Lógica e Sistemas Digitais

Memórias ROM e RAM

Controlo Microprogramado

João Pedro Patriarca (jpatri@cc.isel.ipl.pt)

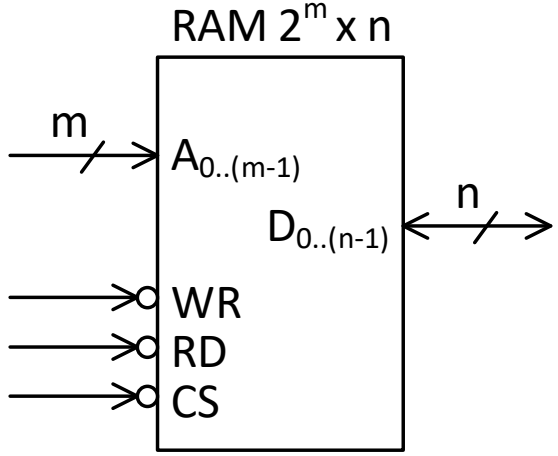
Slides inspirados nos slides do prof. Mário Véstias

Memória

- Uma memória é um dispositivo capaz de armazenar um conjunto de palavras (uma palavra é um conjunto de bits)
- Dois tipos de memórias: voláteis; não voláteis
  - Nas memórias voláteis os dados permanecem armazenados apenas enquanto alimentadas (ex: RAM)
  - Nas memórias não voláteis, os dados persistem mesmo com interrupções na alimentação (ex: ROM, Flash)
- Ao nível da interface física apresenta, tipicamente, um barramento de endereços para identificar a palavra, um barramento de dados para interface com a palavra e um barramento de controlo para controlar o acesso
  - O número de bits do endereço depende do número de palavras da memória
  - O número de bits de dados corresponde à dimensão de cada palavra
  - O número de bits de controlo depende, essencialmente, do tipo de memória

