INF01 118

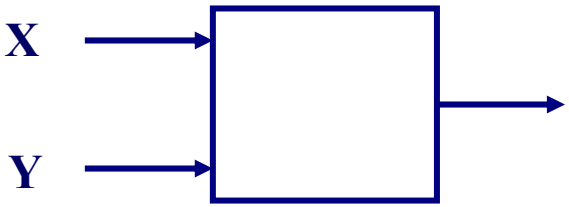
Técnicas Digitais para Computação

Comparadores
ULA Multifunção

UFRGS

Aula 13

Comparador

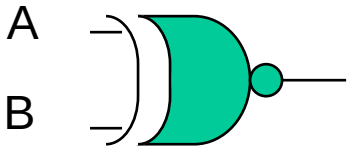


COMP = 1 se X = Y

COMP = 0 se X ≠ Y

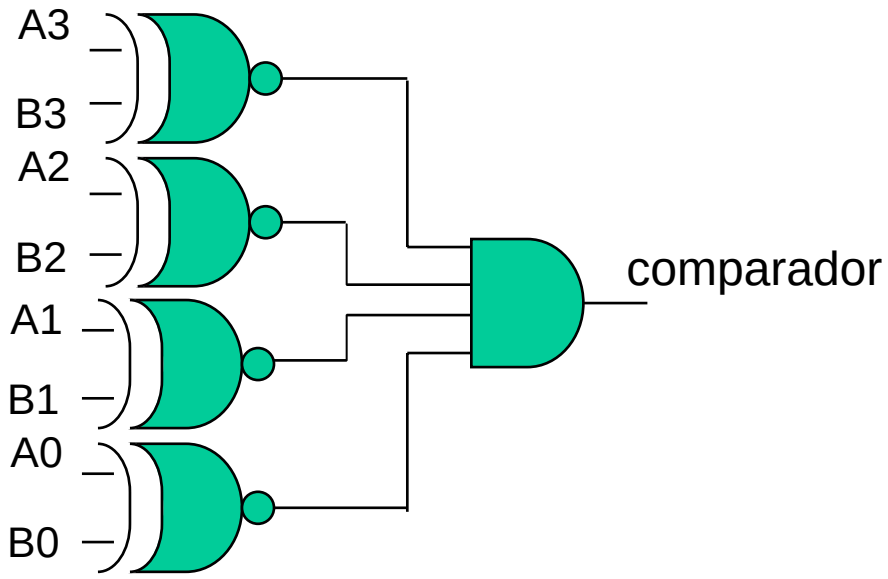
X	Y	COMP
0	0	1
0	1	0
1	0	0
1	1	1

$$COMP = X\,Y + \overline{X}\,\overline{Y} = \overline{X \oplus Y}$$



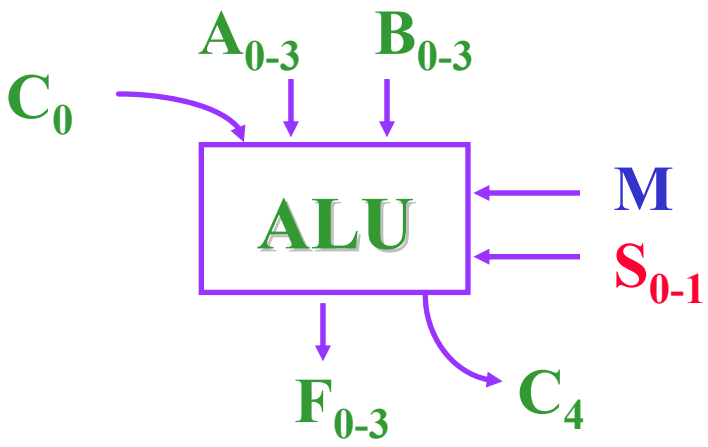
Comparação

Comparador de 4 bits (A3A2A1A0 e B3B2B1B0)



