Sistemas Reconfiguráveis
Eng. de Computação

Profs. Francisco Garcia e Antônio Hamilton Magalhães

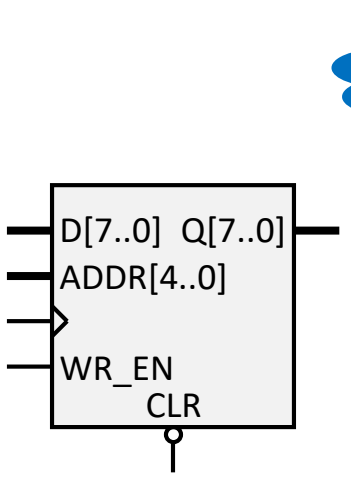
Aula 10 – Código sequencial em VHDL

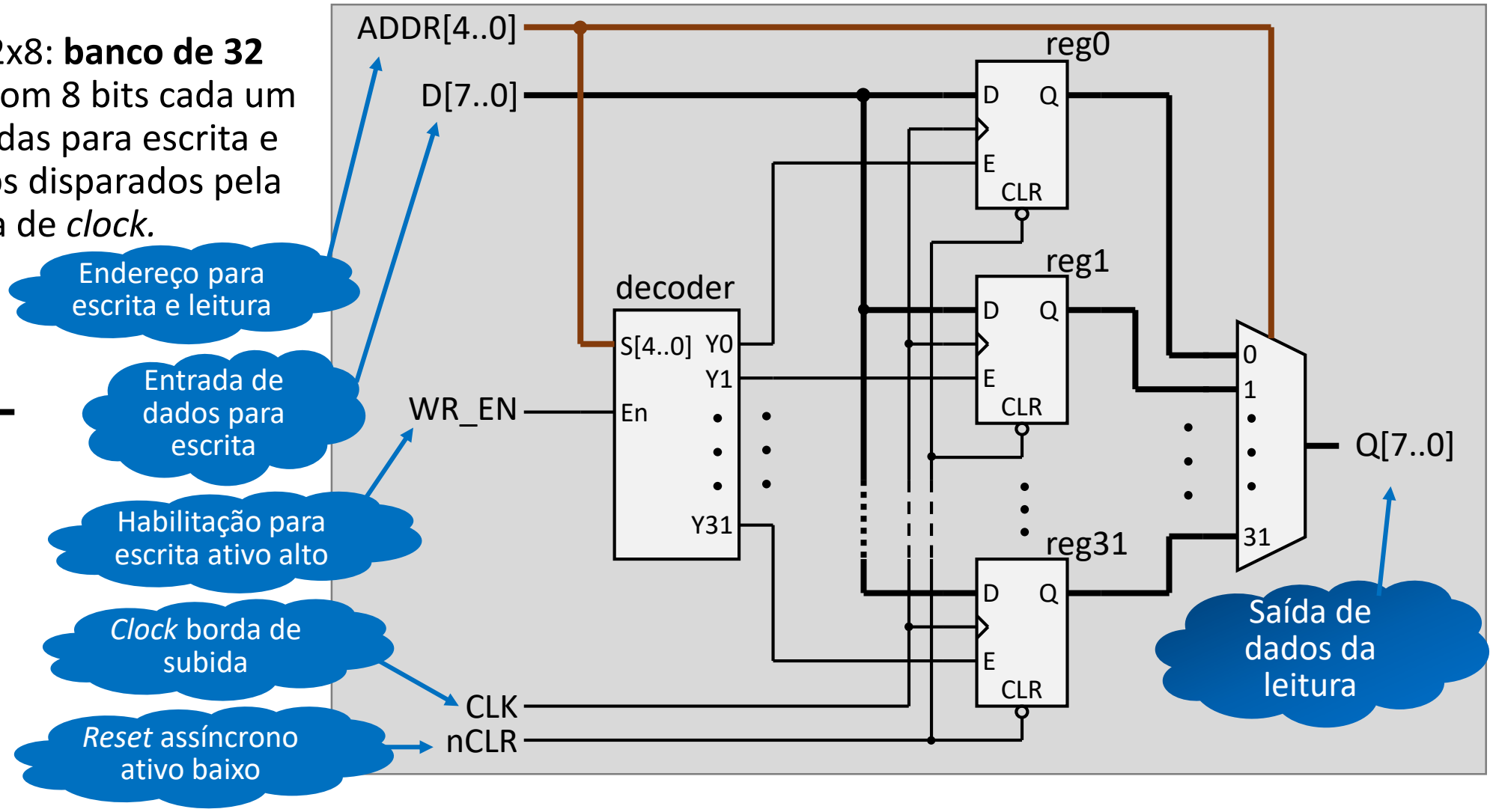
Exemplos: Banco de registradores (memória)

