

## Laboratório 05 - Contadores

### Objetivos

1. Experimentar a descrição em VHDL de circuitos na forma comportamental;
2. Reforçar os conceitos de contadores assíncronos e síncronos;
3. Pôr em prática conceitos aprendidos na disciplina Circuitos Digitais - Teoria.

### Introdução

Na aula de hoje serão apresentados os blocos contadores. Os contadores são divididos em contadores são divididos em contadores síncronos e assíncronos. Os contadores síncronos são aqueles em que a mudança do seu estado ocorre em cada pulso de clock, enquanto os contadores assíncronos não possuem uma entrada de clock. Nessa aula, trataremos apenas dos contadores síncronos, sendo apresentados os contadores crescentes e decrescentes.

### Contadores

Um contador de  $N$  bits é um componente que pode incrementar ou decrementar o próprio valor a cada ciclo de relógio, quando uma entrada de habilitação é 1.

Incrementar significa adicionar 1 a sua contagem e decrementar significa subtrair 1 a sua contagem. Um contador que pode incrementar é conhecido como **contador crescente**, um contador que pode decrementar é conhecido como **contador decrescente** e um contador que pode incrementar e decrementar é conhecido como **contador crescente/decrescente**.

Um contador crescente de quatro bits produz a seguinte sequência: 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111, 0000, 0001, etc. Note que quando o contador atinge o seu valor mais alto (1111), ele retorna para 0.

De forma similar, um contador decrescente dá uma volta completa quando chega a 0 e, em seguida, indo para o valor mais elevado. Na **Fig. 1** é apresentado o diagrama de blocos de um contador crescente de 4 bits. Quando  $cnt=1$ , o contador incrementa seu próprio valor a cada ciclo de relógio. Quando  $cnt=0$ , o contador mantém o seu valor atual.

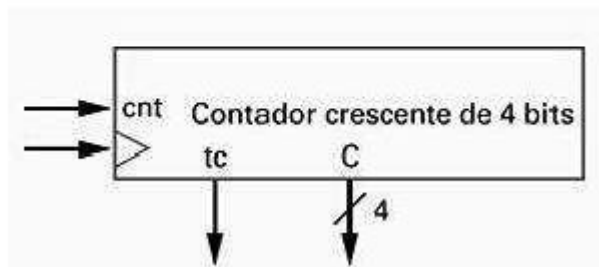


Fig. 1: Diagrama de blocos de um contador crescente de 4 bits.

## Contador crescente

Um contador crescente pode ser implementado utilizando registradores. A **Fig. 2** apresenta um exemplo de implementação, o qual utiliza um bloco incrementador para somar 1 ao valor atual do registrador. Quando  $cnt=0$ , o registrador mantém o seu valor atual e, quando  $cnt=1$ , o valor corrente no registrador é somado de 1.

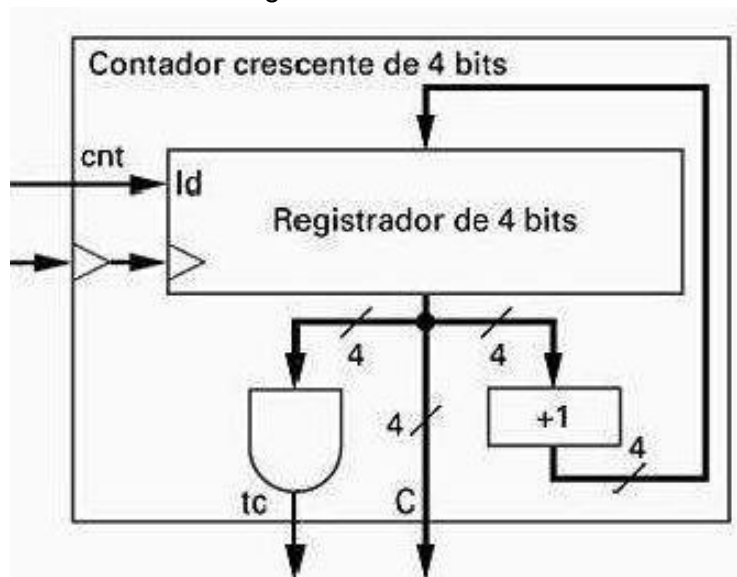


Fig. 2: Contador síncrono crescente

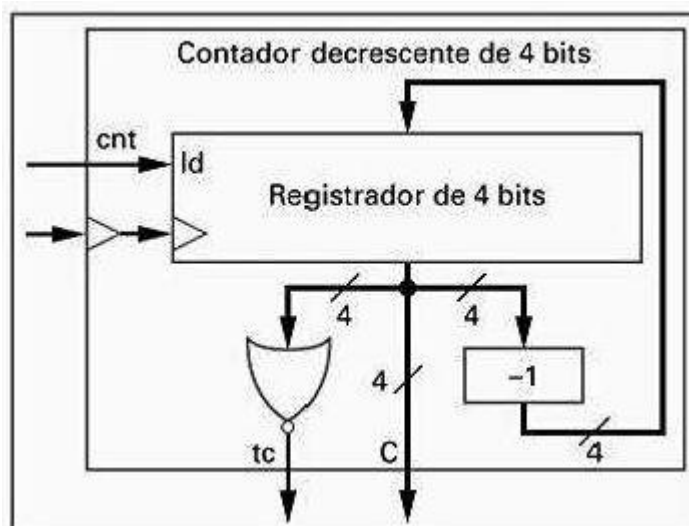
A tabela-verdade de um contador crescente é apresentada na **Fig. 3**. Note que as saídas  $s_0$ ,  $s_1$ ,  $s_2$  e  $s_3$  assumem os valores dos bits de entrada. Quando o contador chega em sua contagem final, as suas saídas retornam para 0.

Entradas				Saídas				
a3	a2	a1	a0	c0	s3	s2	s1	s0
0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	1	0
0	0	1	0	0	0	0	1	1
0	0	1	1	0	0	1	0	0
0	1	0	0	0	0	1	0	1
0	1	0	1	0	0	1	1	0
0	1	1	0	0	0	1	1	1
0	1	1	1	0	1	0	0	0
1	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	1	0
1	0	1	0	0	1	0	1	1
1	0	1	1	0	1	1	0	0
1	1	0	0	0	1	1	0	1
1	1	0	1	0	1	1	1	0
1	1	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	0

**Fig. 3:** Tabela-Verdade de um contador crescente.

## Contador decrescente

Um contador decrescente pode ser projetado de forma semelhante a um contador crescente, substituindo o incrementador por um decrementador. Na **Fig. 4** é apresentada a estrutura de um decrementador de 4 bits.



**Fig. 4:** Estrutura de um contador decrescente de 4 bits

## Descrição Comportamental do VHDL:

**Quadro 1:** Descrição comportamental de um contador de 4 bits com load em paralelo.

```
ENTITY contador IS
PORT(
  clk:  IN BIT; --entrada de clock
  ld:   IN BIT; --carrega os dados
  reset: IN BIT;
  data: IN INTEGER RANGE 15 DOWNT0 0; -- entrada de dados
  q:    OUT INTEGER RANGE 15 DOWNT0 0; --saída de dados
END contador;

ARCHITECTURE comportamento OF contador IS
BEGIN PROCESS(clk,reset)
  VARIABLE qv: INTEGER RANGE 15 DOWNT0 0; --variável para a saída

  BEGIN
    IF(reset = '1') THEN
      qv := 0;
    ELSIF(clk ' event and clk = '1') THEN
      IF(ld = '1') THEN
        qv := data;
      ELSE
        IF(qv >= 9) THEN
          qv := 0;
        ELSE
          qv := qv + 1;
        END IF;
      END IF;
    END IF;
    q <= qv;
  END PROCESS;
END;
```

## Hora de Praticar:

1. Implemente o contador crescente, apresentado no quadro acima, no Quartus II e verifique os resultados no Simulation Waveform Editor;
2. Implemente um contador crescente de 4 bits que conte até sua última contagem e retorne ao seu valor inicial. Adicione o sinal indicativo de término de contagem: tc=1 quando o contador atingir 1111 e tc=0 caso contrário.

- 
3. Implemente um contador decrescente de 4 bits. Adicione o sinal indicativo de término de contagem:  $tc=1$  quando o contador atingir 0000 e  $tc=0$  caso contrário.
  4. Elabore e entregue um relatório contendo a descrição e resultados deste experimento.