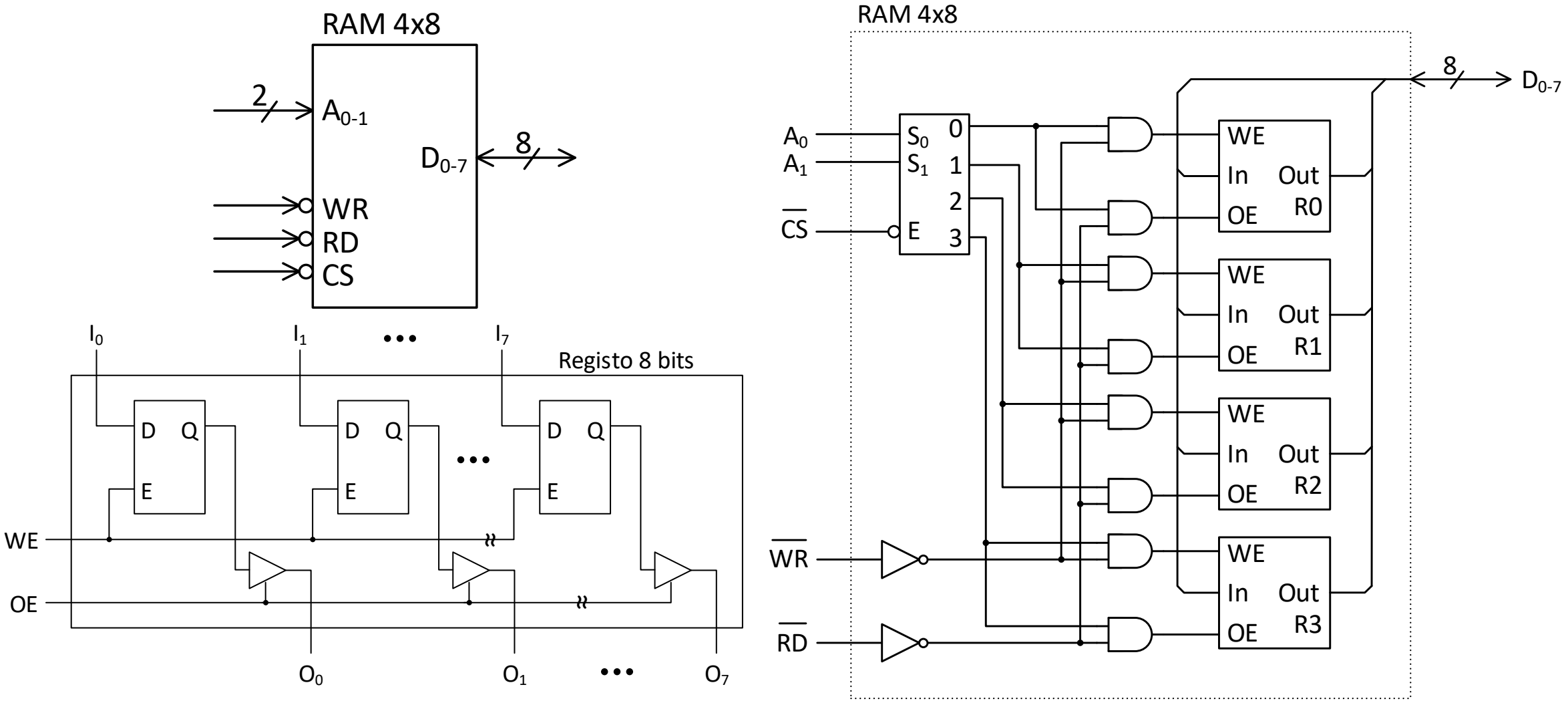
RAM – *Random Access Memory*

- Cada registo é identificado pelo endereço (A – *Address*)
- Barramento de dados bidirecional para interface com os dados (D – *Data*)
- Acessos de leitura e de escrita (RD – *Read* e WR – *Write*, respetivamente)
- Controlo de seleção da unidade (CS – *Chip Select*): quando desativo, os acessos de leitura e de escrita não são consequentes



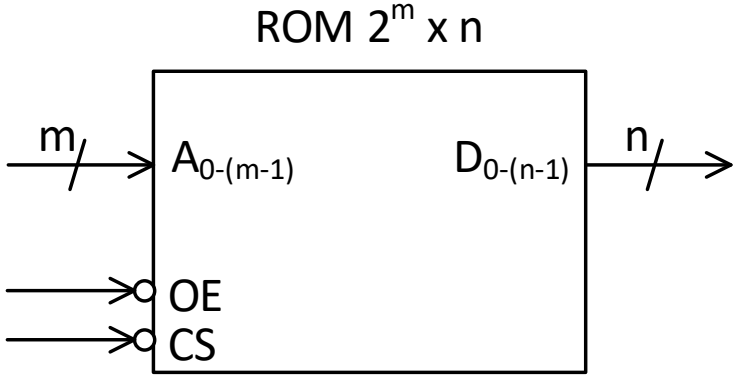
