EASTERN MEDITERRANEAN UNIVERSITY
COMPUTER ENGINEERING DEPARTMENT
CMPE224 **DIGITAL LOGIC SYSTEMS**
VHDL EXPERIMENT V

TITLE: VHDL IMPLEMENTATION OF REGISTERS AND COUNTERS

OBJECTIVES: Implementation of registers, shift-registers, and counters, will be studied. The students are expected to learn how repetitive structures are mapped to VHDL implementations.

Below, you will find detailed examples on the implementation of registers and counters in VHDL. Examine these VHDL codes and implement them in Quartus II environment to understand the underlying ideas better.

Example 1: Multiple Bit Register

This is a simple N-bit register with parallel load consisting of an array of N D flip-flops.

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

entity NBit_Register is          -- Top-level design entity,
generic (n: NATURAL :=4); -- this statement defines a device specific structural
                                 -- parameter for this circuit.
Port (clock, reset: in std_logic;
        D: in std_logic_vector (n-1 downto 0)
        Q: out std_logic_vector (n-1 downto 0));
end NBit_Register;

architecture behavioral of NBit_Register is
begin
P0: process (clock, reset)       -- process that implements sequential behavior
begin
        if (reset = '0') then
                Q <= (others => ‘0’);   -- fills all elements of Q with zero
        elsif (clock'event and clock ='1') then
                Q<=D;
        end if;
end process P0;

end behavioral;
```

Example 2 N-bit Shift Register with Parallel Input

```
library ieee;
```

```
use ieee.std_logic_1164.all;

entity NBit_ShiftRegister is  -- Top-level design entity,
generic (n: NATURAL :=8); -- this statement defines a device specific structural
                          -- parameter for this circuit.
Port (clock, reset: in std_logic;
      Data_In: in std_logic;
      D: in std_logic_vector (n-1 downto 0);
      Q: out std_logic_vector (n-1 downto 0));
end NBit_ShiftRegister;

architecture behavioral of NBit_ShiftRegister is
begin
P0: process (clock, reset)       -- process that implements sequential behavior
variable reg: std_logic_vector (n-1 down to 0); -- defines a local variable for shift
                                                -- operations. Note that you
                                                --cannot read values of Q since it
                                                -- is defined as an output.
begin
      if (reset = '0') then
            Q <= D;   -- fills all elements of Q with parallel input D
      elsif (clock'event and clock ='1') then
            reg:=reg (n-2 downto 0) & Data_In;
                  -- register(n-2 down to 0) is concatenated with Data_In
                 -- from right, bit (n-1) is lost.
            Q<=reg;
      end if;
end process P0;

end behavioral;
```

Example 3 N-bit Register with Multiple Mode of Operations

Assume that there are two control inputs S_1 and S_0 that control the mode of operation as follows:

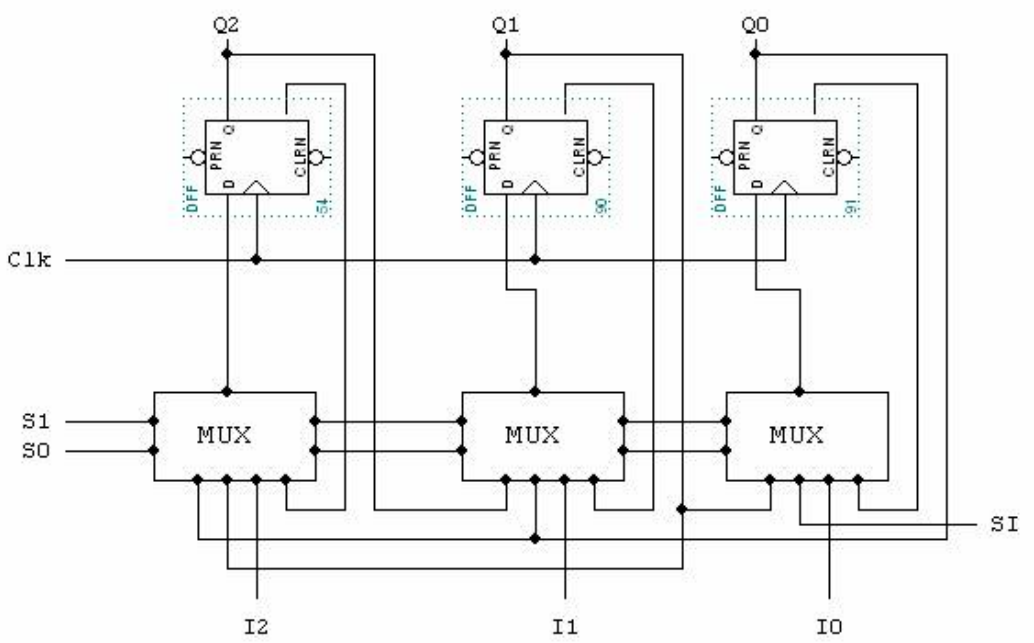
S_1S_0	Operation
00	Hold the current contents
01	Shift right
10	Shift left
11	Parallel load

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

entity MMode_Register is     -- Top-level design entity,
generic (n: NATURAL :=8); -- this statement defines a device specific structural
                          -- parameter for this circuit.
Port (clock, reset: in std_logic;
```

```
        Left_In, Right_In: in std_logic;
        D: in std_logic_vector (n-1 downto 0);
        S: in std_logic_vector(1 downto 0);
        Q: out std_logic_vector (n-1 downto 0));
end MMode_Register;

architecture behavioral of MMode_Register is
begin
P0: process (clock, reset)       -- process that implements sequential behavior
variable reg: std_logic_vector (n-1 down to 0); -- defines a local variable for shift
                                                -- operations. Note that you
                                                --cannot read values of Q since it
                                                -- is defined as an output.
begin
        if (reset = '0') then
                reg := (others =>'0');   -- fills all elements of register with 0
        elsif (clock'event and clock ='1') then
                case S is
                        when "11" =>
                                reg:=D;
                        when "10"=>
                                reg:=reg (n-2 downto 0) & Left_In;
                        when "01"=>
                                reg:=Right_In & reg (n-1 downto 1);
                        when others =>
                                null; -- do nothing
                end case;
        end if;
        Q<=reg;

end process P0;

end behavioral;
```

PRELIMINARY WORK:

Consider the below circuit and implement it in VHDL as illustrated in Example3. Perform all experimental steps including VHDL coding, waveform preparation, and simulations; make them ready before you come to the laboratory.



S_1	S_0	**Circuit Operation**
0	0	***Complement***
0	1	***Parallel Load***
1	0	***Shift Left***
1	1	***Rotate Right***

EXPERIMENTAL WORK:

Demonstrate your preliminary work to the laboratory assistants using Quartus II environment. Be ready for detailed questions on your work.

Good Luck.

Dr. Adnan Acan
Dr. Evgueni Doukhnitch
Dr. Muhammed Salamah.