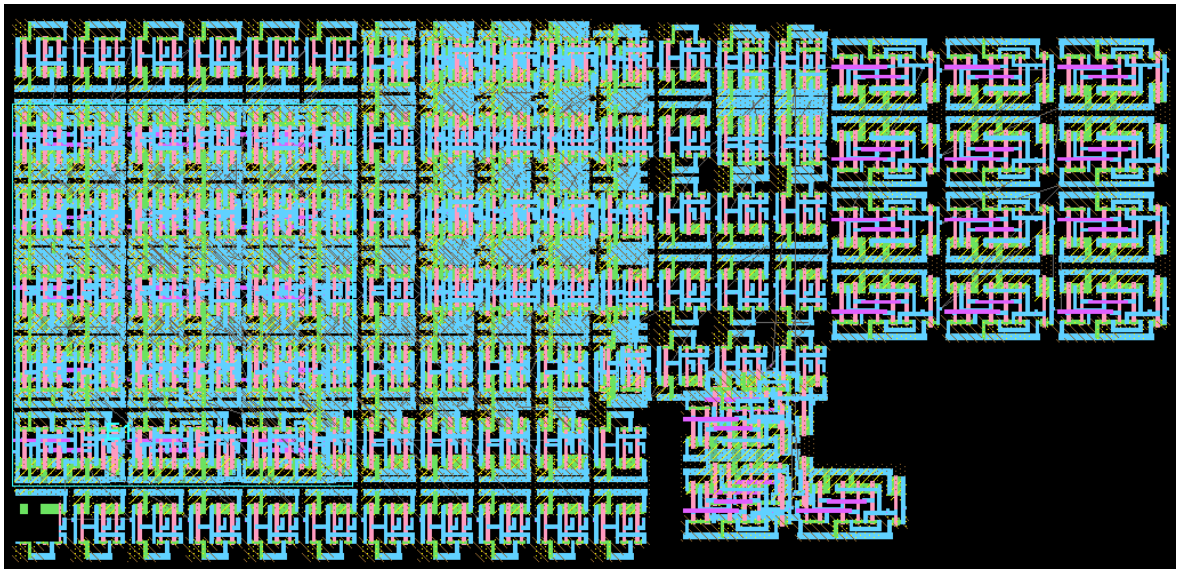
KSA 16bit 3D Layout



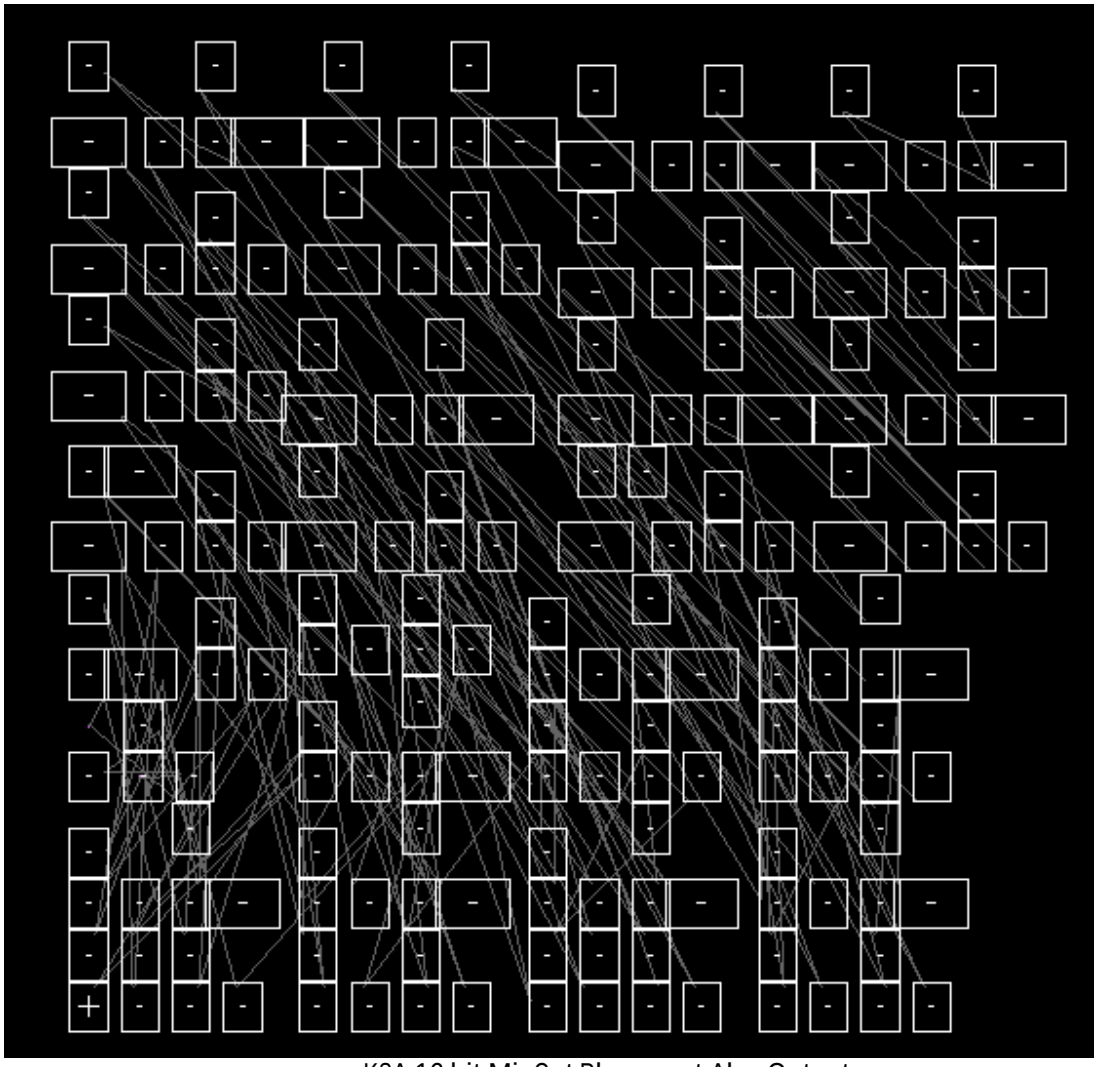
KSA 16bit Floor Plan



KSA 16 bit Floor Plan Full View

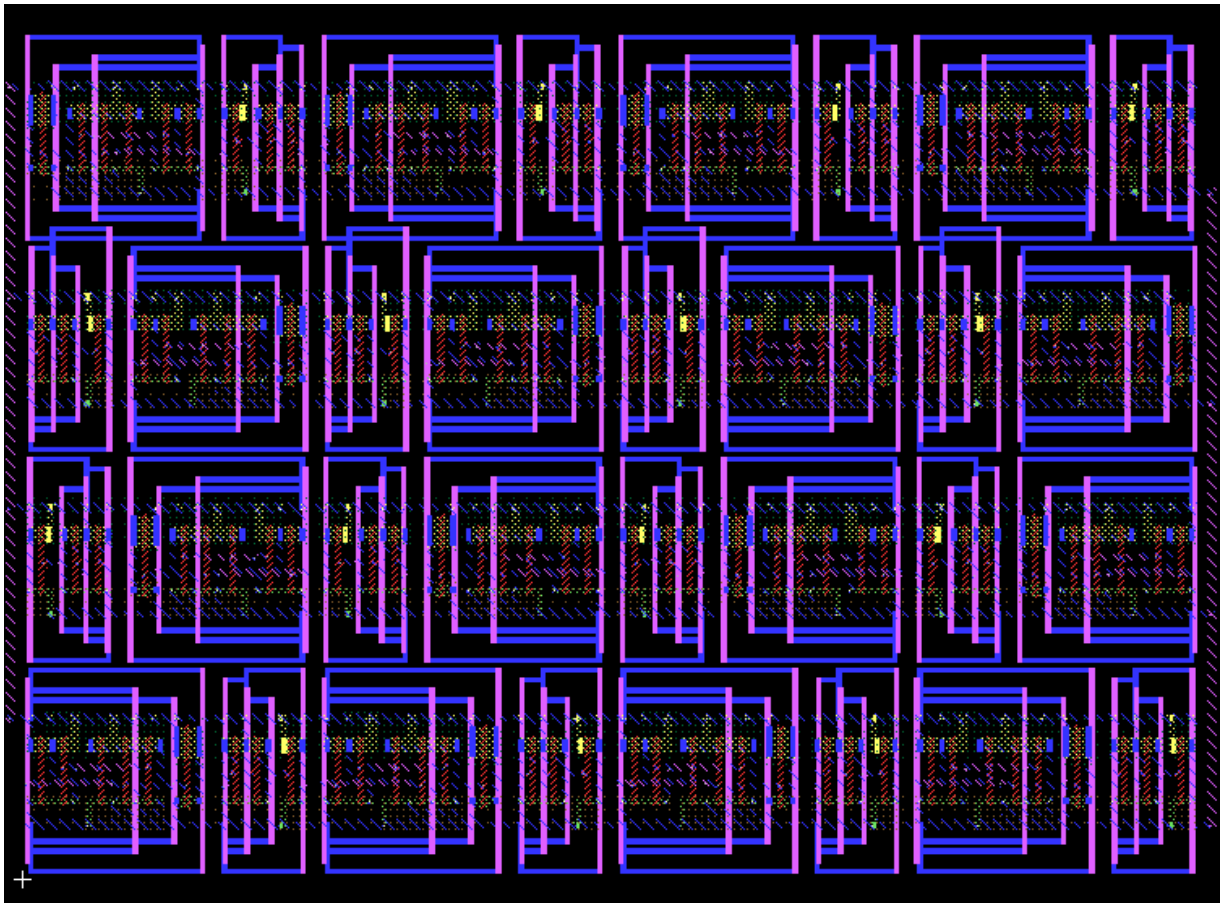


KSA 16bit Full Compact Floor Plan



KSA 16 bit MinCut Placement Algo Output

SIZE: 5792 x 849 | TECH: mocmos (scale=200.0nm,foundry=MOSIS)



KSA Layout in 3D view

Delay estimation for each network is a ratio of the following:

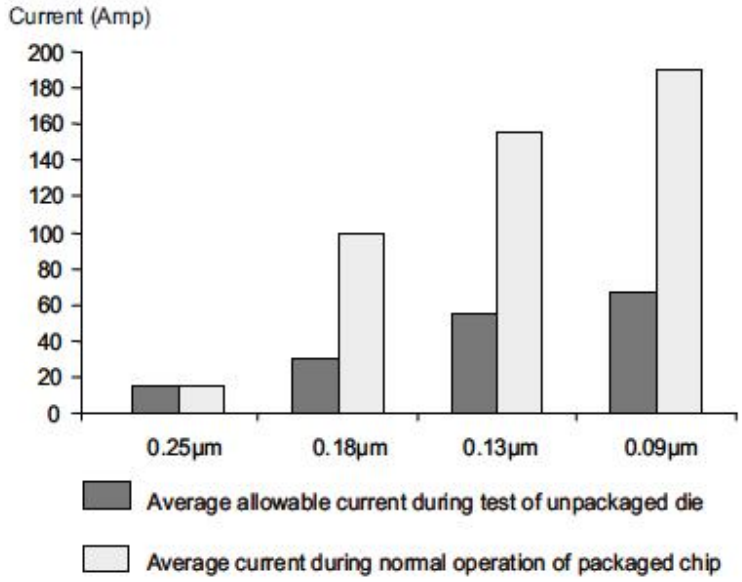
```
  Numerator is the sum of these layers on the network:
    Transistor widths
    Diffusion half-perimeter, weighted by 0.7
    Polysilicon half-perimeter, weighted by 0.25
    Metal half-perimeter, weighted by 0.25
  Denominator is the width of transistors on the network
    Separate results are computed for N and P transistors, as well as
their sum
```

Network 'gnd/vdd' ratio computation:

```
   N Gate width = 3550.0, P Gate width = 10650.0
   Layer Metal-1 half-perimeter is 247358.108 x 0.25 = 61839.527
   Layer Metal-2 half-perimeter is 149180 x 0.25 = 37295
   Layer N-Active half-perimeter is 19444 x 0.7 = 13610.8
   Layer P-Active half-perimeter is 39032 x 0.7 = 27322.4
   Layer Polysilicon-1 half-perimeter is 43320 x 0.25 = 10830
   Numerator is the sum of these factors (165097.727)
   N denominator = 3550.0, ratio = 46.506
   P denominator = 10650.0, ratio = 15.502
   N+P Denominator = 14200.0, ratio = 11.627
```

Power Estimation for Kogge Stone Adder

Power consumed by a digital circuit is very much depends upon the no of transitions generated in each gate of the circuit for various inputs. This is called *weighted switching activity (WSA)*. Lets us consider following figure.



Above figure displays the current limit on a wafer circuit for various sizes. We need to consider this because if we increase the power then our circuit may burn out. To calculate WSA we need to first calculate the transition which is made by our KSA 16 bit circuit. To find the number of transitions we used VPI/PLI with C and Verilog. PLI help us to write modules in traditional programming language (C) which can be directly used by the Verilog modules. Following is the VPI/PLI program with the Verilog module and the output generated by the program.

VPI/PLI and Kogge Stone Adder

KSA.C

```
# include <vpi_user.h>
int count=0,event=0;
int input[200][3];
static int init(){
      int i;
      for(i=0;i<200;i++){
            input[i][0]=0;
            input[i][1]=0;
            input[i][2]=0;
      }
      return 0;
}
void register_init() {
      s_vpi_systf_data data = {vpiSysTask, 0, "$init", init, 0, 0, 0};
      vpi_register_systf(&data);
}
//-------------------------------------------------
static int show(){
      int i,d=0;
      for(i=0;i<200;i++){
            if(input[i][2]>0){
                  vpi_printf("(G=%d,C=%d)\t",i,input[i][2]);
                  d++;
                  if(d%5==0)
                  vpi_printf("\n");
            }

      }
      vpi_printf("\n------------------------\nCount = %d, Event = %d",
count, event);
      return count;
}
void register_show() {
      s_vpi_systf_data data = {vpiSysTask, 0, "$show", show, 0, 0, 0};
      vpi_register_systf(&data);
}
//-------------------------------------------------
static int AND2(char *userdata) {
      vpiHandle systfref, args_iter, argh1,argh2,argh3,argh4;
      struct t_vpi_value argval1,argval2,argval3,argval4;
      int value1,value2,value3,value4;
      // Obtain a handle to the argument list
      systfref = vpi_handle(vpiSysTfCall, NULL);
      args_iter = vpi_iterate(vpiArgument, systfref);
      // Grab the value of the first argument
      argh1 = vpi_scan(args_iter);
      argh2 = vpi_scan(args_iter);
      argh3 = vpi_scan(args_iter);
```

```
      argh4 = vpi_scan(args_iter);

      argval1.format = vpiIntVal;
      argval2.format = vpiIntVal;
      argval3.format = vpiIntVal;
      argval4.format = vpiIntVal;
      vpi_get_value(argh1, &argval1);
      vpi_get_value(argh2, &argval2);
      vpi_get_value(argh3, &argval3);
      vpi_get_value(argh4, &argval4);

      value1 = argval1.value.integer;
      value2 = argval2.value.integer;
      value3 = argval3.value.integer;
      value4 = argval4.value.integer;

      if (input[value4][0] == value2 && input[value4][1] == value3){
            //vpi_printf("If part-%d-%d\n",value2,value3);
      }
      else{
            input[value4][2]++;
            input[value4][0] = value2 ;
            input[value4][1] = value3 ;
            event++;
            //vpi_printf("Else part\n");
      }
      //vpi_printf("VPI routine received %d, %d, %d\t\t",
value1,value2,value3);
      // Increment the value and put it back as first argument
      argval1.value.integer = value1 = (value2 & value3);
      vpi_put_value(argh1, &argval1, NULL, vpiForceFlag);
      //vpi_printf("VPI made it %d and puted it back\n",
argval1.value.integer);
      // Cleanup and return
      ///vpi_printf("Rec=%s\n",userdata);
      vpi_free_object(args_iter);
      count++;
      return count;
}

void register_AND2() {
      s_vpi_systf_data data = {vpiSysTask, 0, "$AND2", AND2, 0, 0, 0};
      vpi_register_systf(&data);
}
//----------------------------------------

static int OR2(char *userdata) {
      vpiHandle systfref, args_iter, argh1,argh2,argh3,argh4;
      struct t_vpi_value argval1,argval2,argval3,argval4;
      int value1,value2,value3,value4;
      // Obtain a handle to the argument list
      systfref = vpi_handle(vpiSysTfCall, NULL);
```

```
      args_iter = vpi_iterate(vpiArgument, systfref);
      // Grab the value of the first argument
      argh1 = vpi_scan(args_iter);
      argh2 = vpi_scan(args_iter);
      argh3 = vpi_scan(args_iter);
      argh4 = vpi_scan(args_iter);

      argval1.format = vpiIntVal;
      argval2.format = vpiIntVal;
      argval3.format = vpiIntVal;
      argval4.format = vpiIntVal;
      vpi_get_value(argh1, &argval1);
      vpi_get_value(argh2, &argval2);
      vpi_get_value(argh3, &argval3);
      vpi_get_value(argh4, &argval4);

      value1 = argval1.value.integer;
      value2 = argval2.value.integer;
      value3 = argval3.value.integer;
      value4 = argval4.value.integer;

      if (input[value4][0] == value2 && input[value4][1] == value3){
            }
      else{
            input[value4][2]++;
            input[value4][0] = value2 ;
            input[value4][1] = value3 ;
            event ++;
      }
      //vpi_printf("VPI routine received %d, %d, %d\t\t",
value1,value2,value3);
      // Increment the value and put it back as first argument
      argval1.value.integer = value1 = (value2 | value3);
      vpi_put_value(argh1, &argval1, NULL, vpiForceFlag);
      //vpi_printf("VPI made it %d and puted it back\t\t",
argval1.value.integer);
      // Cleanup and return
      ///vpi_printf("Rec=%s\n",userdata);
      vpi_free_object(args_iter);
      count++;
      return count;
}
void register_OR2() {
      s_vpi_systf_data data = {vpiSysTask, 0, "$OR2", OR2, 0, 0, 0};
      vpi_register_systf(&data);
}
//-------------------------------------------------------
static int XOR2(char *userdata) {
      vpiHandle systfref, args_iter, argh1,argh2,argh3,argh4;
      struct t_vpi_value argval1,argval2,argval3,argval4;
      int value1,value2,value3,value4;
      // Obtain a handle to the argument list
```

```
      systfref = vpi_handle(vpiSysTfCall, NULL);
      args_iter = vpi_iterate(vpiArgument, systfref);
      // Grab the value of the first argument
      argh1 = vpi_scan(args_iter);
      argh2 = vpi_scan(args_iter);
      argh3 = vpi_scan(args_iter);
      argh4 = vpi_scan(args_iter);

      argval1.format = vpiIntVal;
      argval2.format = vpiIntVal;
      argval3.format = vpiIntVal;
      argval4.format = vpiIntVal;
      vpi_get_value(argh1, &argval1);
      vpi_get_value(argh2, &argval2);
      vpi_get_value(argh3, &argval3);
      vpi_get_value(argh4, &argval4);

      value1 = argval1.value.integer;
      value2 = argval2.value.integer;
      value3 = argval3.value.integer;
      value4 = argval4.value.integer;

      if (input[value4][0] == value2 && input[value4][1] == value3){
            }
      else{
            input[value4][2]++;
            input[value4][0] = value2 ;
            input[value4][1] = value3 ;
            event++;
      }
      //vpi_printf("VPI routine received %d, %d, %d\t\t",
value1,value2,value3);
      // Increment the value and put it back as first argument
      argval1.value.integer = value1 = (value2 ^ value3);
      vpi_put_value(argh1, &argval1, NULL, vpiForceFlag);
      //vpi_printf("VPI made it %d and puted it back\t\t",
argval1.value.integer);
      // Cleanup and return
      ///vpi_printf("Rec=%s\n",userdata);
      vpi_free_object(args_iter);
      count++;
      return count;
}
void register_XOR2() {
      s_vpi_systf_data data = {vpiSysTask, 0, "$XOR2", XOR2, 0, 0, 0};
      vpi_register_systf(&data);
}

// Contains a zero-terminated list of functions that have to be called
at startup
void (*vlog_startup_routines[])() = {
      register_AND2,
```

```
      register_OR2,
      register_show,
      register_init,
      register_XOR2,
      0
};
```

