CPE/EE 422/522 **EXAM II** **Name** KEY

Class _____

Fall Semester 2008

Work should be performed systematically and neatly with the final answer being underlined. This exam is closed book, closed notes, closed neighbor. Allowable items on desk include: exam, data manual, pencils, straight edge, and calculator. All other items must be removed from student's desk. Students have approximately 90 minutes (1 1/2 hours) to complete this exam. Best wishes!

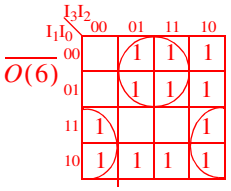
1. [15 points] On the 16L8 Programmable Array Logic, PAL, element shown in the figure what are the Boolean Equation for the point labeled **O6**?

$$\overline{O(6)} = I(2)\bullet\overline{I(1)} + \overline{I(2)}\bullet I(1) + I(3)\bullet\overline{I(1)} + I(1)\bullet\overline{I(0)}$$ ← Direct Relationship

$$= I(2)\bullet\overline{I(1)} + \overline{I(2)}\bullet I(1) + I(3)\bullet\overline{I(2)} + I(1)\bullet\overline{I(0)}$$ but any of these forms

$$= I(2)\bullet\overline{I(1)} + \overline{I(2)}\bullet I(1) + I(3)\bullet\overline{I(1)} + I(2)\bullet\overline{I(0)}$$ ok -- simplification not

$$= I(2)\bullet\overline{I(1)} + \overline{I(2)}\bullet I(1) + I(3)\bullet\overline{I(2)} + I(2)\bullet\overline{I(0)}$$ necessary in this prat



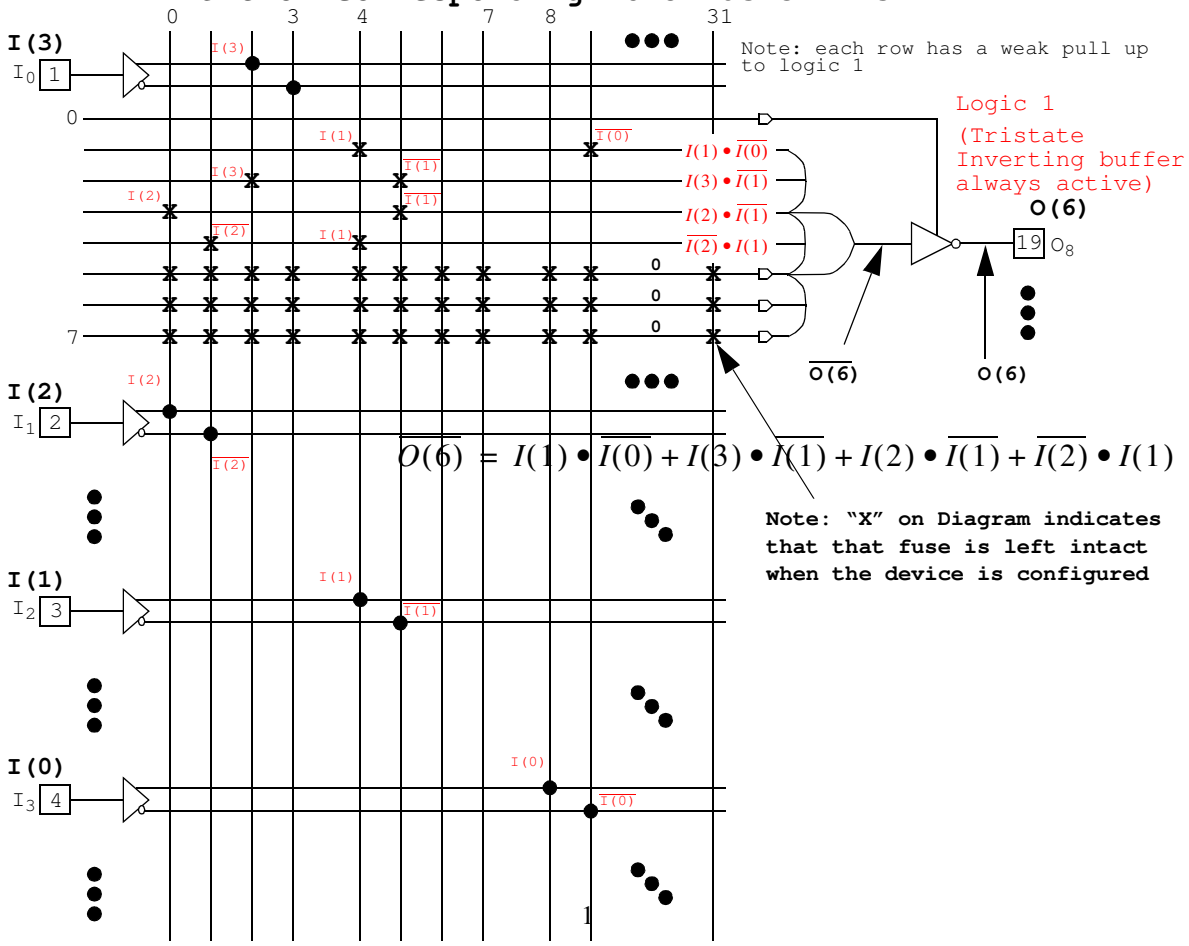
What is the simplified form of the Boolean equation for the output point labeled **O6**? (use **bold** labelings of I/O Pins)

