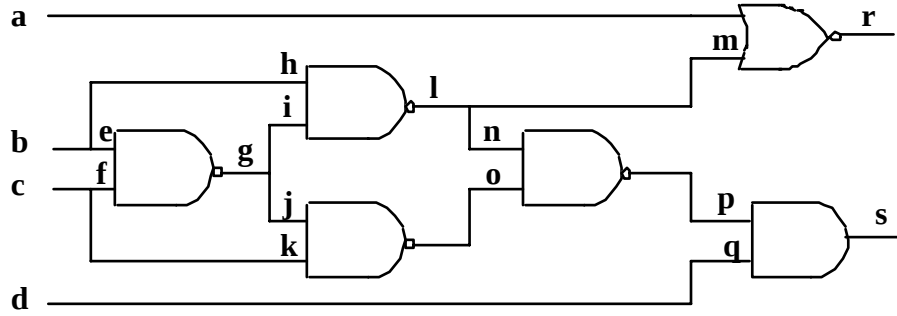


The University of Alabama in Huntsville
ECE Department
CPE/EE 422/522 01
Spring 2004
Final Exam Solution

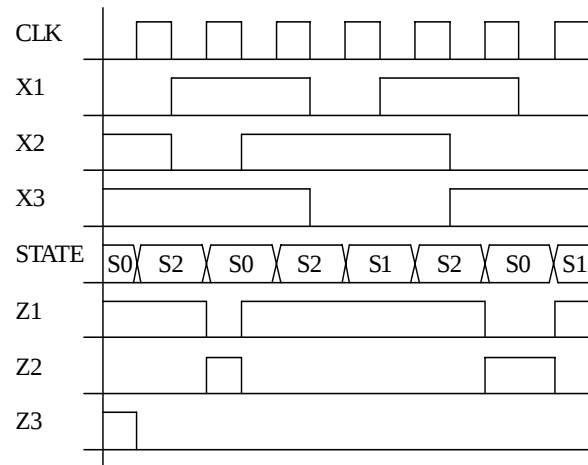
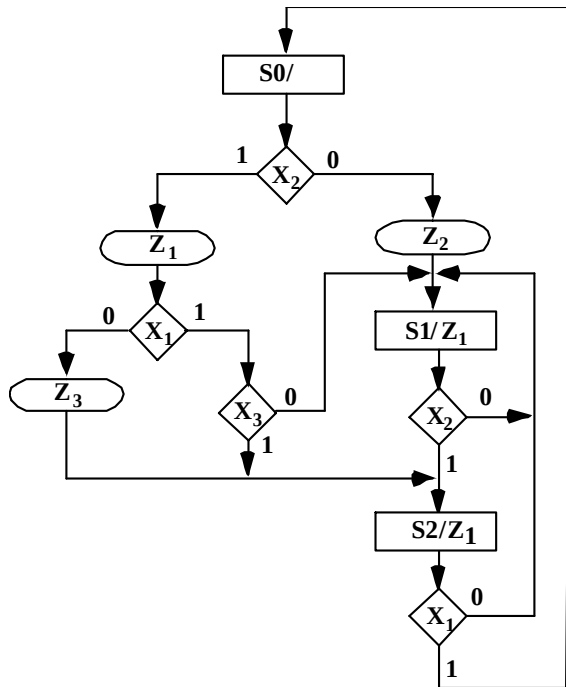
1. (10 points) For the following circuit, find all tests for the faults g s-a-1 and g s-a-0.



a	b	c	d	r	s	g s-a-1 r	g s-a-1 s	g s-a-0 r	g s-a-0 s
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	1	0	1	0	0-detect
0	1	0	0	1	0	1	0	0-detect	0
0	1	0	1	1	1	1	1	0-detect	0-detect
0	1	1	0	0	0	1-detect	0	0	0
0	1	1	1	0	0	1-detect	1-detect	0	0
1	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0
1	0	1	1	0	1	0	1	0	0-detect
1	1	0	0	0	0	0	0	0	0
1	1	0	1	0	1	0	1	0	0-detect
1	1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	1-detect	0	0

2. (2 points) Xilinx and Altera are the two most prominent providers of field programmable gate arrays (FPGAs).
3. (2 points) _Transport_ delay and __inertial_ delay are the two types of delay provided for in VHDL.
4. (15 points) For the following SM chart:

Draw a timing chart that shows the clock, the state (S0, S1 or S2), the inputs (X1, X2 and X3) and the outputs. The input sequence is X1 X2 X3 = 011, 101, 111, 010, 110, 101, 001. Assume that all state changes occur on the rising edge of the clock, and the inputs change on the falling edge of the clock.



