MC542

Organização de Computadores
Teoria e Prática

2006

Prof. Paulo Cesar Centoducatte

ducatte@ic.unicamp.br

www.ic.unicamp.br/~ducatte

MC542

Circuitos Lógicos

Projeto de Máquinas de Estados com VHDL

"Fundamentals of Digital Logic with VHDL Design" - (Capítulo 8)

Projeto de Máquinas de Estados com VHDL
Sumário

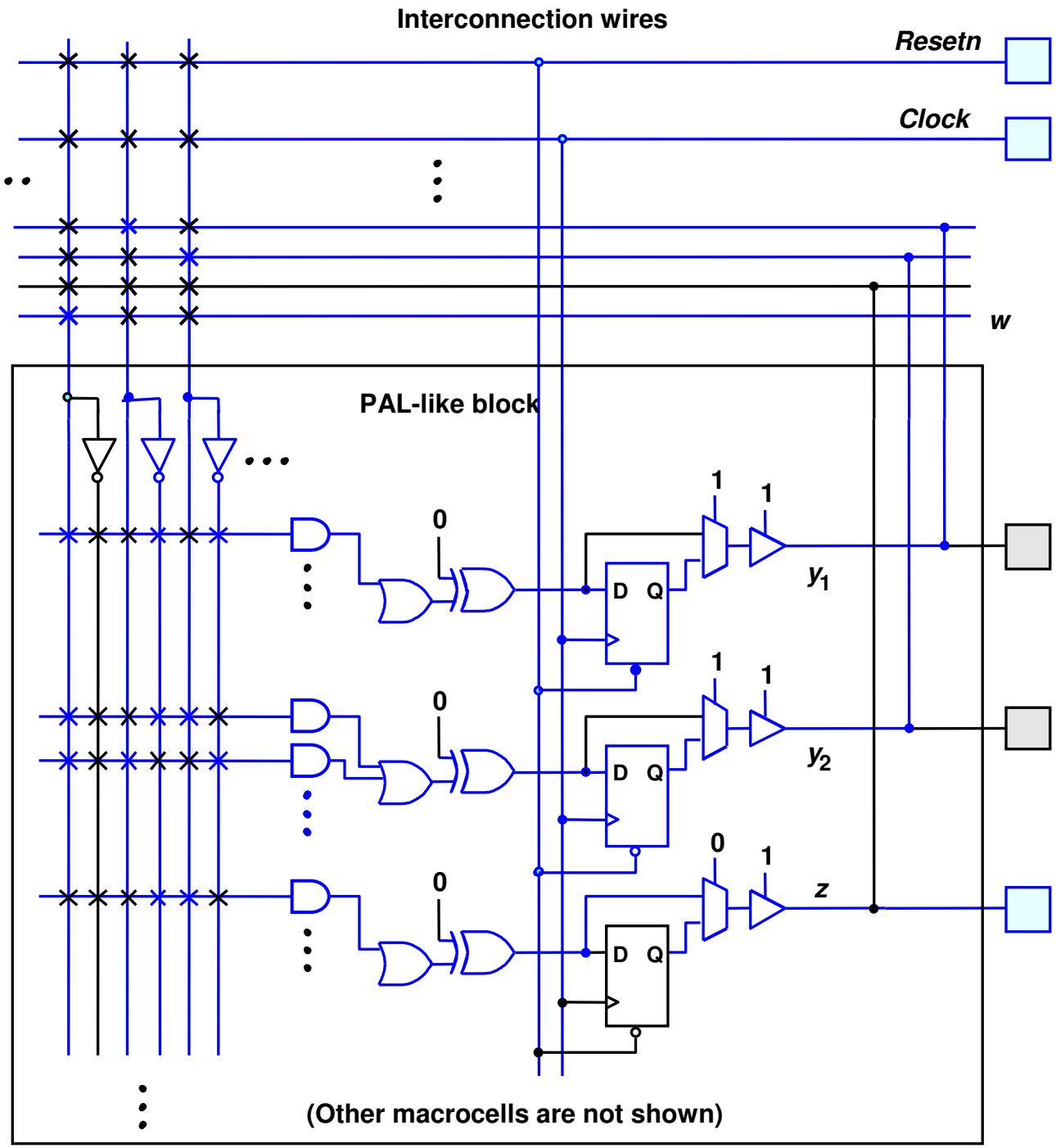
- Código VHDL para FSM de Moore
- Código VHDL para FSM de Mealy