KSA.V

```
module hello;//(sum,carry,a,b,c);
      wire [177:0] out;
      //input[15:0] a,b;
      wire [15:0] a;
      wire [15:0] b;
      //input c;
      wire c;
      output [15:0] sum;
      output carry;
      //wire val1 = 0, val2=0,val3=1;
      assign a=65535;//65535
      assign b=999;
      assign c=1;
      initial
      begin
            $init();

      $AND2 (out[1],a[15],b[15],1);
      $XOR2 (out[2],a[15],b[15],2);
      $AND2 (out[3],a[14],b[14],3);
      $XOR2 (out[4],a[14],b[14],4);
      $AND2 (out[5],a[13],b[13],5);
      $XOR2 (out[6],a[13],b[13],6);
      $AND2 (out[7],a[12],b[12],7);
      $XOR2 (out[8],a[12],b[12],8);
      $AND2 (out[9],a[11],b[11],9);
      $XOR2 (out[10],a[11],b[11],10);
      $AND2 (out[11],a[10],b[10],11);
      $XOR2 (out[12],a[10],b[10],12);
      $AND2 (out[13],a[9],b[9],13);
      $XOR2 (out[14],a[9],b[9],14);
      $AND2 (out[15],a[8],b[8],15);
      $XOR2 (out[16],a[8],b[8],16);
      $AND2 (out[17],a[7],b[7],17);
      $XOR2 (out[18],a[7],b[7],18);
      $AND2 (out[19],a[6],b[6],19);
      $XOR2 (out[20],a[6],b[6],20);
      $AND2 (out[21],a[5],b[5],21);
      $XOR2 (out[22],a[5],b[5],22);
      $AND2 (out[23],a[4],b[4],23);
```

```
      $XOR2 (out[24],a[4],b[4],24);
      $AND2 (out[25],a[3],b[3],25);
      $XOR2 (out[26],a[3],b[3],26);
      $AND2 (out[27],a[2],b[2],27);
      $XOR2 (out[28],a[2],b[2],28);
      $AND2 (out[29],a[1],b[1],29);
      $XOR2 (out[30],a[1],b[1],30);
      $AND2 (out[31],a[0],b[0],31);
      $XOR2 (out[32],a[0],b[0],32);
//-----------------------------------------------------------------------
      $AND2 (out[33],out[3],out[2],33);
      $OR2  (out[34],out[1],out[33],34);
      $AND2 (out[35],out[2],out[4],35);

      $AND2 (out[36],out[5],out[4],36);
      $OR2  (out[37],out[3],out[36],37);
      $AND2 (out[38],out[4],out[6],38);

      $AND2 (out[39],out[7],out[6],39);
      $OR2  (out[40],out[5],out[39],40);
      $AND2 (out[41],out[6],out[8],41);

      $AND2 (out[42],out[9],out[8],42);
      $OR2  (out[43],out[7],out[42],43);
      $AND2 (out[44],out[8],out[10],44);

      $AND2 (out[45],out[11],out[10],45);
      $OR2  (out[46],out[9],out[45],46);
      $AND2 (out[47],out[10],out[12],47);

      $AND2 (out[48],out[13],out[12],48);
      $OR2  (out[49],out[11],out[48],49);
      $AND2 (out[50],out[12],out[14],50);

      $AND2 (out[51],out[15],out[14],51);
      $OR2  (out[52],out[13],out[51],52);
      $AND2 (out[53],out[14],out[16],53);

      $AND2 (out[54],out[17],out[16],54);
      $OR2  (out[55],out[15],out[54],55);
      $AND2 (out[56],out[16],out[18],56);

      $AND2 (out[57],out[19],out[18],57);
      $OR2  (out[58],out[17],out[57],58);
      $AND2 (out[59],out[18],out[20],59);

      $AND2 (out[60],out[21],out[20],60);
      $OR2  (out[61],out[19],out[60],61);
      $AND2 (out[62],out[20],out[22],62);

      $AND2 (out[63],out[23],out[22],63);
      $OR2  (out[64],out[21],out[63],64);
```

```
      $AND2 (out[65],out[22],out[24],65);

      $AND2 (out[66],out[25],out[24],66);
      $OR2  (out[67],out[23],out[66],67);
      $AND2 (out[68],out[24],out[26],68);

      $AND2 (out[69],out[27],out[26],69);
      $OR2  (out[70],out[25],out[69],70);
      $AND2 (out[71],out[26],out[28],71);

      $AND2 (out[72],out[29],out[28],72);
      $OR2  (out[73],out[27],out[72],73);
      $AND2 (out[74],out[28],out[30],74);

      $AND2 (out[75],out[31],out[30],75);
      $OR2  (out[76],out[29],out[75],76);
      $AND2 (out[77],out[30],out[32],77);

      $AND2 (out[78],c,out[32],78);
      $OR2  (out[79],out[31],out[78],79);
//-----------------------------------------------------------------
      $AND2 (out[80],out[40],out[35],80);
      $OR2  (out[81],out[34],out[80],81);
      $AND2 (out[82],out[35],out[41],82);

      $AND2 (out[83],out[43],out[38],83);
      $OR2  (out[84],out[37],out[83],84);
      $AND2 (out[85],out[38],out[44],85);

      $AND2 (out[86],out[46],out[41],86);
      $OR2  (out[87],out[40],out[86],87);
      $AND2 (out[88],out[41],out[47],88);

      $AND2 (out[89],out[49],out[44],89);
      $OR2  (out[90],out[43],out[89],90);
      $AND2 (out[91],out[44],out[50],91);

      $AND2 (out[92],out[52],out[47],92);
      $OR2  (out[93],out[46],out[92],93);
      $AND2 (out[94],out[47],out[53],94);

      $AND2 (out[95],out[55],out[50],95);
      $OR2  (out[96],out[49],out[95],96);
      $AND2 (out[97],out[50],out[56],97);

      $AND2 (out[98],out[58],out[53],98);
      $OR2  (out[99],out[52],out[98],99);
      $AND2 (out[100],out[53],out[59],100);

      $AND2 (out[101],out[61],out[56],101);
      $OR2  (out[102],out[55],out[101],102);
      $AND2 (out[103],out[56],out[62],103);
```

```
      $AND2 (out[104],out[64],out[59],104);
      $OR2  (out[105],out[58],out[104],105);
      $AND2 (out[106],out[59],out[65],106);

      $AND2 (out[107],out[67],out[62],107);
      $OR2  (out[108],out[61],out[107],108);
      $AND2 (out[109],out[62],out[68],109);

      $AND2 (out[110],out[70],out[65],110);
      $OR2  (out[111],out[64],out[110],111);
      $AND2 (out[112],out[65],out[71],112);

      $AND2 (out[113],out[73],out[68],113);
      $OR2  (out[114],out[67],out[113],114);
      $AND2 (out[115],out[68],out[74],115);

      $AND2 (out[116],out[76],out[71],116);
      $OR2  (out[117],out[70],out[116],117);
      $AND2 (out[118],out[71],out[77],118);

      $AND2 (out[119],out[79],out[74],119);
      $OR2  (out[120],out[73],out[119],120);

      $AND2 (out[121],c,out[77],121);
      $OR2  (out[122],out[76],out[121],122);
//-----------------------------------------------------------------
      $AND2 (out[123],out[93],out[82],123);
      $OR2  (out[124],out[81],out[123],124);
      $AND2 (out[125],out[82],out[94],125);

      $AND2 (out[126],out[96],out[85],126);
      $OR2  (out[127],out[84],out[126],127);
      $AND2 (out[128],out[85],out[97],128);

      $AND2 (out[129],out[99],out[88],129);
      $OR2  (out[130],out[87],out[129],130);
      $AND2 (out[131],out[88],out[100],131);

      $AND2 (out[132],out[102],out[91],132);
      $OR2  (out[133],out[90],out[132],133);
      $AND2 (out[134],out[91],out[103],134);

      $AND2 (out[135],out[105],out[94],135);
      $OR2  (out[136],out[93],out[135],136);
      $AND2 (out[137],out[94],out[106],137);

      $AND2 (out[138],out[108],out[97],138);
      $OR2  (out[139],out[96],out[138],139);
      $AND2 (out[140],out[97],out[109],140);

      $AND2 (out[141],out[111],out[100],141);
```

```
      $OR2  (out[142],out[99],out[141],142);
      $AND2 (out[143],out[100],out[112],143);

      $AND2 (out[144],out[114],out[103],144);
      $OR2  (out[145],out[102],out[144],145);
      $AND2 (out[146],out[103],out[115],146);

      $AND2 (out[147],out[117],out[106],147);
      $OR2  (out[148],out[105],out[147],148);
      $AND2 (out[149],out[106],out[118],149);

      $AND2 (out[150],out[120],out[109],150);
      $OR2  (out[151],out[108],out[150],151);

      $AND2 (out[152],out[122],out[112],152);
      $OR2  (out[153],out[111],out[152],153);

      $AND2 (out[154],out[79],out[115],154);
      $OR2  (out[155],out[114],out[154],155);

      $AND2 (out[156],c,out[118],156);
      $OR2  (out[157],out[117],out[156],157);
//-----------------------------------------------------------------
      $AND2 (out[158],out[148],out[125],158);
      $OR2  (out[159],out[124],out[158],159);
      $AND2 (out[160],out[125],out[149],160);

      $AND2 (out[161],out[151],out[128],161);
      $OR2  (out[162],out[127],out[161],162);

      $AND2 (out[163],out[153],out[131],163);
      $OR2  (out[164],out[130],out[163],164);

      $AND2 (out[165],out[155],out[134],165);
      $OR2  (out[166],out[133],out[165],166);

      $AND2 (out[167],out[157],out[137],167);
      $OR2  (out[168],out[136],out[167],168);

      $AND2 (out[169],out[120],out[140],169);
      $OR2  (out[170],out[139],out[169],170);

      $AND2 (out[171],out[122],out[143],171);
      $OR2  (out[172],out[142],out[171],172);

      $AND2 (out[173],out[79],out[146],173);
      $OR2  (out[174],out[145],out[173],174);

      $AND2 (out[175],c,out[149],175);
      $OR2  (out[176],out[148],out[175],176);
//-----------------------------------------------------------------
      $AND2 (out[177],c,out[160],177);
```

```
      $OR2  (carry,out[159],out[177],178);

      $XOR2 (sum[15],out[2],out[162],179);
      $XOR2 (sum[14],out[4],out[164],180);
      $XOR2 (sum[13],out[6],out[166],181);
      $XOR2 (sum[12],out[8],out[168],182);
      $XOR2 (sum[11],out[10],out[170],183);
      $XOR2 (sum[10],out[12],out[172],184);
      $XOR2 (sum[9],out[14],out[174],185);
      $XOR2 (sum[8],out[16],out[176],186);
      $XOR2 (sum[7],out[18],out[151],187);
      $XOR2 (sum[6],out[20],out[153],188);
      $XOR2 (sum[5],out[22],out[155],189);
      $XOR2 (sum[4],out[24],out[157],190);
      $XOR2 (sum[3],out[26],out[120],191);
      $XOR2 (sum[2],out[28],out[122],192);
      $XOR2 (sum[1],out[30],out[79],193);
      $XOR2 (sum[0],out[32],c,194);


            $show();
      $display("\nAdding A=%d, B=%d and CarryIn=%d",a,b,c);
      $display("Sum: %d, Carry: %d\n",sum,carry);
      end
endmodule
```

