Máquinas de estado Finitos

Computação Digital

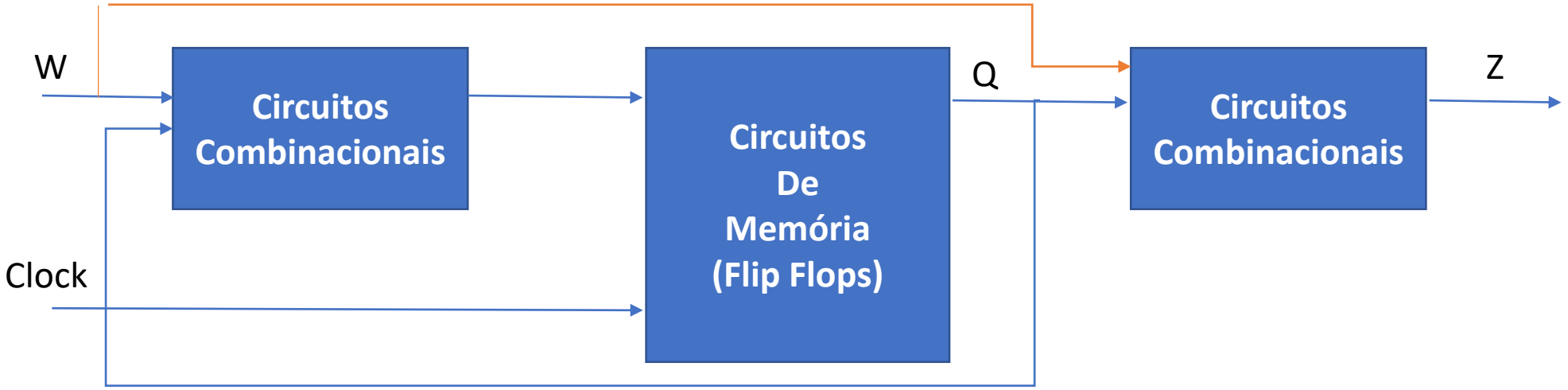
Máquinas de Estados

- Estados
  - Situação particular na qual encontra-se uma Máquina de Estado.
  - Pode trocar de estado em função das entradas e do estado atual.
  - Cada saída apresenta um proposito independente.

- Tipos Máquina de Estados
  - Máquina de estado de Mealy.
  - Maquina de estado de Moore.

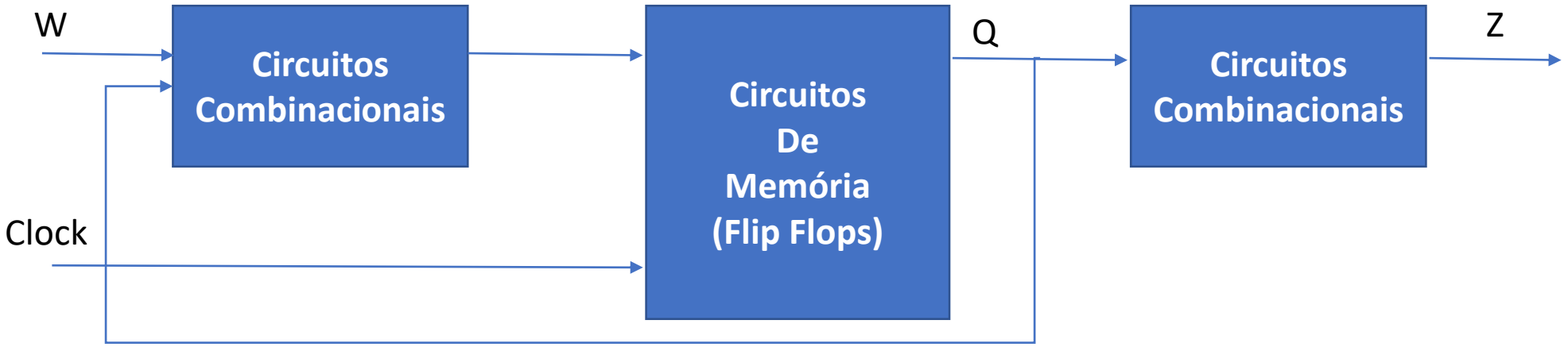
Máquinas de Estados

Máquina de estado de Mealy.



