Universidade Federal de Santa Catarina
Centro Tecnológico – CTC
Departamento de Engenharia Elétrica

http://gse.ufsc.br

“EEL7020 – Sistemas Digitais”

Prof. Eduardo Augusto Bezerra

Eduardo.Bezerra@eel.ufsc.br

Florianópolis, marco de 2013.

Sistemas Digitais

Uso de FSMs no controle do fluxo de execução de sistemas digitais.

Estudo de caso: Projeto de Calculadora.

Objetivos do laboratório

1. Entender o uso de FSMs como estrutura de controle do fluxo de operações de um circuito combinacional.
2. Implementar uma FSM em VHDL para controlar as operações da calculadora desenvolvida nas aulas anteriores.

Tarefa a ser realizada na aula prática

Definição do problema: calculadora com entrada de dados reduzida

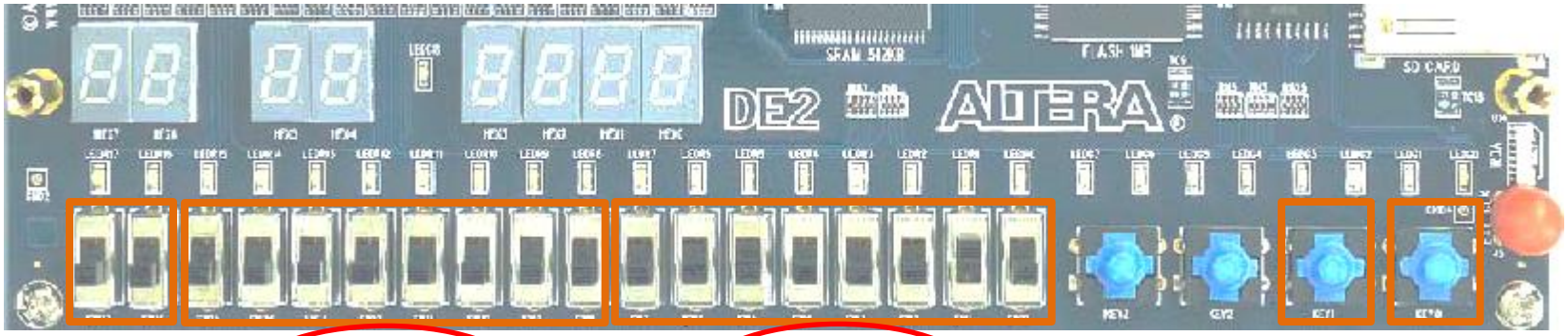
- Para realizar as operações, alguns componentes da calculadora desenvolvida nas aulas anteriores utilizam:
  - as chaves SW(7..0) para leitura do operando A, e
  - as chaves SW(15..8) para leitura do operando B.

- Visando reduzir a quantidade de chaves no projeto de uma placa, solicita-se utilizar apenas um conjunto de 8 chaves, SW(7..0), para leitura dos dois operandos.

- A seleção da operação desejada continua sendo realizada pelas chaves SW(17..16).

Entrada de dados para realizar uma soma

Entrada de dados na calculadora original (aulas anteriores):



