SOLUTION SET

1) Write the VHDL code for a circuit that routes the maximal value of Three input signals (a,b,c) to the output (f). Write only the ARCHITECTURE portion of the code. (15p)

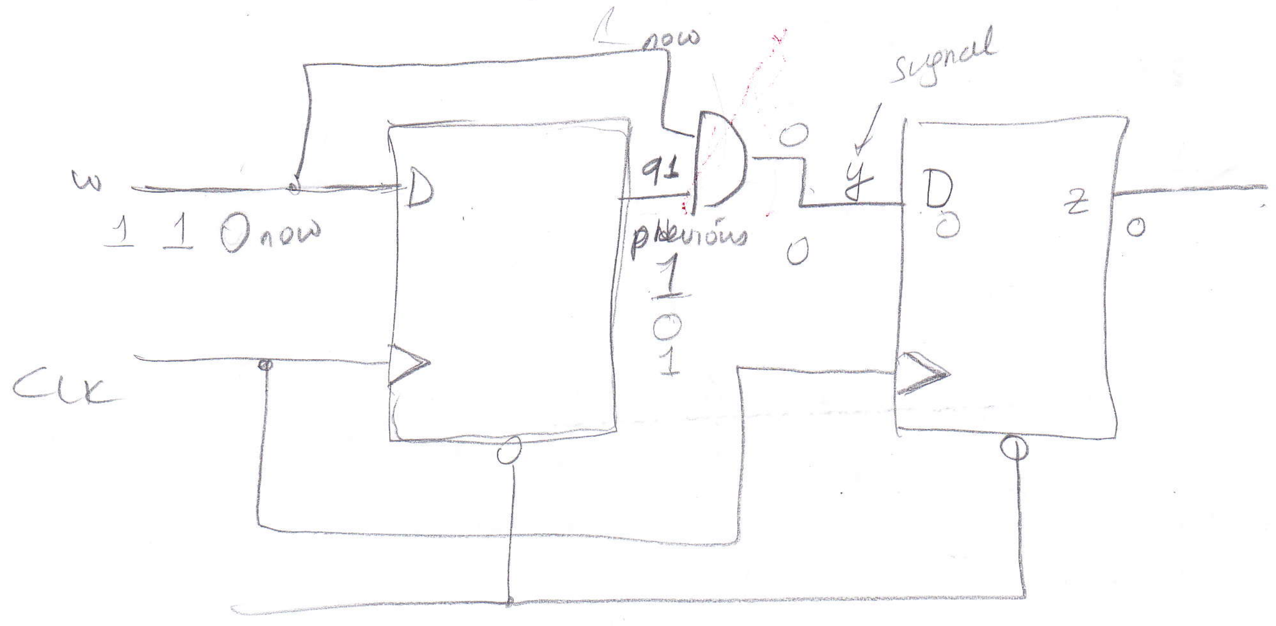
SOL:

```
ARCHITECTURE Behavior OF maximal IS
  BEGIN
  PROCESS (a,b,c)                (2)
    if (a>b) then
       if (a>c) then
           max <= a ;
       else
          max <= c   ;
        end if ;                  (13)
       else
         if (b>c) then
           map <= b   ;
          else
           max <= c;
        end if;
       endif ;
     end process;
   end behavior;
```

Fresnel–Kirchhoff diffraction of spherical waves by a circular aperture," J. Opt. Soc. Am. A **11**, 774–778 (1994).

15. In the case of the plane-wave results, recall that the incident field has already been scaled before taking the source point to infinity, so it is necessary to divide the potential given in Eq. (3.20), for example, by only $\exp(ik\mathbf{u}\cdot\mathbf{r}_O)$, andthis means that the phase factor now becomes $\exp(ik2\mathscr{L}_\infty)$. The same is true of Eq. (4.4), and these results are in keeping with Eqs. (5.1) and (5.3). For the focused field, $\exp(-ik|\mathbf{r}_O - \mathbf{r}_S|)/(4\pi|\mathbf{r}_O - \mathbf{r}_S|)$ is to be divided out for points well before focus, but its complex conjugate is to be used far beyond the focal plane; the phase is small near the focus, so there is nothing to be gained there in this way.

SOL (2) Since we should detect 2 consecutive 1's ~~and~~ in the clock signal and produce a 1 at the next clock cycle we need 2 FFs



when the previous and the current w value is 1 then we can declare that we have 2 1's coming after another.

So we need an AND circuit between the FFs.