FSM de Moore

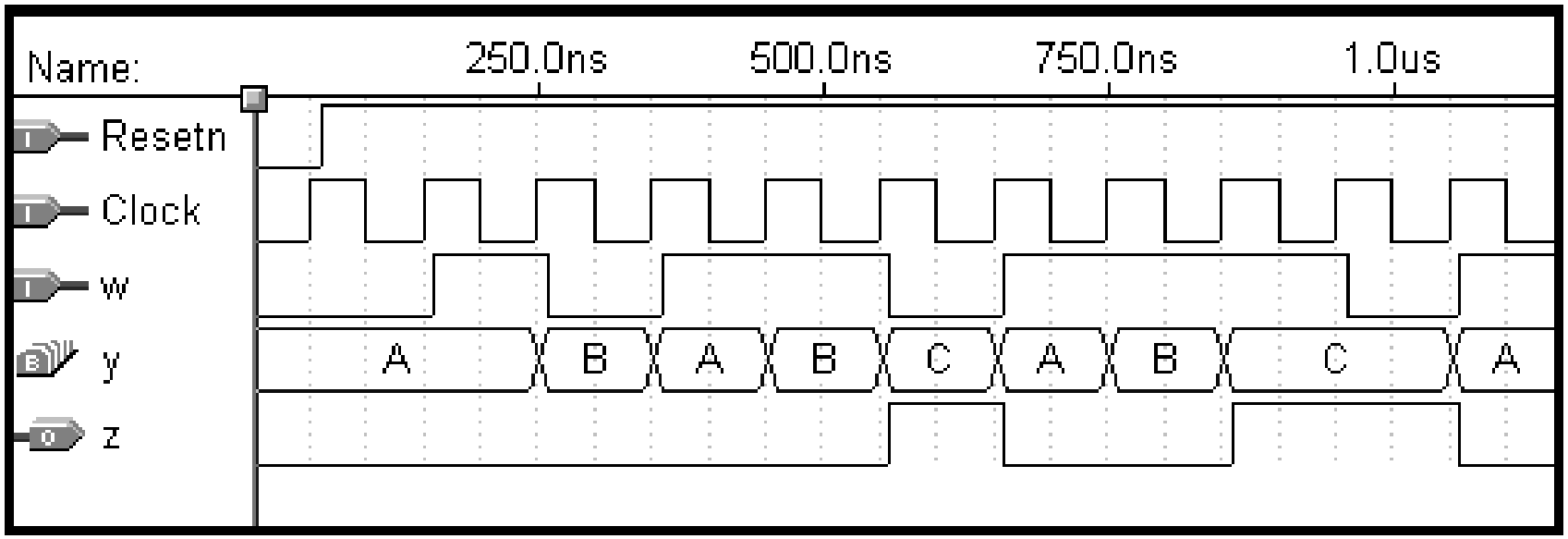
```
USE ieee.std_logic_1164.all ;

ENTITY simple IS
   PORT (Clock, Resetn, w : IN     STD_LOGIC ;
         z                : OUT    STD_LOGIC ) ;
END simple ;

ARCHITECTURE Behavior OF simple IS
   TYPE State_type IS (A, B, C) ;  -- Tipo Enumerado para
                                   -- definir os  Estados
   SIGNAL y : State_type ;
BEGIN
   PROCESS ( Resetn, Clock )
   BEGIN
        IF Resetn = '0' THEN
                y <= A ;
        ELSIF (Clock'EVENT AND Clock = '1') THEN

con't ...
```

