Lista de Exercícios sobre a Linguagem VHDL

1) Dada a descrição do multiplexador 4-1 de dados de 8 bits:

```
Entity Mux8 is
        Port (  select : in bit_vector (0 to 1);
                In_1 : in bit_vector (0 to 7);
                In_2 : in bit_vector (0 to 7);
                In_3 : in bit_vector (0 to 7);
                In_4 : in bit_vector (0 to 7);
                Z_out : in bit_vector (0 to 7));
End Mux8;
```

a) Escreva uma descrição comportamental de sua arquitetura utilizando um processo com comando **case**.
b) Escreva uma descrição comportamental de sua arquitetura utilizando um processo com comando **if-the-esle**.
c) Escreva uma descrição em fluxo de dados (*dataflow*) de sua arquitetura usando designações de sinais concorrentes.
d) Escreva uma descrição em fluxo de dados (*dataflow*) de sua arquitetura usando as designações com **when-else**.

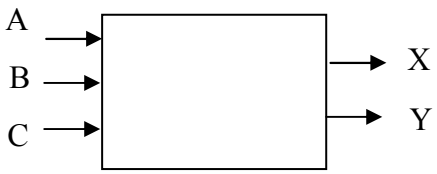
2) Dê o código VHDL de um módulo que recebe como entrada um valor de tipo std_logic_vector e retorne o valor em ordem inversa em tipo unsigned, utilizando-se da construção generate. Por exemplo, recebe ´´001101´´ e retorne ´´101100´´. Deixe evidente as bibliotecas necessárias.

3) Faça o diagrama de tempos para componente abaixo (é dada a sua arquitetura comportamental) e descubra qual a sua função. Enable é um sinal de entrada (Bit) e z de saída (Bit) do módulo. (Sugestão: para ter certeza da resposta, faça simulação no ModelSim).

```
ARCHITECTURE behavioral OF component IS
        BEGIN
                PROCESS
                        VARIABLE lag: BIT := '1';
                BEGIN
                        IF enable = '1' THEN
                                lag := NOT lag;
                        END IF;
                        z <= lag;
                        WAIT FOR 50 ns;
                END PROCESS;
        END behavioral;
```

4) Modifique o contador VHDL, vistos na aula 3, adicionando uma linha de controle **Updown** que faz o seguinte: quando **Updown** =´0´, o contador faz uma contagem regressiva; quando **Updown** =´1´, o contador realiza a contagem progressiva, como no arquivo original.

5) Considere o seguinte bloco de três entradas e duas saídas cuja funcionalidade está descrita na tabela da verdade abaixo.



A	B	C	X	Y
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	0
1	1	0	1	1
1	1	1	1	0

a) Escreva a declaração da entidade.
b) Escreva a arquitetura da entidade usando o estilo estrutural. Use somente a seguinte entidade como componente:

```
Entity NAND2 is
        Port (   A, B : in Bit;
                 Z: out Bit);
End NAND2;
```

c) Escreva a arquitetura da entidade usando o estilo fluxo de dados (*dataflow*).

6) O circuito a ser projetado é um autômato celular de N bits (z[0] a z[N-1]), com regra 90, utilizado freqüentemente na geração de padrões pseudo-aleatórios. Trata-se de uma série de N flip-flops (registradores tipo D, com saídas q[0] a q[N-1]) ligados em série de acordo com o padrão da figura a abaixo. Nas pontas, para os *bits* mais e menos significativo, assume-se que q[-1] e q[N] sejam iguais a ´0´, portanto, apresentando as configurações da figura b, após a simplificação.

a) Faça o projeto do flip-flop tipo D sensível á borda de subida do clock, ou seja a sua descrição VHDL da entidade e da arquitetura comportamental . Use a instrução **generics** para o tempo de atraso (default = 12 ns).

b) Faça o projeto do EXOR, ou seja a sua descrição VHDL da entidade e da arquitetura dataflow . Use a instrução **generics** para o tempo de atraso (default = 8 ns).

c) Faça o projeto do autômato celular de N bits, ou seja a sua descrição VHDL da entidade e de sua arquitetura estrutural. Utilize necessariamente o comando **generate** e os elementos do item a) e b) como componentes (instancie-os, respectivamente, com t_{FF}= 15ns e t_{XOR}= 10ns).