10

Sol ② continued -

The VHDL code for the circuit, using Flip Flops as components;

```
use work.components.all;
ENTITY COUNTING1 IS
 PORT (Clock, Resetn, w  : IN - STD_LOGIC;
                 z       ; OUT STD-LOGIC);


ARCHITECTURE Structure OF COUNTING1 IS
  SIGNAL   y : STD_LOGIC;
  SIGNAL   Q1 : STD-LOGIC;
  flipflop1 : flipflop PORTMAP(w, Resetn, clock, Q1);
  flipflop2 : flipflop PORTMAP (y, Resetn, clock, z);
  y = Q1 AND W;
  END structure;
```

7.5

The D FLIP FLOP is used as component in the code:

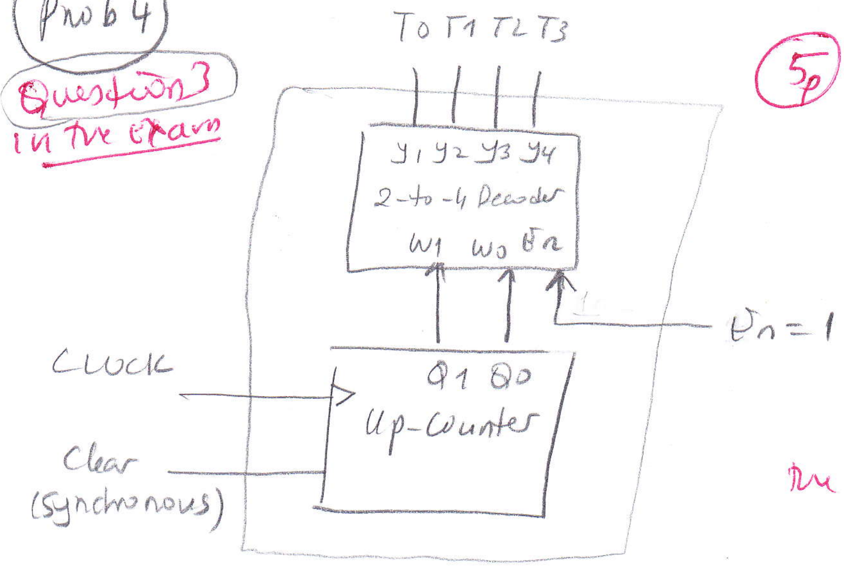
2.5

```
COMPONENT flipflop
 ENTITY flipflop IS
  PORT (D, Resetn, Clock : IN Std_logic;
        Q                ; OUT Std-logic);
  END flipflop;
  ARCHITECTURE Behavior OF flipflop IS
  BEGIN
   PROCESS (Resetn, Clock)
   BEGIN
    IF Resetn = '0' THEN   Q <= '0';
    ELSIF (Clock'EVENT and CLOCK = '1') THEN
     Q <= D;
    END IF;
   END PROCESS;
  END Behavior;
```

Yılmaz	AVCI	200514005	1	0	38	17	34,75	47,90
							31,58	43,53

Prob 4)

Question 3 in the exam



5p

a) This circuit produces a sequence of 1000 and shifts "1" in the sequence at each clock cycle: i.e. in the first clock cycle 1000, then 0100, then 0010, then 0001 and again with 1000. It clears the output when CLR=1.

a) Explain what the circuit given does.

b) Write the VHDL code implementing the circuit. Inputs are clear, clock, En. Outputs are T0, T1, T2, T3. Design the code such that up counter and 2-to-4 decoder are used as components in the code. First write the VHDL code for the components in a components package. Then use these components in the main code by a package implementation.

SOL: First write the VHDL codes for the componments in a package:

```
ENTITY upcounter IS
PORT ( CLOCK, Clear ; IN  Std-logic ;
       Q             ; OUT STD-LOGIC-VECTOR ( 1 DOWNTO0);
END upkounter ;
ARCHITECTURING Behavior of UPCOUNTOR IS
 BEGIN
  PROCESS (Clock).
  BEGIN
```

2,5p

```
IF (Clock'EVENT AND Clock ='1') THEN
 IF Clear ='1' THEN
     Q <= "00";
 ELSE
   Q <= Q+'1';
 END IF;
 END IF;
END PROCESS;
END Behavior;
```

```
ENTITY Dec2to4 IS
PORT ( w  : IN   STD_LOGIC_VECTOR (1 DOWNTO 0);
          En : IN STD-LOGIC;
          Y  : OUT  STD-LOGIC-VECTOR (0 TO 3));
END Dec2to4;
ARCHITECTURE Behavior OF Dec2to4 IS
~~SIGNAL Count : STD_LOGIC_VECTOR~~
BEGIN
PROCESS (w, En)
 BEGIN
    IF En ='1' THEN
       CASE w  IS
          WHEN "00" => y <= "1000";
          WHEN "01" => y <= "0100";
          WHEN "10" => y <= "0010";
          WHEN OTHERS => y <= "0001";
    END CASE;
  ELSE  y <= "0000";  ( i.e. if En=0)
  END IF;
 END PROCESS;
```