$$O(6) = (\overline{I(1)} + I(0))(\overline{I(3)} + I(1))(\overline{I(2)} + I(1))(I(2) + \overline{I(1)}) = \overline{I(3)}\bullet\overline{I(2)}\bullet\overline{I(1)} + I(2)\bullet I(1)\bullet I(0)$$

What is the purpose of the inverting tri-state buffer that drives output **O6**? (i.e. in general why is it there?)

Major purpose of the inverting tri-state buffer is as follows. First it allows this output to have a common connection with one or more outputs allowing multiple output devices to share the same bus. It also allows provides the inversion function that allows the output to be complemented when it is activated.

Part of corresponding 16L8 LOGIC DIAGRAM



2. [10 points] What is the final value of signal **X** in the following VHDL model fragment?

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity TEST_Q1 is
                ⋮
end TEST_Q1;
architecture BEHAVIORAL of TEST_Q1 is

   function F_COUT(A,B: in STD_LOGIC)
      return integer is
      variable F : integer;
      begin
         if ((A or B) = '1') then
            F := 1;
         else      '1' or '0' = '1'
            F := 0;
         end if;
         return F;
   end F_COUT;

   function F_COUT(A,B,CIN: in STD_LOGIC)
      return STD_LOGIC is
      begin
         return (A and B) or (A and CIN) or (B and CIN);
   end F_COUT;

   function F_COUT(A,B: in integer)
      return integer is
      begin     1+1 = 2
         return (A + B);
   end F_COUT;

   signal X : integer;
                ⋮

   begin

   X <= F_COUT(1,F_COUT('1','0'));
                ⋮

end BEHAVIORAL;
```

function F_COUNT returns an integer 1

outer call of F_COUNT also has two input parameters but these are of integer type the F_COUNT function resulting in the lower F_COUNT function being called as F_COUT(1,1)

2

1

inner call of F_COUNT has two input parameters of STD_LOGIC type this matches the above F_COUNT function which returns an integer

X <= F_COUT(1,F_COUT('1','0'));

X = 2

3. [20 points] It is possible to apply the same concepts from Laboratory 3 to another display standard such as the old EGA standard. Using the EGA timing information that is provided in Figures 1-3 (very similar to that presented in Laboratory 4 for the VGA), determine the correct values for the conditional statement shown below. This statement is associated with the generation of the horizontal synchronization pulse.(You will want to refer to the copy of the VHDL model for the EGA SYNC module provided with this exam.)