Operação *SW(17..16)*

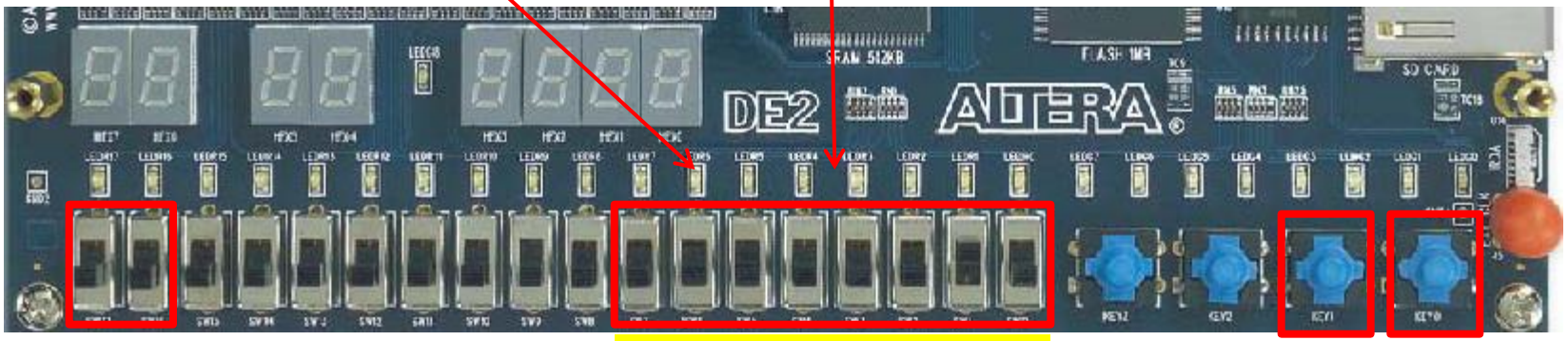
Operando B *SW(15..8)*

Operando A *SW(7..0)*

Enter *Key(1)*

Reset *Key(0)*

Entrada de dados na nova calculadora modificada:



Operação ***SW(17..16)***

Operando A
Operando B
SW(7..0)

Enter ***Key(1)***

Reset ***Key(0)***

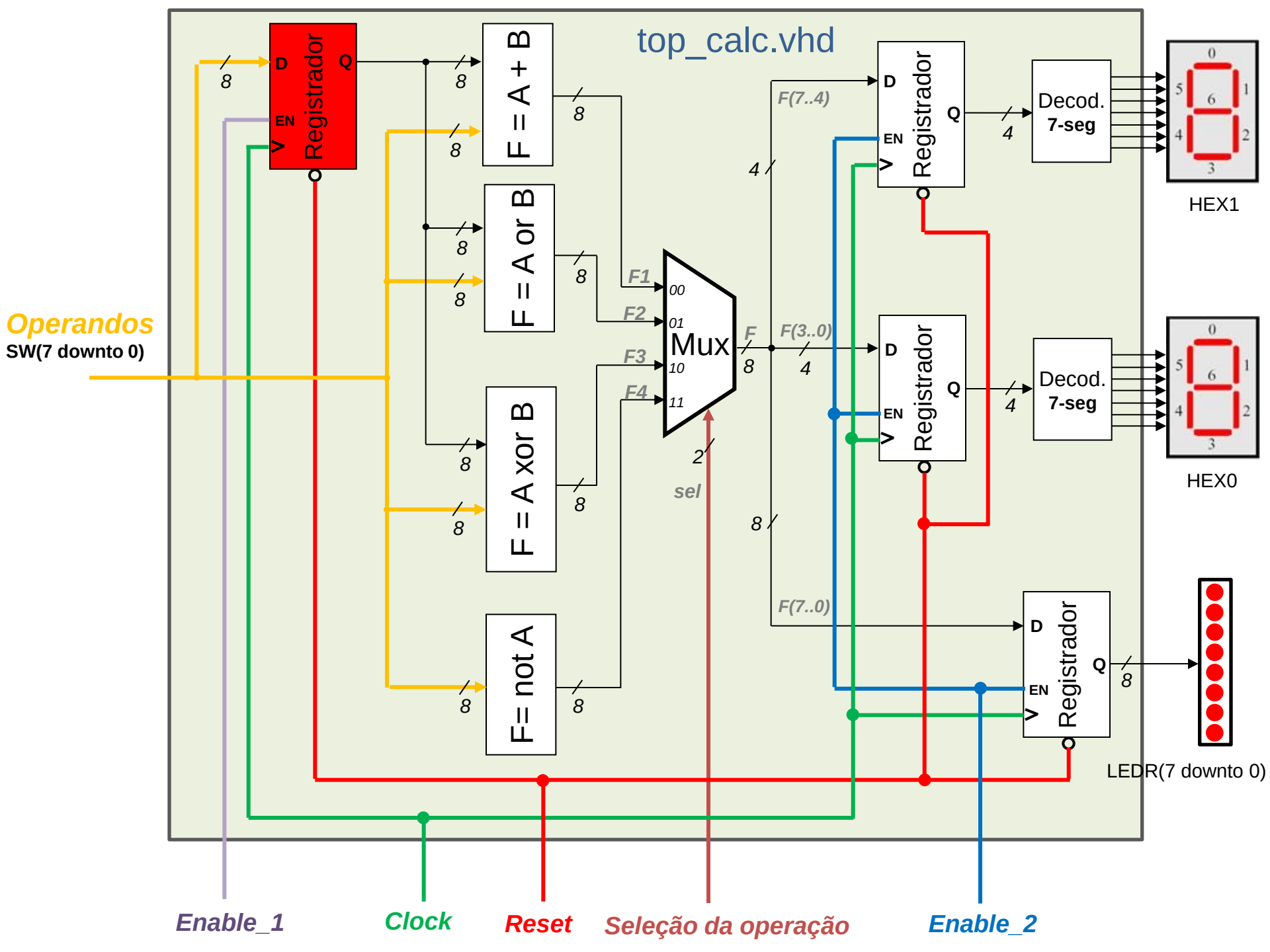
Novo Registrador e FSM para controle do fluxo de operações

