UNIVERSIDADE FEDERAL DE SANTA CATARINA

# Laboratório 3:
# Projeto Hierárquico

## EEL5105 – Circuitos e Técnicas Digitais

# Objetivos

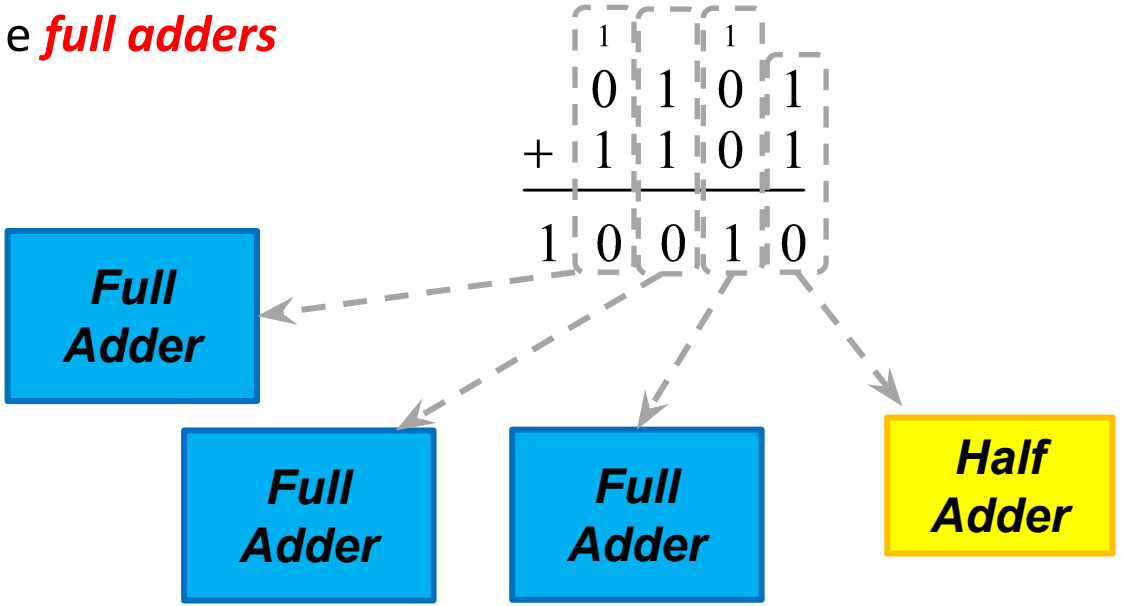
- Entender o conceito de **Projeto Hierárquico**.
- Implementar um **Projeto Hierárquico** em **VHDL** usando **component** e **port map**.
- Realizar **implementações** visando fixar os conceitos e as estruturas estudadas.

# Introdução

- **Projeto Hierárquico**
  - Abordagem de projeto usada não somente em **VHDL**.
  - **Idéia**: compartimentalizar o projeto em múltiplos **componentes** que podem ser **criados separadamente** e depois **integrados** e **reutilizados**.
  - Facilita a **leitura**, **entendimento** e **manutenção** do código.

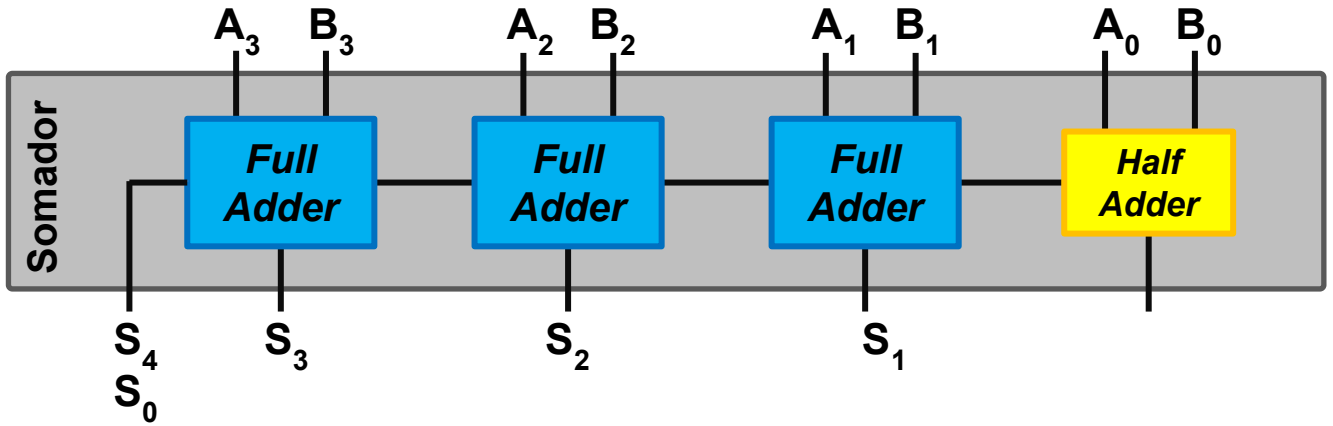
# Introdução

- **Projeto Hierárquico**
  - **Exemplo 1**: somador de números de **4 bits** construído usando um ***half adder*** e ***full adders***