Projeto de Unidade Aritmética e Lógica

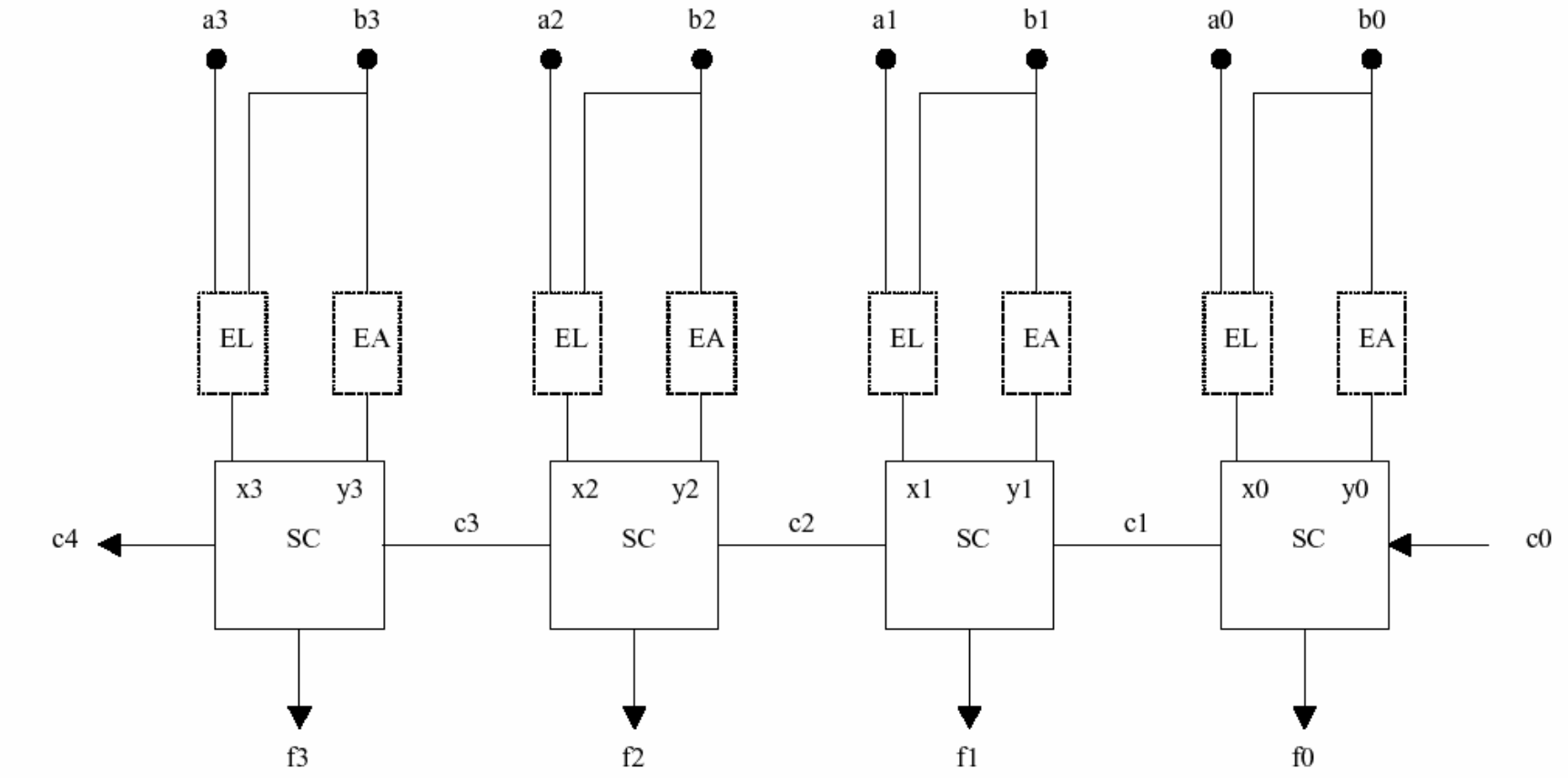
Exemplo de uma ULA Simples



M = Modo
- 1 funções aritméticas
- 0 funções lógicas

S = Seleção da função

Projeto de Unidade Aritmética e Lógica



Projeto de Unidade Aritmética e Lógica

M	S1	S0	nome da função	F	X	Y	C0
0	0	0	complementa	A’	A’	0	0
0	0	1	E	A E B	A E B	0	0
0	1	0	identidade	A	A	0	0
0	1	1	OU	A OU B	A OU B	0	0
1	0	0	decrementa	A-1	A	todos 1s	0
1	0	1	soma	A+B	A	B	0
1	1	0	subtrai	A+B’+1	A	B’	1
1	1	1	incrementa	A+1	A	todos 0s	1