• No slide a seguir, um registrador (em vermelho) é adicionado na entrada para armazenar o primeiro operando para operações que necessitam dois operandos.

• Algumas observações:

- Não é necessário alterar o VHDL de nenhum componente.
- No arquivo “top_calc.vhd”, deve ser incluído um novo componente Registrador, baseado nos existentes.
- Devem ser realizadas alterações em alguns *port maps*, visando a conexão do novo registrador aos demais componentes existentes, e também às entradas.

- Notar também que os sinais de habilitação de registradores, seleção de multiplexador, entre outros, foram desconectados, visando a inclusão de um controle automático do fluxo de operações por intermédio de uma FSM.


top_calc.vhd
Operandos
SW(7 downto 0)
Registrador
D
Q
EN
F = A + B
F = A or B
F = A xor B
F= not A
F1
F2
F3
F4
00
01
10
11
Mux
sel
F
F(7..4)
F(3..0)
F(7..0)
Decod.
7-seg
HEX1
HEX0
LEDR(7 downto 0)
Enable_1
Clock
Reset
Seleção da operação
Enable_2

FSM para controle do fluxo de operações da calculadora

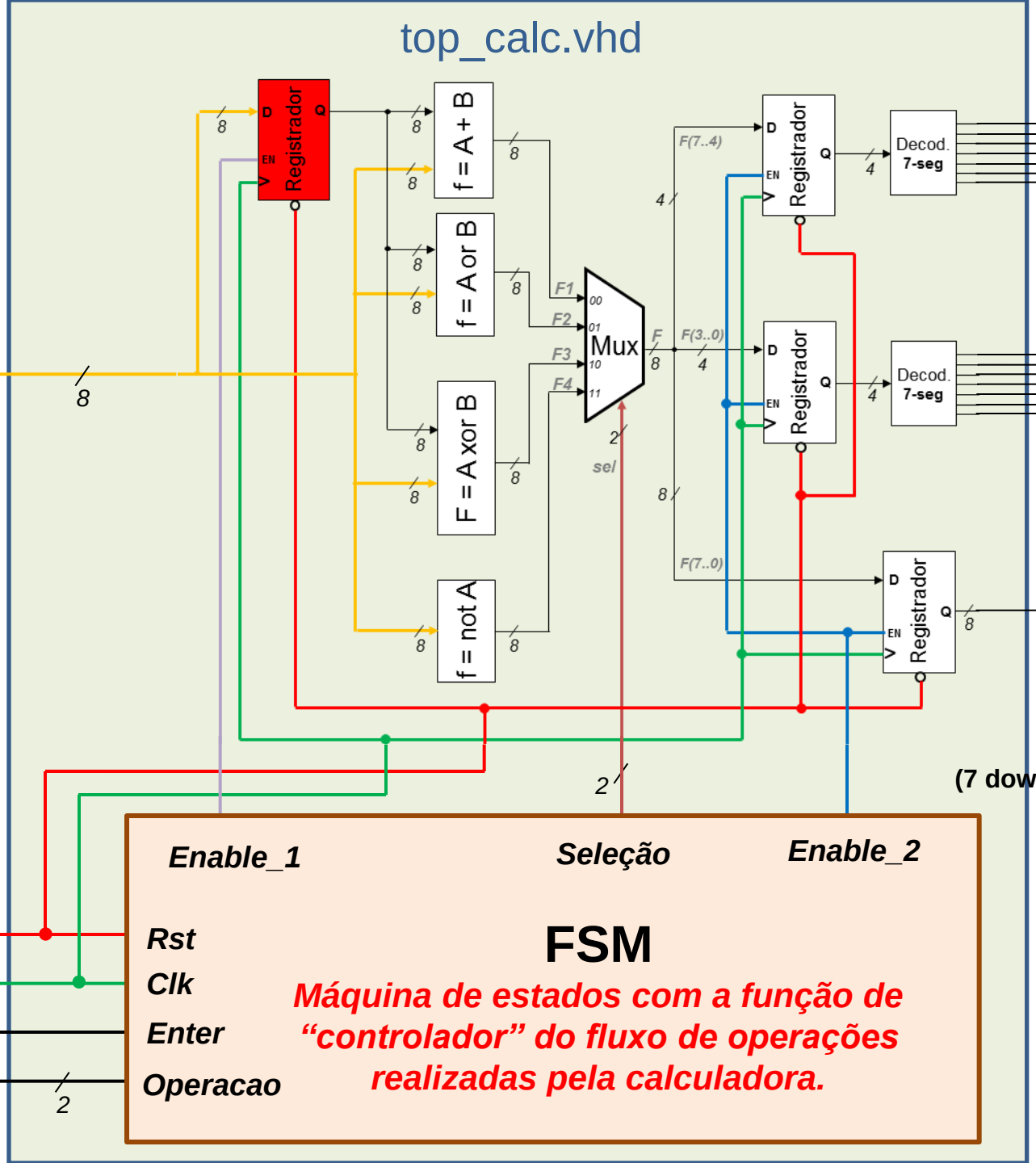
• No slide a seguir é apresentado o diagrama de blocos do circuito, incluindo o controlador (FSM) para gerenciar o fluxo de operações.

