```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity bit_serial_mul is
 port(x:inout std_logic_vector(3 downto 0);
      y:inout std_logic_vector(3 downto 0);
      s3,s2,s1,s0:inout std_logic;
      a:in std_logic;
      r7,r6,r5,r4,r3,r2,r1,r0: out std_logic;
      clk,reset:in std_logic
      );
 end bit_serial_mul;


 architecture beh of bit_serial_mul is
   component fulladder is
    port(a,b,c:in std_logic;
         sum,carry:out std_logic
         );
   end component;
   component diff is
     port(a,clk,reset:in std_logic;
          q:out std_logic
          );
   end component;
   component delay is
    port(a,clk,reset:inout std_logic;
       q:inout std_logic
       );
    end component;
    signal a1,a2,a3:std_logic;
    signal b:std_logic_vector(5 downto 0);
    signal c1,c2,c3,c4:std_logic;
    signal d:std_logic_vector(3 downto 0);
    signal e1,e2,e3,e4:std_logic;
    signal f1,f2,f3,f4:std_logic;
 begin
    u1:delay port map (a,clk,reset,a1);
    u1:delay port map (a,clk,reset,a2);
    u1:delay port map (a,clk,reset,a3);
    u1:delay port map (a,clk,reset,a4);
    for i in 0 to 3 loop
      u5:delay port map(y(i),clk,reset,b(0));
      u6:delay port map(y(0),clk,reset,b(1));
      u7:delay port map(y(1),clk,reset,b(2));
      u8:delay port map(y(2),clk,reset,b(3));
      u9:delay port map(y(3),clk,reset,b(4));
      u10:delay port map(y(4),clk,reset,b(5));
    end loop;
    for i in 0 to 3 loop
    u11: dff port map(x(i),a,0,d(0));
```

```
    u12: dff port map(x(i),a1,0,d(1));
    u13: dff port map(x(i),a2,0,d(2));
    u14: dff port map(x(i),a3,0,d(3));
    end loop;
    c1 <= d(0) and y;
    c2 <= d(1) and b(1);
    c3 <= d(2) and b(3);
    c4 <= d(3) and b(5);
    u15: delay port map (g1,clk,reset,f1);
    u16: delay port map (g2,clk,reset,f2);
    u17: delay port map (g3,clk,reset,f3);
    u18: delay port map (g4,clk,reset,f4);
    u19: delay port map (h1,clk,reset,e1);
    u20: delay port map (h2,clk,reset,e2);
    u21: delay port map (h3,clk,reset,e3);
    u22: delay port map (h4,clk,reset,e4);
    u23: fulladder port map (s0,c1,f1,h1,g1);
    u24: fulladder port map (e1,c2,f2,h2,g2);
    u25: fulladder port map (e2,c3,f3,h3,g3);
    u26: fulladder port map (e3,c4,f4,h4,g4);
    u27: fulladder port map (s1,c1,f1,h1,g1);
    u28: fulladder port map (e1,c2,f2,h2,g2);
    u29: fulladder port map (e2,c3,f3,h3,g3);
    u30: fulladder port map (e3,c4,f4,h4,g4);
    u31: fulladder port map (s2,c1,f1,h1,g1);
    u32: fulladder port map (e1,c2,f2,h2,g2);
    u33: fulladder port map (e2,c3,f3,h3,g3);
    u34: fulladder port map (e3,c4,f4,h4,g4);
    u35: fulladder port map (s3,c1,f1,h1,g1);
    u36: fulladder port map (e1,c2,f2,h2,g2);
    u37: fulladder port map (e2,c3,f3,h3,g3);
    u38: fulladder port map (e3,c4,f4,h4,g4);
end beh;
```

```
library IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
entity fulladder is
 port(a,b,c:in std_logic;
       sum,carry:out std_logic
       );
end fulladder;
architecture fulladder of fulladder is
begin
 sum<=a xor b xor c;
 carry<=(a and b) or (b and c) or (c and a);
end fulladder;

library ieee;
use ieee.std_logic_1164.all;

entity dff is
 port(
   d :inout std_logic;-- Data input
   clk :inout std_logic;-- Clock input
   reset :inout std_logic;-- Reset input
   q :out std_logic;-- Q input
 );
end entity;

architecture rtl of dff is
begin
 process(clk,reset) begin
 if (reset ='0') then
   q<='0';
  elsif (rising_edge(clk)) then
   q<= d;
  end if;
 end process;
end architecture;
```

```
library ieee;
use ieee.std_logic_1164.all;

entity delay is
 port (
   d :inout std_logic;-- Data input
   clk :inout std_logic;-- Clock input
   reset :inout std_logic;-- Reset input
   q :out std_logic;-- Q input
 );
end entity;


architecture rtl of delay is
begin
 process(clk,reset) begin
 if (reset ='0') then
   q<='0';
  elsif (falling_edge(clk)) then
   q<= d;
  end if;
 end process;
end architecture;
```

Ans8. OBSERVABILITY AND CONTROLLABILTY CODE

```
#include<stdio.h>
void main(){
int T[16][5], i, j, c[2], c_in[2], sum, o, o_op,temp;
for(i=0;i<16;i++)
for(j=0;j<5;j++)
T[i][j]=0;
for(i=1;i<16;i+=2)
T[i][3]=i;
for(i=2;i<16;i+=4)
T[i][2]=T[i+1][2]=1;
for(i=4;i<16;i+=8)
T[i][1]=T[i+1][1]=T[i+2][1]=T[i+3][1]=1;
for(i=8;i<16;i++)
T[i][0]=1;
printf("Enter Truth Table\n");
for(i=0;i<16;i++)
for(j=0;j<4;j++)
scanf("%d ",&T[i][j]);
scanf("Enter CCO and CC1 of input: %d %d", &c_in[0], &c_in[1]);
scanf("Enter observability of output: %d", &o_op);
temp=T[0][4];
i=0;
while(i<8)
if(T[i][4]==temp)
i++;
else
break;
k=8;
temp=T[8][4];
while(k<16)
if(T[k][4]==temp)
i++;
else
break;
c[0]=c[1]=-1;
if(i==8)
c[T[0][4]]=c_in[0]+1;
if(k==8)
c[T[8][4]]=c_in[1]+1;
if(c[0]==-1){
for(i=0;i<16;i++){
if(T[i][4]===0){
```

```
sum=0;
for(j=0;j<4;j++){
if(c[0]>sum+1 || c[0]==-1)
c[0]=sum+1;
}
}
}
}
if(c[0]==-1){
for(i=0;i<16;i++){
if(T[i][4]===0){
sum=0;
for(j=0;j<4;j++){
if(c[1]>sum+1 || c[1]==-1)
c[1]=sum+1;
}
}
}
}
o=-1;
for(i=0;i<8;i++){
sum=0;
if(T[i][4]!=T[i+8][4]){
for(j=0;j<4;j++)
sum+=T[i][k];
sum+=(1+o_op);
if(o>sum || o==-1)
o=sum;
}
}
printf("%d %d %d\n", c[0], c[1], o);
return;
}
```