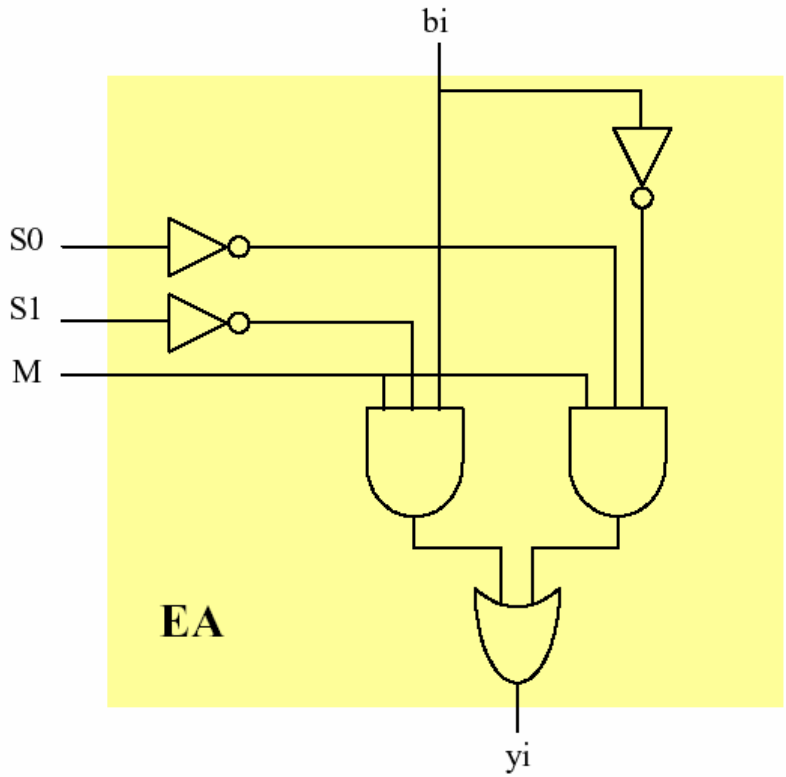
Projeto de Unidade Aritmética e Lógica

Tabela 2 – Tabela-verdade para o extensor aritmético.

S1	S0	bi	yi
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Tabela 3 – Mapa de Karnaugh para o extensor aritmético.

S1S0 bi	00	01	11	10
0	**1**	0	0	**1**
1	**1**	**1**	0	0



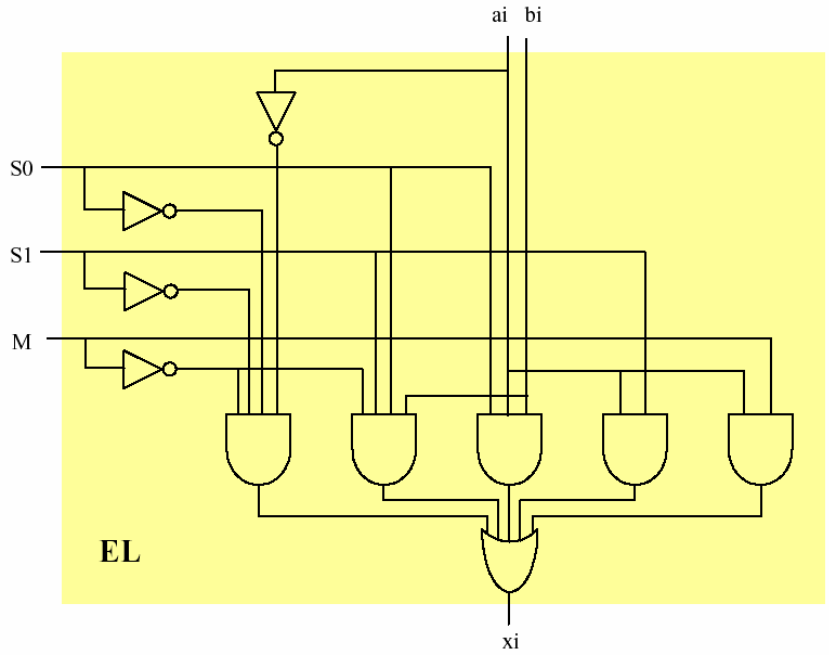
Projeto de Unidade Aritmética e Lógica

Tabela 4 – Tabela-verdade para o extensor lógico.