KSA_tb.v

```
module hello_tb;
      /*reg [15:0]a,b;
      reg c;
      wire [15:0]sum;
      wire carry;
      hello my_ksa(sum,carry,a,b,c);
      initial
      begin
            $dumpfile("hello_tb.vcd");
            $dumpvars(0, hello_tb);
            a=65535; b=0; c=0;
            #15;
            b=1;
            #20;

      end*/
endmodule
```

RUN.SH

```
$ clear
$ echo "\nCompiling C Code File\n-------------------------"
$ iverilog-vpi hello.c
$ echo "\nCompiling V Code File\n-------------------------"
$ iverilog -o hello.vvp hello.v hello_tb.v
$ echo "Compilation Done! Running the Code\n
################--OUTPUT--################\n"
$ vvp -M. -mhello hello.vvp
$ echo "\n##################-DONE-##################\n"
```

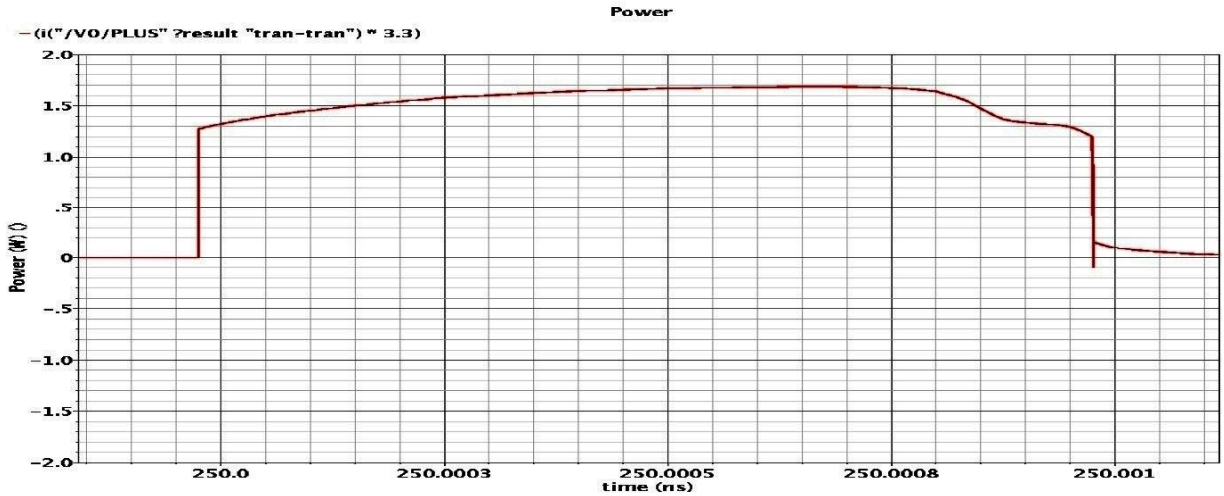
OUTPUT

```
Compiling C Code File
-------------------------
Compiling hello.c...
Making hello.vpi from  hello.o...

Compiling V Code File
-------------------------
Compilation Done! Running the Code
################--OUTPUT--################

(G=1,C=1)  (G=2,C=1)  (G=3,C=1)  (G=4,C=1)  (G=5,C=1)
(G=6,C=1)  (G=7,C=1)  (G=8,C=1)  (G=9,C=1)  (G=10,C=1)
(G=11,C=1) (G=12,C=1) (G=13,C=1) (G=14,C=1) (G=15,C=1)
(G=16,C=1) (G=17,C=1) (G=18,C=1) (G=19,C=1) (G=20,C=1)
(G=21,C=1) (G=22,C=1) (G=23,C=1) (G=24,C=1) (G=25,C=1)
(G=26,C=1) (G=27,C=1) (G=28,C=1) (G=29,C=1) (G=30,C=1)
(G=31,C=1) (G=32,C=1) (G=33,C=1) (G=35,C=1) (G=36,C=1)
(G=38,C=1) (G=39,C=1) (G=41,C=1) (G=42,C=1) (G=44,C=1)
(G=45,C=1) (G=47,C=1) (G=48,C=1) (G=49,C=1) (G=50,C=1)
(G=51,C=1) (G=52,C=1) (G=54,C=1) (G=55,C=1) (G=57,C=1)
(G=58,C=1) (G=60,C=1) (G=61,C=1) (G=64,C=1) (G=65,C=1)
(G=66,C=1) (G=68,C=1) (G=69,C=1) (G=70,C=1) (G=71,C=1)
(G=72,C=1) (G=73,C=1) (G=75,C=1) (G=76,C=1) (G=78,C=1)
(G=79,C=1) (G=80,C=1) (G=82,C=1) (G=83,C=1) (G=85,C=1)
(G=86,C=1) (G=88,C=1) (G=89,C=1) (G=90,C=1) (G=91,C=1)
(G=92,C=1) (G=93,C=1) (G=94,C=1) (G=95,C=1) (G=96,C=1)
(G=98,C=1) (G=99,C=1) (G=101,C=1) (G=102,C=1) (G=104,C=1)
(G=105,C=1) (G=108,C=1) (G=109,C=1) (G=110,C=1) (G=111,C=1)
(G=113,C=1) (G=114,C=1) (G=115,C=1) (G=116,C=1) (G=117,C=1)
(G=119,C=1) (G=120,C=1) (G=121,C=1) (G=122,C=1) (G=123,C=1)
(G=124,C=1) (G=125,C=1) (G=126,C=1) (G=127,C=1) (G=128,C=1)
```

```
(G=129,C=1) (G=130,C=1) (G=131,C=1) (G=132,C=1) (G=133,C=1)
(G=135,C=1) (G=136,C=1) (G=138,C=1) (G=139,C=1) (G=141,C=1)
(G=142,C=1) (G=144,C=1) (G=145,C=1) (G=147,C=1) (G=148,C=1)
(G=150,C=1) (G=151,C=1) (G=152,C=1) (G=153,C=1) (G=154,C=1)
(G=155,C=1) (G=156,C=1) (G=157,C=1) (G=158,C=1) (G=159,C=1)
(G=161,C=1) (G=162,C=1) (G=163,C=1) (G=164,C=1) (G=165,C=1)
(G=166,C=1) (G=167,C=1) (G=168,C=1) (G=169,C=1) (G=170,C=1)
(G=171,C=1) (G=172,C=1) (G=173,C=1) (G=174,C=1) (G=175,C=1)
(G=176,C=1) (G=177,C=1) (G=178,C=1) (G=179,C=1) (G=180,C=1)
(G=181,C=1) (G=182,C=1) (G=183,C=1) (G=184,C=1) (G=185,C=1)
(G=186,C=1) (G=187,C=1) (G=188,C=1) (G=189,C=1) (G=190,C=1)
(G=191,C=1) (G=192,C=1) (G=193,C=1) (G=194,C=1)
------------------------
Count = 194, Event = 164
Adding A=65535, B=  999 and CarryIn=1
Sum:   999, Carry: 1

##################-DONE-#################
```

$$WSA_{gk} = d_k(\tau_k + \phi_k f_k), \text{ where}$$
$$d_k = \begin{cases} 1, & \text{Transition occurs} \\ 0, & \text{No transition} \end{cases}$$

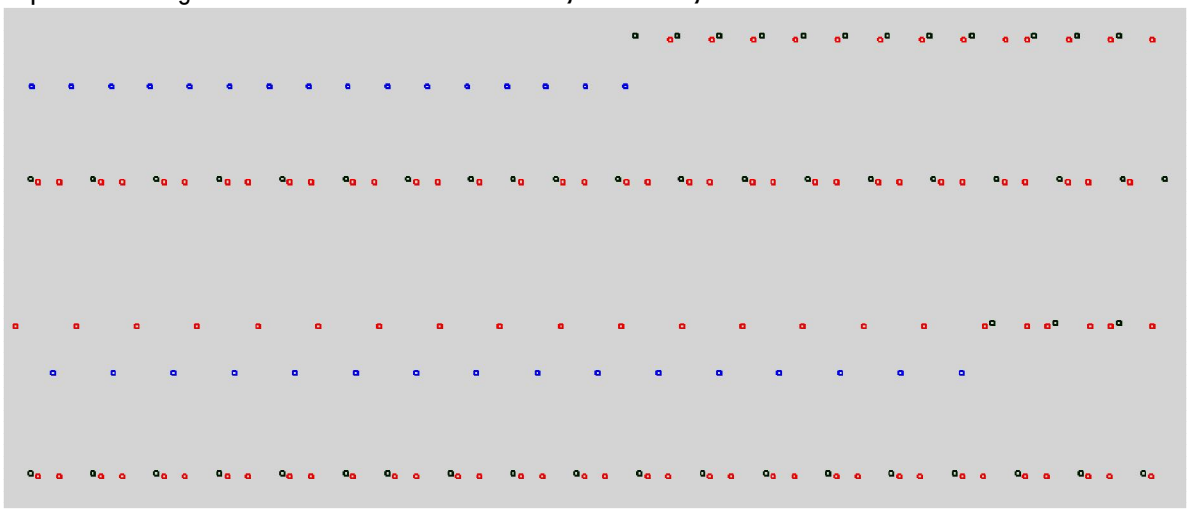
Above equation is used to perform power calculation (*University of Connecticut)*. Following is the graph for power calculation done on KSA.



