Universidade Federal da Fronteira Sul
Curso de Ciência da Computação
Disciplina: Sistemas Digitais
Professor: Geomar A. Schreiner

Matrícula: 2023000 1178 **Nota:** 7,1

1. (1,0) Diferencie um contador síncrono de um assíncrono. Em sua resposta considere apresentar exemplos de aplicação, e se necessário desenhos para auxiliar na explicação.

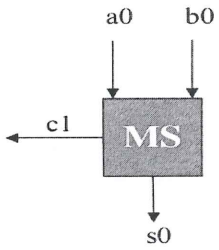
0,6

2. (2,0) Utilizando Flip-Flops JK, crie um contador assíncrono capaz de contar até 26, ou seja, faça um circuito contador de módulo 26.

1,8

3. (3) Considerando a arquitetura de um meio somador apresentada a baixo, construa um somador completo de 2 Bits em VHDL. Você pode criar um componente de somador completo e reutilizá-lo ou fazer um conjunto de arranjos com o meio somador.

1,2



4. (1,0) Considerando o programa em VHDL abaixo apresente as saídas conforme a tabela.

Local em toda prova

```
entity teste is
port(
    a,b,c: in std_logic;
    s0,s1: out std_logic
);
end teste;

architecture behav2 of teste is
    signal x: std_logic := '0';
begin
    process(a,b,c)
        variable y: std_logic := '0';
    begin
        x <= (a and b) xor c;
        y := (a and b) xor c;

        s0<=y;
        s1<=x;
    end process;
end behav2;
```

t	a	b	c	s0	s1
0	0	0	1	0	1
1	0	1	1	1	1
2	1	1	0	1	1
3	1	1	1	1	0
4	0	0	0	0	0

5. (3,0) Projete e descreva em VHDL um circuito codificador BCD para 7-segmentos, ou seja, um circuito que recebe uma palavra código BCD (de 4 bits) e apresenta na saída a palavra de 7 bits que ativa os segmentos correspondentes no display de 7 segmentos. Utilize a seguinte entidade para descrever sua solução em VHDL:

1,5

```
entity bcd7seg is
    port (
        EN : in std_logic; -- Enable
        D : in std_logic_vector (3 downto 0); -- Encoder Input
        S : out std_logic_vector (6 downto 0) -- Encoder Output
    );
end entity bcd7seg;
```