```
   -- generate positive going horizontal synchronization pulse
if (H_COUNT <= ???) and (H_COUNT >= ???) then
       H_SYNC <= '1';
else
       H_SYNC <= '0';
end if;
```

Place Answer below.

a) Fill in the correct numbers (assuming the slowest possible system clock)

if (H_COUNT <= 724) and (H_COUNT >= 640) then

Note: horizonal sync pulse is $\frac{5.15us}{t_{pixel}} = \frac{5.15us}{6.1x10^{-8}} \approx 84$clocks long

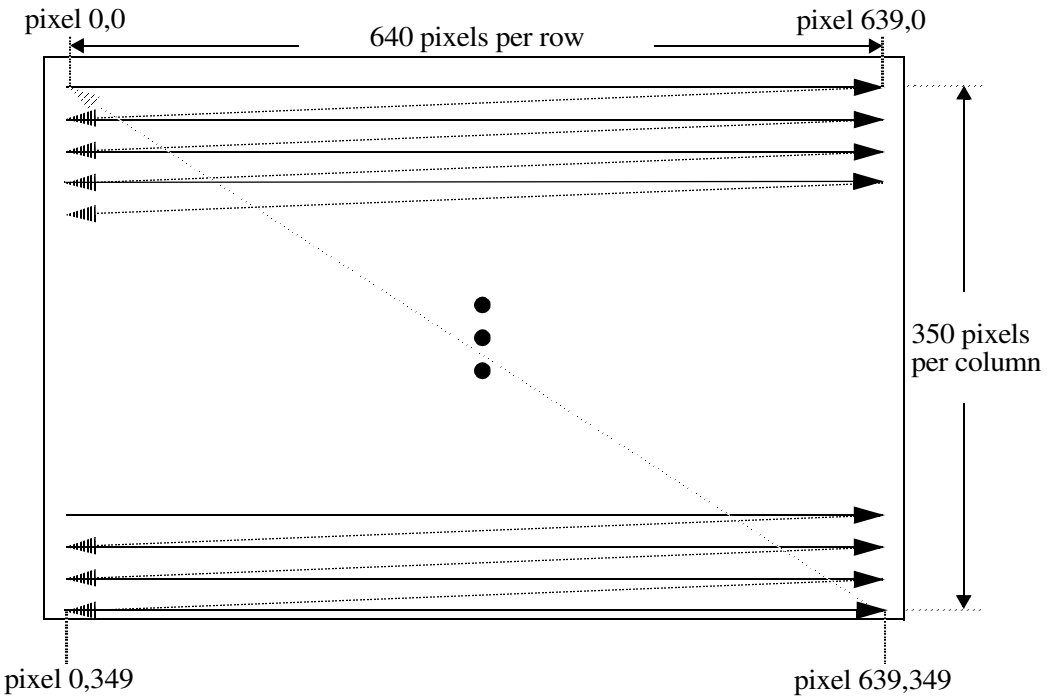


Figure 1: EGA Pixel Layout and Tracing for 640 x 350 Resolution

b) What is the slowest clock frequency needed to supply the pixel data?

$$t_{pixel} = \frac{\text{time to write pixels per row}}{\text{number of pixels per row}} = \frac{45.76us - 5.15us - 1.53us}{640} \approx 61ns$$

$$f_{pixel} = \frac{1}{t_{pixel}} \approx 16.38Mhz$$

minimum clock rate

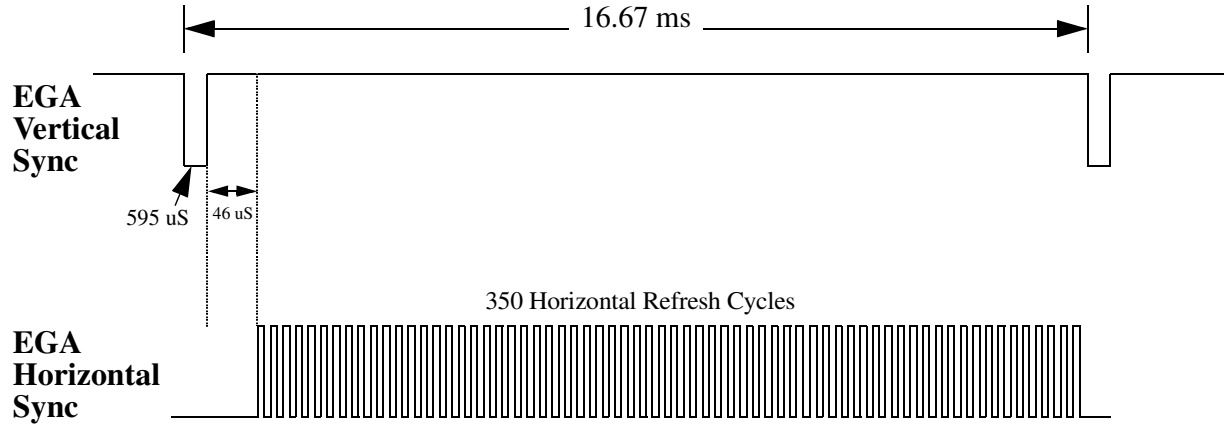


Figure 2: EGA Vertical Refresh Cycle

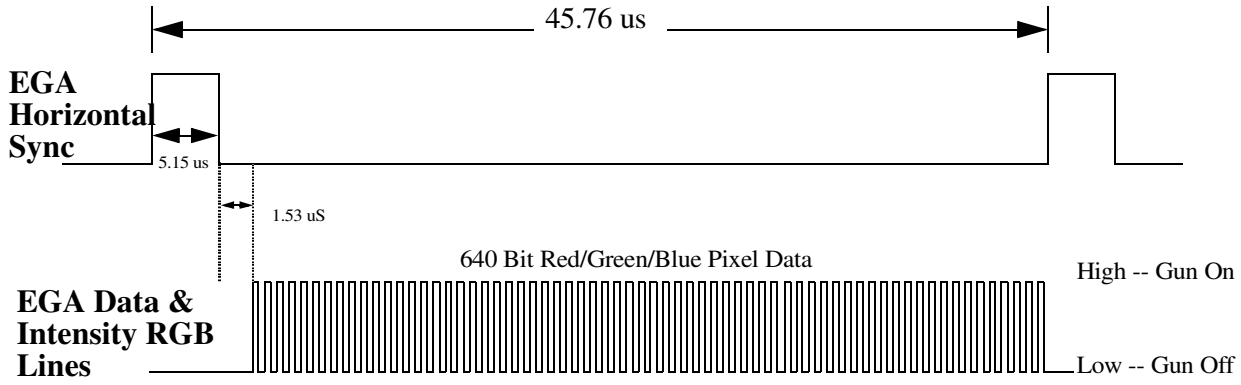


Figure 3: EGA Horizontal Refresh Cycle

4. [5 points] Describe what is meant by the term *operator overloading* in VHDL. Give one example of how or when this concept might be used.

Operator overloading in VHDL is a mechanism by which the function of the built in mathematical operators that are part of the VHDL language can be extended to provide different functions on different data types. The actual operation that is performed will be determined in this case by a careful matching of the operation type, and the types and numbers of the parameters. An example of when operation overloading could be useful