# Introdução

- **Projeto Hierárquico**
  - **Exemplo 1**: somador de números de **4 bits** construído usando um ***half adder*** e ***full adders***
    - **Componentes** internos são primeiramente projetados: **Full Adder** **Half Adder**
    - Em seguida, são **integrados** para construir o somador desejado:



# Introdução

- **Projeto Hierárquico**
  - **Exemplo 1** em **VHDL**: Componentes internos como na aula anterior.

fulladder.vhd

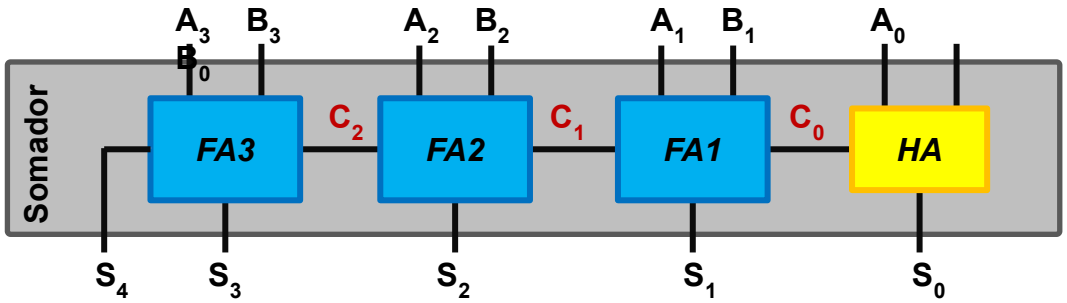
```
library IEEE;
use IEEE.Std_Logic_1164.all;

entity fulladder is
port (A: in std_logic;
      B: in std_logic;
      Cin: in std_logic;
      S: out std_logic;
      Cout: out std_logic
      );
end fulladder;

architecture ...
```


halfadder.vhd

```
library IEEE;
use IEEE.Std_Logic_1164.all;

entity halfadder is
port (A: in std_logic;
      B: in std_logic;
      S: out std_logic;
      Cout: out std_logic
      );
end halfadder;

architecture ...
```

# Introdução

- **Exemplo 1** em **VHDL**:

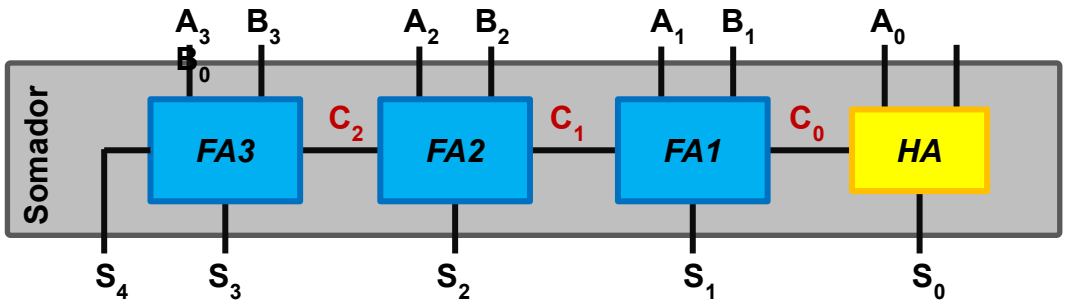



```
library IEEE;
use IEEE.Std_Logic_1164.all;
entity somador is
port (A,B: in std_logic_vector(3 downto 0);
      S: out std_logic_vector(4 downto 0));
end somador;
architecture soma4 of somador is
 signal C0,C1,C2: std_logic;
 component halfadder is
  port (A: in std_logic;
        B: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;
 component fulladder is
  port (A: in std_logic;
        B: in std_logic;
        Cin: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;
```

```
begin
 HA: halfadder port map (A => A(0),
                         B => B(0),
                         S => S(0),
                         Cout => C0);
 FA1: fulladder port map (A => A(1),
                          B => B(1),
                          Cin => C0,
                          S => S(1),
                          Cout => C1);
 FA2: fulladder port map (A => A(2),
                          B => B(2),
                          Cin => C1,
                          S => S(2),
                          Cout => C2);
 FA3: fulladder port map (A => A(3),
                          B => B(3),
                          Cin => C2,
                          S => S(3),
                          Cout => S(4));
end soma4;
```