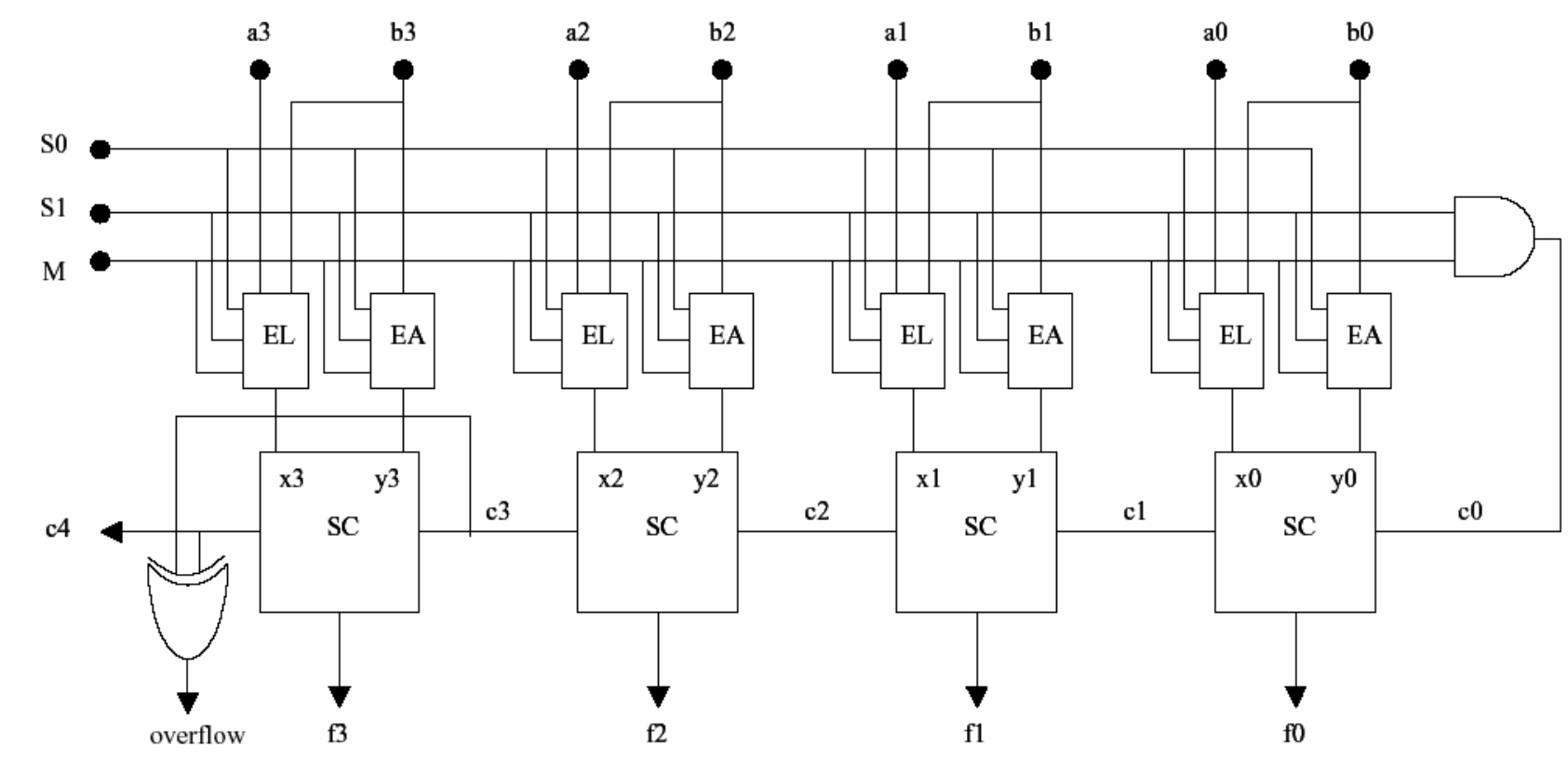
M	S1	S0	xi
0	0	0	ai'
0	0	1	ai.bi
0	1	0	ai
0	1	1	ai+bi
1	?	?	ai

Tabela 5 – Mapa de Karnaugh para o extensor aritmético.