20P

Now declare these components in a Package:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
 PACKAGE components IS
  COMPONENT upcounter
   PORT ( CLOCK, CLEAR : IN   STD_LOGIC;
             Q              : OUT STD_LOGIC_VECTOR(1 DOWN TO 0));
   END COMPONENT;

   COMPONENT  Dec 2to4
    PORT ( W  : IN    STD_LOGIC_VECTOR (1 DOWN TO 0);
           En : IN   STD_LOGIC;
           Y  : OUT   STD_LOGIC_VECTOR (0 TO 3));
    END component;

    END components;
```

2 p

Now the main program:

7 p

```
 LIBRARY ieee;
 USE iee.std_logic_1164.all;
 USE work.components.all;
 ENTITY main IS
     PORT ( CLOCK, CLEAR, En : IN STD_LOGIC;
            T                  : OUT STD_LOGIC_VECTOR(0TO3));
  END main;
 ARCHITECTURE Structure OF main IS
  SIGNAL : count : STD_LOGIC_VECTOR (1 DOWN TO 0);
  BEGIN
   Counter: upcounter PORT MAP (Clock, clear, Count);
```

```
dec : dec2to4 PORTMAP (count, 1, T);

END Structure;
```

Exam Question 5 top

4 Explain with an example why OVERFLOW occurs in SIGNED arithmetic and NOT in unsigned arithmetic. Show how the OVERFLOW condition is detected.

if a and b are 4-bit signed numbers, when they have different sign no overflow occurs, since the result is not out of the range. However, when the two numbers are of the same sign then OVERFLOW may occur if the result exceeds the range that can be represented by the number of bits in the signed representation (i.e. MSB is representing the sign of the number) 3

Lets explain this with an example:

$a = +7$

$b = +2$

$$a+b = \begin{array}{r} +7 \\ +2 \\ \hline +9 \end{array} \Rightarrow \begin{array}{r} 0111 \\ +0010 \\ \hline 1001 \end{array}$$

$+9 \xleftrightarrow{\times} 1001$

$C_3 = 1$
$C_4 = 0$

a negative number! (9 does not fit in the range that can be represented by a signed 4-bit number.)

$\Rightarrow$ C_3 and C_4 are different!

$C_4 \oplus C_3 = 1$ 3

If $a = -7$
$b = -2$

$$\Rightarrow \begin{array}{rl} 1001 & (-7) \\ 1110 & (-2) \\ \hline 10111 & \end{array}$$

a positive number! WRONG.

$-7-2=-9$

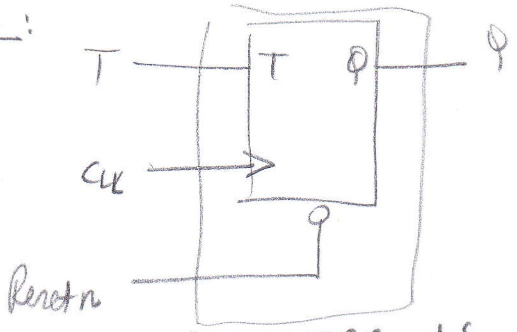
$C_4 = 1$
$C_3 = 0$
$C_4 \oplus C_3 = 1$ 4

Therefore overflow can be detecteds by $\boxed{V = C_4 \oplus C_3}$

(*) Exam Question 4.(15p)

Write the behavioral VHDL code for an T flip flop. with an asynchronous reset .

SOL :



```
ENTITY TFF IS
PORT (T, Clock, Resetn : IN STD-LOGIC ;
                      Q : OUT STD-LOGIC);        2

END Tff ;

ARCHITECTURE Behavior OF Tff IS             (Note : Behavior)
BEGIN
PROCESS (Clock, Resetn)
BEGIN
  IF Resetn ='0' THEN                  } Note : Asynchronous Reset *
   Q <= '0';                           }             8

  ELSIF (Clock'EVENT AND Clock ='1') THEN
    IF T ='1' THEN
       Q <= NOT Q ;