# Introdução

- **Exemplo 1** em **VHDL**:



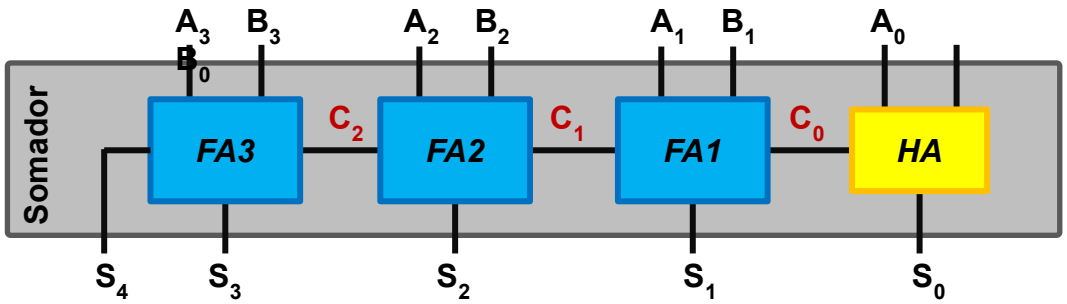

```
library IEEE;
use IEEE.Std_Logic_1164.all;
entity somador is
port (A,B: in std_logic_vector(3 downto 0);
      S: out std_logic_vector(4 downto 0));
end somador;
architecture soma4 of somador is
 signal C0,C1,C2: std_logic;

 component halfadder is
  port (A: in std_logic;
        B: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;

 component fulladder is
  port (A: in std_logic;
        B: in std_logic;
        Cin: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;
```

```
begin
 HA: halfadder port map (A => A(0),
                         B => B(0),
                         S => S(0),
                         Cout => C0);

 FA1: fulladder port map (A => A(1),
                          B => B(1),
                          Cin => C0,
                          S => S(1),
                          Cout => C1);
 FA2:                     A => A(2),
                          B => B(2),
                          Cin => C1,
                          S => S(2),
                          Cout => C2);

 FA3: fulladder port map (A => A(3),
                          B => B(3),
                          Cin => C2,
                          S => S(3),
                          Cout => S(4));

end soma4;
```

**Declarações dos componentes já existentes (de outros arquivos)**

# Introdução

- **Exemplo 1** em **VHDL**:

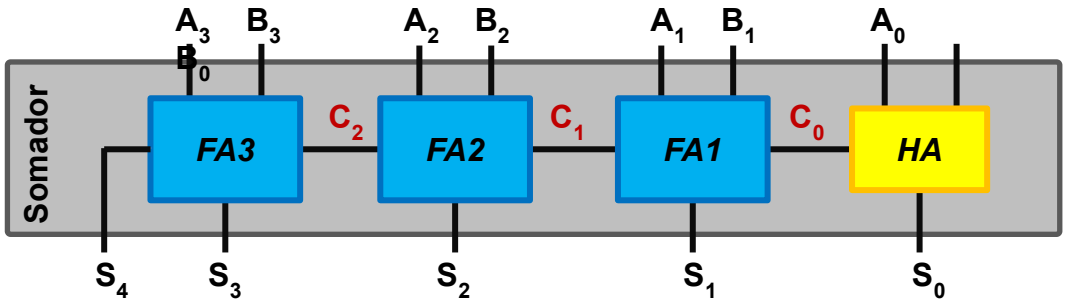



```
library IEEE;
use IEEE.Std_Logic_1164.all;
entity somador is
port (A,B: in std_logic_vector(3 downto 0);
      S: out std_logic_vector(4 downto 0));
end somador;
architecture soma4 of somador is
 signal C0,
 component
  port (A:
        B:
        S:
        Cou
 end compon
 component fulladder is
  port (A: in std_logic;
        B: in std_logic;
        Cin: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;
```

**Descrição das conexões dos componentes (note que múltiplas instâncias de um mesmo componente podem ser utilizadas)**

```
begin
HA: halfadder port map (A => A(0),
                        B => B(0),
                        S => S(0),
                        Cout => C0);
FA1: fulladder port map (A => A(1),
                         B => B(1),
                         Cin => C0,
                         S => S(1),
                         Cout => C1);
FA2: fulladder port map (A => A(2),
                         B => B(2),
                         Cin => C1,
                         S => S(2),
                         Cout => C2);
FA3: fulladder port map (A => A(3),
                         B => B(3),
                         Cin => C2,
                         S => S(3),
                         Cout => S(4));
end soma4;
```