5. (8 points) For the following state table, make a suitable state assignment.

Present State		Next State						Output Z_1Z_2			
State	$X_1X_2=$	00	01	10	11	$X_1X_2=$		00	01	10	11
S0		S3	S2	S1	S0			00	10	11	01
S1		S0	S1	S2	S3			10	10	11	11
S2		S3	S0	S1	S1			00	10	11	01
S3		S2	S2	S1	S0			00	00	01	01

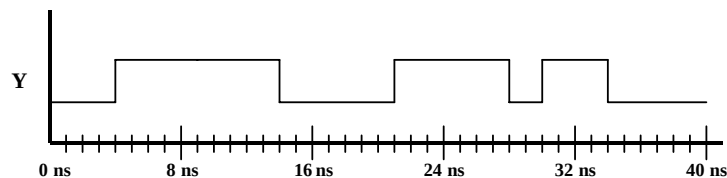
Guideline 1: (S0, S2), (S0, S3), (S0, S2, S3)

Guideline 2: (S0, S1, S2, S3), (S0, S1, S3), (S0, S1, S2)

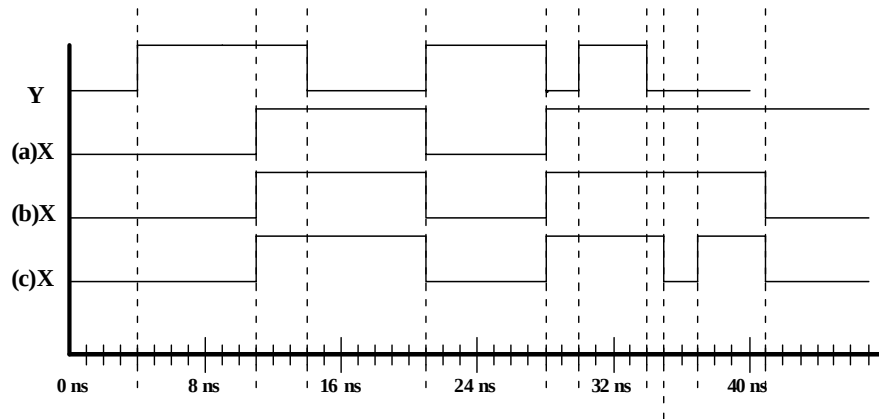
Guideline 3: (S0, S1, S2), (S0, S2, S3)

		Q2
	S0	S3
Q1	S2	S1

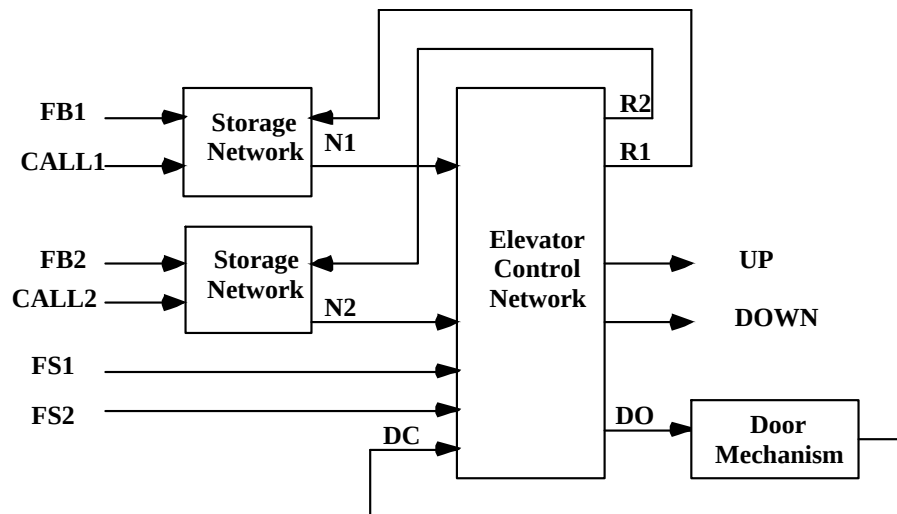
6. (9 points) Draw waveforms for the following signal assignments given the input waveform below. Be sure to clearly label the time when an event occurs. X and Y are both of type bit.