- Possível arquitetura interna considerando uma RAM estática assíncrona, constituída por 4 registos de 8 bits cada (próxima página)

RAM – *Random Access Memory* – arquitetura interna



ROM – *Read Only Memory*

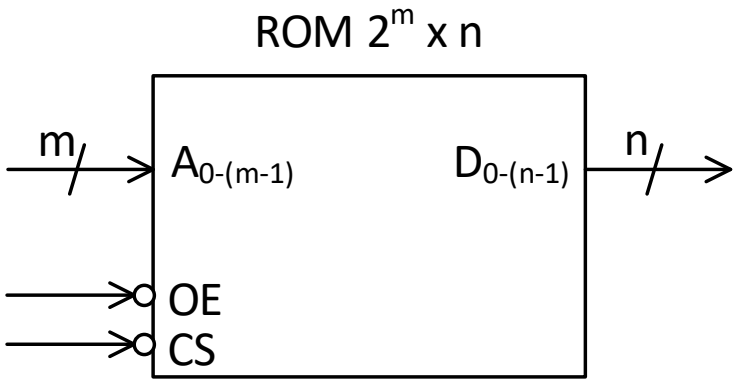
- Cada palavra é identificada pelo endereço (A – *Address*)
- Barramento de dados unidirecional (D – *Data*)
- Permite apenas acessos de leitura (OE – *Output Enable*)
  - Necessário programar a ROM previamente