• Sugestão de interface para o novo componente (FSM):

```
component FSMctrl
    port (
            Clk, Rst, Enter          : IN STD_LOGIC;
            Operacao                 : IN STD_LOGIC_VECTOR(1 downto 0);
            Selecao                  : OUT STD_LOGIC_VECTOR(1 downto 0);
            Enable_1, Enable_2 : OUT STD_LOGIC
  );
  end component;
```

• O top_calc.vhd passará a ter 12 componentes instanciados! (9 "*Components*" no total).

TOPO: SUGESTÃO I



Funcionamento da FSM (controlador)

• No estado inicial, “Esperando”, as saídas da FSM estão desabilitadas (*Enable_1*=‘0’, *Enable_2*=‘0’, *Selecao*=“XX”).

• Isso garante que nenhuma atividade ocorrerá na calculadora, enquanto o usuário não fornecer os operandos e a operação.

• Quando *Enter* (*Key(1)*) for ‘0’, a FSM avança para o próximo estado, para aguardar até o *Enter* voltar para ‘1’ (botão “não pressionado”).

• No estado “Operação” é realizada uma transição para o próximo estado, de acordo com a operação selecionada.

• Se a operação for “*not A*” (apenas um operando), a operação é realizada, o resultado é armazenado no registrador de saída (Enable_2=‘1’) e a FSM retorna para o estado inicial “Esperando”.