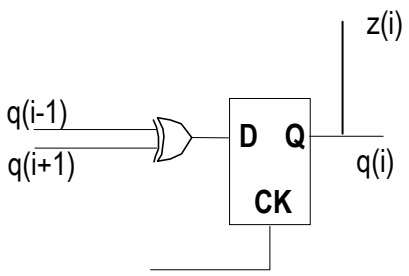
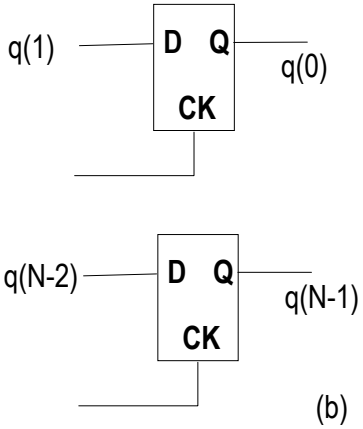


Figura (a)



7) Um projeto é desenvolvido e os códigos VHL de um circuito *LFSR* são apresentados abaixo. Cada arquivo VHDL é alocado em uma biblioteca diferente.

a) Monte um diagrama com as bibliotecas e seus conteúdos, e defina as dependências entre entidades e pacotes.

b) Há algumas incorreções nos códigos VHDL no que refere-se ao acesso às bibliotecas, que impedem a simulação do circuito. Faça as mudanças necessárias para que possa funcionar (sugestão: faça a simulação no Modelsim para confirmação).

===ARQUIVO LFSR_pack_1.vhd==========

```
PACKAGE LFSR_PACK IS

        COMPONENT xor2
                GENERIC(t_xor : time := 4 ns);
                PORT (x, y : IN BIT;
                        z : OUT BIT);
        END COMPONENT;

END;


-- descricao do xor


ENTITY xor2 IS
        GENERIC(t_xor : time := 4 ns);
        PORT( x, y: IN BIT;
                z: OUT BIT);
END xor2;
```

```
ARCHITECTURE dataflow OF xor2 IS

BEGIN
        z <= x XOR y AFTER t_xor;
END dataflow;
```

=== **Fim do ARQUIVO LFSR_pack_1.vhd**==========

===**ARQUIVO LFSR_pack_2.vhd**==========

```
PACKAGE LFSR_PACK IS

                COMPONENT dff
                GENERIC(t_q : time := 12 ns; t_qbar : time := 14 ns);
                PORT( D, clock: IN Bit;
                                        q: OUT Bit;
                                        qbar: OUT Bit);
        END COMPONENT;
END;


-- descricao do dff

ENTITY dff IS
        GENERIC(t_q : time := 12 ns; t_qbar : time := 14 ns);
        PORT( D, clock: IN Bit;
                q: OUT Bit:='1';
                qbar: OUT Bit);
END dff;



ARCHITECTURE with_package OF dff IS

BEGIN

        PROCESS (clock)
        BEGIN
                IF (clock'EVENT AND clock ='0') THEN
                        p_latch( d, t_qbar, q, qbar);
                END IF;
        END PROCESS;

END with_package;
```

=== **Fim do ARQUIVO LFSR_pack_2.vhd**==========

====**ARQUIVO LATCH_pack.vhd**==========

```
PACKAGE LATCH_PACK IS
        PROCEDURE p_latch (SIGNAL d : IN BIT;
                        CONSTANT tempo :IN time;
                        SIGNAL q, qbar: OUT BIT);
END;


PACKAGE BODY LATCH_PACK IS
        PROCEDURE p_latch ( SIGNAL d : IN BIT;
                        CONSTANT tempo :IN time;
                        SIGNAL q, qbar: OUT BIT) IS
        BEGIN
                q <= d AFTER tempo;
                qbar <= NOT d AFTER tempo;
        END ;
END ;
```

=== **Fim do ARQUIVO LATCH_pack.vhd** ==========

== **ARQUIVO LFSR.vhd** ==========

```
ENTITY LFSR IS
        PORT( clock: IN BIT;
                o: OUT     BIT_VECTOR (2 DOWNTO 0));
END LFSR;


ARCHITECTURE structure OF LFSR IS

        COMPONENT dff
        GENERIC(t_q : time := 12 ns; t_qbar : time := 14 ns);
        PORT( D, clock: IN Bit;
                q: OUT Bit;
                qbar: OUT Bit);
        END COMPONENT;

        COMPONENT xor2
                GENERIC(t_xor : time := 4 ns);
                PORT (x, y : IN BIT;
                        z : OUT BIT);
        END COMPONENT;

        signal xor_out, q0, q1, q2, qbar :BIT;
```

(arquivo continua na próxima página)

```
BEGIN
        FF0 : dff GENERIC MAP (11 ns, 13 ns) PORT MAP (q1, clock, q0, qbar);
        FF1 : dff GENERIC MAP (11 ns, 13 ns) PORT MAP (q=>q1, d=> xor_out,
clock=>clock);
        FF2 : dff GENERIC MAP (11 ns, 13 ns) PORT MAP (clock=>clock, q=>q2,
d=>q0,);
        X0 : xor2 PORT MAP (q2, q0, xor_out);
        o <= (q2,q1,q0);
END structure;
```

== **Fim do ARQUIVO LFSR.vhd** ==========