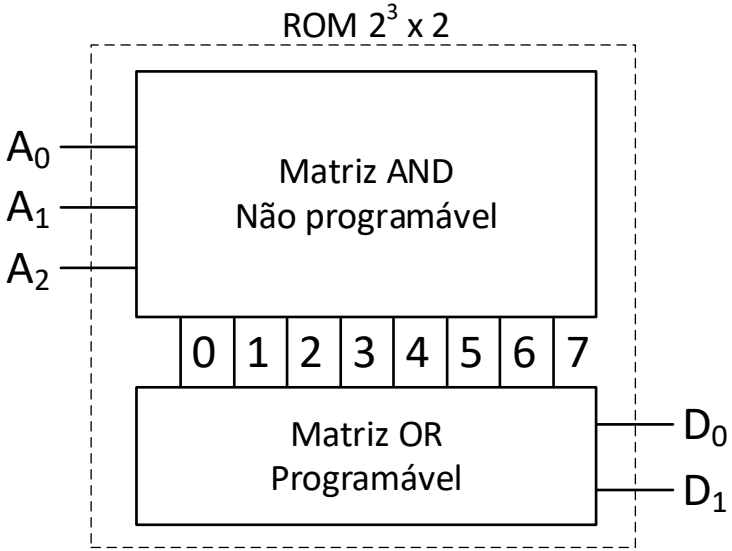


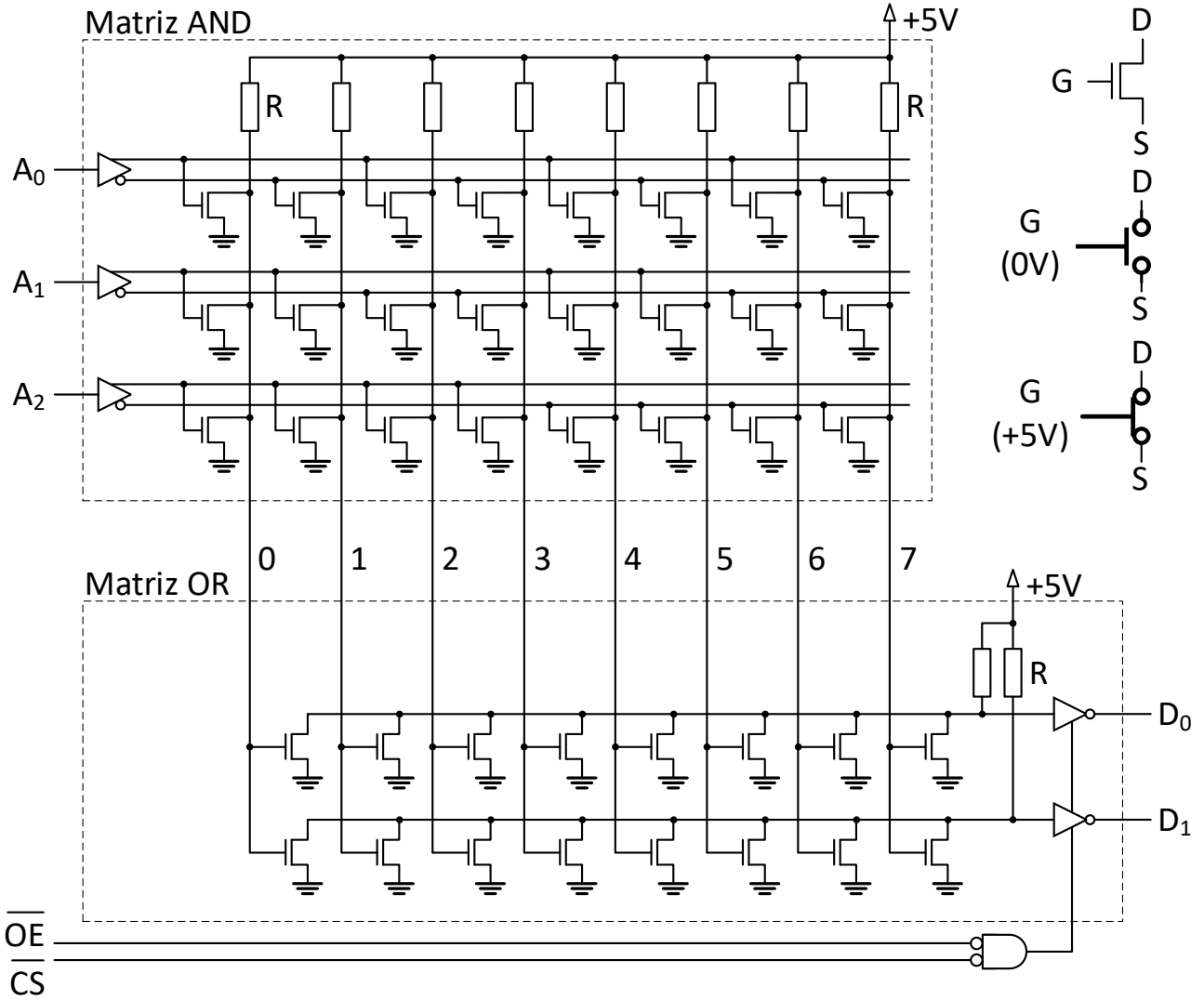
- Controlo de seleção da unidade (CS – *Chip Select*): quando desativo, os acessos de leitura não são consequentes

