The University of Alabama in Huntsville
ECE Department
CPE 526 01
Final Exam Solution
Spring 2004

1. (15 points) An old Thunderbird car has three left and three right tail lights, which flash in unique patterns to indicate left and right turns.

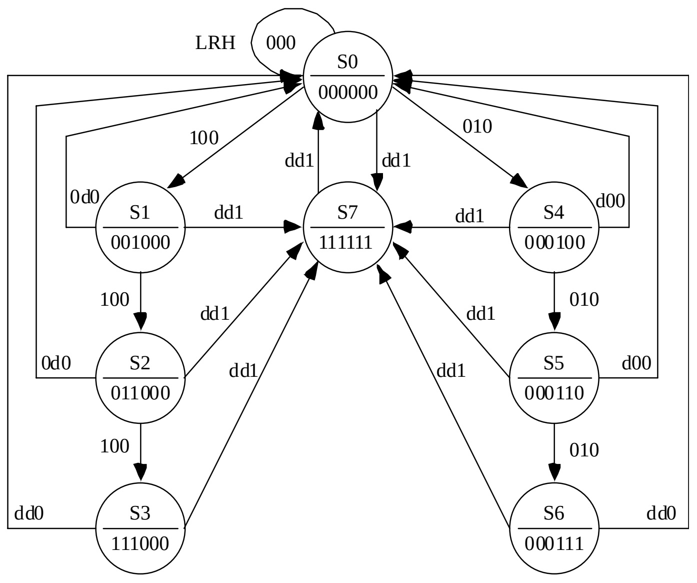
Left-turn pattern:

LC	LB	LA	RA	RB	RC
○	○	○	○	○	○
○	○	●	○	○	○
○	●	●	○	○	○
●	●	●	○	○	○

Right-turn pattern:

LC	LB	LA	RA	RB	RC
○	○	○	○	○	○
○	○	○	●	○	○
○	○	○	●	●	●
○	○	○	●	●	●

Design a Moore sequential network to control these lights using VHDL. The network has three inputs, LEFT, RIGHT, and HAZ. LEFT and RIGHT come from driver's turn-signal switch and cannot be 1 at the same time. As indicated above, when LEFT = 1, the lights flash in a pattern LA on, LA and LB on, LA, LB, and LC on and all off; then the sequence repeats. When RIGHT = 1, the light sequence is similar. IF a switch from LEFT to RIGHT (or vice versa) occurs in the middle of a flashing sequence, the network should immediately go to the IDLE state (lights off) and then start the new sequence. HAZ comes from the hazard switch, and when HAZ = 1, all six lights flash on and off in unison. HAZ takes precedence if LEFT or RIGHT is also on. Assume that a clock signal is available with a frequency equal to the desired flashing rate.



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

entity THUNDERBIRD is
  port (L, R, H, CLK : in std_logic;
        LC, LB, LA, RA, RB, RC : out std_logic);
end THUNDERBIRD;
```

```
architecture BEHAVE of THUNDERBIRD is
  type STATE_TYPE is (S0, S1, S2, S3, S4, S5, S6, S7);
  signal CURRENT_STATE, NEXT_STATE : STATE_TYPE;
begin
  process(CURRENT_STATE, L, R, H)
    variable INPUTS : std_logic_vector(2 downto 0);
  begin
    INPUTS := L&R&H;
    case CURRENT_STATE is
      when S0 => if (INPUTS = "000") then
                   NEXT_STATE <= S0;
                 elsif (INPUTS = "100") then
                   NEXT_STATE <= S1;
                 elsif (INPUTS = "010") then
                   NEXT_STATE <= S4;
                 else
                   NEXT_STATE <= S7;
                 end if;
      when S1 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "100") then
                   NEXT_STATE <= S2;
                 else
                   NEXT_STATE <= S0;
                 end if;
      when S2 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "100") then
                   NEXT_STATE <= S3;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S3|S6 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S4 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "010") then
                   NEXT_STATE <= S5;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S5 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "010") then
                   NEXT_STATE <= S6;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S7 => NEXT_STATE <= S0;
     end case;
   end process;
   process (CLK)
   begin
     if (CLK'event and CLK = '1') then
       CURRENT_STATE <= NEXT_STATE;
     end if;
   end process;
   process(CURRENT_STATE)
   begin
     LC <= '0'; LB <= '0'; LA <= '0';
     RA <= '0'; RB <= '0'; RC <= '0';
     case CURRENT_STATE is
       when S0 => null;
       when S1 => LA <= '1';
       when S2 => LA <= '1'; LB <= '1';
       when S3 => LA <= '1'; LB <= '1'; LC <= '1';
       when S4 => RA <= '1';
       when S5 => RA <= '1'; RB <= '1';
       when S6 => RA <= '1'; RB <= '1'; RC <= '1';
       when S7 => LA <= '1'; LB <= '1'; LC <= '1';
                  RA <= '1'; RB <= '1'; RC <= '1';
     end case;
   end process;
 end BEHAVE;
```

2. (1 point) ____Routing______ is the process of making the connections between standard cells.

3. (5 points) If the NRE costs for FPGA and CBIC circuits are \$21,000 and \$187,000, respectively, and the cost of individual parts for FPGA and CBIC circuits are \$15 and \$7, respectively, what is the break-even manufacturing volume for these two types of circuits?

Let x be the break-even volume.

\$21,000 +x * \$15 = \$187,000 + x * \$7
x * \$8 = \$166,000
x = 20,750

4. (5 points) What kind of hardware element will be inferred by a synthesis tool from the following model? Answer: A flip-flop (because of edge behavior) with synchronous reset.