# Introdução

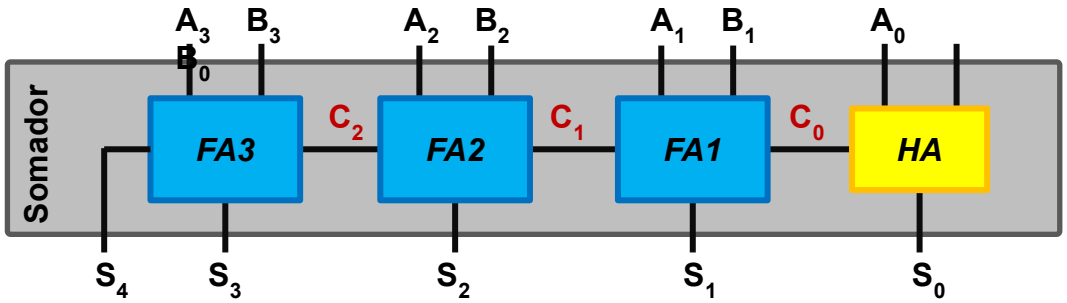



- **Projeto Hierárquico**
  - **Exemplo 1** em **VHDL** (forma alternativa para **port map**):

```
library IEEE;
use IEEE.Std_Logic_1164.all;

entity somador is
port (A,B: in std_logic_vector(3 downto 0);
      S: out std_logic_vector(4 downto 0)
     );
end somador;

architecture soma4 of somador is
 signal C0,C1,C2: std_logic;
 component halfadder is
  port (A: in std_logic;
        B: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;
```

```
 component fulladder is
   port (A: in std_logic;
         B: in std_logic;
         Cin: in std_logic;
         S: out std_logic;
         Cout: out std_logic);
  end component;

 begin
  HA: halfadder port map (A(0),B(0),S(0),C0);
  FA1: fulladder port map (A(1),B(1),C0,S(1),C1);
  FA2: fulladder port map (A(2),B(2),C1,S(2),C2);
  FA3: fulladder port map (A(3),B(3),C2,S(3),S(4));
 end soma4;
```

# Introdução



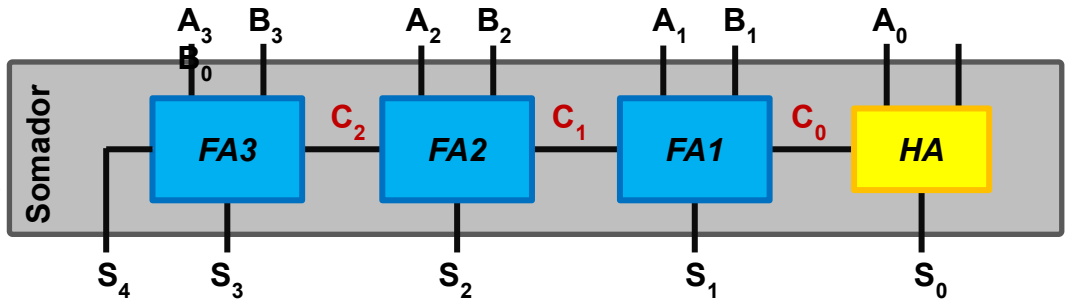

- **Projeto Hierárquico**
  - **Exemplo 1** em **VHDL** (forma alternativa para **port map**):

**Declarações dos componentes já existentes (de outros arquivos)**

```
library IEEE;
use IEEE.Std_Logic_1164.all;

entity somador
port (A,B: in s                    nto 0);
      S: out st                    to 0)
     );
end somador;

architecture soma4 of somador is

 signal C0,C1,C2: std_logic;

 component halfadder is
  port (A: in std_logic;
        B: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;
```

```
 component fulladder is
   port (A: in std_logic;
         B: in std_logic;
         Cin: in std_logic;
         S: out std_logic;
         Cout: out std_logic);
 end component;

begin
  HA: halfadder port map (A(0),B(0),S(0),C0);
  FA1: fulladder port map (A(1),B(1),C0,S(1),C1);
  FA2: fulladder port map (A(2),B(2),C1,S(2),C2);
  FA3: fulladder port map (A(3),B(3),C2,S(3),S(4));
end soma4;
```

# Introdução




- **Projeto Hierárquico**
  - **Exemplo 1** em **VHDL** (forma alternativa para **port map**):