- Possível arquitetura interna considerando uma ROM constituída por 8 palavras de 2 bits cada (próxima página)

ROM – *Read Only Memory – arquitetura interna*



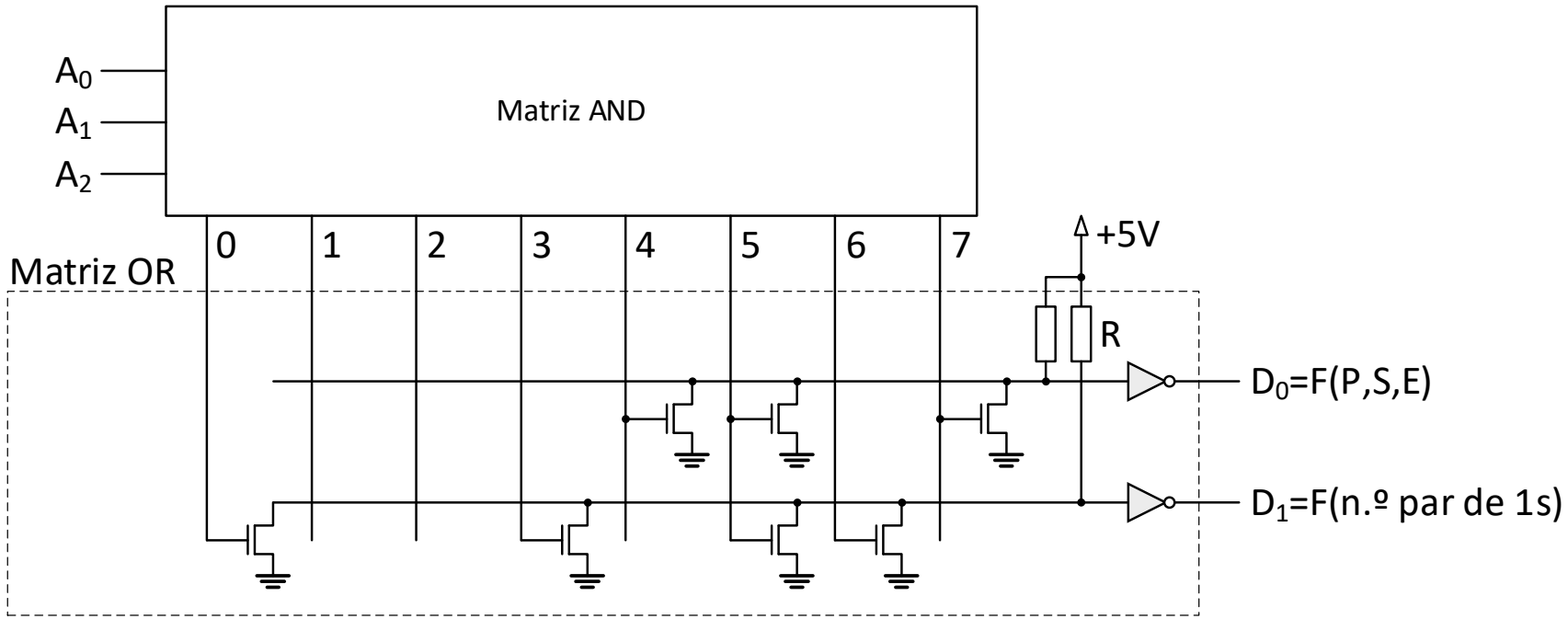




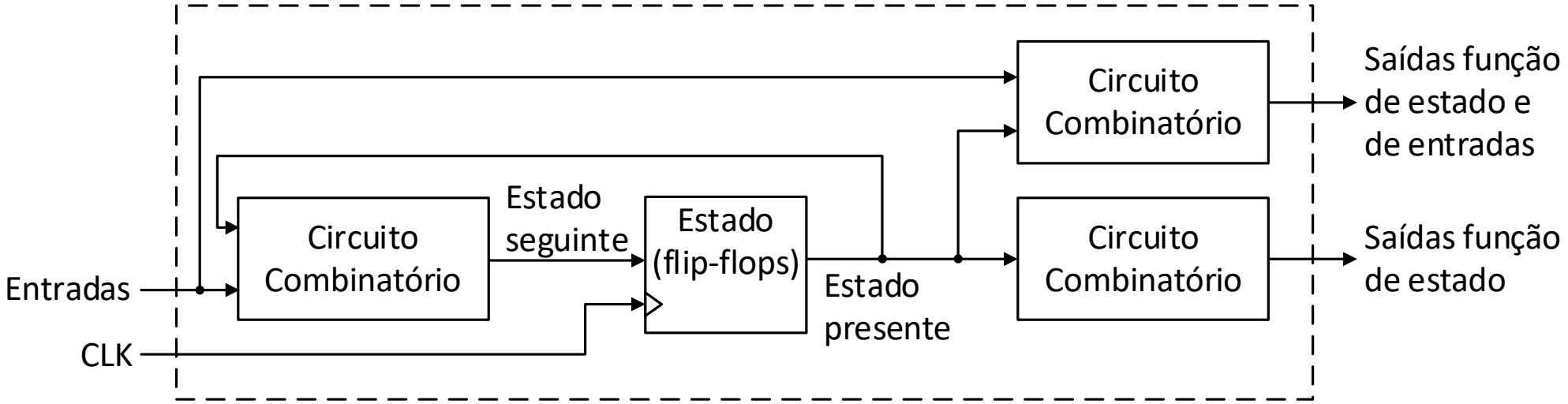
Implementação de funções lógicas com ROM

A_2	A_1	A_0	D_1	D_0
0	0	0	1	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	1
1	0	1	1	1
1	1	0	1	0
1	1	1	0	1

- D_0 = Controlo de estore automático baseado nos sensores de presença (A_2), luz solar (A_1) e estore fechado (A_0)
- D_1 = Nº par de bits com o valor lógico 1