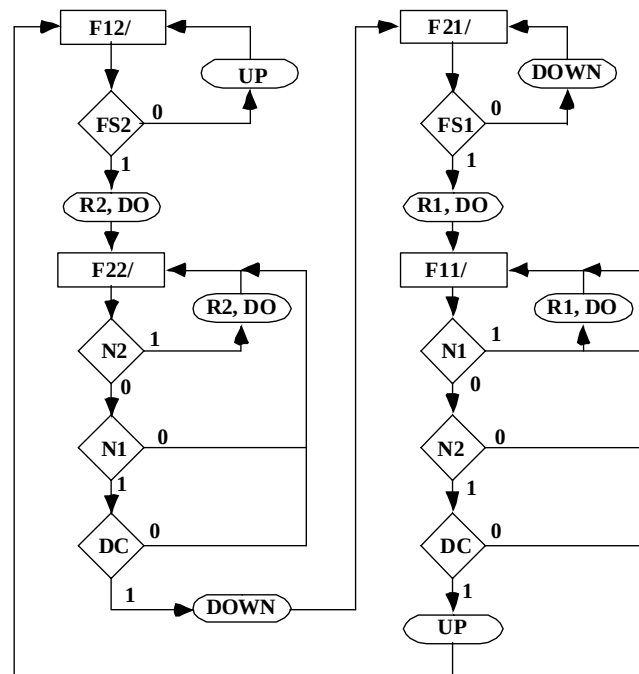


- `X <= Y` after 7 ns;
- `X <= reject 3 ns inertial Y` after 7 ns;
- `X <= transport Y` after 7 ns;



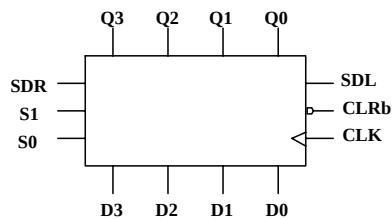
7. (10 points) The block diagram for an elevator controller for a building with two floors is shown below. The inputs FB1 and FB2 are floor buttons in the elevator. The inputs CALL1 and CALL2 are call buttons in the hall. The inputs FS1 and FS2 are floor switches that output a 1 when the elevator is at the first or second floor landing. Outputs UP and DOWN control the motor, and the elevator is stopped when $UP = DOWN = 0$. N1 and N2 are flip-flops that indicate when the elevator is needed on the first or second floor. R1 and R2 are signals that reset these flip-flops. $DO = 1$ causes the door to open, and $DC = 1$ indicates that the door is closed. Draw an SM chart for the elevator controller (four states).





8. (15 points) A description of a 74194 4-bit bidirectional shift register follows:

The CLRb input is asynchronous and active low and overrides all the other control inputs. All other state changes occur following the rising edge of the clock. If the control inputs $S1 = S0 = 1$, the register is loaded in parallel. If $S1 = 1$ and $S0 = 0$, the register is shifted right and SDR (serial data right) is shifted into Q3. If $S1 = 0$ and $S0 = 1$, the register is shifted left and SDL is shifted into Q0. If $S1 = S0 = 0$, no action occurs. Write a VHDL model for the shift register, including an entity (4 points) and an architecture (11 points).



(a)

```
entity SHIFTREG is
  port(DATA_IN: in BIT_VECTOR(3 downto 0);
        CLK, CLRb, S1, S0: in BIT; SDL, SDR: in BIT;
        Q: out BIT_VECTOR(3 downto 0));
end SHIFTREG;
```

(b)

```

architecture BEHAVE of SHIFTREG is
  signal QTEMP : BIT_VECTOR(3 downto 0);
begin
  process (CLK)
  begin
    if (CLRB = '0') then
      QTEMP <= "0000";
    elsif (CLK'event and CLK = '1') then
      if (S1 = '1' and S0 = '1') then
        QTEMP <= DATA_IN;
      elsif (S1='0' and S0='1') then
        QTEMP <= QTEMP(2 downto 0) & SDL;
      elsif (S1='1' and S0='0') then
        QTEMP <= SDR & QTEMP(3 downto 1);
      end if;
    end if;
  end process;
  Q <= QTEMP;
end BEHAVE;

```

9. (10 points) Consider the following fragment of VHDL.

```

type My4 is ('d', 'c', 'b', 'a');
type My4_vector is array(natural range <>) of My4;
function myresolve(s: My4_vector) return My4;
subtype My4R is myresolve My4;
...

signal R : My4R;

```

The resolution function **myresolve** realizes the signal resolution using the following table.