     ELSE                                         10
       Q <= Q;
     END IF;
   END IF;
 END PROCESS;
 END Behavior
```

Exam Question ⑥ 20p

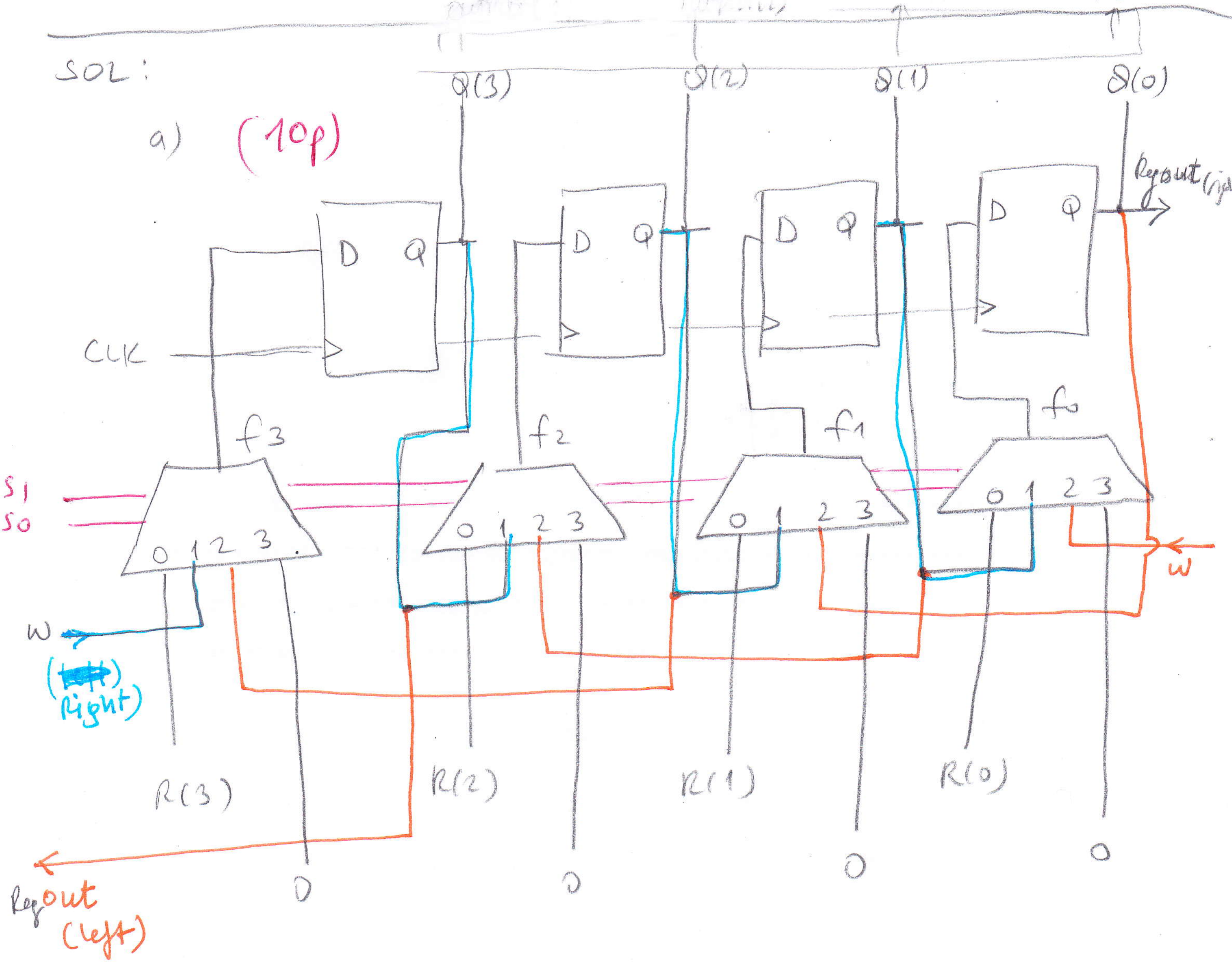
(*) a) Design a universal shift register (4-bit) with load and clear capability by using D Flip flops and 4-to-1 multiplexers.

S1	So	Function
0	0	Load Registers
0	1	Shift Right
1	0	Shift left
1	1	Load 0 ("0000")

b) Write the VHDL code to implement this circuit on an FPGA. (Structural code)

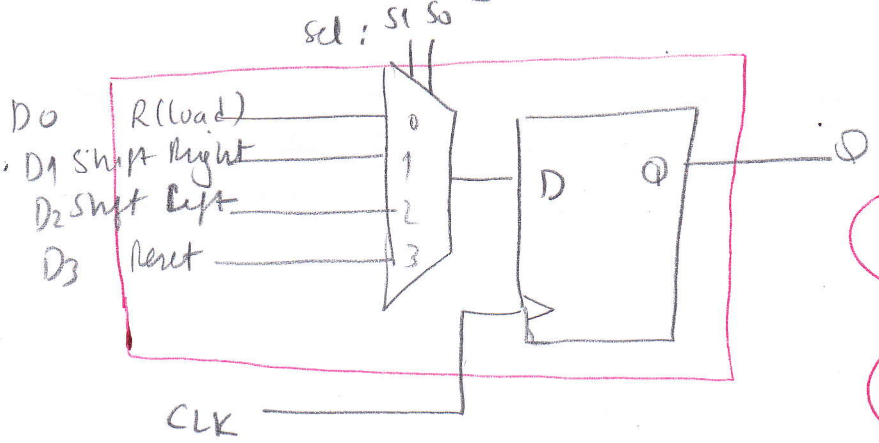
SOL:

a) (10p)



b) The VHDL code (structural)

First lets configure a mux with a D FF as a subcircuit: (component)



2,5

2.5