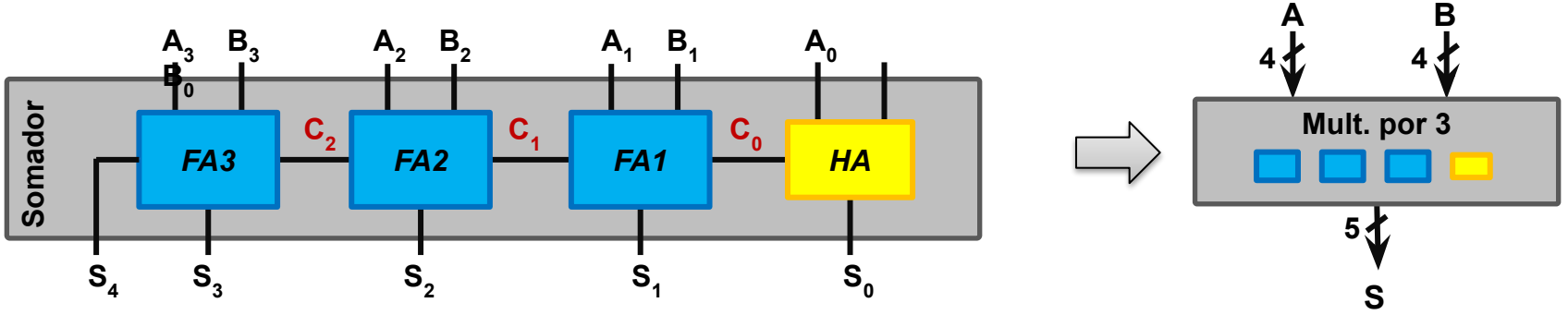
```
library IEEE;
use IEEE.Std_Logic_1164.all;

entity somador is
port (A,B: in std_logic_vector(3 downto 0);
      S: out std_lo                to 0)
     );
end somador;

architecture soma4 of somador is
 signal C0,C1,C2: std_logic;
 component halfadder is
  port (A: in std_logic;
        B: in std_logic;
        S: out std_logic;
        Cout: out std_logic);
 end component;
```

**Descrição das conexões dos componentes**

```
 component fulladder is
   port (A: in std_logic;
         B: in std_logic;
         Cin: in std_logic;
         S: out std_logic;
         Cout: out std_logic);
  end component;

begin
 HA: halfadder port map (A(0),B(0),S(0),C0);
 FA1: fulladder port map (A(1),B(1),C0,S(1),C1);
 FA2: fulladder port map (A(2),B(2),C1,S(2),C2);
 FA3: fulladder port map (A(3),B(3),C2,S(3),S(4));
end soma4;
```

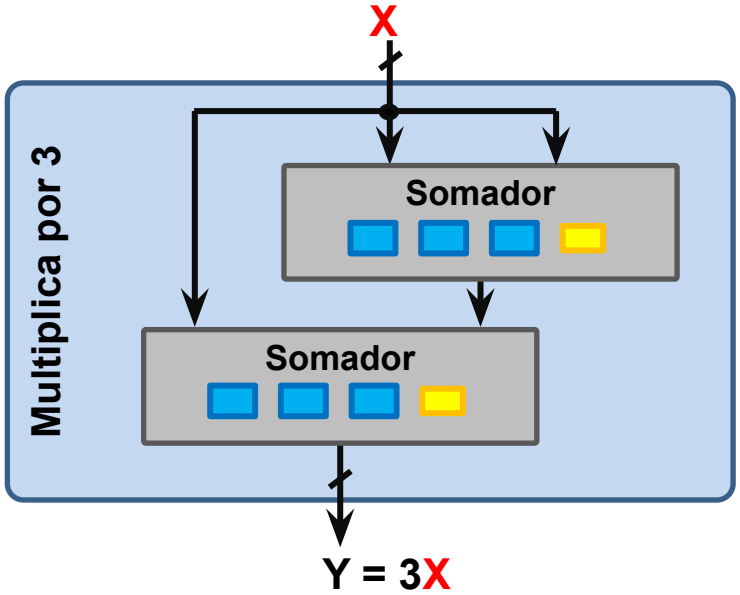
# Introdução

- **Projeto Hierárquico**
  - **Exemplo 2**: multiplicador por 3 construído usando somadores, que por sua vez foram construídos com ***half adder*** e ***full adders***



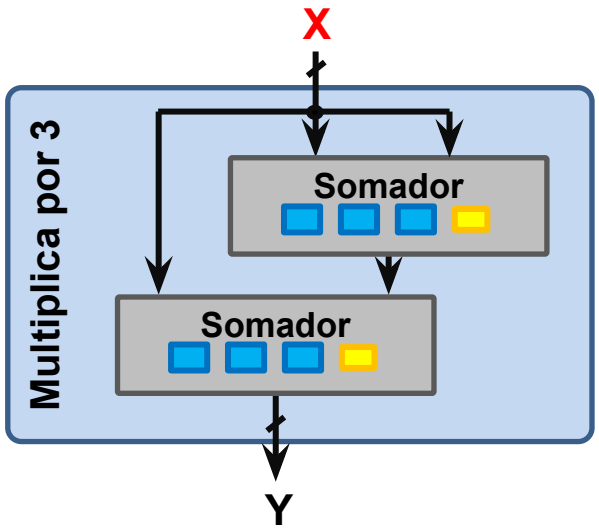
# Introdução

- **Projeto Hierárquico**
  - **Exemplo 2**: multiplicador por 3 construído usando somadores, que por sua vez foram construídos com ***half adder*** e ***full adders***