FSM de Moore

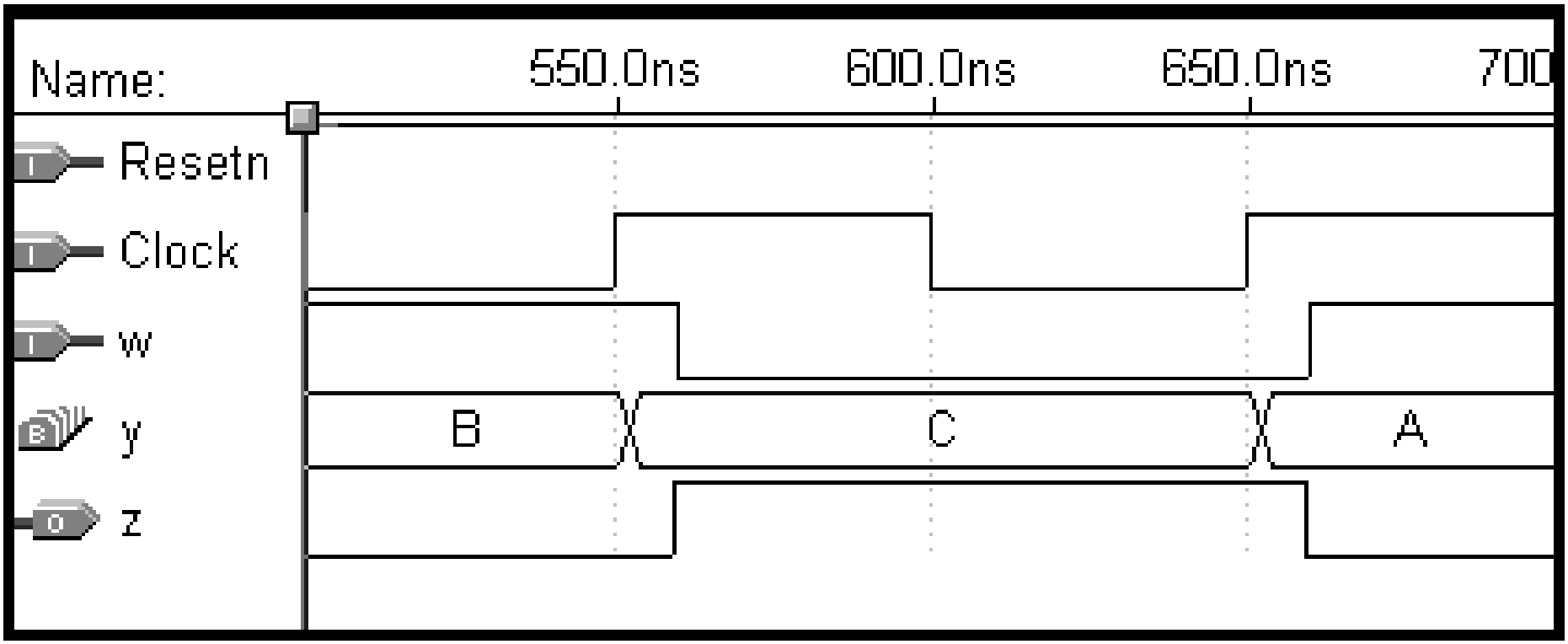
```
        CASE y IS
                WHEN A =>
                   IF w = '0'
                     THEN  y <= A ;
                      ELSE   y <= B ;
                    END IF ;
                WHEN B =>
                   IF w = '0'
                     THEN  y <= A ;
                     ELSE   y <= C ;
                   END IF ;
                WHEN C =>
                   IF w = '0'
                     THEN   y <= A ;
                     ELSE    y <= C ;
                   END IF ;
                END CASE ;
        END IF ;
END PROCESS ;
  z <= '1' WHEN y = C ELSE '0' ;
END Behavior ;
```


Interconnection wires
Resetn
Clock
w
PAL-like block
0
1
1
D Q
y_1
0
1
1
D Q
y_2
0
1
0
D Q
z
(Other macrocells are not shown)

FSM de Moore
Simulação



FSM de Moore
Simulação



FSM de Moore
Codificação Alternativa

```
USE ieee.std_logic_1164.all ;

ENTITY simple IS
  PORT (Clock, Resetn, w : IN    STD_LOGIC ;
        z                : OUT   STD_LOGIC ) ;
END simple ;

ARCHITECTURE Behavior OF simple IS
  TYPE State_type IS (A, B, C) ;
  SIGNAL y_present, y_next : State_type ;
```