This is show power vs time graph of power consumed by the KSA.

Power requires on different area on wafer is depend upon the number of gates in the grid. Following is the image which shows the number of gates in the various area of the wafer.

Output of Parsing the KSA16bit def file: **XOR: blue**, **AND: Red**, **OR: Green**

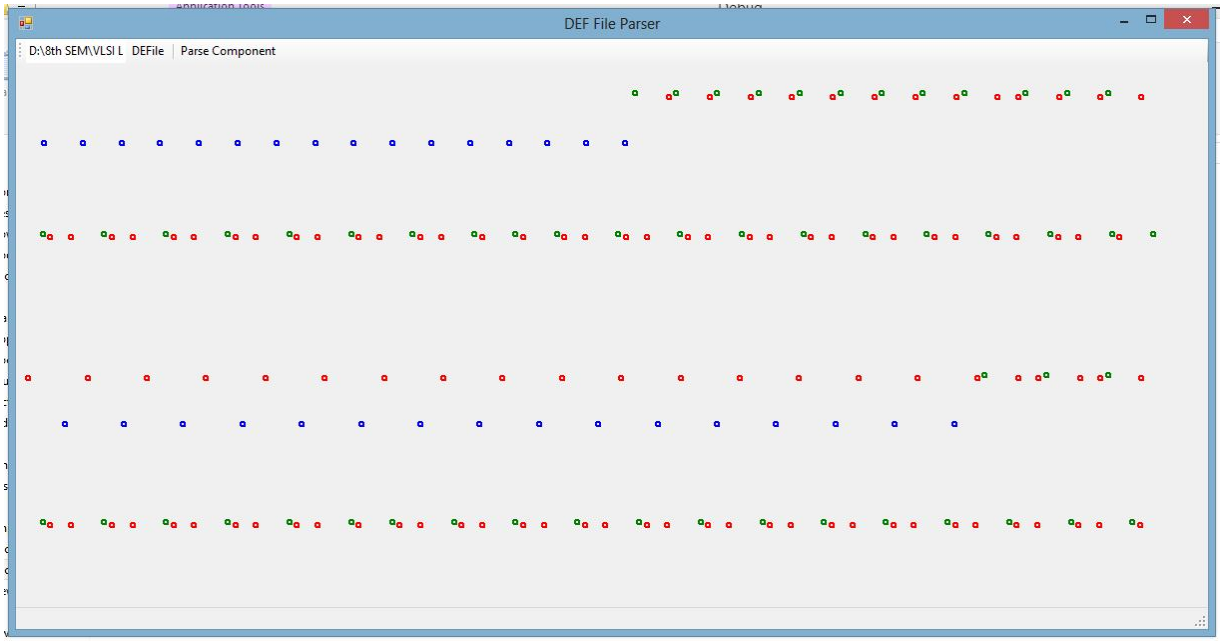


Code: Parser.vb

```
Imports System.Drawing
Imports System.Drawing.Drawing2D
Imports System.Windows.Forms
Public Class Form1

    Private Sub ToolStripButton1_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ToolStripButton1.Click
        opn.ShowDialog()
    End Sub

    Private Sub opn_FileOk(ByVal sender As System.Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles opn.FileOk
        Filename.Text = opn.FileName
    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub
    Dim c(500) As Component
    Dim n As Integer = -1
    Dim g As System.Drawing.Graphics
    Dim pxor As New System.Drawing.Pen(Color.Blue, 2)
    Dim pand As New System.Drawing.Pen(Color.Red, 2)
    Dim por As New System.Drawing.Pen(Color.Green, 2)
    Dim pnot As New System.Drawing.Pen(Color.Wheat, 2)
    Private Sub ToolStripButton2_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ToolStripButton2.Click
        Dim i As Integer = 0
        Dim maxX As Integer = -1
        Dim maxY As Integer = -1

        Dim str() As String
```

```
        str = IO.File.ReadAllLines(Filename.Text)
        Dim s As String
        Dim look As Boolean = False
        For Each s In str

            If s.Contains("COMPONENTS") = True Then
                look = True
            End If
            If s.Contains("END COMPONENTS") Then
                look = False
            End If
            If look Then
                If s.StartsWith("-") Then
                    Dim x As New Component()
                    x.name = s.Substring(2, s.IndexOf(" ", 3) - 2)
                    If s.Contains("xor2 +") Then
                        x.type = Gate.XOR2
                    ElseIf s.Contains("and2 +") Then
                        x.type = Gate.AND2
                    ElseIf s.Contains("or2 +") Then
                        x.type = Gate.OR2
                    ElseIf s.Contains("not1 +") Then
                        x.type = Gate.NOT1
                    Else
                        x.type = Gate.Undef
                    End If
                    Dim place As Integer = s.IndexOf("PLACED ( ")
                    Dim nextSpace As Integer = s.IndexOf(" ", place + 9)
                    x.x = s.Substring(place + 9, nextSpace - place - 9)
                    x.y = Integer.Parse(s.Substring(nextSpace, s.IndexOf(" ",
nextSpace + 1) - nextSpace))
                    maxX = IIf(maxX < x.x, x.x, maxX)
                    maxY = IIf(maxY < x.y, x.y, maxY)
                    c(i) = x
                    i += 1
                    n = i - 1
                    g = Panel1.CreateGraphics
                    Dim px, py As Integer
                    px = x.x
                    py = x.y
                    px = (px / 20000).ToString.Substring(0, ((px /
20000).ToString & ".").IndexOf("."))
                    py = (py / 6000).ToString.Substring(0, ((py /
6000).ToString & ".").IndexOf("."))
                    If x.type = Gate.AND2 Then
                        g.DrawEllipse(pand, px, py, 5, 5)
                    ElseIf x.type = Gate.NOT1 Then
                        g.DrawEllipse(pnot, px, py, 5, 5)
                    ElseIf x.type = Gate.OR2 Then
                        g.DrawEllipse(por, px, py, 5, 5)
                    ElseIf x.type = Gate.XOR2 Then
                        g.DrawEllipse(pxor, px, py, 5, 5)
                    End If

                    'PictureBox1.Refresh()
                    'MsgBox(x.toString)
                End If
```

```
            End If

        Next
        'Parsing Complete

    End Sub
End Class
Enum Gate
    AND2
    OR2
    XOR2
    NOT1
    Undef
End Enum
Class Component

    Public x, y As Integer
    Public name As String
    Public type As Gate
    Public Sub New()
        x = 0
        y = 0
        name = ""
        type = Gate.Undef
    End Sub


    Public Overrides Function toString() As String
        Dim s As String = "Component: " & name & ", Type: "
        If type = Gate.AND2 Then
            s &= "AND2, "
        ElseIf type = Gate.NOT1 Then
            s &= "NOT2, "
        ElseIf type = Gate.OR2 Then
            s &= "OR2, "
        ElseIf type = Gate.Undef Then
            s &= "Undefined, "
        ElseIf type = Gate.XOR2 Then
            s &= "XOR2, "
        End If
        s &= "(" & x & ", " & y & ")"
        Return s
    End Function

End Class
```


