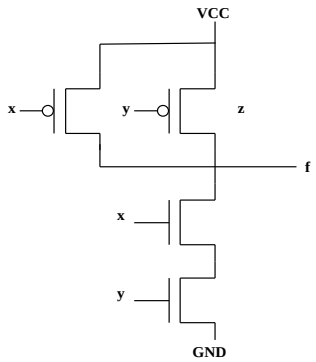
The University of Alabama in Huntsville
ECE Department
CPE 526 01
Final Exam Solution
Fall 2007

1. (5 points) Draw the transistor-level diagram of a two-input CMOS NAND gate.



2. (10 points) Write a VHDL entity (3 points) and architecture (7 points) of a D latch with the generics, TPDQ, which reflects the time for a change on D to appear at the outputs and TGDQ, which reflects the time for a change on the gate input, D, to appear at the outputs.

```
entity DLATCH is
  generic (TPDQ, TPGQ : time);
  port (D, G : in std_logic;
        Q, QB : out std_logic);
end DLATCH;

architecture DLATCH of DLATCH is
  signal QTEMP : std_logic;
begin
  process (D, G)
  begin
    if (G'event) then
      if (G = '1') then
        QTEMP <= D after TPGQ;
      end if;
    elsif (D'event) then
      if (G = '1') then
         QTEMP <= D after TPDQ;
      end if;
    end if;
  end process;
  Q <= QTEMP;
  QB <= not QTEMP;
end DLATCH;
```

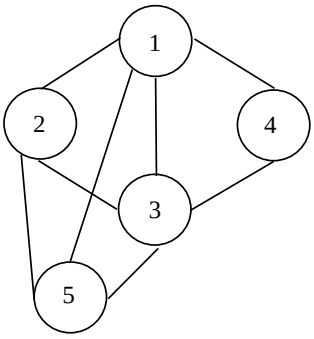
3. (1 point) _Scheduling_ consists of assigning each operation to a control step.

4. (1 point) The ___utilization_ of a component is the ratio of the busy time for the component to the execution time for the process.

5. (5 points) If the NRE costs for FPGA and ASIC circuits are $25,000 and $575,000, respectively, and the cost of individual parts for FPGA and ASIC circuits are $22 and $7, respectively, what is the break-even manufacturing volume for these two types of circuits?

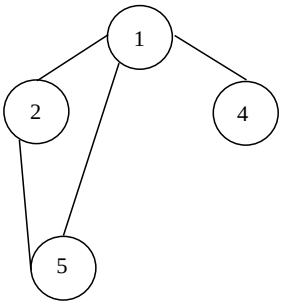
   Let x be the volume
   $25,000 + ($22x) = $575,000 + ($7x)
   15x = 550,000
   x = 36,667

6. (10 points) Consider the dataflow graph shown below. As part of the cluster partitioning algorithm, determine which two nodes should be merged and show the dataflow graph that results from the merger.



	Common Neighbors
(1, 2)	2
(1, 3)	3
(1, 4)	1
(1, 5)	2
(2, 3)	2
(2, 5)	2
(3, 4)	1
(3, 5)	2

So, 1 and 3 have the most common neighbors, merge them.



7. (4 points) List the four types of paths that must be considered when doing timing analysis of sequential circuits.

__primary inputs to primary outputs_____

__primary inputs to inputs of storage elements_____

__outputs of storage elements to primary outputs_____

__outputs of storage elements to inputs of storage elements____

8. (1 point) _Synthesis_ is the process of transforming a behavioral description into a structural gate-level circuit.

9. (1 point) __Routing__ is the process of making the connections between standard cells.

10. (10 points) For the data lifetime chart shown, use the left edge algorithm to obtain an efficient register allocation.

	A	B	C	D	E	F	G	H	I	J	K	L
S1				X			X			X		
S2	X			X			X		X	X		
S3	X	X		X		X			X	X	X	
S4	X	X		X		X		X	X		X	
S5	X	X			X	X		X			X	X
S6		X	X		X	X		X				X
S7			X									X

Sorting by earliest start and then by earliest end times,

	G	J	D	I	A	K	B	F	H	E	L	C
S1	X	X	X									
S2	X	X	X	X	X							
S3		X	X	X	X	X	X	X				
S4			X	X	X	X	X	X	X			
S5					X	X	X	X	X	X	X	
S6							X	X	X	X	X	X
S7											X	X