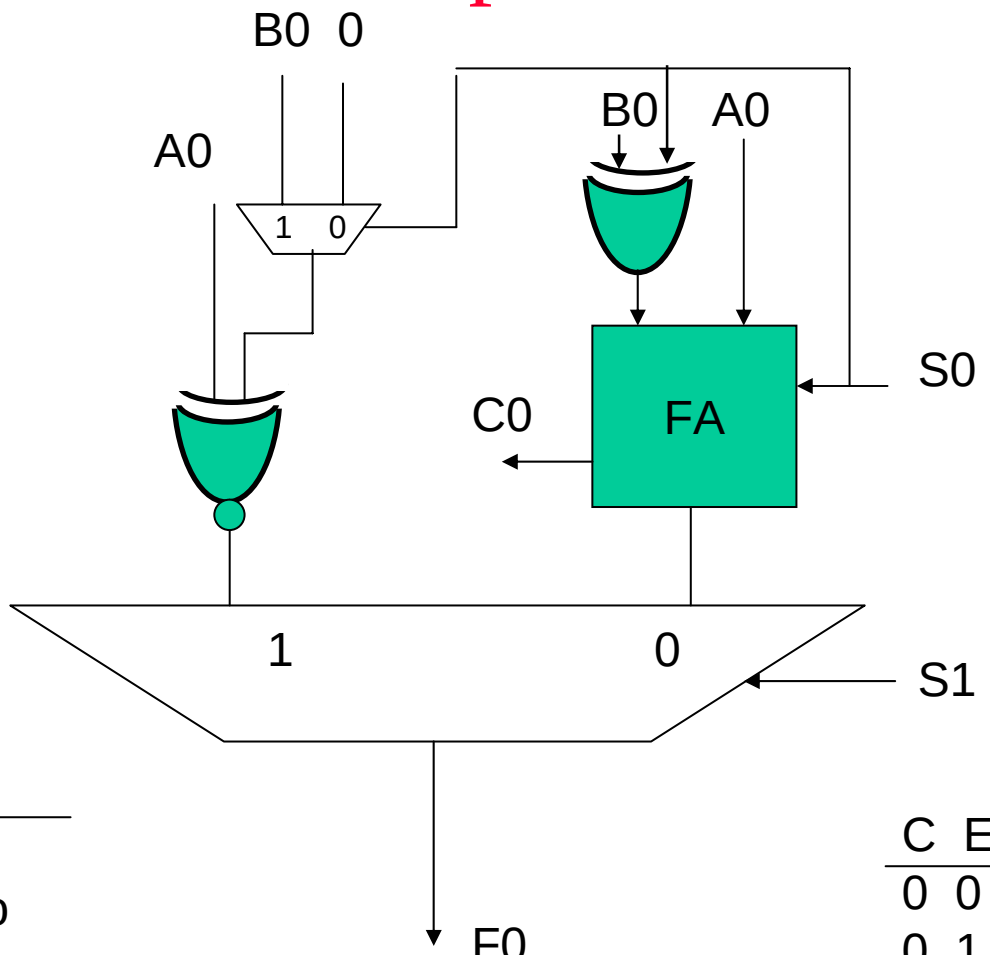
	M=0				M=1			
S1S0 / ai bi	00	01	11	10	00	01	11	10
00	**1**							
01	**1**		**1**					
11		**1**	**1**	**1**	**1**	**1**	**1**	**1**
10			**1**	**1**	**1**	**1**	**1**	**1**