DEF File Parser
D:\8th SEM\VLSI L
DEFile
Parse Component

Kogge Stone Adder Verilog and netlist code

```
module KSA8bit( sum, carry, a, b, c);
        wire [177:0] out;
        input [15:0] a, b;
        input c;
        output [15:0] sum;
        output carry;
        //---------------------------------------------------
        and2 gate1(out[1],a[15],b[15]);
        xor2 gate2(out[2],a[15],b[15]);
        and2 gate3(out[3],a[14],b[14]);
        xor2 gate4(out[4],a[14],b[14]);
        and2 gate5(out[5],a[13],b[13]);
        xor2 gate6(out[6],a[13],b[13]);
        and2 gate7(out[7],a[12],b[12]);
        xor2 gate8(out[8],a[12],b[12]);
        and2 gate9(out[9],a[11],b[11]);
        xor2 gate10(out[10],a[11],b[11]);
        and2 gate11(out[11],a[10],b[10]);
        xor2 gate12(out[12],a[10],b[10]);
        and2 gate13(out[13],a[9],b[9]);
        xor2 gate14(out[14],a[9],b[9]);
        and2 gate15(out[15],a[8],b[8]);
        xor2 gate16(out[16],a[8],b[8]);
        and2 gate17(out[17],a[7],b[7]);
        xor2 gate18(out[18],a[7],b[7]);
        and2 gate19(out[19],a[6],b[6]);
        xor2 gate20(out[20],a[6],b[6]);
        and2 gate21(out[21],a[5],b[5]);
        xor2 gate22(out[22],a[5],b[5]);
        and2 gate23(out[23],a[4],b[4]);
        xor2 gate24(out[24],a[4],b[4]);
        and2 gate25(out[25],a[3],b[3]);
        xor2 gate26(out[26],a[3],b[3]);
        and2 gate27(out[27],a[2],b[2]);
        xor2 gate28(out[28],a[2],b[2]);
        and2 gate29(out[29],a[1],b[1]);
        xor2 gate30(out[30],a[1],b[1]);
        and2 gate31(out[31],a[0],b[0]);
        xor2 gate32(out[32],a[0],b[0]);
//-------------------------------------------------------------------------------------------------------------------
        and2 gate33(out[33],out[3],out[2]);
        or2  gate34(out[34],out[1],out[33]);
        and2 gate35(out[35],out[2],out[4]);

        and2 gate36(out[36],out[5],out[4]);
```

```
or2  gate37(out[37],out[3],out[36]);
and2 gate38(out[38],out[4],out[6]);

and2 gate39(out[39],out[7],out[6]);
or2  gate40(out[40],out[5],out[39]);
and2 gate41(out[41],out[6],out[8]);

and2 gate42(out[42],out[9],out[8]);
or2  gate43(out[43],out[7],out[42]);
and2 gate44(out[44],out[8],out[10]);

and2 gate45(out[45],out[11],out[10]);
or2  gate46(out[46],out[9],out[45]);
and2 gate47(out[47],out[10],out[12]);

and2 gate48(out[48],out[13],out[12]);
or2  gate49(out[49],out[11],out[48]);
and2 gate50(out[50],out[12],out[14]);

and2 gate51(out[51],out[15],out[14]);
or2  gate52(out[52],out[13],out[51]);
and2 gate53(out[53],out[14],out[16]);

and2 gate54(out[54],out[17],out[16]);
or2  gate55(out[55],out[15],out[54]);
and2 gate56(out[56],out[16],out[18]);

and2 gate57(out[57],out[19],out[18]);
or2  gate58(out[58],out[17],out[57]);
and2 gate59(out[59],out[18],out[20]);

and2 gate60(out[60],out[21],out[20]);
or2  gate61(out[61],out[19],out[60]);
and2 gate62(out[62],out[20],out[22]);

and2 gate63(out[63],out[23],out[22]);
or2  gate64(out[64],out[21],out[63]);
and2 gate65(out[65],out[22],out[24]);

and2 gate66(out[66],out[25],out[24]);
or2  gate67(out[67],out[23],out[66]);
and2 gate68(out[68],out[24],out[26]);

and2 gate69(out[69],out[27],out[26]);
or2  gate70(out[70],out[25],out[69]);
and2 gate71(out[71],out[26],out[28]);

and2 gate72(out[72],out[29],out[28]);
```

```
        or2  gate73(out[73],out[27],out[72]);
        and2 gate74(out[74],out[28],out[30]);

        and2 gate75(out[75],out[31],out[30]);
        or2  gate76(out[76],out[29],out[75]);
        and2 gate77(out[77],out[30],out[32]);

        and2 gate78(out[78],c,out[32]);
        or2  gate79(out[79],out[31],out[78]);
//------------------------------------------------------------------------
        and2 gate80(out[80],out[40],out[35]);
        or2  gate81(out[81],out[34],out[80]);
        and2 gate82(out[82],out[35],out[41]);

        and2 gate83(out[83],out[43],out[38]);
        or2  gate84(out[84],out[37],out[83]);
        and2 gate85(out[85],out[38],out[44]);

        and2 gate86(out[86],out[46],out[41]);
        or2  gate87(out[87],out[40],out[86]);
        and2 gate88(out[88],out[41],out[47]);

        and2 gate89(out[89],out[49],out[44]);
        or2  gate90(out[90],out[43],out[89]);
        and2 gate91(out[91],out[44],out[50]);

        and2 gate92(out[92],out[52],out[47]);
        or2  gate93(out[93],out[46],out[92]);
        and2 gate94(out[94],out[47],out[53]);

        and2 gate95(out[95],out[55],out[50]);
        or2  gate96(out[96],out[49],out[95]);
        and2 gate97(out[97],out[50],out[56]);

        and2 gate98(out[98],out[58],out[53]);
        or2  gate99(out[99],out[52],out[98]);
        and2 gate100(out[100],out[53],out[59]);

        and2 gate101(out[101],out[61],out[56]);
        or2  gate102(out[102],out[55],out[101]);
        and2 gate103(out[103],out[56],out[62]);

        and2 gate104(out[104],out[64],out[59]);
        or2  gate105(out[105],out[58],out[104]);
        and2 gate106(out[106],out[59],out[65]);

        and2 gate107(out[107],out[67],out[62]);
        or2  gate108(out[108],out[61],out[107]);
```

```
        and2 gate109(out[109],out[62],out[68]);

        and2 gate110(out[110],out[70],out[65]);
        or2  gate111(out[111],out[64],out[110]);
        and2 gate112(out[112],out[65],out[71]);

        and2 gate113(out[113],out[73],out[68]);
        or2  gate114(out[114],out[67],out[113]);
        and2 gate115(out[115],out[68],out[74]);

        and2 gate116(out[116],out[76],out[71]);
        or2  gate117(out[117],out[70],out[116]);
        and2 gate118(out[118],out[71],out[77]);

        and2 gate119(out[119],out[79],out[74]);
        or2  gate120(out[120],out[73],out[119]);

        and2 gate121(out[121],c,out[77]);
        or2  gate122(out[122],out[76],out[121]);
//-------------------------------------------------------------------------
        and2 gate123(out[123],out[93],out[82]);
        or2  gate124(out[124],out[81],out[123]);
        and2 gate125(out[125],out[82],out[94]);

        and2 gate126(out[126],out[96],out[85]);
        or2  gate127(out[127],out[84],out[126]);
        and2 gate128(out[128],out[85],out[97]);

        and2 gate129(out[129],out[99],out[88]);
        or2  gate130(out[130],out[87],out[129]);
        and2 gate131(out[131],out[88],out[100]);

        and2 gate132(out[132],out[102],out[91]);
        or2  gate133(out[133],out[90],out[132]);
        and2 gate134(out[134],out[91],out[103]);

        and2 gate135(out[135],out[105],out[94]);
        or2  gate136(out[136],out[93],out[135]);
        and2 gate137(out[137],out[94],out[106]);

        and2 gate138(out[138],out[108],out[97]);
        or2  gate139(out[139],out[96],out[138]);
        and2 gate140(out[140],out[97],out[109]);

        and2 gate141(out[141],out[111],out[100]);
        or2  gate142(out[142],out[99],out[141]);
        and2 gate143(out[143],out[100],out[112]);
```

```
        and2 gate144(out[144],out[114],out[103]);
        or2  gate145(out[145],out[102],out[144]);
        and2 gate146(out[146],out[103],out[115]);

        and2 gate147(out[147],out[117],out[106]);
        or2  gate148(out[148],out[105],out[147]);
        and2 gate149(out[149],out[106],out[118]);

        and2 gate150(out[150],out[120],out[109]);
        or2  gate151(out[151],out[108],out[150]);

        and2 gate152(out[152],out[122],out[112]);
        or2  gate153(out[153],out[111],out[152]);

        and2 gate154(out[154],out[79],out[115]);
        or2  gate155(out[155],out[114],out[154]);

        and2 gate156(out[156],c,out[118]);
        or2  gate157(out[157],out[117],out[156]);
//-------------------------------------------------------------------------------
        and2 gate158(out[158],out[148],out[125]);
        or2  gate159(out[159],out[124],out[158]);
        and2 gate160(out[160],out[125],out[149]);

        and2 gate161(out[161],out[151],out[128]);
        or2  gate162(out[162],out[127],out[161]);

        and2 gate163(out[163],out[153],out[131]);
        or2  gate164(out[164],out[130],out[163]);

        and2 gate165(out[165],out[155],out[134]);
        or2  gate166(out[166],out[133],out[165]);

        and2 gate167(out[167],out[157],out[137]);
        or2  gate168(out[168],out[136],out[167]);

        and2 gate169(out[169],out[120],out[140]);
        or2  gate170(out[170],out[139],out[169]);

        and2 gate171(out[171],out[122],out[143]);
        or2  gate172(out[172],out[142],out[171]);

        and2 gate173(out[173],out[79],out[146]);
        or2  gate174(out[174],out[145],out[173]);

        and2 gate175(out[175],c,out[149]);
        or2  gate176(out[176],out[148],out[175]);
//-------------------------------------------------------------------------------
```

```
        and2 gate177(out[177],c,out[160]);
        or2  gate178(carry,out[159],out[177]);
        xor2 gate179(sum[15],out[2],out[162]);
        xor2 gate180(sum[14],out[4],out[164]);
        xor2 gate181(sum[13],out[6],out[166]);
        xor2 gate182(sum[12],out[8],out[168]);
        xor2 gate183(sum[11],out[10],out[170]);
        xor2 gate184(sum[10],out[12],out[172]);
        xor2 gate185(sum[9],out[14],out[174]);
        xor2 gate186(sum[8],out[16],out[176]);
        xor2 gate187(sum[7],out[18],out[151]);
        xor2 gate188(sum[6],out[20],out[153]);
        xor2 gate189(sum[5],out[22],out[155]);
        xor2 gate190(sum[4],out[24],out[157]);
        xor2 gate191(sum[3],out[26],out[120]);
        xor2 gate192(sum[2],out[28],out[122]);
        xor2 gate193(sum[1],out[30],out[79]);
        xor2 gate194(sum[0],out[32],c);

endmodule
```