• Para as demais operações, a leitura do segundo operando é realizada em um estado adicional - *Enable_1 e Enable_2 precisam ser devidamente controlados em cada estado*.

• Para essas operações, quando o *Enter* for pressionado pela segunda vez (*Key(1) = ‘0’*), o resultado é escrito nos registradores de saída (Enable_1=‘1’), e a FSM retorna para o estado inicial “Esperando”.

Sugestão de FSM

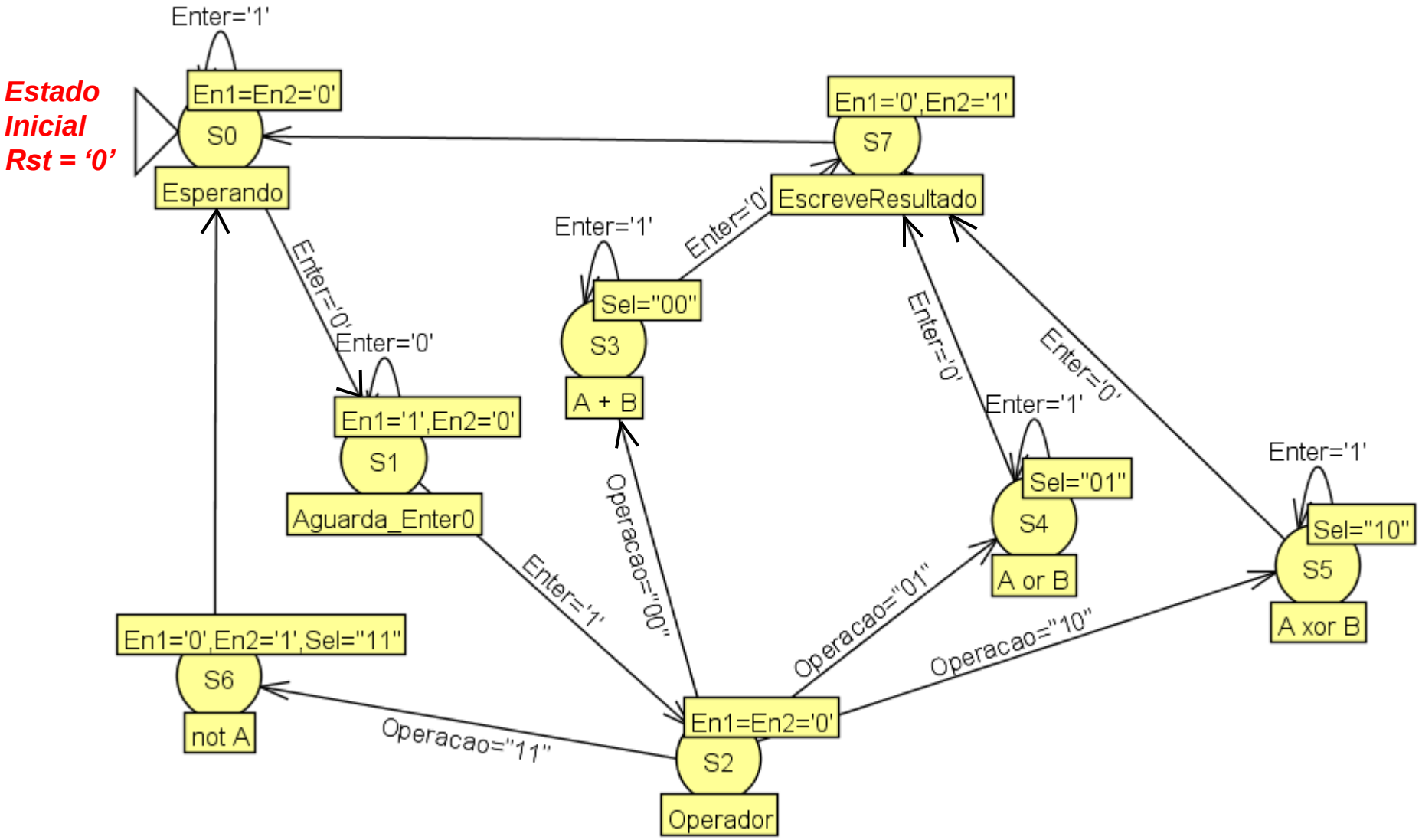


Tabela de transição de estados

Entradas				Saídas			
Reset	Enter	Operação	EA	PE	Enable_1	Seleção	Enable_2
0	X	XX	S0	S0	0	00	0
1	1	XX	S0	S0	0	00	0
1	0	XX	S0	S1	0	00	0
1	0	XX	S1	S1	1	00	0
1	1	XX	S1	S2	1	00	0
1	X	00	S2	S3	0	00	0
1	X	01	S2	S4	0	00	0
1	X	10	S2	S5	0	00	0
1	X	11	S2	S6	0	00	0
1	0	XX	S3	S7	0	00	0
1	1	XX	S3	S3	0	00	0
1	0	XX	S4	S7	0	01	0
1	1	XX	S4	S4	0	01	0
1	0	XX	S5	S7	0	10	0
1	1	XX	S5	S5	0	10	0
1	X	XX	S6	S0	0	11	1
1	X	XX	S7	S0	0	00	1

Sugestão de FSM em VHDL

```
library  ieee; use  ieee.std_logic_1164.all;
entity FSMctrl is
port ( Clk, Rst, Enter : in std_logic;
 Operacao: in std_logic_vector(1 downto 0);
 Selecao: out std_logic_vector(1 downto
 Enable_1, Enable_2: out std_logic
);
end FSMctr;
architecture FSM_beh of FSMctrl is
 type states is (S0, S1, S2, S3, S4, S5,
 signal EA, PE: states;
 signal clock: std_logic;
 signal reset: std_logic;
begin
 clock <= Clk;
 reset <= Rst;