# Introdução

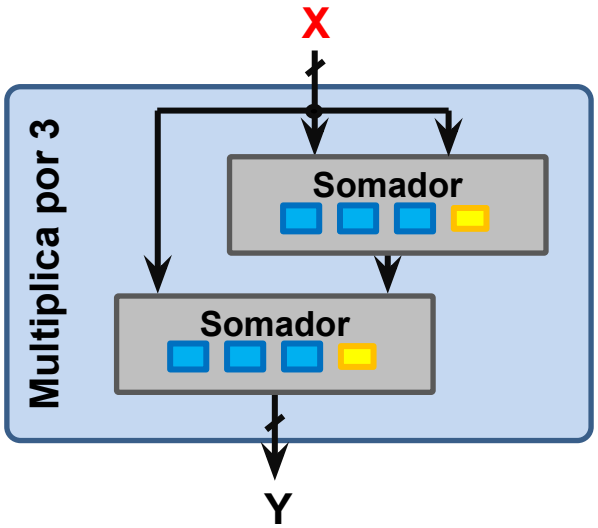
- **Projeto Hierárquico**
  - **Exemplo 2**:




```
library IEEE;
use IEEE.Std_Logic_1164.all;

entity mult3 is
port (X: in std_logic_vector(3 downto 0);
      Y: out std_logic_vector(4 downto 0) );
end mult3;

architecture mult3arch of mult3 is

 signal S: std_logic_vector(4 downto 0);

 component somador is
  port (A,B: in std_logic_vector(3 downto 0);
        S: out std_logic_vector(4 downto 0) );
 end component;

begin

 SUM1: somador port map (A => X,
                         B => X,
                         S => S);

 SUM2: somador port map (A => X,
                         B => S(3 downto 0),
                         S => Y);

end mult3arch;
```

# Introdução

- **Projeto Hierárquico**
  - **Exemplo 2** em **VHDL**:

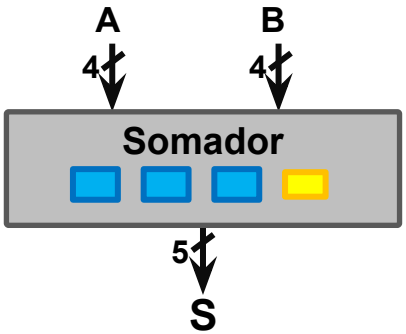



```
library IEEE;
use IEEE.Std_Logic_1164.all;

entity mult3 is
port (X: in std_logic_vector(3 downto 0);
      Y: out std_logic_vector(4 downto 0));
end mult3;

architecture mult3arch of mult3 is

 signal S: std_logic_vector(4 downto 0);

 component somador is
  port (A,B: in std_logic_vector(3 downto 0);
        S: out std_logic_vector(4 downto 0) );
 end component;

begin

 SUM1: somador port map (X, X, S);

 SUM2: somador port map (X, S(3 downto 0), Y);

end mult3arch;
```

# Tarefas

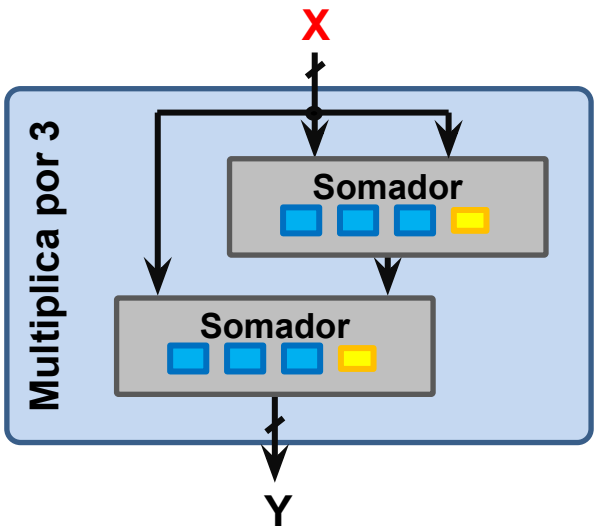
# Tarefa 1

- Implemente o **Somador de 4 Bits** seguindo os seguintes passos:
  - Faça a implementação do **half adder** no arquivo **halfadder.vhd**.
  - Faça a implementação do **full adder** no arquivo **fulladder.vhd**.
  - Faça a implementação do **somador de 4 bits** mostrado anteriormente no arquivo **somador.vhd**.
  - Finalmente, crie um arquivo **usertop.vhd**,contendo uma entity **usertop**, visandoconectar as **chaves** e **LEDRs** do kit/emulador ao **somador de 4 bits**.
    - **Ports** devem ser **SW** e **LEDR** com 18 bits.
    - Declarar **component somador** para poder usá-lo.
    - Fazer conexões via **port map**.