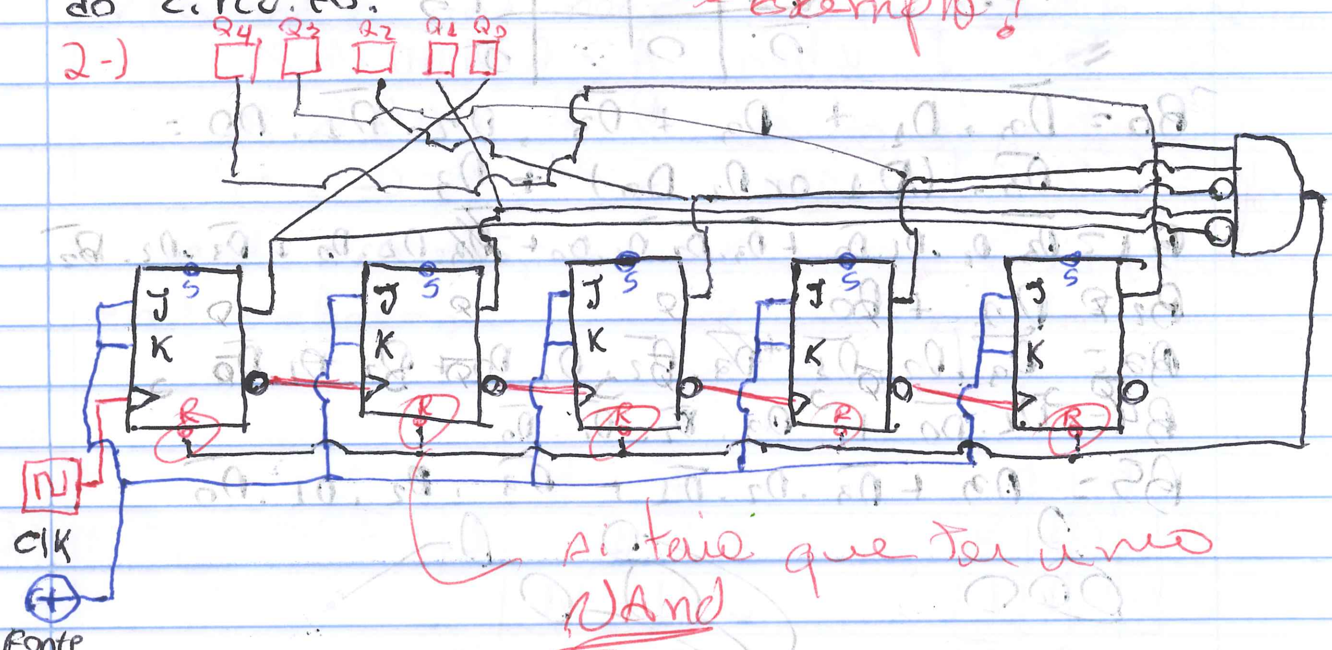
2023000 1178

1-) A diferença entre um contador síncrono e um assíncrono é que neste não há sinal de clock comum a todos os flip-flops, o clock não é aplicado diretamente a todos os flip-flops mas sim ao flip-flop que representa o bit de menor valor da contagem, as entradas de clock dos demais flip-flops são acionadas de acordo com a saída do flip-flop anterior.

Por outro lado, o clock do contador síncrono é comum a todos os flip-flops. Ou seja, todos os flip-flops são acionados simulteaneamente sempre que o clock do circuito for acionado, o que vai determinar a mudança de estado da saída é o valor armazenado em cada flip-flop e proveniente das saídas dos demais flip-flops do circuito.

↳ Exemplo!

2-)



3-)

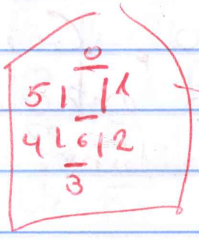
```
entity adder is
Port (
    a, b : in std_logic_vector(1 downto 0);
    c : out std_logic_vector(2 downto 1);
    s : out std_logic_vector(1 downto 0);
    );
end entity adder;

architecture behave of adder is
begin
  process (a, b)
    variable x : std_logic := '0';
    variable y : std_logic := '0';
  begin
    x := (c AND a) or (c AND b) or (a AND b AND c);
    y := (a AND not b and not c) or (not a and not b and c)
or (a and b and not c) or (not a and b and not c);

    s <= x;
    c <= y;
  end process;
end behave;
```

Entendo o ponto mas esse sinal não foi a soma, C é uma saída não uma entrada como vai considerar?

5-) architecture behave of bcd7seg is begin
process (EN, D)

begin
S[0] <= ~D[3] and (D[1] or (D[2] and D[0])) or D[3];
S[1] <= (~D[3] and D[2] and D[1] and ~D[0]) or
(~D[3] and ~D[2] and ~D[1] and D[0]) or (D[2] And D[0]) or
(D[3] and ~D[2] and D[1]);
S[2] <= D[2] or D[1];
S[3] <= D[3] and (D[2] and ~D[1] or ~D[2] and D[1]
and D[0] or D[2] and D[1] and ~D[0]);
S[4] <= D[0] and (D[1] or D[3] and ~D[2] and ~D[1]);
S[5] <= D[3] or (D[2] and (D[1] or D[0] and D[0]));
S[6] <= D[3] or (D[1] and (D[0] or (~D[3] and ~D[2])
or (~D[3] and D[2] and ~D[1])));
if EN = '1'
end process;
end behave;

Não sei qual o padrão que tu usou mas considero no o que coloquei no quadro



não estão de acordo!