NETLIST CODE:

```
!************************************************
!  QUISC Command file
!
!  File Creation:    Wed Feb 25, 2015 18:08:12
!------------------------------------------------

create cell KSA8bit
create instance gate1 and2
create instance gate2 xor2
create instance gate3 and2
create instance gate4 xor2
create instance gate5 and2
create instance gate6 xor2
create instance gate7 and2
create instance gate8 xor2
create instance gate9 and2
create instance gate10 xor2
create instance gate11 and2
create instance gate12 xor2
create instance gate13 and2
create instance gate14 xor2
create instance gate15 and2
create instance gate16 xor2
create instance gate17 and2
create instance gate18 xor2
```

```
create instance gate19 and2
create instance gate20 xor2
create instance gate21 and2
create instance gate22 xor2
create instance gate23 and2
create instance gate24 xor2
create instance gate25 and2
create instance gate26 xor2
create instance gate27 and2
create instance gate28 xor2
create instance gate29 and2
create instance gate30 xor2
create instance gate31 and2
create instance gate32 xor2
create instance gate33 and2
create instance gate34 or2
create instance gate35 and2
create instance gate36 and2
create instance gate37 or2
create instance gate38 and2
create instance gate39 and2
create instance gate40 or2
create instance gate41 and2
create instance gate42 and2
create instance gate43 or2
create instance gate44 and2
create instance gate45 and2
create instance gate46 or2
create instance gate47 and2
create instance gate48 and2
create instance gate49 or2
create instance gate50 and2
create instance gate51 and2
create instance gate52 or2
create instance gate53 and2
create instance gate54 and2
create instance gate55 or2
create instance gate56 and2
create instance gate57 and2
create instance gate58 or2
create instance gate59 and2
create instance gate60 and2
create instance gate61 or2
create instance gate62 and2
create instance gate63 and2
create instance gate64 or2
create instance gate65 and2
create instance gate66 and2
```

```
create instance gate67 or2
create instance gate68 and2
create instance gate69 and2
create instance gate70 or2
create instance gate71 and2
create instance gate72 and2
create instance gate73 or2
create instance gate74 and2
create instance gate75 and2
create instance gate76 or2
create instance gate77 and2
create instance gate78 and2
create instance gate79 or2
create instance gate80 and2
create instance gate81 or2
create instance gate82 and2
create instance gate83 and2
create instance gate84 or2
create instance gate85 and2
create instance gate86 and2
create instance gate87 or2
create instance gate88 and2
create instance gate89 and2
create instance gate90 or2
create instance gate91 and2
create instance gate92 and2
create instance gate93 or2
create instance gate94 and2
create instance gate95 and2
create instance gate96 or2
create instance gate97 and2
create instance gate98 and2
create instance gate99 or2
create instance gate100 and2
create instance gate101 and2
create instance gate102 or2
create instance gate103 and2
create instance gate104 and2
create instance gate105 or2
create instance gate106 and2
create instance gate107 and2
create instance gate108 or2
create instance gate109 and2
create instance gate110 and2
create instance gate111 or2
create instance gate112 and2
create instance gate113 and2
create instance gate114 or2
```

```
create instance gate115 and2
create instance gate116 and2
create instance gate117 or2
create instance gate118 and2
create instance gate119 and2
create instance gate120 or2
create instance gate121 and2
create instance gate122 or2
create instance gate123 and2
create instance gate124 or2
create instance gate125 and2
create instance gate126 and2
create instance gate127 or2
create instance gate128 and2
create instance gate129 and2
create instance gate130 or2
create instance gate131 and2
create instance gate132 and2
create instance gate133 or2
create instance gate134 and2
create instance gate135 and2
create instance gate136 or2
create instance gate137 and2
create instance gate138 and2
create instance gate139 or2
create instance gate140 and2
create instance gate141 and2
create instance gate142 or2
create instance gate143 and2
create instance gate144 and2
create instance gate145 or2
create instance gate146 and2
create instance gate147 and2
create instance gate148 or2
create instance gate149 and2
create instance gate150 and2
create instance gate151 or2
create instance gate152 and2
create instance gate153 or2
create instance gate154 and2
create instance gate155 or2
create instance gate156 and2
create instance gate157 or2
create instance gate158 and2
create instance gate159 or2
create instance gate160 and2
create instance gate161 and2
create instance gate162 or2
```

```
create instance gate163 and2
create instance gate164 or2
create instance gate165 and2
create instance gate166 or2
create instance gate167 and2
create instance gate168 or2
create instance gate169 and2
create instance gate170 or2
create instance gate171 and2
create instance gate172 or2
create instance gate173 and2
create instance gate174 or2
create instance gate175 and2
create instance gate176 or2
create instance gate177 and2
create instance gate178 or2
create instance gate179 xor2
create instance gate180 xor2
create instance gate181 xor2
create instance gate182 xor2
create instance gate183 xor2
create instance gate184 xor2
create instance gate185 xor2
create instance gate186 xor2
create instance gate187 xor2
create instance gate188 xor2
create instance gate189 xor2
create instance gate190 xor2
create instance gate191 xor2
create instance gate192 xor2
create instance gate193 xor2
create instance gate194 xor2
connect gate3 ARG1 gate33 ARG2
connect gate33 ARG2 gate37 ARG2
connect gate45 ARG1 gate46 ARG3
connect gate25 ARG2 gate26 ARG2
connect gate57 ARG1 gate58 ARG3
connect gate69 ARG1 gate70 ARG3
connect gate31 ARG3 gate32 ARG3
connect gate7 ARG3 gate8 ARG3
connect gate113 ARG1 gate114 ARG3
connect gate125 ARG1 gate158 ARG3
connect gate158 ARG3 gate160 ARG2
connect gate137 ARG1 gate167 ARG3
connect gate149 ARG1 gate160 ARG3
connect gate160 ARG3 gate175 ARG3
connect gate21 ARG1 gate60 ARG2
connect gate60 ARG2 gate64 ARG2
```

connect gate33 ARG1 gate34 ARG3
connect gate101 ARG1 gate102 ARG3
connect gate4 ARG1 gate35 ARG3
connect gate35 ARG3 gate36 ARG3
connect gate36 ARG3 gate38 ARG2
connect gate38 ARG2 gate180 ARG2
connect gate27 ARG2 gate28 ARG2
connect gate34 ARG1 gate81 ARG2
connect gate46 ARG1 gate86 ARG2
connect gate86 ARG2 gate93 ARG2
connect gate58 ARG1 gate98 ARG2
connect gate98 ARG2 gate105 ARG2
connect gate5 ARG3 gate6 ARG3
connect gate124 ARG1 gate159 ARG2
connect gate136 ARG1 gate168 ARG2
connect gate148 ARG1 gate158 ARG2
connect gate158 ARG2 gate176 ARG2
connect gate10 ARG1 gate44 ARG3
connect gate44 ARG3 gate45 ARG3
connect gate45 ARG3 gate47 ARG2
connect gate47 ARG2 gate183 ARG2
connect gate22 ARG1 gate62 ARG3
connect gate62 ARG3 gate63 ARG3
connect gate63 ARG3 gate65 ARG2
connect gate65 ARG2 gate189 ARG2
connect gate90 ARG1 gate133 ARG2
connect gate100 ARG1 gate131 ARG3
connect gate131 ARG3 gate141 ARG3
connect gate141 ARG3 gate143 ARG2
connect gate112 ARG1 gate143 ARG3
connect gate143 ARG3 gate152 ARG3
connect gate5 ARG1 gate36 ARG2
connect gate36 ARG2 gate40 ARG2
connect gate11 ARG2 gate12 ARG2
connect gate23 ARG1 gate63 ARG2
connect gate63 ARG2 gate67 ARG2
connect gate27 ARG3 gate28 ARG3
connect gate35 ARG1 gate80 ARG3
connect gate80 ARG3 gate82 ARG2
connect gate47 ARG1 gate88 ARG3
connect gate88 ARG3 gate92 ARG3
connect gate92 ARG3 gate94 ARG2
connect gate59 ARG1 gate100 ARG3
connect gate100 ARG3 gate104 ARG3
connect gate104 ARG3 gate106 ARG2
connect gate3 ARG3 gate4 ARG3
connect gate21 ARG2 gate22 ARG2
connect gate135 ARG1 gate136 ARG3

connect gate147 ARG1 gate148 ARG3
connect gate159 ARG1 gate178 ARG2
connect gate11 ARG1 gate45 ARG2
connect gate45 ARG2 gate49 ARG2
connect gate111 ARG1 gate141 ARG2
connect gate141 ARG2 gate153 ARG2
connect gate91 ARG1 gate132 ARG3
connect gate132 ARG3 gate134 ARG2
connect gate123 ARG1 gate124 ARG3
connect gate9 ARG2 gate10 ARG2
connect gate78 ARG2 gate121 ARG2
connect gate121 ARG2 gate156 ARG2
connect gate156 ARG2 gate175 ARG2
connect gate175 ARG2 gate177 ARG2
connect gate177 ARG2 gate194 ARG3
connect gate6 ARG1 gate38 ARG3
connect gate38 ARG3 gate39 ARG3
connect gate39 ARG3 gate41 ARG2
connect gate41 ARG2 gate181 ARG2
connect gate12 ARG1 gate47 ARG3
connect gate47 ARG3 gate48 ARG3
connect gate48 ARG3 gate50 ARG2
connect gate50 ARG2 gate184 ARG2
connect gate29 ARG3 gate30 ARG3
connect gate24 ARG1 gate65 ARG3
connect gate65 ARG3 gate66 ARG3
connect gate66 ARG3 gate68 ARG2
connect gate68 ARG2 gate190 ARG2
connect gate36 ARG1 gate37 ARG3
connect gate48 ARG1 gate49 ARG3
connect gate23 ARG2 gate24 ARG2
connect gate1 ARG3 gate2 ARG3
connect gate92 ARG1 gate93 ARG3
connect gate146 ARG1 gate173 ARG3
connect gate158 ARG1 gate159 ARG3
connect gate110 ARG1 gate111 ARG3
connect gate122 ARG1 gate152 ARG2
connect gate152 ARG2 gate171 ARG2
connect gate171 ARG2 gate192 ARG3
connect gate80 ARG1 gate81 ARG3
connect gate134 ARG1 gate165 ARG3
connect gate7 ARG1 gate39 ARG2
connect gate39 ARG2 gate43 ARG2
connect gate13 ARG1 gate48 ARG2
connect gate48 ARG2 gate52 ARG2
connect gate25 ARG1 gate66 ARG2
connect gate66 ARG2 gate70 ARG2
connect gate37 ARG1 gate84 ARG2

connect gate49 ARG1 gate89 ARG2
connect gate89 ARG2 gate96 ARG2
connect gate17 ARG2 gate18 ARG2
connect gate23 ARG3 gate24 ARG3
connect gate81 ARG1 gate124 ARG2
connect gate157 ARG1 gate167 ARG2
connect gate167 ARG2 gate190 ARG3
connect gate93 ARG1 gate123 ARG2
connect gate123 ARG2 gate136 ARG2
connect gate169 ARG1 gate170 ARG3
connect gate15 ARG2 gate16 ARG2
connect gate121 ARG1 gate122 ARG3
connect gate133 ARG1 gate166 ARG2
connect gate145 ARG1 gate174 ARG2
connect gate8 ARG1 gate41 ARG3
connect gate41 ARG3 gate42 ARG3
connect gate42 ARG3 gate44 ARG2
connect gate44 ARG2 gate182 ARG2
connect gate14 ARG1 gate50 ARG3
connect gate50 ARG3 gate51 ARG3
connect gate51 ARG3 gate53 ARG2
connect gate53 ARG2 gate185 ARG2
connect gate26 ARG1 gate68 ARG3
connect gate68 ARG3 gate69 ARG3
connect gate69 ARG3 gate71 ARG2
connect gate71 ARG2 gate191 ARG2
connect gate19 ARG2 gate20 ARG2
connect gate38 ARG1 gate83 ARG3
connect gate83 ARG3 gate85 ARG2
connect gate25 ARG3 gate26 ARG3
connect gate70 ARG1 gate110 ARG2
connect gate110 ARG2 gate117 ARG2
connect gate168 ARG1 gate182 ARG3
connect gate82 ARG1 gate123 ARG3
connect gate123 ARG3 gate125 ARG2
connect gate94 ARG1 gate125 ARG3
connect gate125 ARG3 gate135 ARG3
connect gate135 ARG3 gate137 ARG2
connect gate120 ARG1 gate150 ARG2
connect gate150 ARG2 gate169 ARG2
connect gate169 ARG2 gate191 ARG3
connect gate132 ARG1 gate133 ARG3
connect gate144 ARG1 gate145 ARG3
connect gate156 ARG1 gate157 ARG3
connect gate9 ARG1 gate42 ARG2
connect gate42 ARG2 gate46 ARG2
connect gate11 ARG3 gate12 ARG3
connect gate15 ARG1 gate51 ARG2