FSM de Moore
Codificação Alternativa

```
BEGIN
  PROCESS ( w, y_present )
  BEGIN
       CASE y_present IS
            WHEN A =>
                 IF w = '0' THEN
                      y_next <= A ;
                 ELSE
                      y_next <= B ;
                 END IF ;
            WHEN B =>
                 IF w = '0' THEN
                      y_next <= A ;
                 ELSE
                      y_next <= C ;
                 END IF ;
```

FSM de Moore
Codificação Alternativa

```
                WHEN C =>
                        IF w = '0' THEN
                                y_next <= A ;
                        ELSE
                                y_next <= C ;
                        END IF ;
        END CASE ;
    END PROCESS ;

    PROCESS (Clock, Resetn)
    BEGIN
        IF Resetn = '0' THEN
                y_present <= A ;
        ELSIF (Clock'EVENT AND Clock = '1') THEN
                y_present <= y_next ;
        END IF ;
    END PROCESS ;

    z <= '1' WHEN y_present = C ELSE '0' ;
END Behavior ;
```

FSM
O Usuário Especificando a Atribuição de Estados

```
ARCHITECTURE Behavior OF simple IS

 TYPE State_TYPE IS (A, B, C) ;

 ATTRIBUTE ENUM_ENCODING : STRING ;

 ATTRIBUTE ENUM_ENCODING OF State_type : TYPE IS "00 01 11";

 SIGNAL y_present, y_next : State_type ;
BEGIN

 con't ...
```

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY simple IS
        PORT (  Clock, Resetn, w  : IN      STD_LOGIC ;
                        z                                       : OUT
        STD_LOGIC ) ;
END simple ;

ARCHITECTURE Behavior OF simple IS
        SIGNAL y_present, y_next : STD_LOGIC_VECTOR(1 DOWNTO 0);
        CONSTANT A   : STD_LOGIC_VECTOR(1 DOWNTO 0) := "00" ;
        CONSTANT B   : STD_LOGIC_VECTOR(1 DOWNTO 0) := "01" ;
        CONSTANT C   : STD_LOGIC_VECTOR(1 DOWNTO 0) := "11" ;
BEGIN
        PROCESS ( w, y_present )
        BEGIN
                CASE y_present IS
                        WHEN A =>
                                IF w = '0' THEN y_next <= A ;
                                ELSE y_next <= B ;
                                END IF ;
```

… con't

```
                    WHEN B =>
                            IF w = '0' THEN y_next <= A ;
                            ELSE y_next <= C ;
                            END IF ;
                    WHEN C =>
                            IF w = '0' THEN y_next <= A ;
                            ELSE y_next <= C ;
                            END IF ;
                    WHEN OTHERS =>
                            y_next <= A ;
            END CASE ;
    END PROCESS ;

    PROCESS ( Clock, Resetn )
    BEGIN
            IF Resetn = '0' THEN
                    y_present <= A ;
            ELSIF (Clock'EVENT AND Clock = '1') THEN
                    y_present <= y_next ;
            END IF ;
    END PROCESS ;
    z <= '1' WHEN y_present = C ELSE '0' ;
END Behavior ;
```

FSM de Mealy

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY mealy IS
        PORT ( Clock, Resetn, w         : IN    STD_LOGIC ;
                z                       : OUT   STD_LOGIC ) ;
END mealy ;

ARCHITECTURE Behavior OF mealy IS
        TYPE State_type IS (A, B) ;
        SIGNAL y : State_type ;
BEGIN
        PROCESS ( Resetn, Clock )
        BEGIN
                IF Resetn = '0' THEN
                        y <= A ;
                ELSIF (Clock'EVENT AND Clock = '1') THEN
                        CASE y IS
                                WHEN A =>
                                        IF w = '0' THEN y <= A ;
                                        ELSE y <= B ;
                                        END IF ;
```

… con't

FSM de Mealy

```
                        WHEN B =>
                                IF w = '0' THEN y <= A ;
                                ELSE y <= B ;
                                END IF ;
                END CASE ;
        END IF ;
END PROCESS ;

PROCESS ( y, w )
BEGIN
        CASE y IS
                WHEN A =>
                        z <= '0' ;
                WHEN B =>
                        z <= w ;
        END CASE ;
END PROCESS ;
END Behavior ;
```

FSM de Mealy

- Estudar 8.5 a 8.8 (exemplos de projetos)