would be to allow the addition of a STD_LOGIC_VECTOR with and integer to produce the same size STD_LOGIC_VECTOR. You have probably utilized this form of operator overloading when you performed the VGA laboratory. Many other examples of operator overloading exist.Operator overloading is implemented in VHDL by making the VHDL function name the same as the desired operator by enclosing the symbol for the operator in double quotes. The data types determine when the overloaded function will be activated.

5. [5 points] Describe how combinational logic implemented inside the logic blocks in most FPGAs. Describe two common approaches.

Combinational logic in FPGAs is not implemented using discrete gates but is usually implemented using a set of look up tables and or multiplexers. Sometimes PAL like programmable and/or logic arrays are also employed but this is usually for the smaller devices.

6. [30 points] You are to implement on a particular island-style FPGA device a two-bit version of the parallel adder described in the text whose functional diagram is shown in Figure 4 below.

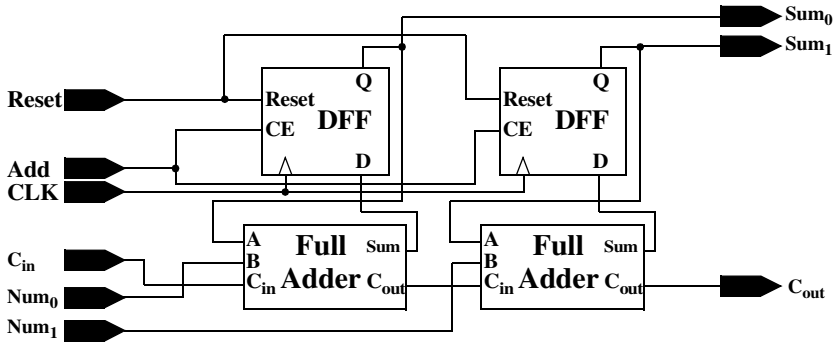


Figure 4: 2-bit Parallel Adder with Accumulator

The adder represents an accumulator style circuit (i.e. the current value stored gets added to the new value whenever the **Add** input is a logic one before the active clock edge. The synchronous **Reset** signal is used to clear the results). Figure 5 illustrates the targeted island-style FPGA architecture which is be used to implement the design.

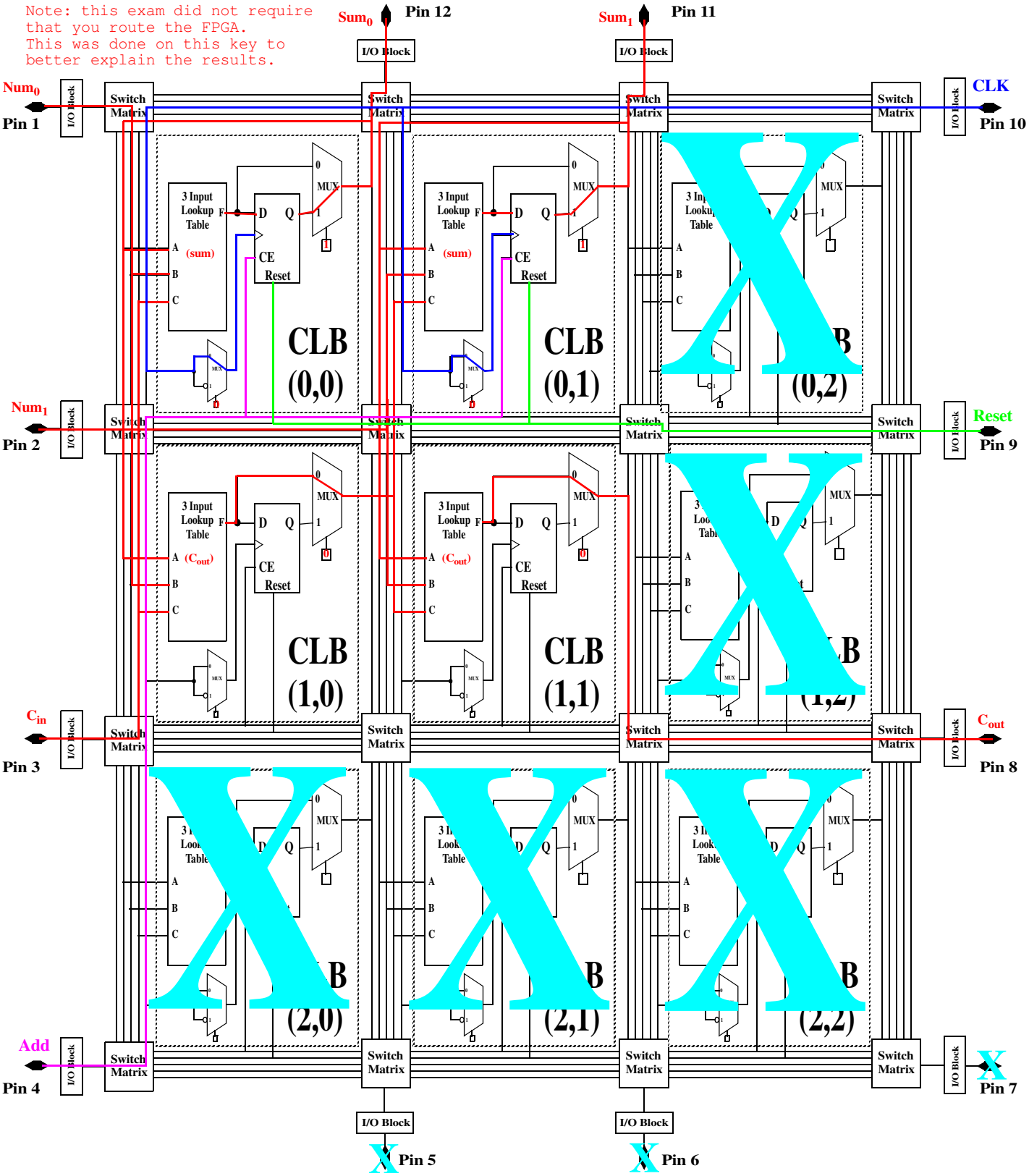