```

```
P1: process (clock, reset)
 begin
  if reset = '0' then
        EA <= S0;
  elsif clock'event and clock = '1' then
        EA <= PE;
  end if;
 end process;
```

Essa architecture continua na coluna da direita ...

```
P2: process (EA, Enter)
begin
 case EA is
  when S0 =>
     if Enter = '1' then
       PE <= S0;
     else
       PE <= S1;
     end if;
     Enable_1 <= '0';
     Enable_2 <= '0';
  when S1 =>  ... (a fazer)
  when S2 =>      -- Operador
     Enable_1 <= '0';
     Enable_2 <= '0';
     if Operacao = "00" then
        PE <= S3;  -- Fazer soma
     elsif Operacao = "01" then
        PE <= S4;  -- Fazer OR
     elsif ... (a fazer)
   end case;
 end process;
end FSM_beh; -- fim da architecture
```

Controle do fluxo de operações da calculadora

- Para utilizar a calculadora é necessário:

1. Selecionar a operação desejada nas chaves SW(17..16).
2. Fornecer um valor nas chaves SW(7..0) - operando A.
3. Pressionar *Enter* - o botao KEY(1) será ‘0’ quando pressionado.
4. Se for a operação *“not A”*, o resultado será apresentado nos displays de 7-segmentos e LEDs.
5. Se for operação de *soma*, *xor* ou *or*, fornecer o segundo operando nas chaves SW(7..0), e pressionar *Enter*.
6. Após apresentado o resultado, essa sequência é reiniciada, voltando ao passo 1.

IMPLEMENTAÇÃO ALTERNATIVA DO TOPO!

