1. (10 points) Modify the following VHDL model by adding a parameter that sets the number of flip-flops in the counter. Also, add an input which is loaded with an asynchronous load input signal which is active low. (Changes are included in red)

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity UPCOUNT is
  generic (N : integer);
  port (CLOCK, RESETN, E : in std_logic;
        LD : in std_logic;
        LD_INPUT : in std_logic_vector (N-1 downto 0);
        Q : out std_logic_vector (N-1 downto 0));
end UPCOUNT;

architecture BEHAVIOR of UPCOUNT is
  signal COUNT : std_logic_vector (N-1 downto 0);
begin
  process (CLOCK, RESETN, LD)
  begin
    if RESETN = '0' then
      COUNT <= (OTHERS => '0');
    if (LD = '0') then
      COUNT <= LD_INPUT;
    elsif (CLOCK'event and CLOCK = '1') then
      if E = '1' then
        COUNT <= COUNT + 1;
      else
        COUNT <= COUNT;
      end if;
    end if;
  end process
  Q <= COUNT;
end BEHAVIOR;
2. (5 points) What kind of hardware element will be inferred by a synthesis tool from the following model? Answer: a latch, since the behavior is level sensitive

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

entity WIDGET is
  Port (A, B : in SIGNED (0 to 2);
        CLK, RESET : in std_logic;
        Z : out SIGNED(0 to 2));
end WIDGET;

architecture EXAMPLE of WIDGET is
begin
  process (CLK, RESET)
  begin
    if (RESET = '1' then)
      Z <= '0';
    elsif (CLK = '1') then
      Z <= A nor B;
    end if;
  end process;
end EXAMPLE;
```

3. (8 points) Draw the transistor-level diagram of a CMOS three-input NAND gate.
Consider the following VHDL code:

```
-- Entity declaration
---------------------------------------
entity SCHED2 is
  port (A, B, C, D, E, F: in INTEGER;
       CLK : in BIT;
       W, X, Y: out INTEGER);
end SCHED2;
---------------------------------------
-- Architecture declaration
---------------------------------------
architecture HIGH_LEVEL of SCHED2 is
  signal Z: INTEGER;
begin
  X <= (A – B) * C * D;
  Y <= (A * B) + (E + F)/D;
  W <= (C + F) * B
end HIGH_LEVEL;
```

4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```