Doing the allocation

	R1	R2	R3	R4	R5	R6	R7
S1	G	J	D				
S2	G	J	D	I	A		
S3	K	J	D	I	A	B	F
S4	K	H	D	I	A	B	F
S5	K	H	E	L	A	B	F
S6	C	H	E	L		B	F
S7	C			L			

11. (6 points) Translate the following VHDL use (a) block statement(s) instead of a process:

```
process (S1, S4, DI, S3, Q)
  begin
    if (S1 = '1' and S4 = '1') then
      Q <= DI after FFDEL;
    elsif (S1 = '0' and S4 = '0') then
      Q <="00000000" after FFDEL;
    end if;
    if (S3 = '1') then
      DO <= Q after BUFDEL;
    else
      DO <= "ZZZZZZZZ" after BUFDEL;
    end if;
```

```
  B1 : block
  begin
    Q <= DI after FFDEL when (S1 = '1' and S4 = '1') else
         "00000000" after FFDEL when (S1 = '0' and S4 = '0');
    DO <= Q after BUFDEL when (S3 = '1') else
          "ZZZZZZZZ" after BUFDEL;
  end block B1;
```

12. (9 points) (a) (5 points) Write a single VHDL model which represents an AND gate with an arbitrary number of inputs, N. (b) (4 points) Use that model as a component in an entity that represents a four input AND gate with inputs a, b, c, d and output f

```
entity N_AND is
  generic (N : integer);
  port (I : in std_logic_vector(N-1 downto 0);
        O : out std_logic);
end N_AND;

architecture N_AND of N_AND is
begin
  process(I)
    variable TEMP : std_logic := '1';
  begin
    for j in I'range loop
      TEMP := TEMP and I(j);
    end loop;
    O <= TEMP;
  end process;
end N_AND;

entity UPPER is
end UPPER;

architecture UPPER of UPPER is
  component N_AND is
    generic (N : integer);
    port (I : in std_logic_vector(N-1 downto 0);
          O : out std_logic);
  end component;
  signal a, b, c, d, f : std_logic;
begin
  U1 : N_AND generic map (N => 4)
             port map (I(0) => a,
                       I(1) => b,
                       I(2) => c,
                       I(3) => d,
                       O => f);
end UPPER;
```

13. (12 points) A sequential network with one input and one output is used to stretch the first two bits of a 4-bit sequence as follows:

Input	Output
00dd	0000
01dd	0011
10dd	1100
11dd	1111

After every four bits, the network resets. Model this network in VHDL as a Moore state machine. a.(3 points) Write an entity. b. (9 points) Write an architecture.

```
entity STRETCH is
  port (X, CLK, RST : in std_logic;
        Z : out std_logic);
end STRETCH;
```

```
architecture STRETCH of STRETCH is
  type STATE is (S0, S1, S2, S3, S4, S5, S6, S7, S8);
  signal CURRENT_STATE, NEXT_STATE : STATE;
  signal TEMP : std_logic;
begin
  process(X, CURRENT_STATE)
  begin
    case CURRENT_STATE is
      when S0 => if (X = '0') then       -- initial state
                   NEXT_STATE <= S1;     -- pick state to go to based on the
                 else                    -- first input
                   NEXT_STATE <= S2;
                 end if;
       when S1 => TEMP <= X;             -- have received a '0', store next
                  NEXT_STATE <= S3       -- input, output a '0', go to the
                                         -- state where we output another '0'
      when S2 => TEMP <= X;              -- have received a '1', store next
                 NEXT_STATE <= S4;       -- input, output a '1', go to the
                                         -- state where we output another '1'
      when S3 => if (TEMP <= '0') then   -- output a '0', pick state to go to
                   NEXT_STATE <= S5;     -- based on the value of TEMP
                 else NEXT_STATE <= S6;
                 end if;
      when S4 => if (TEMP <= '0') then   -- output a '1', pick state to go to
                    NEXT_STATE <= S5;    -- based on the value of TEMP
                 else NEXT_STATE <= S6;
                 end if;
      when S5 => NEXT_STATE <= S7;       -- output first '0' for second input
      when S6 => NEXT_STATE <= S8;       -- output first '1' for second input
      when S7 => if (X = '0') then       -- output second '0', pick state to
                   NEXT_STATE <= S1;     -- go to based on input (now first
                 else                    -- bit of next four)
                   NEXT_STATE <= S2;
                 end if;
      when S8 => if (X = '0') then       -- output second '1', pick state to
                   NEXT_STATE <= S1;     -- go to based on input (now first
                 Else                    -- bit of next four)
                   NEXT_STATE <= S2;
                 end if;
    end case;
  end process;
  process(CLK, RST)
  begin
    if (RST = '1')
      then CURRENT_STATE <= S0;
    elsif (CLK'event and CLK = '1') then
      CURRENT_STATE <= NEXT_STATE;
    end if;
  end process;
  process(CURRENT_STATE)
  begin
    case CURRENT_STATE is
      when S0|S1|S3|S5|S7 => Z <= '0';
      when S2|S4|S6|S8 => Z <= '1';
    end case;
  end process;
end STRETCH;
```