Máquinas de Estados

Maquina de estado de Moore.

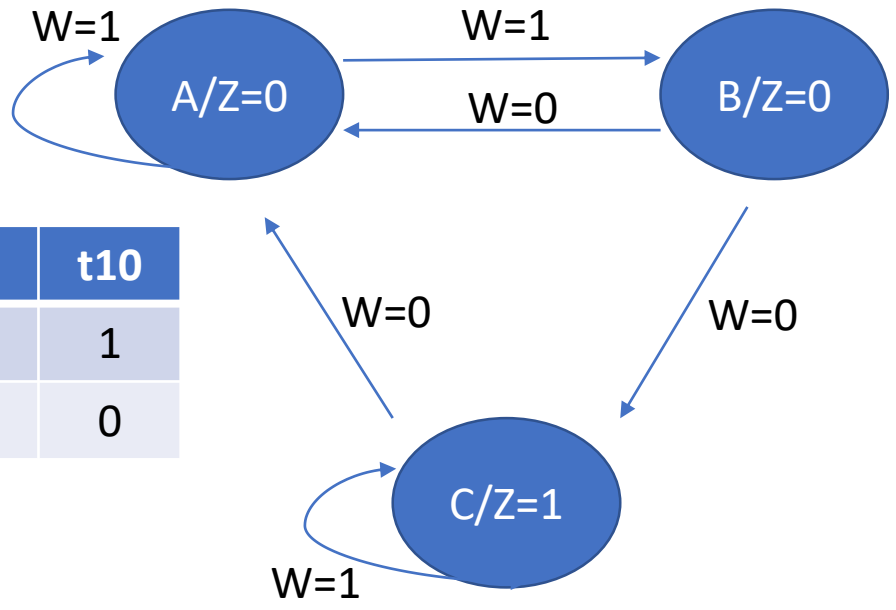


Máquinas de Estados

Diagrama de uma máquina de estado.

- Entrada : W
- Saída : Z

clock	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
W	0	1	0	1	1	0	1	1	1	0	1
Z	0	0	0	0	0	1	0	0	1	1	0



Máquinas de Estados

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
--use IEEE.NUMERIC_STD.ALL;
entity Maq_Estados is
    Port ( clk,reset : in  STD_LOGIC;
           W : in  STD_LOGIC;
           Z : out  STD_LOGIC);
end Maq_Estados;
architecture Behavioral of Maq_Estados is
  type STATE_TYPE is (A, B, C);
  signal estado: STATE_TYPE;
```

```vhdl
begin
process(clk, reset)
begin
   if reset = '1' then
      estado <= A;
   elsif clk'event and clk = '1' then
   --Máquina de estados
      case estado is
      when A =>
         Z <= '0';
         if W = '1' then
            estado <= B;
         end if;
      when B =>
         Z <= '0';
         if W = '0' then
            estado <= A;
         else
            estado <=C;
         end if;
      when C =>
         Z <= '1';
      if W = '0' then
         estado <= A;
      end if;
      end case;
   end if;
end process;
end Behavioral;
```

Máquinas de Estados: Testbench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY TB_ME IS
END TB_ME;
ARCHITECTURE behavior OF TB_ME IS
    -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT Maq_Estados
    PORT(
         clk : IN  std_logic;
         reset : IN  std_logic;
         W : IN  std_logic;
         Z : OUT  std_logic
        );
    END COMPONENT;
   --Inputs
   signal clk : std_logic := '0';
   signal reset : std_logic := '0';
   signal W : std_logic := '0';
   --Outputs
   signal Z : std_logic;
   -- Clock period definitions
   constant clk_period : time := 10 ns;
BEGIN
   -- Instantiate the Unit Under Test (UUT)
   uut: Maq_Estados PORT MAP (
          clk => clk,
          reset => reset,
          W => W,
          Z => Z
        );

   -- Clock process definitions
   clk_process :process
   begin
      clk <= '0';
      wait for clk_period/2;
      clk <= '1';
      wait for clk_period/2;
   end process;


   -- Stimulus process
   stim_proc: process
   begin
      -- hold reset state for 100 ns.
      wait for 100 ns;

      wait for clk_period*10;

      -- insert stimulus here

      wait;
   end process;

END;
```

```
-- Stimulus process
stim_proc: process
begin
   reset <= '1','0' after clk_period/2;
   W <= '0';
   wait for 100 ns;
   W <= '1';
   wait for 100 ns;
   W <= '0';
   wait for 200 ns;
   reset <= '0';
end process;
```