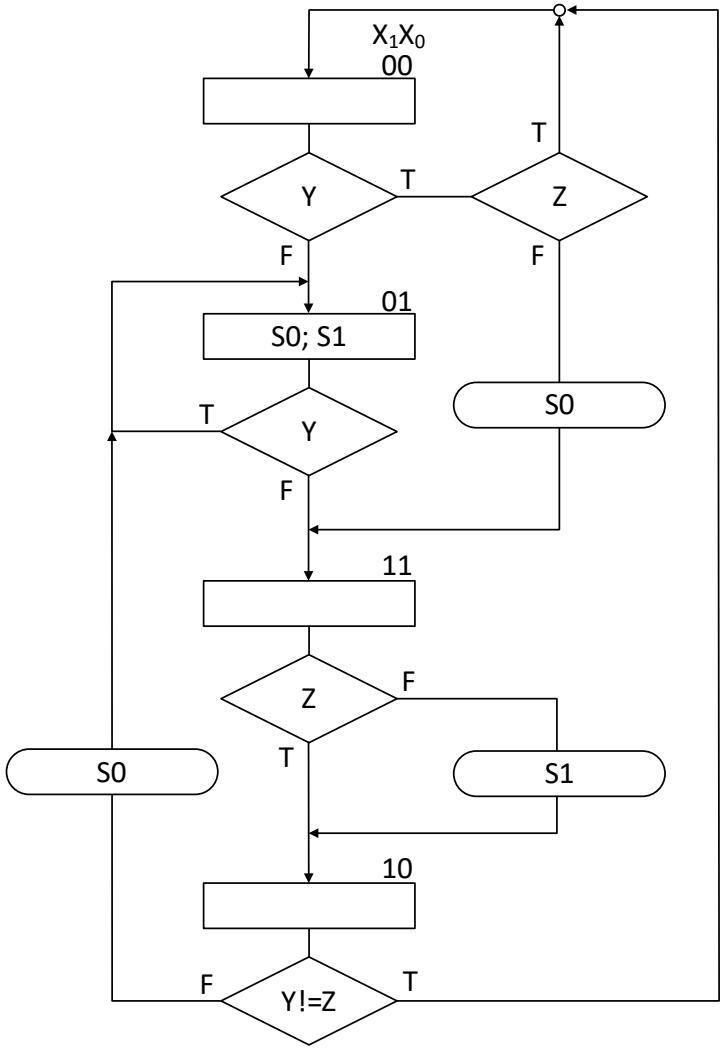


Controlo microprogramado / modelo geral de uma máquina de estados



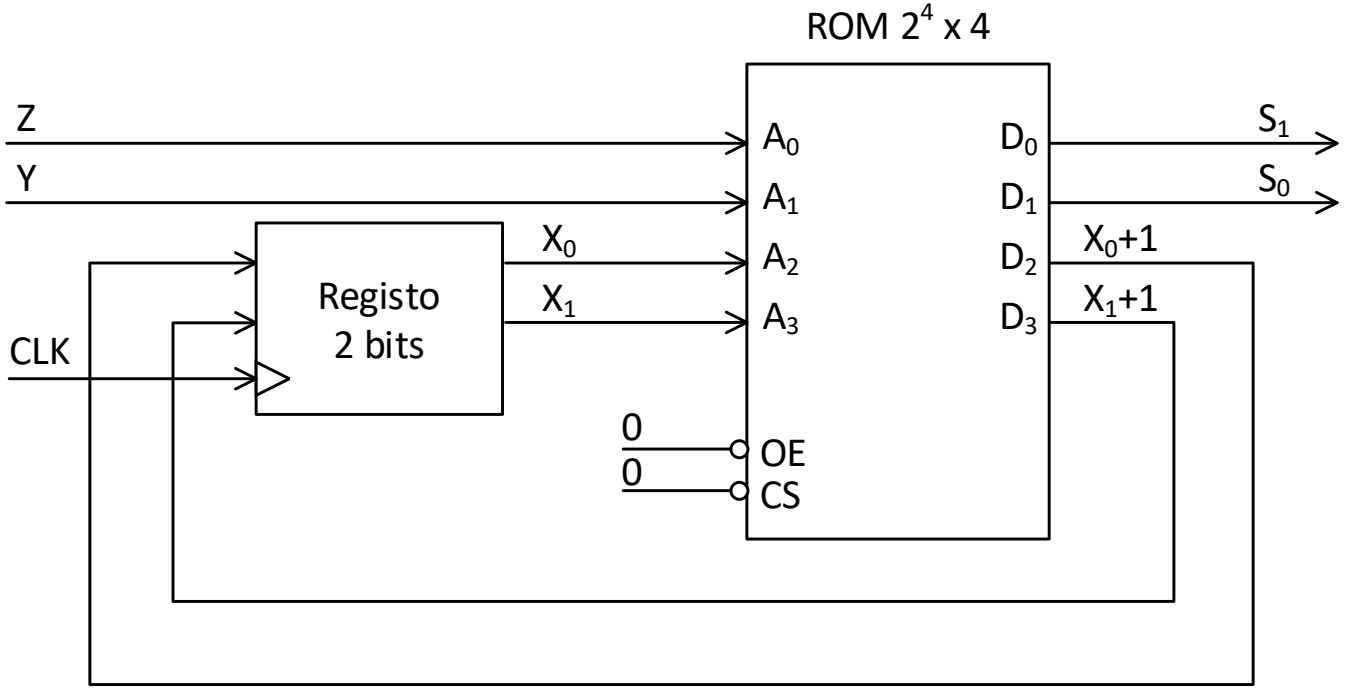
- Os blocos “Circuito Combinatório” podem ser implementados com uma ROM
  - O “Estado presente” e “Entradas” definem o endereço de acesso à ROM
  - As “Saídas função de estado” e “Saídas função de estado e de entradas” são definidas pela palavra da ROM selecionada