Consider the following VHDL code:

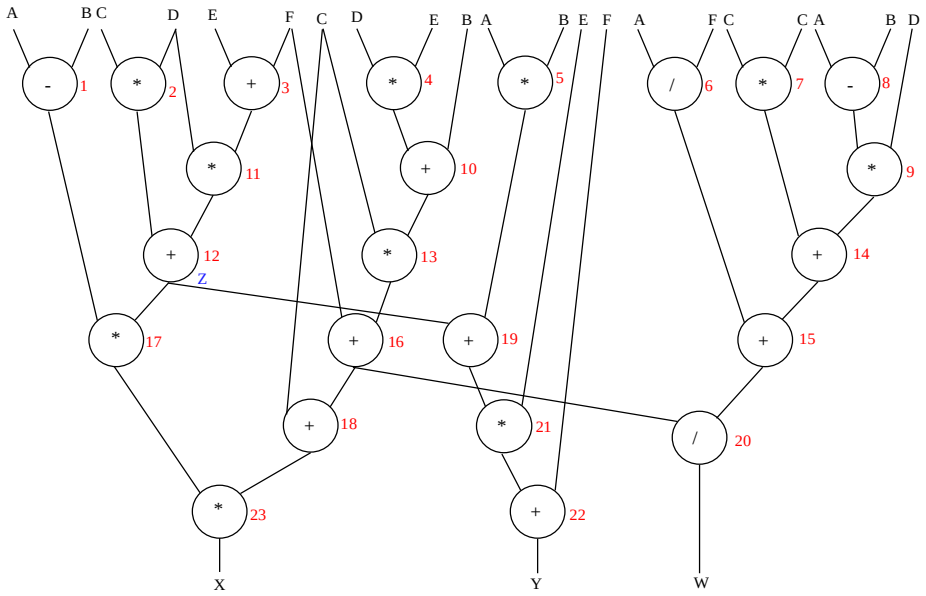
```
---------------------------------------
-- Entity declaration
---------------------------------------

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 V, Z: INTEGER;
begin
  X <= ((A - B) * Z) * (C + V);
  Y <= ((A * B) + Z) * E + F;
  Z <= (C * D) + D * (E + F);
  W <= (A/F + C*C + D* (A - B))/V;
  V <= (C * (B + D*E)) + F;
end HIGH_LEVEL;
```

16. (15 points) The following task refers to the VHDL code above. Assume that all operations are done in an ALU module and there are two ALU modules available. Derive a list schedule using the critical path priority metric for the operations.

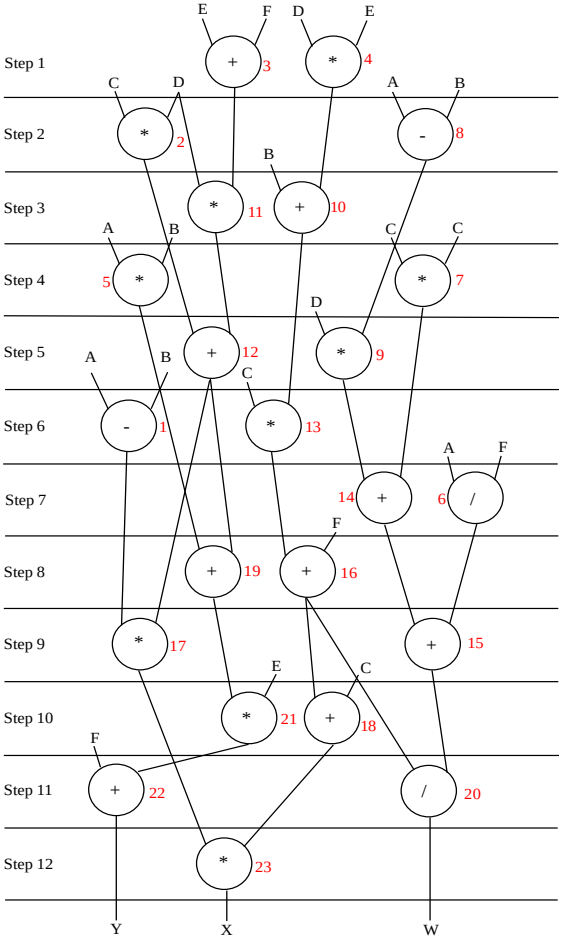


Node	Distance
1	3
2	5
3	6
4	6
5	4
6	3
7	4
8	5
9	4
10	5
11	5
12	4
13	4
14	3
15	2
16	3
17	2
18	2
19	3
20	1
21	2
22	1
23	1