	'a'	'b'	'c'	'd'
'a'	'a'	'b'	'a'	'a'
'b'	'b'	'b'	'b'	'b'
'c'	'a'	'b'	'c'	'c'
'd'	'a'	'b'	'c'	'd'

Give values of the signal R in time interval from 0 ns to 20 ns assuming that it's driven by the following concurrent signal assignment statements and that these statements are executed once at 0 ns.

```

R <= transport 'a' after 3 ns, 'b' after 10 ns;
R <= transport 'c' after 2 ns, 'd' after 6 ns;
R <= transport 'b' after 1 ns, 'c' after 5 ns;

```

Time	S0	S1	S2	R
0 ns	d	d	d	d
1 ns	d	d	b	b
2 ns	d	c	b	b
3 ns	a	c	b	b
4 ns	a	c	b	b
5 ns	a	c	c	a
6 ns	a	d	c	a
7 ns	a	d	c	a
8 ns	a	d	c	a
9 ns	a	d	c	a
10 ns	b	d	c	b
11ns – 20 ns	b	d	c	b

10. (10 points) Write a VHDL function to compare two IEEE std_logic_vectors to see whether they are equal. Report an error if any bit in either vector is not '0', '1', or '-' (don't care), or if the lengths of the vector are not the same. The function call should pass only the vectors. The function should return TRUE if the vectors are equal, else FALSE. When comparing the vectors, consider that '0' = '-' and '1' = '-'. Make no assumptions about the index range of the two vectors.

```
function "=" (L, R : std_logic_vector) return boolean is
    variable EQUAL : boolean;
    alias LEFT : std_logic_vector(L'length-1 downto 0) is L;
    alias RIGHT : std_logic_vector(L'length-1 downto 0) is R;
begin
    assert L'length = R'length
    report "Vectors are not the same length"
    severity error;
    EQUAL := TRUE;
    for I in LEFT'range loop
        case LEFT(I) is
            when '0' => case RIGHT(I) is
                            when '0'|'-' => null;
                            when others => EQUAL := FALSE;
                        end case;
            when '1' => case RIGHT(I) is
                            when '1'|'-' => null;
                            when others => EQUAL := FALSE;
                        end case;
            when '-' => case RIGHT(I) is
                            when '0'|'1' => null;
                            when others => EQUAL := FALSE;
                        end case;
            when others => assert FALSE
                            report "First vector has invalid value"
                            severity error;
        end case;
        assert (RIGHT(I)='0' or RIGHT(I)='1' or RIGHT(I)='-')
        report "Second vector has invalid value"
        severity error;
    end loop;
    return EQUAL;
end "=";
```

11. (1 point) The term __static__ RAM means that once data is stored in the RAM, the data remains there until the power is turned off.
12. (1 point) ____Communication____ is the hardest problem.

13. (7 points) Draw the state diagram for the following state machine. Is it a Moore machine or a Mealy machine? Mealy

```

ENTITY state_machine IS
  PORT (sig_in ; IN BIT; clk : IN BIT;
        sig_out : OUT BIT);
END state_machine;

ARCHITECTURE state_machine OF state_machine IS
  TYPE state_type IS (a, b, c, d, e);
  SIGNAL state: state_type;
BEGIN
  PROCESS (clk)
  BEGIN
    IF (clk'event and clk = '1') THEN
      sig_out <= '0';
      state <= b;
      CASE state
        WHEN a =>
          IF sig_in = '0' THEN
            state <= a;
            sig_out <= '1';
          ELSE
            state <= d;
          END IF;
        WHEN b =>
          IF sig_in = '0' THEN
            state <= e;
          ELSE
            state <= c;
            sig_out <= '1';
          END IF;
        WHEN c =>
          sig_out <= '1';
          IF sig_in = '1' THEN
            state <= c;
          ELSE
            state <= b;
          END IF;
        WHEN d =>
          IF sig_in = '0' THEN
            state <= e;
            sig_out <= '1';
          END IF;
        WHEN e =>
          IF sig_in = '1' THEN
            state <= c;
          END IF;
      END CASE;
    END IF;
  END PROCESS;
END state_machine;

```