Figure 5: Floor-plan of Targeted island-style FPGA Device

6. continued.

Figure 5 shows an architecture that is similar to the Xilinx 3000 and 4000 devices. It contains CLBs, I/O blocks, and general routing resources as shown.

a. On Figure 5, label the inputs and outputs pins using the same naming convention that is used in Figure 4. Then cross out (put an X on) all CLBs that are not being used in this design. Also cross out (put an X on) any I/O pin that is not being used.

See colored markings on this key.

b. Specify below the contents of the 3 Input Lookup Table for each of the CLBs that is used in the design. Make sure to clearly label this table with the CLB number and label the table inputs.

A (A)	B (B)	C (C_{in})	F (Sum)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

CLB (0,0) and CLB (0,1)

A (A)	B (B)	C (C_{in})	F (C_{out})
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

CLB (1,0) and CLB (1,1)

c. What is the total number of D flip-flops used in your design?

2 D Flip-flops are used

What is the total number of 3-Input look-up tables?

4 Look-up tables are required (resulting in 4 CLBs being needed)

d. If you were able to expand this design, what bit size is the maximum that the FPGA could support? What is the limiting factor.

Examining the FPGA resources in terms of I/O Pins and CLBs (do not have to consider routing resources in this problem). Limiting factor will be what runs out first. Each additional bit requires two additional pins and two additional CLBs. We have three I/O pins and 5 CLBs that are unused. This means that we can add one more bit to our design before we run out of I/O pins. This means the maximum bit size that could be supported in this FPGA is 3 bits and the limiting factor is the I/O pins.