connect gate51 ARG2 gate55 ARG2
connect gate27 ARG1 gate69 ARG2
connect gate69 ARG2 gate73 ARG2
connect gate19 ARG3 gate20 ARG3
connect gate39 ARG1 gate40 ARG3
connect gate71 ARG1 gate112 ARG3
connect gate112 ARG3 gate116 ARG3
connect gate116 ARG3 gate118 ARG2
connect gate17 ARG3 gate18 ARG3
connect gate83 ARG1 gate84 ARG3
connect gate95 ARG1 gate96 ARG3
connect gate131 ARG1 gate163 ARG3
connect gate143 ARG1 gate171 ARG3
connect gate155 ARG1 gate165 ARG2
connect gate165 ARG2 gate189 ARG3
connect gate167 ARG1 gate168 ARG3
connect gate9 ARG3 gate10 ARG3
connect gate16 ARG1 gate53 ARG3
connect gate53 ARG3 gate54 ARG3
connect gate54 ARG3 gate56 ARG2
connect gate56 ARG2 gate186 ARG2
connect gate21 ARG3 gate22 ARG3
connect gate28 ARG1 gate71 ARG3
connect gate71 ARG3 gate72 ARG3
connect gate72 ARG3 gate74 ARG2
connect gate74 ARG2 gate192 ARG2
connect gate60 ARG1 gate61 ARG3
connect gate72 ARG1 gate73 ARG3
connect gate13 ARG2 gate14 ARG2
connect gate84 ARG1 gate127 ARG2
connect gate96 ARG1 gate126 ARG2
connect gate126 ARG2 gate139 ARG2
connect gate130 ARG1 gate164 ARG2
connect gate142 ARG1 gate172 ARG2
connect gate154 ARG1 gate155 ARG3
connect gate166 ARG1 gate181 ARG3
connect gate17 ARG1 gate54 ARG2
connect gate54 ARG2 gate58 ARG2
connect gate29 ARG1 gate72 ARG2
connect gate72 ARG2 gate76 ARG2
connect gate13 ARG3 gate14 ARG3
connect gate61 ARG1 gate101 ARG2
connect gate101 ARG2 gate108 ARG2
connect gate73 ARG1 gate113 ARG2
connect gate113 ARG2 gate120 ARG2
connect gate85 ARG1 gate126 ARG3
connect gate126 ARG3 gate128 ARG2
connect gate97 ARG1 gate128 ARG3

connect gate128 ARG3 gate138 ARG3
connect gate138 ARG3 gate140 ARG2
connect gate109 ARG1 gate140 ARG3
connect gate140 ARG3 gate150 ARG3
connect gate141 ARG1 gate142 ARG3
connect gate153 ARG1 gate163 ARG2
connect gate163 ARG2 gate188 ARG3
connect gate165 ARG1 gate166 ARG3
connect gate177 ARG1 gate178 ARG3
connect gate18 ARG1 gate56 ARG3
connect gate56 ARG3 gate57 ARG3
connect gate57 ARG3 gate59 ARG2
connect gate59 ARG2 gate187 ARG2
connect gate50 ARG1 gate91 ARG3
connect gate91 ARG3 gate95 ARG3
connect gate95 ARG3 gate97 ARG2
connect gate15 ARG3 gate16 ARG3
connect gate62 ARG1 gate103 ARG3
connect gate103 ARG3 gate107 ARG3
connect gate107 ARG3 gate109 ARG2
connect gate74 ARG1 gate115 ARG3
connect gate115 ARG3 gate119 ARG3
connect gate86 ARG1 gate87 ARG3
connect gate98 ARG1 gate99 ARG3
connect gate108 ARG1 gate138 ARG2
connect gate138 ARG2 gate151 ARG2
connect gate140 ARG1 gate169 ARG3
connect gate152 ARG1 gate153 ARG3
connect gate164 ARG1 gate180 ARG3
connect gate176 ARG1 gate186 ARG3
connect gate19 ARG1 gate57 ARG2
connect gate57 ARG2 gate61 ARG2
connect gate51 ARG1 gate52 ARG3
connect gate63 ARG1 gate64 ARG3
connect gate75 ARG1 gate76 ARG3
connect gate107 ARG1 gate108 ARG3
connect gate87 ARG1 gate130 ARG2
connect gate119 ARG1 gate120 ARG3
connect gate99 ARG1 gate129 ARG2
connect gate129 ARG2 gate142 ARG2
connect gate151 ARG1 gate161 ARG2
connect gate161 ARG2 gate187 ARG3
connect gate163 ARG1 gate164 ARG3
connect gate175 ARG1 gate176 ARG3
connect gate40 ARG1 gate80 ARG2
connect gate80 ARG2 gate87 ARG2
connect gate52 ARG1 gate92 ARG2
connect gate92 ARG2 gate99 ARG2

connect gate106 ARG1 gate137 ARG3
connect gate137 ARG3 gate147 ARG3
connect gate147 ARG3 gate149 ARG2
connect gate64 ARG1 gate104 ARG2
connect gate104 ARG2 gate111 ARG2
connect gate118 ARG1 gate149 ARG3
connect gate149 ARG3 gate156 ARG3
connect gate76 ARG1 gate116 ARG2
connect gate116 ARG2 gate122 ARG2
connect gate88 ARG1 gate129 ARG3
connect gate129 ARG3 gate131 ARG2
connect gate150 ARG1 gate151 ARG3
connect gate162 ARG1 gate179 ARG3
connect gate174 ARG1 gate185 ARG3
connect gate7 ARG2 gate8 ARG2
connect gate89 ARG1 gate90 ARG3
connect gate105 ARG1 gate135 ARG2
connect gate135 ARG2 gate148 ARG2
connect gate41 ARG1 gate82 ARG3
connect gate82 ARG3 gate86 ARG3
connect gate86 ARG3 gate88 ARG2
connect gate117 ARG1 gate147 ARG2
connect gate147 ARG2 gate157 ARG2
connect gate53 ARG1 gate94 ARG3
connect gate94 ARG3 gate98 ARG3
connect gate98 ARG3 gate100 ARG2
connect gate129 ARG1 gate130 ARG3
connect gate65 ARG1 gate106 ARG3
connect gate106 ARG3 gate110 ARG3
connect gate110 ARG3 gate112 ARG2
connect gate77 ARG1 gate118 ARG3
connect gate118 ARG3 gate121 ARG3
connect gate161 ARG1 gate162 ARG3
connect gate173 ARG1 gate174 ARG3
connect gate5 ARG2 gate6 ARG2
connect gate78 ARG1 gate79 ARG3
connect gate104 ARG1 gate105 ARG3
connect gate116 ARG1 gate117 ARG3
connect gate30 ARG1 gate74 ARG3
connect gate74 ARG3 gate75 ARG3
connect gate75 ARG3 gate77 ARG2
connect gate77 ARG2 gate193 ARG2
connect gate128 ARG1 gate161 ARG3
connect gate42 ARG1 gate43 ARG3
connect gate54 ARG1 gate55 ARG3
connect gate66 ARG1 gate67 ARG3
connect gate160 ARG1 gate177 ARG3
connect gate172 ARG1 gate184 ARG3

connect gate3 ARG2 gate4 ARG2
connect gate170 ARG1 gate183 ARG3
connect gate67 ARG1 gate107 ARG2
connect gate107 ARG2 gate114 ARG2
connect gate1 ARG1 gate34 ARG2
connect gate79 ARG1 gate119 ARG2
connect gate119 ARG2 gate154 ARG2
connect gate154 ARG2 gate173 ARG2
connect gate173 ARG2 gate193 ARG3
connect gate29 ARG2 gate30 ARG2
connect gate103 ARG1 gate134 ARG3
connect gate134 ARG3 gate144 ARG3
connect gate144 ARG3 gate146 ARG2
connect gate115 ARG1 gate146 ARG3
connect gate146 ARG3 gate154 ARG3
connect gate127 ARG1 gate162 ARG2
connect gate139 ARG1 gate170 ARG2
connect gate31 ARG1 gate75 ARG2
connect gate75 ARG2 gate79 ARG2
connect gate43 ARG1 gate83 ARG2
connect gate83 ARG2 gate90 ARG2
connect gate55 ARG1 gate95 ARG2
connect gate95 ARG2 gate102 ARG2
connect gate171 ARG1 gate172 ARG3
connect gate1 ARG2 gate2 ARG2
connect gate2 ARG1 gate33 ARG3
connect gate33 ARG3 gate35 ARG2
connect gate35 ARG2 gate179 ARG2
connect gate56 ARG1 gate97 ARG3
connect gate97 ARG3 gate101 ARG3
connect gate101 ARG3 gate103 ARG2
connect gate68 ARG1 gate109 ARG3
connect gate109 ARG3 gate113 ARG3
connect gate113 ARG3 gate115 ARG2
connect gate31 ARG2 gate32 ARG2
connect gate102 ARG1 gate132 ARG2
connect gate132 ARG2 gate145 ARG2
connect gate114 ARG1 gate144 ARG2
connect gate144 ARG2 gate155 ARG2
connect gate126 ARG1 gate127 ARG3
connect gate138 ARG1 gate139 ARG3
connect gate20 ARG1 gate59 ARG3
connect gate59 ARG3 gate60 ARG3
connect gate60 ARG3 gate62 ARG2
connect gate62 ARG2 gate188 ARG2
connect gate32 ARG1 gate77 ARG3
connect gate77 ARG3 gate78 ARG3
connect gate78 ARG3 gate194 ARG2

```
connect gate44 ARG1 gate85 ARG3
connect gate85 ARG3 gate89 ARG3
connect gate89 ARG3 gate91 ARG2
export gate178 ARG1 carry output
export gate78 ARG2 c input
!********* End of command file ********************
```