# Tarefa 2

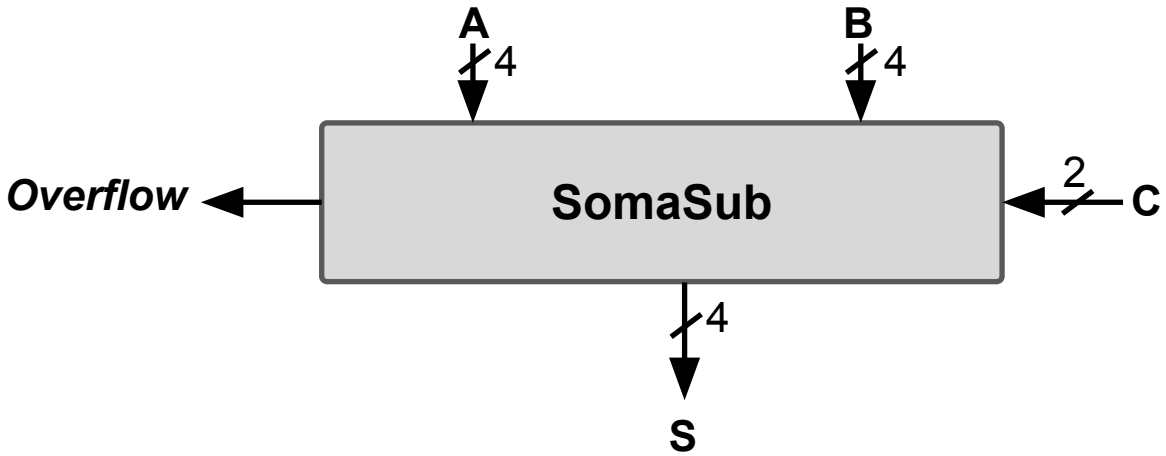
- Faça a implementação do **Multiplicador por 3** discutido anteriormente.
  - Para tal, passos similares aos usados para o **Somador de 4 Bits** devem ser seguidos.
  - Com o circuito e funcionamento, observe que esse multiplicador por 3 produz resultados incorretos para **X > 7**. **Por quê?**



# Tarefa 3

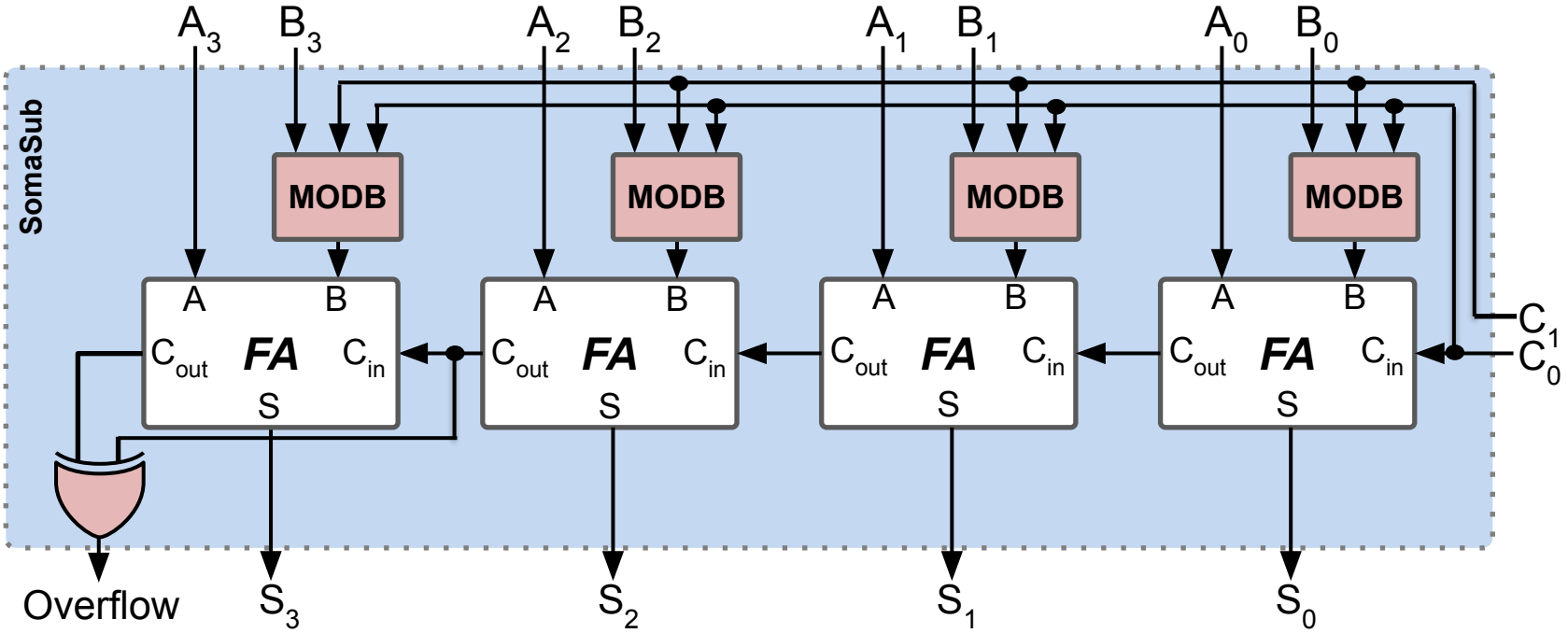
- Implementar um **circuito somador/subtrator de 4 bits** capaz de realizar **soma** ou **subtração** com dois operandos, ou ainda **incremento** e **decremento** de um dos operandos.