7. [15 points] For the VHDL model shown below, specify the times(s) at which each signal will change and the value to which it will change. List these changes in chronological order. List any delta delays as a separate entry.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity BUFS is
    port (A,B,C,D : in STD_LOGIC;
          F:    out STD_LOGIC);
end BUFS;

architecture DATAFLOW of BUFS is
    begin
       F <= not A when B = '1' else 'Z';
       F <= not C when D = '1' else 'Z';
end DATAFLOW;

library IEEE;
use IEEE.std_logic_1164.all;

entity TESTBENCH is
end TESTBENCH;

architecture DATAFLOW2 of TESTBENCH is
    signal A, B, C, D, F: STD_LOGIC := '0';
    component BUFS is
    port (A,B,C,D : in STD_LOGIC;
          F:    out STD_LOGIC);
    end component;
    begin
        A <= '0' after 0 ns, '1' after 10 ns, '0' after 40 ns;
        B <= '0' after 0 ns, '1' after 20 ns, '0' after 60 ns;
        C <= '0' after 0 ns, '1' after 30 ns;
        D <= '0' after 0 ns, '1' after 10 ns, '0' after 50 ns;

        C1:BUFS port map (A,B,C,D,F);
end DATAFLOW2;
```

Time	A	B	C	D	F
0 ns	0	0	0	0	U
0 ns + Δ	0	0	0	0	Z
10 ns	1	0	0	1	Z
10 ns + Δ	1	0	0	1	1
20 ns	1	1	0	1	1
20 ns + Δ	1	1	0	1	X
30 ns	1	1	1	1	X
30 ns + Δ	1	1	1	1	0
40 ns	0	1	1	1	0
40 ns + Δ	0	1	1	1	X
50 ns	0	1	1	0	X
50 ns + Δ	0	1	1	0	1
60 ns	0	0	1	0	1
60 ns + Δ	0	0	1	0	Z