```
ENTITY   MUXDIFF   IS
 PORT ( D       :  IN    STD_LOGIC_VECTOR ( 0 TO 3 ) ;
        Clock, sel : IN  STD-LOGIC ;
        Q       : OUT    STD_LOGIC ) ;
END   MUXDIFF
  ARCHITECTURE  Behavior  OF  MUXDIFF  IS          ( Note: Behavior )
  BEGIN
  PROCESS ( sel, clock )
   BEGIN
   WAIT UNTIL  Clock'EVENT AND CLOCK = '1' ;
   WITH  sel  SELECT
         Q  <=  D0  WHEN  "00",
                D1  WHEN  "01",
                D2  WHEN  "10",
                D3  WHEN  OTHERS ;
    END PROCESS
    END BEHAVIOR
```

MAIN CODE using MUXDIFF as components:

```
ENTITY   UniversalReg   IS
 PORT ( CLOCK, W  :  IN    STD-LOGIC ;
          R       :  IN    STD-LOGIC-VECTOR ( 3 DOWNTO 0 ) ;
          sel     :  IN    STD-LOGIC-VECTOR ( 1 DOWN TO 0 ) ;
          Q       :  BUFFER  STD-LOGIC VECTOR ( 0 TO 3 ) ;
        RegOut    :  OUT   STD-LOGIC ) ;
END  UniversalReg
```

If instead we write a VHDL Code (Structural):

```
ARCHITECTURE structure OF UniversalReg IS
COMPONENT muxDIFF
  PORT ( D    : IN  STD-LOGIC-VECTOR  (0 TO 3);
         sel  : IN  STD-LOGIC-VECTOR  (1 DOWN TO 0)
         Q    : OUT STD-LOGIC);
END COMPONENT;
```

5

```
Stage 3 : muxdiff PORTMAP ( R(3) , W , Q(2) , '0', sel, clock, Q(3));
Stage 2 : muxdiff PORTMAP (R(2), Q(3), Q(1), '0', sel, clock, Q(2));
stage1 : muxdiff PORTMAP (R(1), Q(2), Q(0), '0', sel, clock, Q(1));
stage 0 : muxdiff PORTMAP (R(0), Q(1), W , '0', sel, clock, Q(0));
Process (CLOCK, sel)
 BEGIN
  WAIT UNTIL CLOCK'EVENT AND CLOCK = '1';
  CASE sel IS
    WHEN "00" => Regout <= 0;
    WHEN "01" => Regout <= Q(0);
    WHEN "10" => Regout <= Q(3);
    WHEN OTHERS => Regout <= R(0);
   END CASE;
  END PROCESS
```

ICEAA '09

HOME
GENERAL INFORMATION
CALL FOR PAPERS
DEADLINES
TOPICS
ABSTRACT SUBMISSION
Closed
ICEAA AWARD
FINAL PAPER SUBMISSION
FINAL PROGRAM
SHORT COURSES
REGISTRATION
CONFERENCE SECRETARIAT
CONFERENCE VENUE
EXHIBITORS
COMMITTEES
HOTEL & TOURISTIC INFO
SYMPOSIUM TOURS
CONFERENCE BANQUET
ICEAA '07 GALLERY
ICEAA '05 GALLERY
ICEAA '03 GALLERY
CONTACT
STAFF ONLY