```
   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 (CLK’event and CLK = ‘1’) then
         if (RESET = ‘1’) then
```

```
          Z <= ‘0’;
        else
          Z <= A nor B;
        end if;
      end if;
    end process;
  end EXAMPLE;
```

5. (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		
S3		X				X	X		X	X	X	
S4	X	X				X			X		X	X
S5	X				X	X		X	X		X	X
S6	X				X			X	X			X
S7					X				X			X

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

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

6. (1 point) Typically, a ___test bench__________ is developed to validate a VHDL behavioral model.

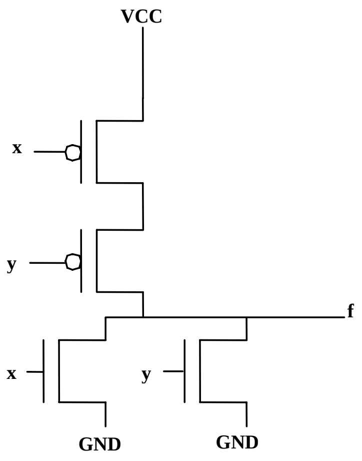
7. (1 point) A(n) ___SDF file____ provides the gate-level circuit with accurate timing backannotated from the layout.

8. (1 point) __Pragmas___ are inserted into VHDL models to give instructions to synthesis or other tools.

9. (15 points) Create a VHDL entity named `en_dec_328` that represents a 3-to-8 decoder with an active-low enable input which has an architecture which uses a case statement to represent the functionality of the decoder. Create a second entity and its accompanying architecture that represents a 4-to-16 decoder by using two instances of the `en_dec_328` entity.

```
entity EN_DEC_328 is
  port (EN :  in std_logic;
        I  : in std_logic_vector (2 downto 0);
        O  :  out std_logic_vector (7 downto 0));
end EN_DEC_328;

architecture BEHAV of EN_DEC_3TO8 is
begin
  process(EN, I)
  begin
    case EN is
      when '0' =>
        case I is
          when "000" => O <= "00000001";
          when "001" => O <= "00000010";
          when "010" => O <= "00000100";
          when "011" => O <= "00001000";
          when "100" => O <= "00010000";
          when "101" => O <= "00100000";
          when "110" => O <= "01000000";
          when "111" => O <= "10000000";
          when others => O <= "00000000";
        end case;
      when others =>
        O <= "00000000";
    end case;
  end process;
end BEHAV;


library ieee;
use ieee.std_logic_1164.all;

entity DEC_4TO16 is
  port (I : in std_logic_vector(3 downto 0);
        O : out std_logic_vector(15 downto 0));
end DEC_4TO16;

architecture STRUCT of DEC_4TO16 is
  signal I3BAR : std_logic;
  component EN_DEC_3TO8C
    port (EN : in std_logic;
          I : in std_logic_vector(2 downto 0);
          O : out std_logic_vector(7 downto 0));
  end component;
  for all : EN_DEC_3TO8C use entity work.EN_DEC_3TO8(BEHAV);
begin
  I3BAR <= not I(3);
  U1 : EN_DEC_3TO8C
         port map (I(3), I(2 downto 0), O(7 downto 0));
  U2 : EN_DEC_3TO8C
         port map(I3BAR, I(2 downto 0), O(15 downto 8));
end;
```

10. (5 points) Draw the transistor-level diagram of a CMOS two-input NOR gate.



11. (15 points) Modify the following VHDL model to use block(s) instead of processes.

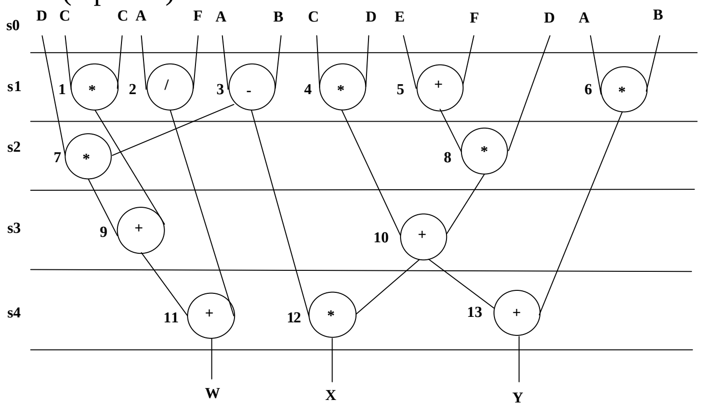
```
library ieee;
use ieee.std_logic_1164.all;

entity BUFF_REG is
 generic (STRB_DEL, EN_DEL, ODEL: TIME);
 port (DI: in std_logic_vector (1 to 8);
       DS1, NDS2, STRB : in std_logic;
       DO: out std_logic_vector (1 to 8));
end BUFF_REG;

architecture THREE_PROC of BUFF_REG is
  signal REG : std_logic_vector (1 to 8);
  signal ENBLD : std_logic;
begin
  PREG: process (STRB)
  begin
    if (STRB = ‘1’) then
      REG <= DI after STRB_DEL;
    end if;
  end process PREG;