Exemplo de ASM com tabela de transição de estado e de saídas



EP		Entradas		ES		Saídas	
X_1	X_0	Y	Z	X_1+1	X_0+1	S_0	S_1
0	0	0	-	0	1	0	0
0	0	1	0	1	1	1	0
0	0	1	1	0	0	0	0
0	1	0	-	1	1	1	1
0	1	1	-	0	1	1	1
1	0	0	0	0	1	1	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	1	1	0
1	1	-	0	1	0	0	1
1	1	-	1	1	0	0	0

Controlo microprogramado



Endereços				Dados			
A_3	A_2	A_1	A_0	D_3	D_2	D_1	D_0
X_1	X_0	Y	Z	$X_1 + 1$	$X_0 + 1$	S_0	S_1
0	0	0	0	0	1	0	0
0	0	0	1	0	1	0	0
0	0	1	0	1	1	1	0
0	0	1	1	0	0	0	0
0	1	0	0	1	1	1	1
0	1	0	1	1	1	1	1
0	1	1	0	0	1	1	1
0	1	1	1	0	1	1	1
1	0	0	0	0	1	1	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	1
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Controlo microprogramado em VHDL – entidade e FFD (1 de 3)

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity ASM_Microprogrammed_i2223 is
port (MCLK, Y, Z, RST: in std_logic;
      S1, S0: out std_logic
      );
end ASM_Microprogrammed_i2223;

architecture ASM_Microprogrammed_i2223_arch of ASM_Microprogrammed_i2223 is

   component FFD
   port( CLK, RESET, SET, D, EN : in std_logic;
         Q : out std_logic
         );
   end component;
   signal ADDR: std_logic_vector (3 downto 0);
   signal DATA: std_logic_vector (3 downto 0);
   signal EP: std_logic_vector(1 downto 0);
```

Controlo microprogramado em VHDL – ROM (2 de 3)

```
-- A(3) = X1; A(2) = X0; A(1) = Y; A(0) = Z
-- D(3) = X1+1; D(2) = X0+1; D(1) = S0; D(0) = S1
type ROM_type is array (0 to 15) of std_logic_vector(3 downto 0);

constant ROM: ROM_type := (0  => "0100",
                           1  => "0100",
                           2  => "1110",
                           3  => "0000",
                           4  => "1111",
                           5  => "1111",
                           6  => "0111",
                           7  => "0111",
                           8  => "0110",
                           9  => "0000",
                           10 => "0000",
                           11 => "0110",
                           12 => "1001",
                           13 => "1000",
                           14 => "1001",
                           15 => "1000");
```

Controlo microprogramado em VHDL – Circuito (3 de 3)

```
begin

    UX0: FFD port map (CLK => MCLK, RESET => RST, SET => '0',
                       D => DATA(2), EN => '1', Q => EP(0));
    UX1: FFD port map (CLK => MCLK, RESET => RST, SET => '0',
                       D => DATA(3), EN => '1', Q => EP(1));
    ADDR <= EP(1) & EP(0) & Y & Z;
    DATA <= ROM( to_integer(unsigned(ADDR)));
    S0 <= DATA(1);
    S1 <= DATA(0);

end ASM_Microprogrammed_i2223_arch;
```