Projeto de Unidade Aritmética e Lógica



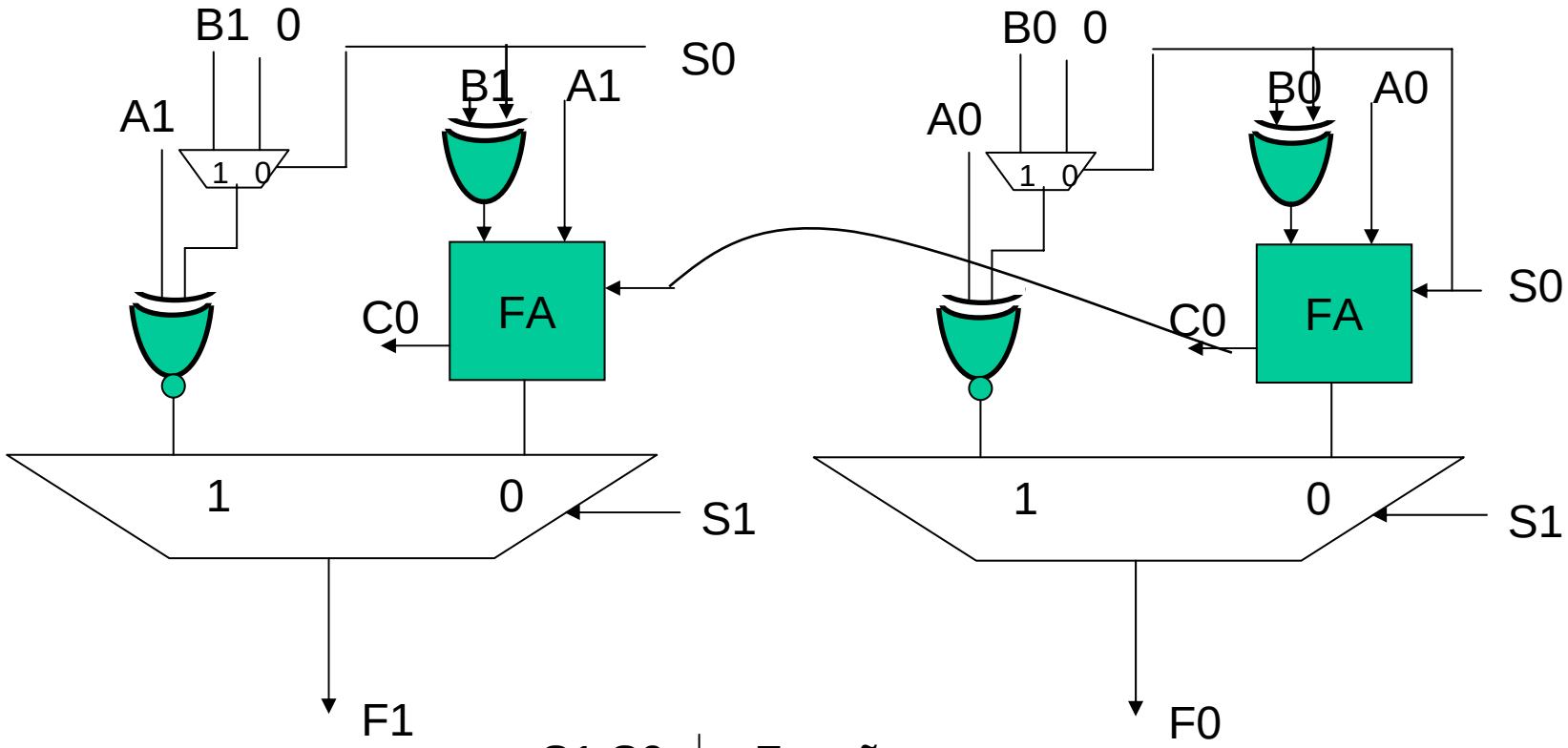
Projeto de ULA de 1 bit
Alternativa de implementacao 2



S1	S0	Função
0	0	soma
0	1	subtração
1	0	inversão
1	1	comparação

C	E	XOR	XNOR
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

ULA de 2 bits



S1	S0	Função
0	0	soma
0	1	subtração
1	0	inversão
1	1	comparação

ULA em VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ULA is
    Port ( A : in  STD_LOGIC_VECTOR (3 downto 0);
           B : in  STD_LOGIC_VECTOR (3 downto 0);
           C : in  STD_LOGIC_VECTOR (1 downto 0);
           F : out  STD_LOGIC_VECTOR (3 downto 0));
end ULA;

architecture Behavioral of ULA is
Begin

process (A, B, C)
begin
CASE C is
WHEN "00" => F <= A+B;
WHEN "01" => F <= A-B;
WHEN "10" => F <= A xor B;
WHEN others => F <= not A;
END CASE;
end process;

end Behavioral;
```