List priority {3, 4, 2, 8, 10, 11, 5, 7, 9, 12, 13, 1, 6, 14, 16, 19, 15, 17, 18, 21, 20, 22, 23}

Step 1	**Ready {1, 2, 3, 4, 5, 6, 7, 8}**	**Schedule {3, 4}**
Step 2	**Ready {1, 2, 5, 6, 7, 8, 10, 11}**	**Schedule {2, 8}**
Step 3	**Ready {1, 5, 6, 7, 9, 10, 11}**	**Schedule {10, 11}**

Step 4	**Ready {1, 5, 6, 7, 9, 12, 13}**	**Schedule {5, 7}**
Step 5	**Ready {1, 6, 9, 12, 13}**	**Schedule {9, 12}**
Step 6	**Ready {1, 6, 13, 14, 19}**	**Schedule {1, 13}**
Step 7	**Ready {6, 14, 16, 17, 19}**	**Schedule {6, 14}**
Step 8	**Ready {15, 16, 17, 19}**	**Schedule {16, 19}**
Step 9	**Ready {15, 17, 18, 21}**	**Schedule {15, 17}**
Step 10	**Ready {18, 20, 21}**	**Schedule {18, 21}**
Step 11	**Ready {20, 22, 23}**	**Schedule {20, 22}**
Step 12	**Ready {23}**	**Schedule {23}**



17. (10 points) Write a VHDL model for a shift register module that includes a 16-bit shift register, a controller, and a 4-bit down counter. The shifter can shift a variable number of bits depending on a count provided to the shifter module. Inputs to the module are a number N (indicating a shift count) in the range 1 to 15, a 16-bit vector par_in, a clock and a start signal, St. When St = '1', N is loaded in the down counter, and par_in is loaded into the shift register. Then the shift register does a left shift N times and the controller returns to the start state. Assume that St is only '1' for one clock time. All operations are synchronous on the falling edge of the clock.

```
entity SHIFT is
  port (PAR_IN : in std_logic_vector(15 downto 0);
          CLK, ST : std_logic;
        N : in integer range 1 to 15;
        PAR_OUT : out std_logic_vector(15 downto 0));
end SHIFT;
```

```
architecture SHIFT of SHIFT is
begin
  process(CLK)
    variable TEMP : std_logic_vector(15 downto 0);
    variable COUNT : integer range 1 to 15;
    variable READY : std_logic;
  begin
    if (CLK'event and CLK = '0') then
      if (ST = '1' and READY = '1') then
        COUNT := N;
        TEMP := PAR_IN;
        READY := '0';
      else
        if (COUNT > 0) then
          COUNT := COUNT - 1;
          TEMP := TEMP(14 downto 0) & '0';
        else
          READY := '1';
        end if;
      end if;
    end if;
    PAR_OUT <= TEMP;
  end process;
end SHIFT;
```

18. (10 points) A Mealy sequence detector detects a sequence of four consecutive 1 inputs. The detector has a single binary input, X, and a single binary output, Z. Signal Z should be logic 1 if and only if the last four inputs were all logic 1. Here is an example input-output sequence:

    X 01011111101101011110
    Z 00000011110000000010

    Model this detector in VHDL.

```
entity FOUR_ONES is
  port (CLK, RST, X : in std_logic;
        Z : out std_logic);
end FOUR_ONES;

architecture FOUR_ONES of FOUR_ONES is
begin
  process(CLK, RST, X)
    variable TEMP : std_logic_vector(3 downto 0);
  begin
    if (RST = '1') then
      TEMP := "0000";
    elsif (CLK'event and CLK = '1') then
      TEMP := TEMP(2 downto 0) & X;
      if (TEMP = "1111") then
        Z <= '1';
      else
        Z <= '0';
      end if;
    end if; --RST = '1'
  end process;
end FOUR_ONES;
```