

# Laboratório de Sistemas Digitais

## Aula Teórico-Prática 5

Ano Letivo 2023/24

Modelação em VHDL de registos e  
módulos combinatórios de  
deslocamento



# Conteúdo

- Modelação em VHDL
  - Registos de deslocamento
  - Módulos combinatórios de deslocamento (*shifters*)

# Operações de Deslocamento

Deslocamento	Operando	Resultado (deslocam. de 1 bit)	Resultado (deslocam. de 2 bits)
À esquerda <u>lógico ou aritmético</u> (introduz 0's)	0 <b>100</b>	<b>100</b> <u>0</u>	<b>00</b> <u>00</u>
	0 <b>101</b>	<b>101</b> <u>0</u>	<b>01</b> <u>00</u>
À direita <u>lógico</u> (introduz 0's)	<b>00</b> 11	<u>000</u> 1	<u>0000</u>
	<b>10</b> 11	<u>010</u> 1	<u>0010</u>
À direita <u>aritmético</u> (preserva o sinal)	<b>00</b> 11	<u>000</u> 1	<u>0000</u>
	<b>10</b> 11	<u>110</u> 1	<u>1110</u>

## Aplicações típicas:

Conversão de dados paralelo  $\leftrightarrow$  série em sistemas computacionais de/para as interfaces Ethernet, SATA, PCIe, etc.

Algoritmos de deteção e correção de erros em sistemas de comunicação, etc.

## Abordagens / implementações típicas:

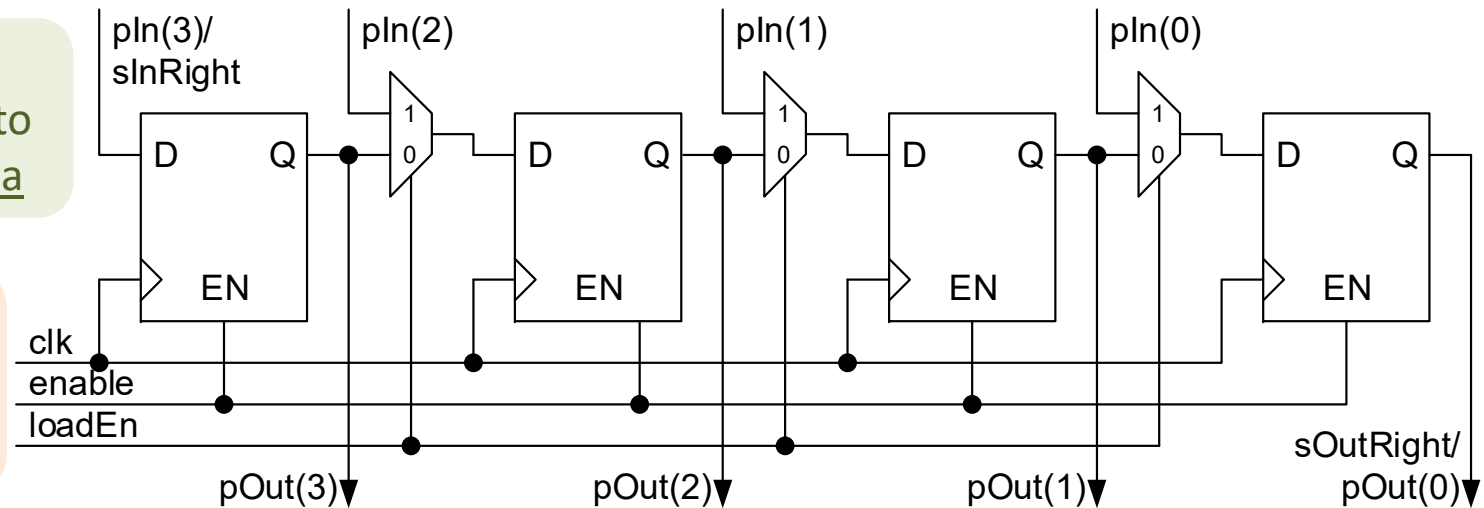
- Iterativa (registo de deslocamento – c/clock)
- Paralela (combinatória - *barrel shifter*)

Deslocar  $i$  bits à esq.  $\Leftrightarrow \times 2^i$   
Deslocar  $i$  bits à direita  $\Leftrightarrow \div 2^i$

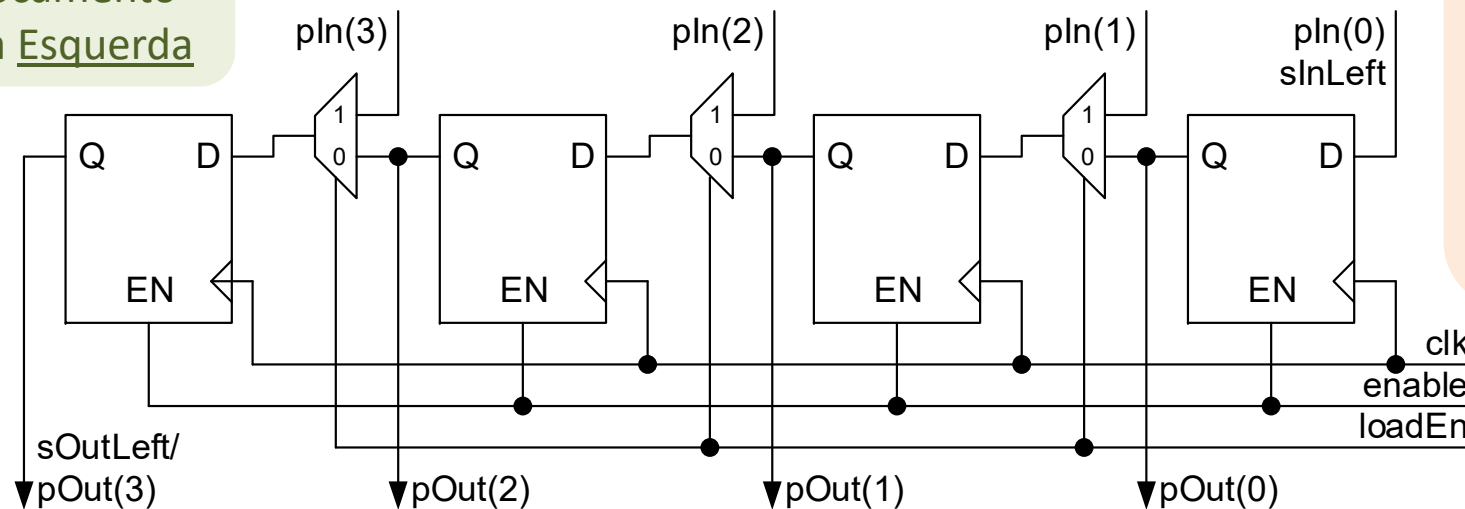
# Interface e Estrutura de um Registro de Deslocamento

Registro de Deslocamento para a Direita

Qual a função de cada sinal de controlo e sincronização?



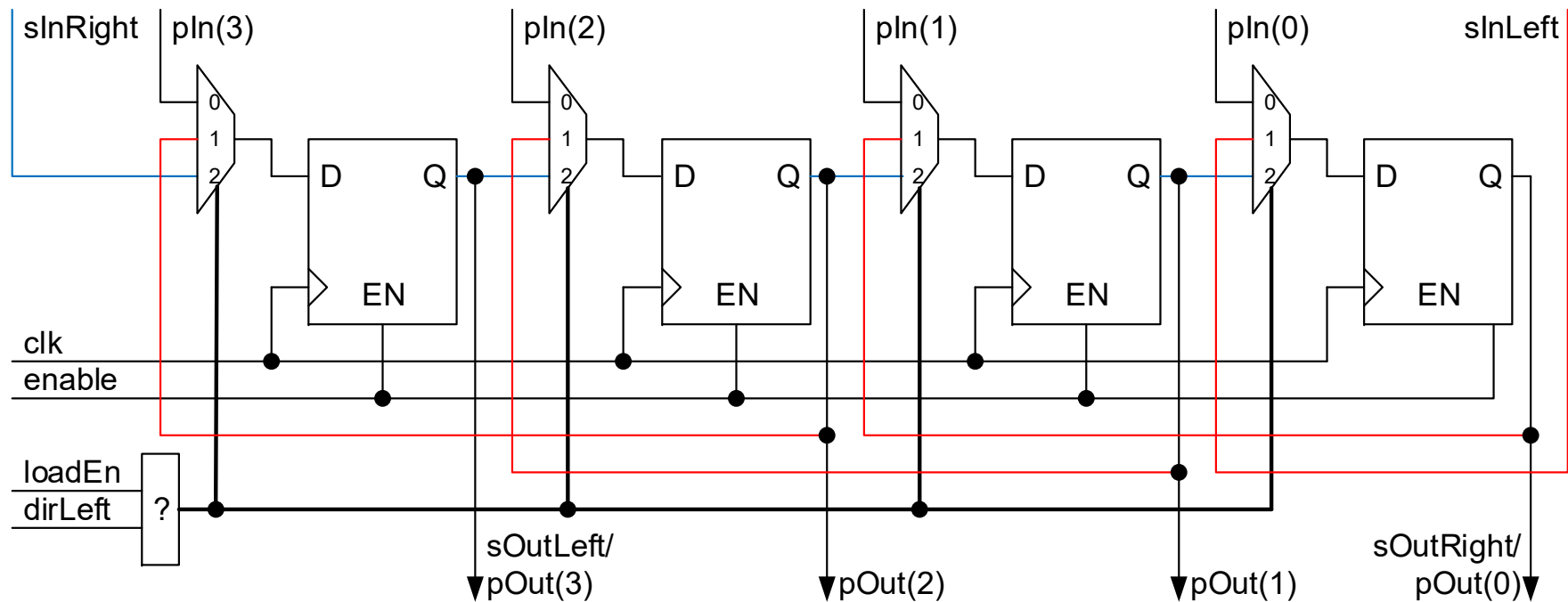
Registro de Deslocamento para a Esquerda



Na maior parte dos casos são necessárias apenas algumas destas entradas/saídas série/paralelas



# Interface e Estrutura de um Registro de Deslocamento Bidirecional



enable	loadEn	dirLeft	Operação
0	-	-	Nenhuma (registro inalterado)
1	1	-	Carregamento paralelo
1	0	1	Deslocamento p/ a esquerda
1	0	0	Deslocamento p/ a direita

Determine a função lógica de cada sinal de seleção dos multiplexadores em função das entradas "loadEn" e "dirLeft"

# Exemplo de Registo de Deslocamento

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

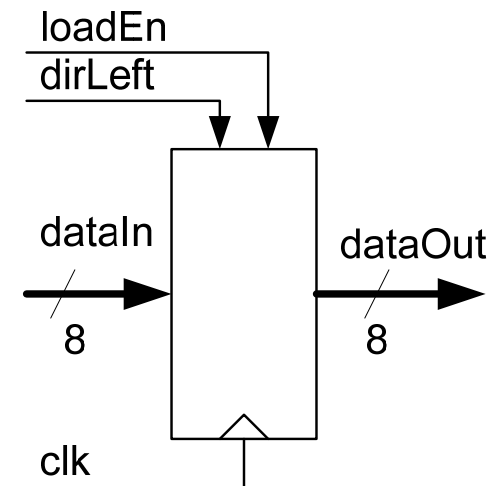
entity ShiftReg is
    port(clk      : in  std_logic;
          loadEn   : in  std_logic;
          dataIn   : in  std_logic_vector(7 downto 0);
          dirLeft  : in  std_logic;
          dataOut  : out std_logic_vector(7 downto 0));
end ShiftReg;

architecture Behavioral of ShiftReg is
    signal s_shiftReg : std_logic_vector(7 downto 0);
begin
    process(clk)
    begin
        if (rising_edge(clk)) then
            if (loadEn = '1') then
                s_shiftReg <= dataIn;
            elsif (dirLeft = '1') then
                s_shiftReg <= s_shiftReg(6 downto 0) & '0';
            else
                s_shiftReg <= '0' & s_shiftReg(7 downto 1);
            end if;
        end if;
    end process;

    dataOut <= s_shiftReg;

end Behavioral;
```

Exemplo com carregamento paralelo de uma “palavra” e o seu deslocamento bit-a-bit de forma síncrona com o *clock*

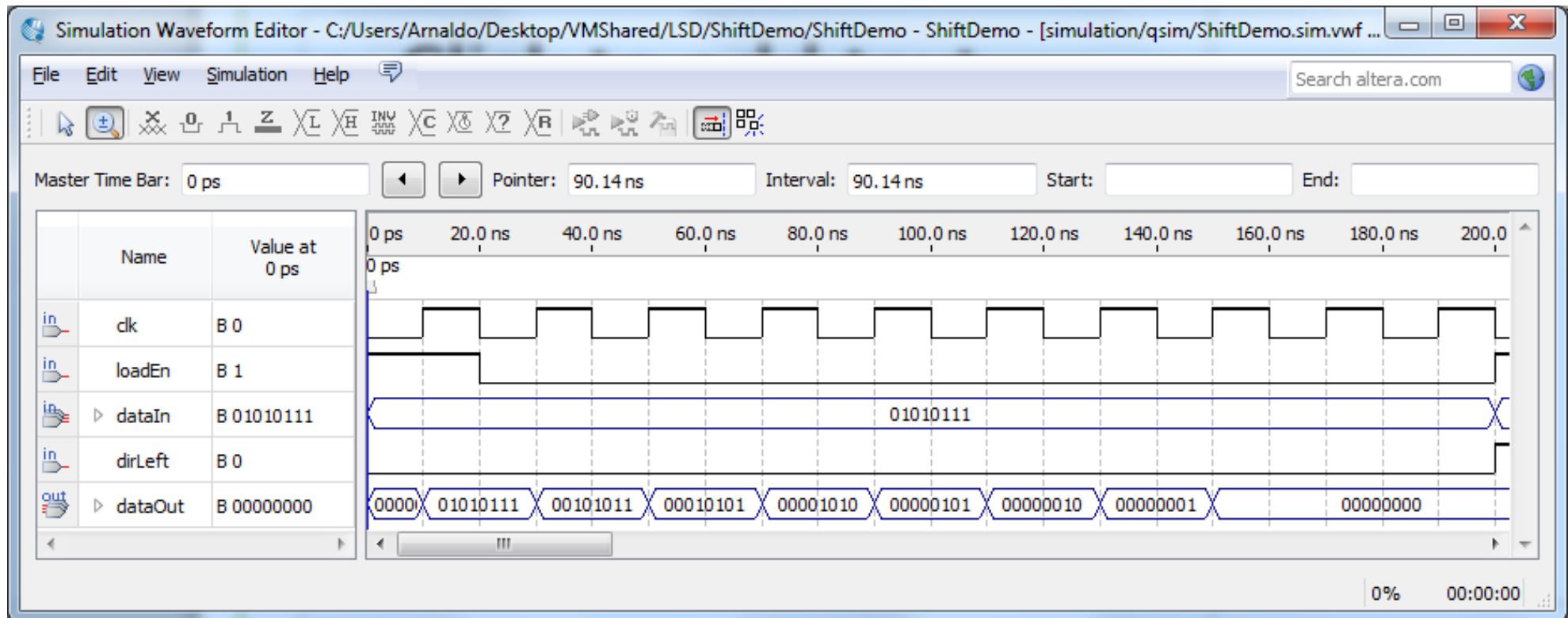


loadEn	dirLeft	Operação
1	-	Carregamento paralelo
0	1	Deslocamento p/ a esquerda
0	0	Deslocamento p/ a direita

Como realizar um deslocamento aritmético para a direita?  
(ex. 1010 >> 1 = 1101)  
Como realizar rotações?

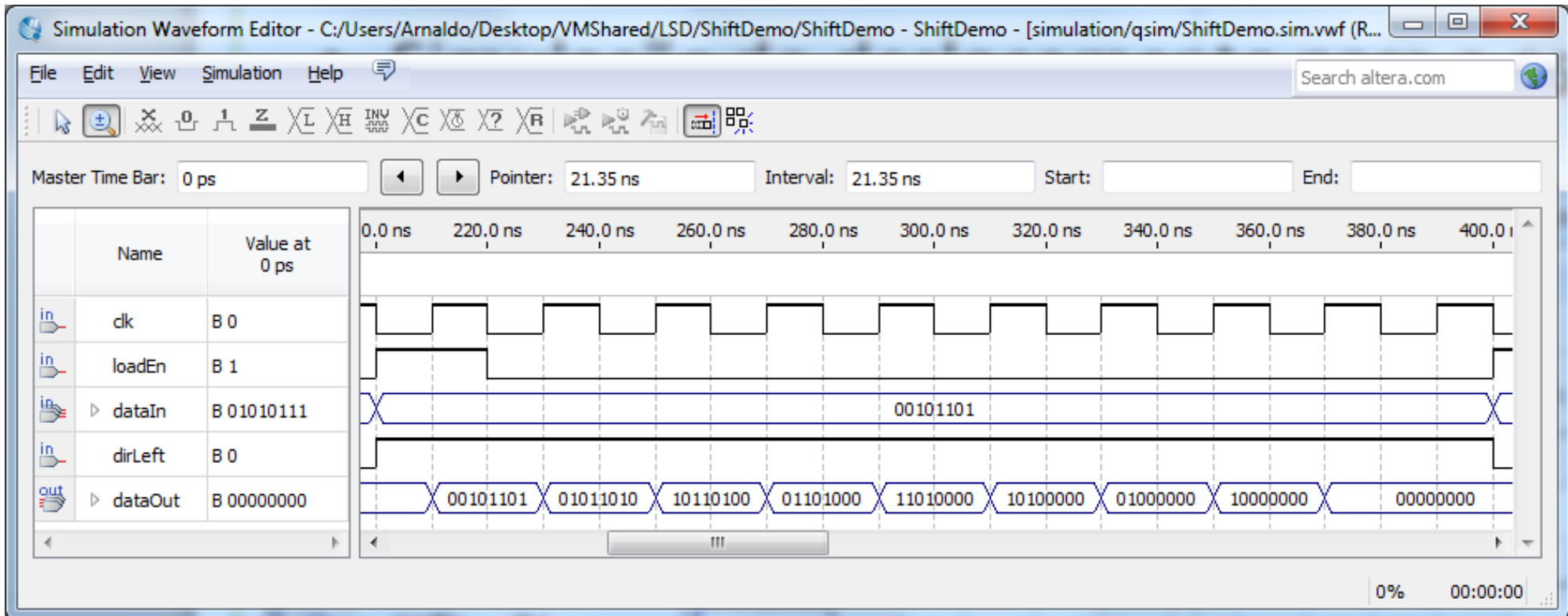
# Simulação do Registo de Deslocamento

- Simulação do carregamento paralelo e deslocamento para a direita (loadEn = '1' -> loadEn = '0'; dirLeft = '0')



# Simulação do Registo de Deslocamento

- Simulação do carregamento paralelo e deslocamento para a esquerda (loadEn = '1' -> loadEn = '0'; dirLeft = '1')





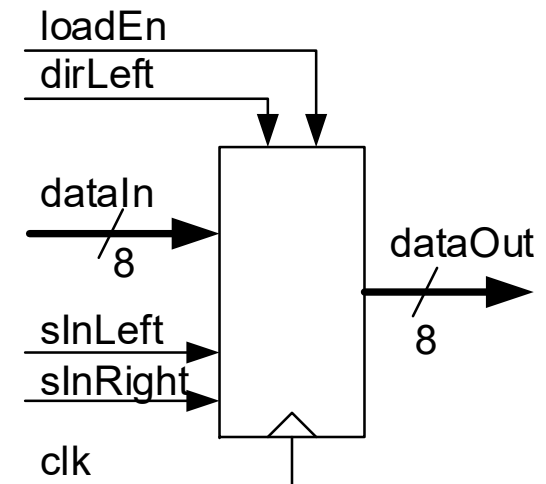
# Mais um Exemplo de um Registo de Deslocamento

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity IterShifter is
  port(clk      : in  std_logic;
        loadEn   : in  std_logic;
        sInLeft  : in  std_logic;
        sInRight : in  std_logic;
        dataIn   : in  std_logic_vector(7 downto 0);
        dirLeft  : in  std_logic;
        dataOut  : out std_logic_vector(7 downto 0));
end IterShifter;

architecture Behavioral of IterShifter is
  signal s_shiftReg : std_logic_vector(7 downto 0);
begin
  process(clk)
  begin
    if (rising_edge(clk)) then
      if (loadEn = '1') then
        s_shiftReg <= dataIn;
      elsif (dirLeft = '1') then
        s_shiftReg <= s_shiftReg(6 downto 0) & sInLeft;
      else
        s_shiftReg <= sInRight & s_shiftReg(7 downto 1);
      end if;
    end if;
  end process;

  dataOut <= s_shiftReg;
end Behavioral;
```



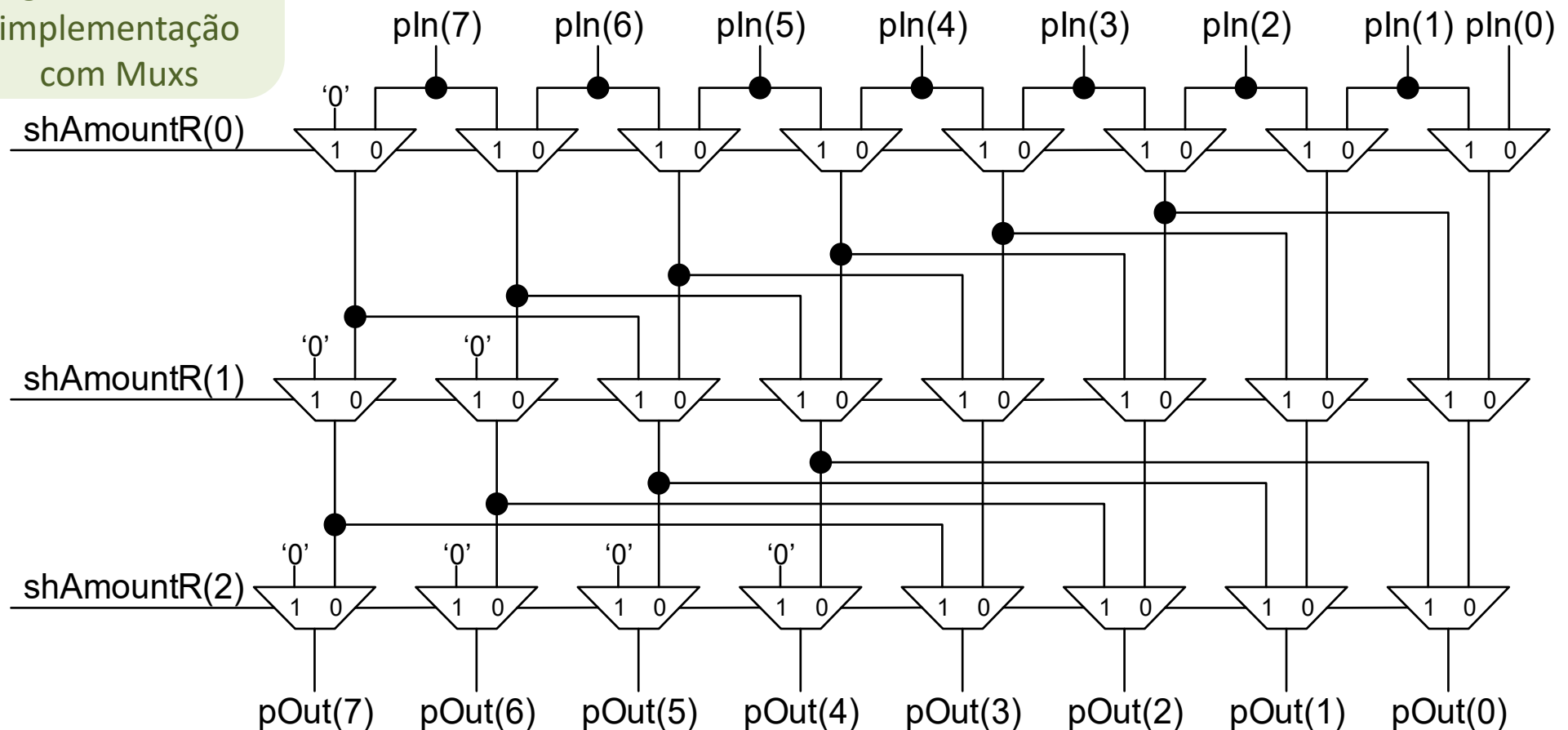
loadEn	dirLeft	Operação
1	-	Carregamento paralelo
0	1	Deslocamento p/ a esquerda
0	0	Deslocamento para a direita

Exemplo com entradas série e também de carregamento paralelo de uma “palavra” e o seu deslocamento bit-a-bit de forma síncrona com o *clock*



# Interface e Estrutura de um *Barrel Shifter* (Combinatório)

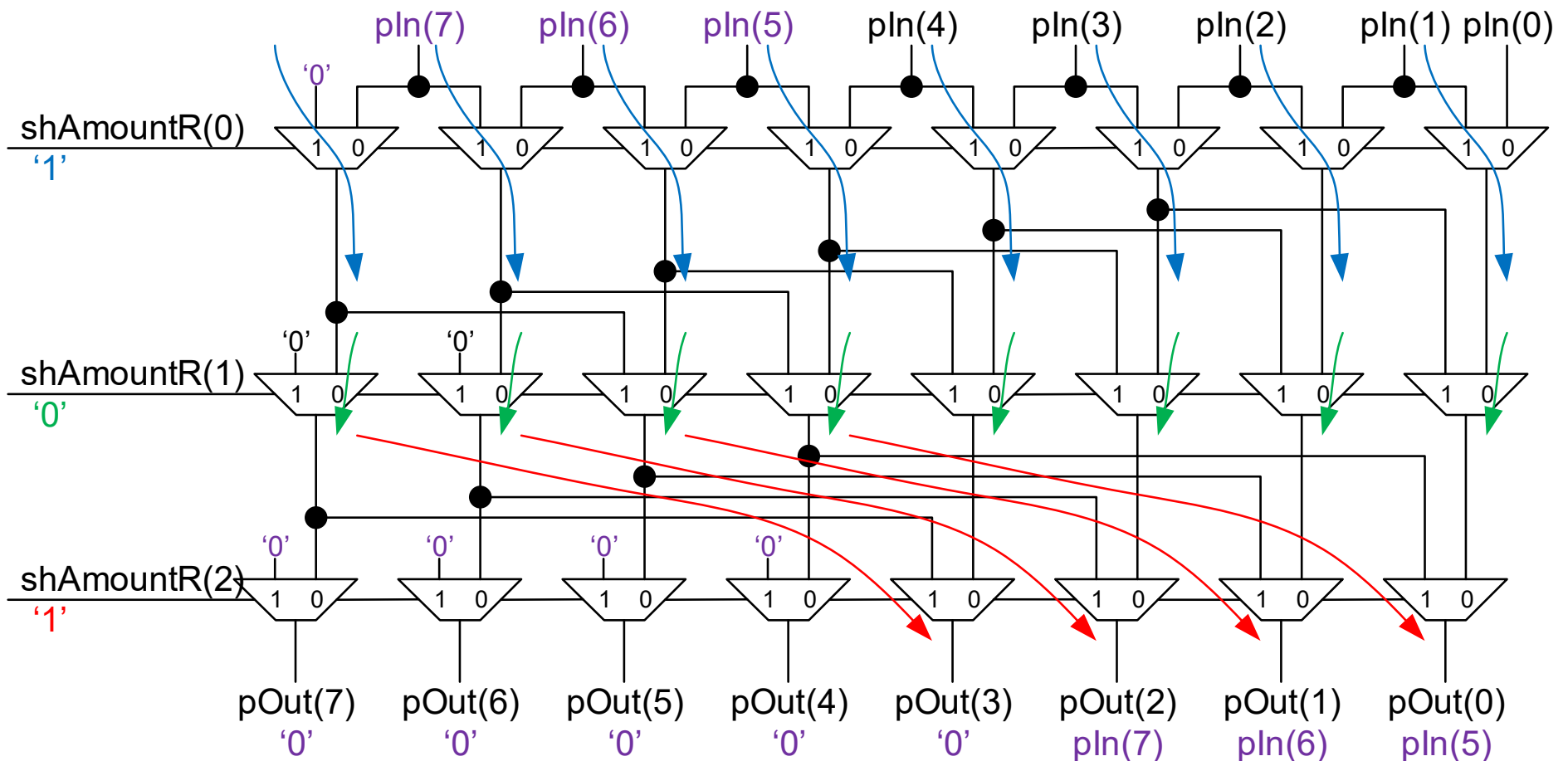
Exemplo de deslocamento lógico à direita e implementação com Muxs



Deslocamento realizado de forma combinatória (sem *clock*)  
Entrada  $shAmountR(i) = '1'$  provoca um deslocamento de  $2^i$

Realiza o deslocamento de “qualquer” número de bits sem necessitar de um sinal de relógio (de forma combinatória)

# Exemplo de Operação de um *Barrel Shifter* (Combinatório)



TPC:

Como adaptar o circuito para realizar deslocamentos aritméticos?

Como estender o circuito para suportar também deslocamentos à esquerda?

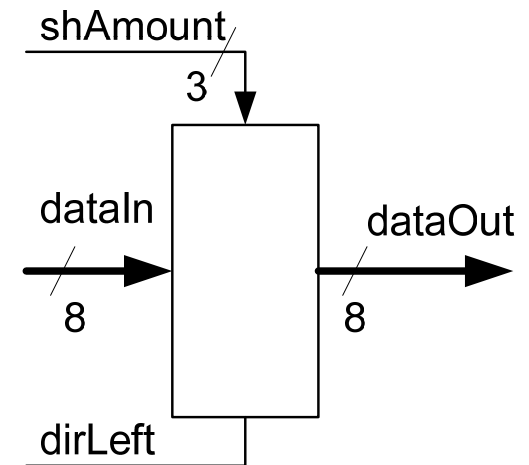
# Exemplo em VHDL de um Módulo Combinatório de Deslocamento

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.all;

entity CombShifter is
  port(dataIn      : in  std_logic_vector(7 downto 0);
        dirLeft    : in  std_logic;
        shAmount   : in  std_logic_vector(2 downto 0);
        dataOut     : out std_logic_vector(7 downto 0));
end CombShifter;
```

```
architecture Behavioral of CombShifter is
  signal s_shAmount : integer;
begin
  s_shAmount <= to_integer(unsigned(shAmount));

  process(dataIn, dirLeft, s_shAmount)
  begin
    if (dirLeft = '1') then
      dataOut <= std_logic_vector(shift_left(unsigned(dataIn), s_shAmount));
    else
      dataOut <= std_logic_vector(shift_right(unsigned(dataIn), s_shAmount));
    end if;
  end process;
end Behavioral;
```



## Deslocamento Lógico

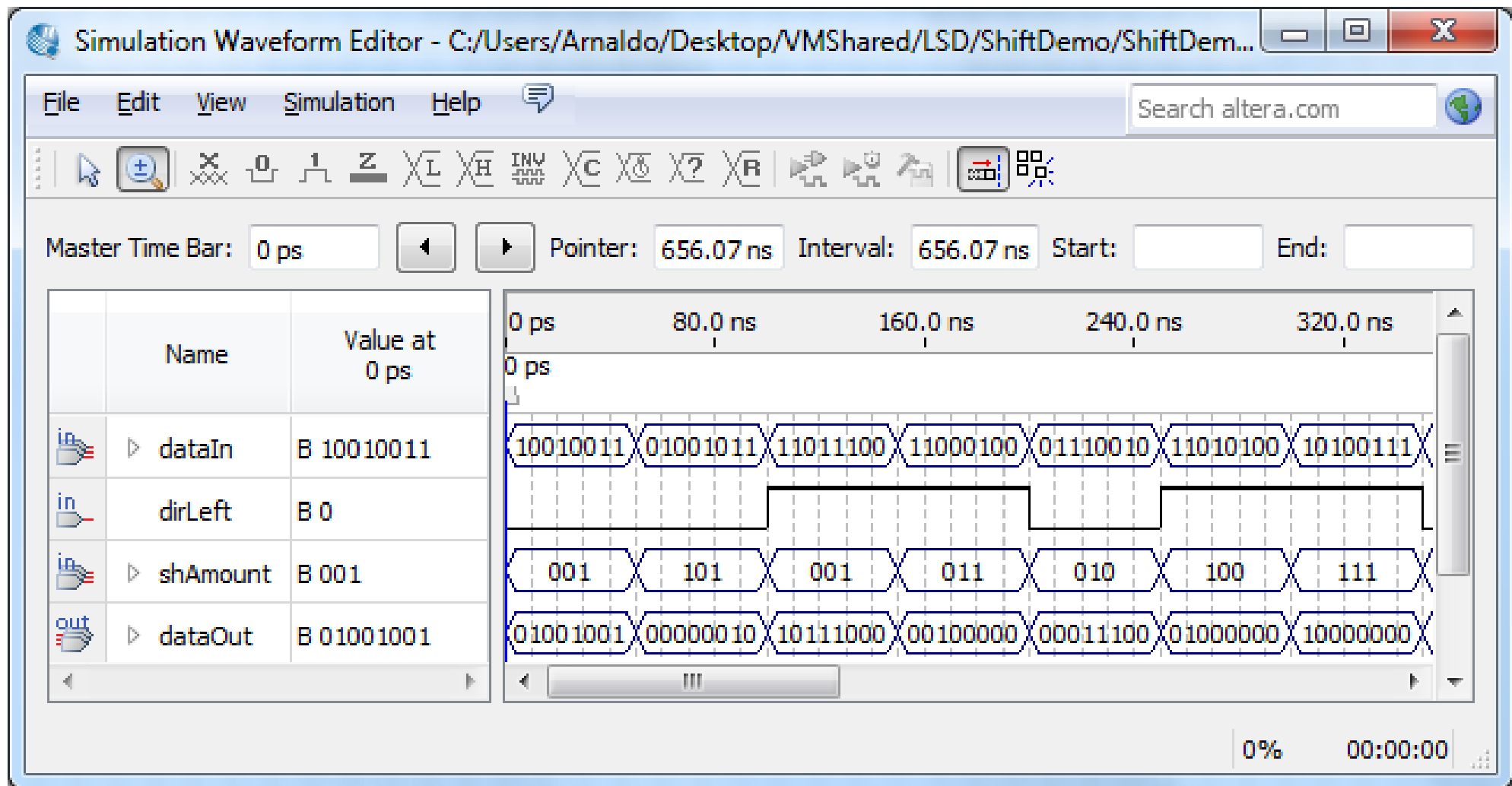
shift\_left(unsigned, integer)  
shift\_right(unsigned, integer)

## Deslocamento Aritmético

shift\_right(signed, integer)

A síntese deste módulo resulta num *Barrel Shifter*

# Simulação do Módulo Combinatório de Deslocamento



# Comentários Finais

- No final desta aula e do trabalho prático 6 de LSD, deverá ser capaz de:
  - Modelar em VHDL módulos de deslocamento
    - Sequenciais
    - Combinatórios

(o trabalho prático 5 é sobre parametrização de componentes em VHDL – abordada nas aulas TP 3 e 4)