| C | Operação |
|---|---|
| 0 0 | A + B |
| 0 1 | A + 1 |
| 1 0 | A – 1 |
| 1 1 | A – B |



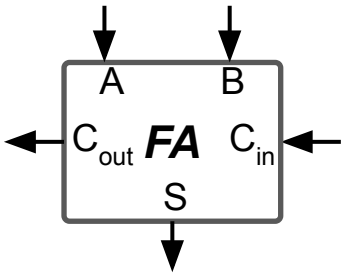
# Tarefa 3

- **Somador/subtrator** a ser implementado:
  (***FA** = **Full Adder***)

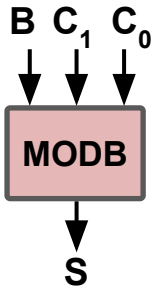


# Tarefa 3

- Implementação do **Somador/subtrator**
  - Primeiramente implementar **componentes** básicos:



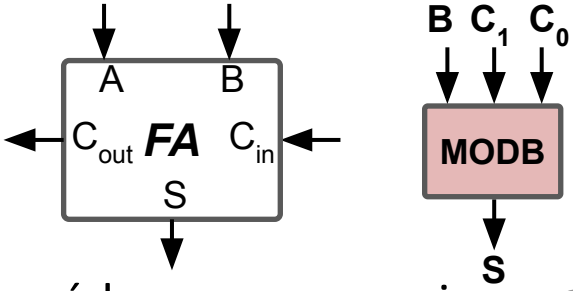

| A B $C_{in}$ | S | $C_{out}$ |
|---|---|---|
| 0 0 0 | 0 | 0 |
| 0 0 1 | 1 | 0 |
| 0 1 0 | 1 | 0 |
| 0 1 1 | 0 | 1 |
| 1 0 0 | 1 | 0 |
| 1 0 1 | 0 | 1 |
| 1 1 0 | 0 | 1 |
| 1 1 1 | 1 | 1 |



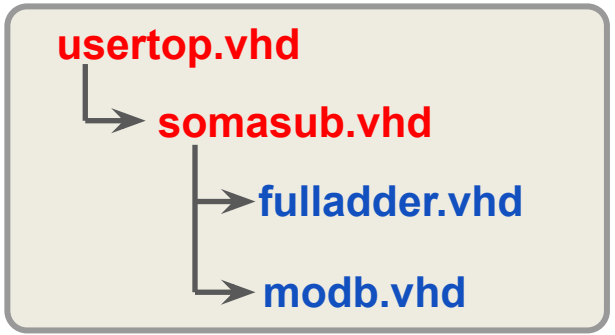

| $C_1$ $C_0$ B | S |
|---|---|
| 0 0 0 | 0 |
| 0 0 1 | 1 |
| 0 1 0 | 0 |
| 0 1 1 | 0 |
| 1 0 0 | 1 |
| 1 0 1 | 1 |
| 1 1 0 | 1 |
| 1 1 1 | 0 |

# Tarefa 3

- Implementação do **Somador/subtrator**
  - Primeiramente implementar **componentes** básicos:




  - Em seguida, integrá-los em um arquivo **somasub.vhd**.
  - Finalmente, usar **usertop.vhd** para conectar o **somasub** com as chaves e leds do kit.



## Sugestão de Estudo Avançado

- Implemente um **somador de números de 8 bits** usando um **half adder** e **full adders**. Para tal, pesquise sobre a estrutura **for generate** disponível em **VHDL** e use-a para fazer essa implementação.