Máquinas de Estados

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
--use IEEE.NUMERIC_STD.ALL;
entity ME_Cont is
    Port ( clk : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           Control : in STD_LOGIC_VECTOR (1 downto 0);
           SaidaLeds : out STD_LOGIC_VECTOR (3 downto 0);
           Display7 : out  STD_LOGIC_VECTOR (6 downto 0);
           c : out  STD_LOGIC);
end ME_Cont;

architecture Behavioral of ME_Cont is
   COMPONENT Mod_Display
   PORT( clk   :  IN STD_LOGIC;
         hex1 :  IN STD_LOGIC_VECTOR (3 DOWNTO 0);
         hex2 :  IN STD_LOGIC_VECTOR (3 DOWNTO 0);
         c :  OUT   STD_LOGIC;
         hex_out :  OUT   STD_LOGIC_VECTOR (6 DOWNTO 0);
         Reset   :  IN STD_LOGIC);
   END COMPONENT;

   type STATE_TYPE is (E0,E1, E2,E3);
   signal estado: STATE_TYPE := E0;
   signal hex_aux : STD_LOGIC_VECTOR (3 DOWNTO 0):="0000";
   signal q : STD_LOGIC_VECTOR (3 DOWNTO 0);
   signal up: STD_LOGIC :='0';
   signal tick: STD_LOGIC :='0';
   signal clk_aux: STD_LOGIC :='0';
begin
timer: entity work.mod_m_counter(arch)
   GENERIC MAP(
    N =>26,M =>50000000)
   PORT MAP(
      clk => clk,
      reset => reset,
      max_tick => tick
   );
   Display: Mod_Display PORT MAP(
      clk => clk,
      hex1 => "1110",
      hex2 => hex_aux,
      c => c,
      hex_out => Display7,
      Reset => reset
   );
   counter: entity work.counter_up_down(arch)
      GENERIC MAP(N =>4)
      PORT MAP(
         clk => clk_aux,
         reset => reset,
         up => up,
         q => q
      );
ME: process(clk, reset)
begin
   if reset = '1' then
      estado <= E0;
      hex_aux <= "0000";
   elsif clk'event and clk = '1' then
   --Máquina de estados
      case estado is
      when E0 =>
         hex_aux <= "0000";
         if Control = "00" then
            estado <= E0;
            SaidaLeds<= "1010";
         else
            estado <= E1;
         end if;
      when E1 =>
         hex_aux <= "0001";
         if Control = "01" then
            estado <= E1;
            SaidaLeds<= "0101";
         else
            estado <= E2;
         end if;
      when E2 =>
         hex_aux <= "0010";
         if Control = "10" then
            estado <= E2;
            up <='0';
            SaidaLeds<= q;
         else
            estado <= E3;
         end if;
      when E3 =>
         hex_aux <= "0011";
         if Control = "11" then
            estado <= E3;
            up <='1';
            SaidaLeds<= q;
         else
            estado <= E0;
         end if;
      end case;
   end if;
end process;

tempo: process (tick)
begin
   if (tick'event and tick='1') then
      clk_aux <= not clk_aux;
   end if;
end process;

end Behavioral;
```