```
4. (15 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints.
   a. (5 points) Draw a data flow graph.
```
b. (5 points) Derive an ASAP schedule.

\[ \text{S1} \]
\[ A \quad B \quad C \quad D \quad A \quad B \quad D \quad E \quad F \quad C \quad F \quad B \]

\[ S2 \]

\[ S3 \]

\[ \text{c. (5 points) Derive an ALAP schedule.} \]

\[ \text{S1} \]
\[ A \quad B \quad C \quad D \quad A \quad B \quad + \quad C \quad F \quad B \]

\[ S2 \]

\[ S3 \]

\[ X \quad Y \quad W \]
5. (10 points) Derive a schedule using the freedom-directed method for the VHDL code above, using the following hardware constraint; all operations are done in an ALU module and there are two ALU modules available.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Earliest ASAP</th>
<th>Latest ALAP</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

For two ALUs,

- Step 1 \{1, 2, 4, 5, 8\} schedule 5, 1
- Step 2 \{2, 4, 8, 6\} schedule 6, 2
- Step 3 \{4, 8, 3\} schedule 4, 8
- Step 4 \{3, 7, 9\} schedule 7, 3
- Step 5 \{9\} schedule 9

6. (4 points) List the four types of paths that must be considered when doing timing analysis of sequential circuits.

- __________ inputs to inputs of storage elements________________________
- __________ inputs to outputs________________________________________
- __________ outputs of storage elements to inputs of storage element____
- __________ outputs of storage elements to outputs____________________
7. (10 points) For the data lifetime chart shown, use the left edge algorithm to obtain an efficient register allocation.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>S2</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>S3</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>S4</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>S5</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S6</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>S7</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>S8</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>S9</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S10</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

8. (1 point) A(n) ______ ASIC________________ is an integrated circuit produced for a specific application and produced in relatively small volumes.

9. (1 point) _______VHDL_____________ is an annoyingly strongly typed language.
10. (15 points) Create a VHDL entity named `mux4to1` that represents a 4-to-1 multiplexer which has an architecture which uses a case statement to represent the functionality of the multiplexer. Create a second entity and its accompanying architecture that represents a 16-to-1 decoder by using four instances of the `mux4to1` entity.

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

entity MUX4TO1 is
    port (DATA_IN : in std_logic_vector (3 downto 0);
          SEL : in std_logic_vector (1 downto 0);
          DATA_OUT : out std_logic);
end MUX4TO1;

architecture CASEMUX of MUX4TO1 is
begin
    process (DATA_IN, SEL)
    begin
        case SEL is
            when "00"    => DATA_OUT <= DATA_IN(0);
            when "01"    => DATA_OUT <= DATA_IN(1);
            when "10"    => DATA_OUT <= DATA_IN(2);
            when "11"    => DATA_OUT <= DATA_IN(3);
            when others => DATA_OUT <= 'X';
        end case;
    end process;
end CASEMUX;

library ieee;
use ieee.std_logic_1164.all;
use work.all;

entity MUX16TO1 is
    port (DATA_IN  : in std_logic_vector (15 downto 0);
          SEL      : in std_logic_vector (3 downto 0);
          DATA_OUT : out std_logic);
end MUX16TO1;

architecture STRUCT of MUX16TO1 is
    component MUX4TO1C
        port (DATA_IN  : in std_logic_vector (3 downto 0);
              SEL      : in std_logic_vector (1 downto 0);
              DATA_OUT : out std_logic);
    end component;
    for all : MUX4TO1C use entity MUX4TO1(CASEMUX);
    signal INTERNAL : std_logic_vector (3 downto 0);
begin
    U1 : MUX4TO1C port map (DATA_IN(3 downto 0), SEL(1 downto 0), INTERNAL(3));
    U2 : MUX4TO1C port map (DATA_IN(7 downto 4), SEL(1 downto 0), INTERNAL(2));
    U3 : MUX4TO1C port map (DATA_IN(11 downto 8), SEL(1 downto 0), INTERNAL(1));
    U4 : MUX4TO1C port map (DATA_IN(15 downto 12), SEL(1 downto 0), INTERNAL(0));
    U5 : MUX4TO1C port map (INTERNAL(3 downto 0), SEL(3 downto 2), DATA_OUT);
end STRUCT;
```
11. (10 points) Develop a VHDL entity and architecture for a D flip-flop with synchronous reset. Use a generic to represent TPCQ, the time it takes for a change to appear on Q after C undergoes a positive transition.

library ieee;
use ieee.std_logic_1164.all;

entity DFF is
  generic (TPCQ : time);
  port (D : in std_logic;
       C : in std_logic;
       RESET : in std_logic;
       Q, QB : out std_logic);
end DFF;

architecture BEHAVE of DFF is
begin
  process (C)
  begin
    if (C'event and C = '1') then
      if (RESET = '1') then
        Q <= '0' after TPCQ;
        QB <= '1' after TPCQ;
      else
        Q <= D after TPCQ;
        QB <= not D after TPCQ;
      end if;
    end if;
  end process;
end BEHAVE;

12. (6 points) If the NRE costs for FPGA and CBIC circuits are $25,000 and $166,000, respectively, and the cost of individual parts for FPGA and CBIC circuits are $20 and $6, respectively, what is the break-even manufacturing volume for these two types of circuits?

Let x be the number of parts. Then,

$25,000 + $20X = $166,000 + $6x
14x = 141,000
x = 10071

13. (2 points) ____Standard cells____ are primitives that are all the same height and varying widths.

14. (3 points) The three primary types of design units are ___entities___, __configurations____, and ___p: