

## Laboratório 5 – Calculadora

### Objetivos:

1. Experimentar a descrição em VHDL de circuitos digitais;
2. Reforçar os conceitos de somadores e subtratores.
3. Pôr em prática conceitos aprendidos na disciplina teórica.

### Introdução:

Nesta aula iremos implementar uma calculadora que realiza somas e subtrações, a partir da utilização de componentes somadores/subtratores.

### Somadores:

Os somadores são um bloco combinacional de grande importância em projetos de circuitos digitais, já que a adição é uma das operações mais comuns. Um somador de largura  $N$  é responsável por somar dois números binários  $A$  e  $B$  de  $N$  bits, além de administrar o *carry* ("vai um")  $C$  de um bit.

Uma das formas de projetar este circuito é através da construção da tabela verdade, onde podemos extrair as equações booleanas para cada saída do somador. O problema com esta abordagem é que, a medida em que a largura do somador aumenta, a tabela verdade e a complexidade do circuito crescem de forma exponencial. Um somador de quatro bits, por exemplo, teria duas entradas de quatro bits, uma entrada de *carry* (opcional), quatro saídas de soma e uma saída de *carry*. Ou seja, no exemplo citado acima temos uma tabela verdade extensa, com ao menos 13 colunas e 256 linhas.

Uma segunda abordagem para a construção dos somadores seria replicar a forma como as adições são realizadas manualmente, através de uma coluna por vez. Por exemplo, vamos considerar a soma ilustrada na Figura 1. Na primeira coluna é feita a soma dos dois bits de entrada, gerando um *carry* e um resultado de saída. Nas colunas restantes é realizada a soma dos bits de entrada juntamente com o *carry*, que é repassado para a coluna seguinte caso exista.

$$\begin{array}{r}
 \phantom{0}1 \leftarrow \text{Carry (vai um)} \\
 1101 \\
 + 0111 \\
 \hline
 0 \leftarrow \text{Soma}
 \end{array}$$

**Figura 1:** Processo de soma em coluna, apresentando o resultado e o *carry*.

Ao final do processo de soma, o bit de *carry* se torna o último bit que compõe o resultado final (o MSB). O circuito somador que opera neste estilo é conhecido como *carry-ripple*. A Figura 2 ilustra o processo completo de soma para o exemplo descrito anteriormente.

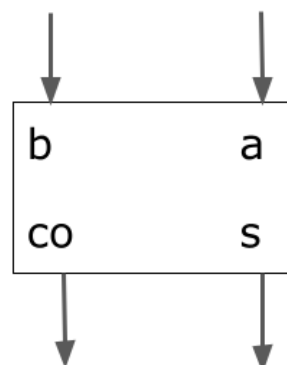
$$\begin{array}{r}
 111 \\
 1101 \\
 +0111 \\
 \hline
 10100
 \end{array}$$

Carry de Saída

**Figura 2:** Processo completo de soma, com o *carry* de saída sendo o MSB do resultado.

A partir deste ponto podemos projetar os componentes combinacionais para realizar o processo de adição em colunas. A construção deste circuito digital é realizada através de meio somadores e somadores completos. A Figura 3 ilustra um bloco meio somador e sua tabela verdade. Este bloco possui como entradas os bits *a* e *b* que serão somados, o resultado *s* e o *carry* de saída *co*.

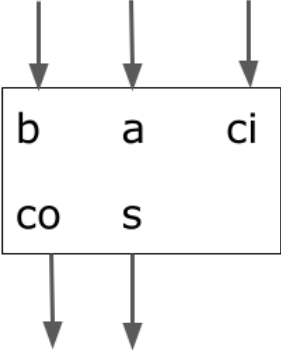
Entradas		Saídas	
a	b	co	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



**Figura 3:** Representação do meio somador e tabela verdade.

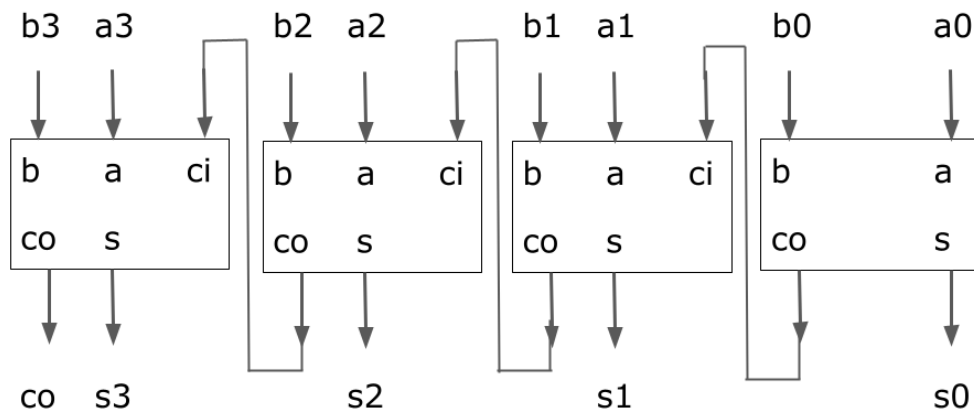
A Figura 4 ilustra um bloco somador completo e sua tabela verdade. Neste caso temos como entradas os bits  $a$ ,  $b$  e  $ci$  que serão somados, o resultado  $s$  e o  $carry$  de saída  $co$ . Como pode ser visto, a diferença entre o somador completo e o meio somador é a presença do  $carry$  de entrada  $ci$ .

Entradas			Saídas	
a	b	ci	co	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



**Figura 4:** Representação do somador completo e tabela verdade.

É importante notar que os componentes apresentados acima realizam somas de um bit. Para somar um número binário de largura  $N$  maior que um é possível combinar os blocos somadores através de instanciação. Esta configuração permite que cada bloco receba uma coluna dos bits  $a$  e  $b$  que serão somados, e o  $carry$  de saída é propagado para o  $carry$  de entrada do bloco seguinte. A Figura 5 ilustra um somador de quatro bits, criado a partir de um meio somador e três somadores completos.



**Figura 5:** Representação de uma soma de quatro bits utilizando meio somador e somadores completos.

## Descrição Comportamental do VHDL:

A linguagem de descrição de hardware VHDL permite a utilização da operação de soma de forma comportamental. Neste tipo de descrição, não é necessário preocupar-se com portas lógicas. Porém, durante o *design*, é importante gerenciar o comportamento do *carry* corretamente, além de evitar problemas de *overflow*. Este erro acontece quando não existem bits suficientes para computar o resultado da soma. No quadro 1, um somador de 16 bits com *carry* de entrada é implementado por meio de descrição comportamental.

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

entity SOMADOR is
generic(
    WIDTH : integer := 16
);
port(
    CIN    : in  std_logic;
    A      : in  std_logic_vector(WIDTH - 1 downto 0);
    B      : in  std_logic_vector(WIDTH - 1 downto 0);
    S      : out std_logic_vector(WIDTH - 1 downto 0);
    COUT   : out std_logic
);
end SOMADOR;

architecture BEHAVIOR of SOMADOR is

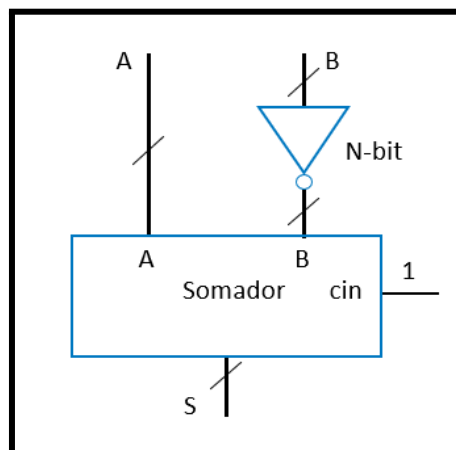
    signal SUM: std_logic_vector(WIDTH downto 0);
```

```
begin
    SUM <= ('0' & A) + ('0' & B) + CIN;
    S <= SUM(WIDTH - 1 downto 0);
    COUT <= SUM(WIDTH);
end BEHAVIOR;
```

**Quadro 1:** Descrição comportamental de um somador de 16 bits com *carry* de entrada.

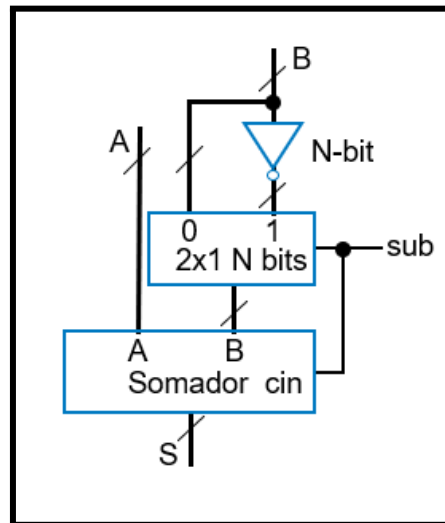
## Construindo um subtrator usando um somador e complemento de dois:

Com o conhecimento da representação em complemento de dois, podemos ver como fazer a subtração usando um somador. Para computar  $A - B$ , calculamos  $A + (-B)$ , que é o mesmo que  $A + B' + 1$  porque  $-B$  pode ser calculado como  $B' + 1$  em complemento de dois. Assim, para realizar a subtração, invertamos  $B$  e colocamos 1 na entrada de “vem um” de um somador, como mostrado na Figura 6.



**Figura 6:** Subtrator construído a partir de somador e complemento de dois.

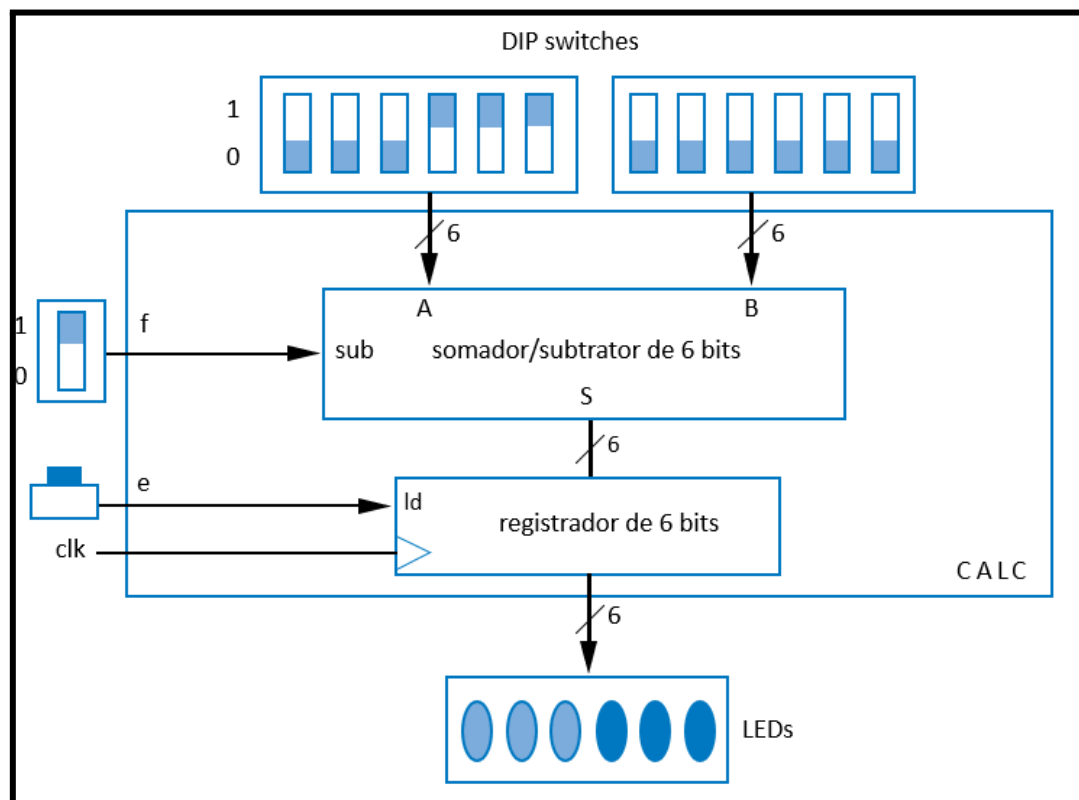
**Somador/Subtrator** - Uma forma direta de se projetar um componente subtrator/somador consiste em uma entrada *sub* tal que, quando *sub* = 1, o componente subtrai, mas, quando *sub* = 0, o componente soma, como mostrado na Figura 7. O multiplexador 2x1 de N bits deixa passar  $B$  quando *sub* = 0 e deixa passar  $B'$  quando *sub* = 1. O sinal *sub* também é conectado à entrada *cin* de “vem um”, de modo que *cin* é 1 na subtração.



**Figura 7:** Componente somador/subtrator.

## Calculadora de 6 bits:

Vamos projetar uma calculadora muito simples que pode somar dois números binários de seis bits e produzir um resultado de seis bits. Os números binários de entrada virão de chaves DIP de seis botões (DIP *switches*) e a saída será mostrada através de seis LEDs, como mostrado na Figura 8. Uma chave DIP (*Dual Inline Package*) de seis bits é um componente digital simples com botões ou chaves que um usuário pode mover para cima ou para baixo. Em cima, é gerado 1 no bit correspondente e, em baixo, é gerado 0. Um LED (*light-emitting diode*) é apenas uma pequena lâmpada que acende, quando a entrada do LED é 1, e apaga, quando a entrada é 0. O botão *e* significa “igual a” e indica quando o novo resultado deve ser exibido. Apertaremos o botão *e* e somente após ter configurado as duas chaves DIP com os novos valores que devem ser somados ou subtraídos. A entrada *clk* será conectada no pino *ld* de um registrador com carga em paralelo, para evitar que os LEDs fiquem piscando até estabilizar o resultado. Observe que o valor exibido será uma representação binária de 6 bits, com complemento de dois.



**Figura 8:** Calculadora de 6 bits que realiza somas e subtrações.

---

## **Atividades:**

1. Implemente a calculadora da Figura 8 em VHDL e simule seu comportamento;
2. Sintetize o código VHDL no kit FPGA disponível e apresente o resultado em sala de aula;
3. Entregue um relatório descrevendo a execução dos itens 1 e 2.

### Observações:

- O somador implementado deve ser do tipo *carry-ripple*, com descrição por portas lógicas.
- O circuito somador/subtrator deve ser implementado segundo a Figura 7.