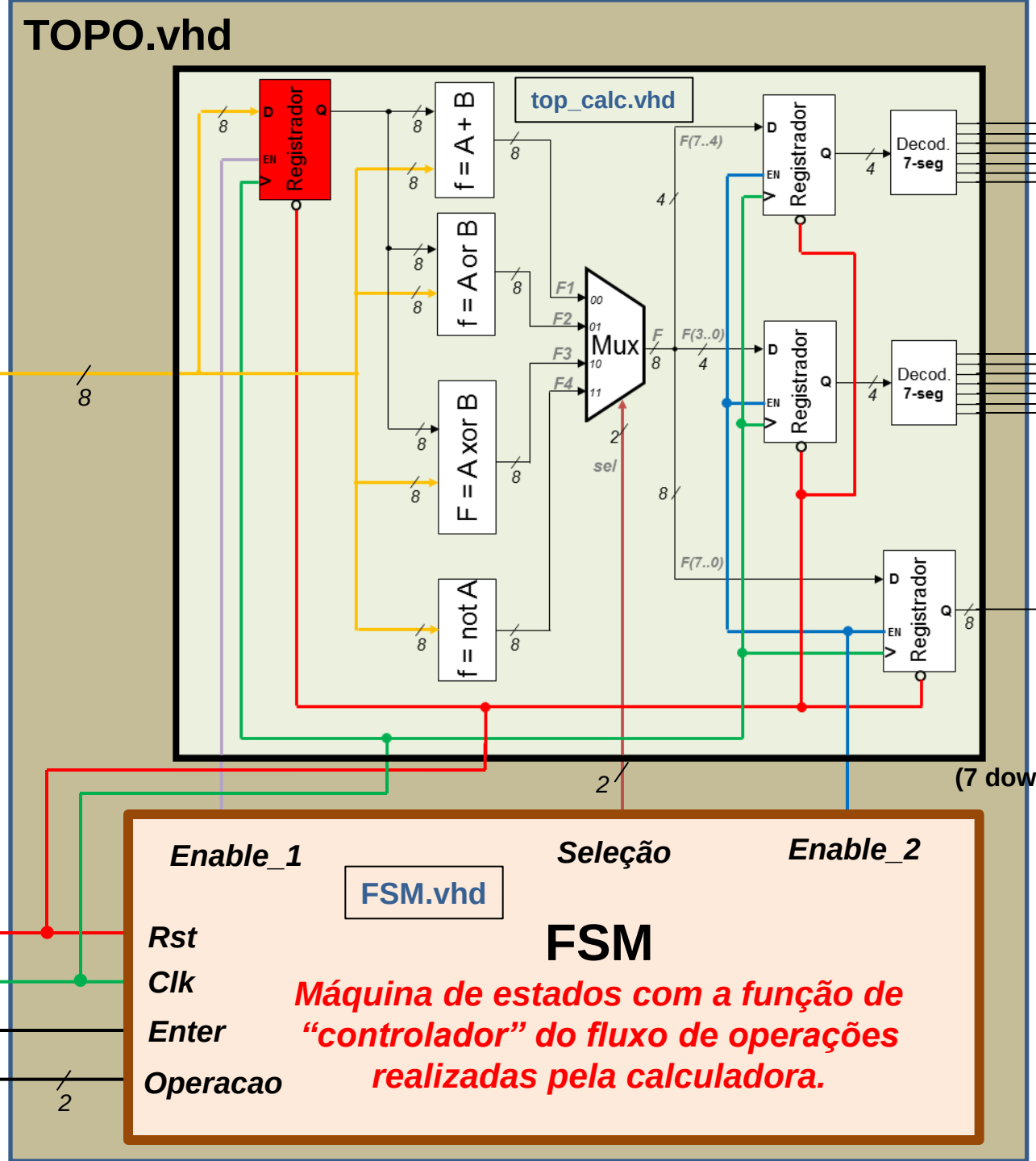
Outra opção de implementação seria criar um novo TOPO (um novo arquivo TOPO.vhd), contendo apenas dois componentes:

- A FSM (arquivo FSM.vhd)
- Os demais componentes (arquivo top_calc.vhd)

Essa alternativa de implementação está apresentada no diagrama de blocos a seguir.

Notar que nessa implementação alternativa, pode ser necessário renomear os sinais de entrada e saída do top_calc.vhd, para evitar o uso de SW, LEDR, HEX0 e HEX1, que passarão a ser os integrantes da nova entity do novo arquivo TOPO.vhd

TOPO: SUGESTÃO II



TOPO: SUGESTÃO II