  ENABLE : process (DS1, NDS2)
  begin
    ENBLD <= DS1 and not NDS2 after EN_DEL;
  end process ENABLE;

  OUTPUT : process (REG, ENBLD)
  begin
    if (ENBLD = ‘1’) then
      DO <= REG after ODEL;
    else
      DO <= “ZZZZZZZZ” after ODEL;
    end if;
  end process OUTPUT;
end THREE_PROC;
```

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) * Z;
  Y <= (A * B) + Z;
  Z <= (C * D) + D * (E + F);
  W <= A/F + C*C + D* (A – B);
end HIGH_LEVEL;
```

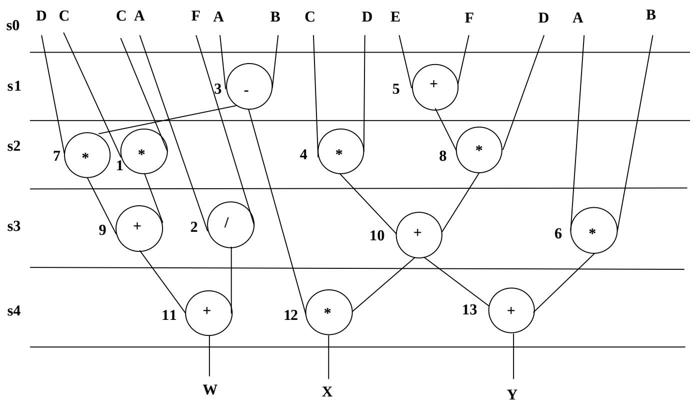
12. (14 points) The following tasks refer to the VHDL code above. Assume that there are no hardware constraints..

a. (7 points) Derive an ASAP schedule.



Ready	Schedule
{1, 2, 3, 4, ,5, 6}	{1, 2, 3, 4, 5, 6}
{7, 8}	{7, 8}
{9, 10}	{9, 10}
{11, 12, 13}	{11, 12, 13}

b. (7 points) Derive an ALAP schedule.

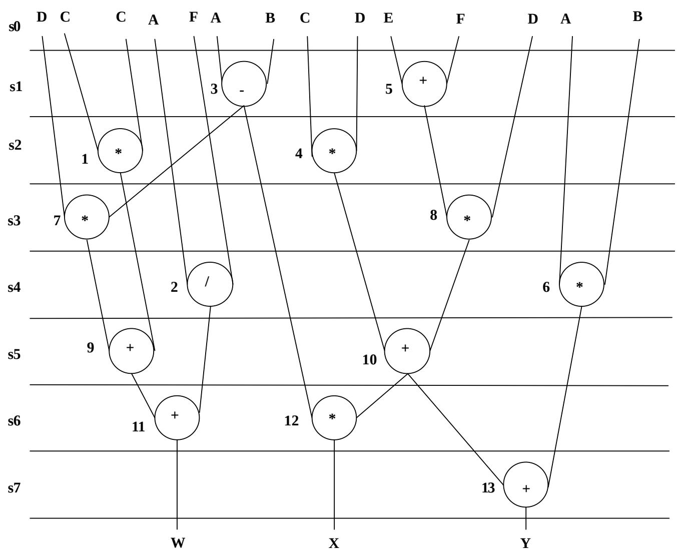


Ready	Schedule
{11, 12, 13}	{11, 12, 13}
{2, 6, 9, 10}	{2, 6, 9, 10}
{1, 4, 7, 8}	{1, 4, 7, 8}
{3, 5}	{3, 5}

13. (10 points) Derive a list schedule using the critical path priority metric 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.

Solution:

Op	Length
1	3
2	2
3	4
4	3
5	4
6	2
7	3
8	3
9	2
10	2
11	1
12	1
13	1



List {3, 5, 1, 4, 7, 8, 2, 6, 9, 10, 11, 12, 13}

Ready	Schedule
{1, 2, 3, 4, 5, 6}	{3, 5}
{1, 2, 4, 6, 7, 8}	{1, 4}
{2, 6, 7, 8}	{7, 8}
{2, 6, 9, 10}	{2, 6}
{9, 10}	{9, 10}
{11, 12, 13}	{11, 12}
{13}	{13}

14. (1 point) __Allocation, scheduling, compilation_________ is one algorithmic-level synthesis task.

15. (1 point) ______Communication_________ is the hardest problem.