rev. B – 31/10/23

Exemplo reg32x8
Banco de registradores

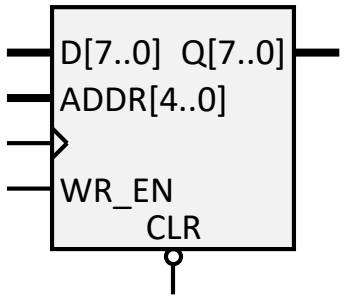
Exemplo reg_32x8: **banco de 32 registradores**, com 8 bits cada um e portas separadas para escrita e leitura. Flip-flops disparados pela borda de subida de *clock*.





Exemplo reg32x8
Banco de registradores

Exemplo reg_32x8: **banco de 32 registradores**, com 8 bits cada um e portas separadas para escrita e leitura. Flip-flops disparados pela borda de subida de *clock*.



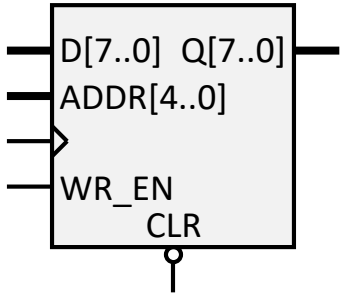

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;
-------------------------------------------------
ENTITY reg32x8 IS
    PORT (
        nclr  : IN STD_LOGIC;
        clk   : IN STD_LOGIC;
        wr_en : IN STD_LOGIC;
        d     : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
        addr  : IN STD_LOGIC_VECTOR(4 DOWNTO 0);
        q     : OUT STD_LOGIC_VECTOR(7 DOWNTO 0)
    );
END ENTITY;
-------------------------------------------------
```

Usado para a conversão de tipos

Continua na próxima página

Exemplo reg32x8
Banco de registradores

Exemplo reg_32x8: **banco de 32 registradores**, com 8 bits cada um e portas separadas para escrita e leitura. Flip-flops disparados pela borda de subida de *clock*.

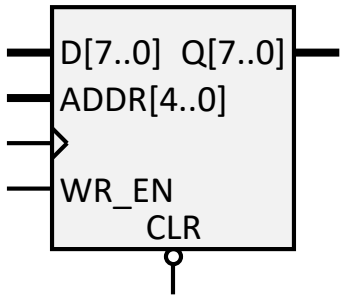



```
ARCHITECTURE arch1 OF reg32x8 IS
    TYPE mem_type IS ARRAY(0 TO 31) OF
        STD_LOGIC_VECTOR(7 DOWNTO 0);
    SIGNAL mem_reg : mem_type;
    SIGNAL addr_int : INTEGER RANGE 0 TO 31;
BEGIN
    ----------- Conversão:
    addr_int <= TO_INTEGER(UNSIGNED(addr));
```

Continua na próxima página

Exemplo reg32x8
Banco de registradores

Exemplo reg_32x8: **banco de 32 registradores**, com 8 bits cada um e portas separadas para escrita e leitura. Flip-flops disparados pela borda de subida de *clock*.

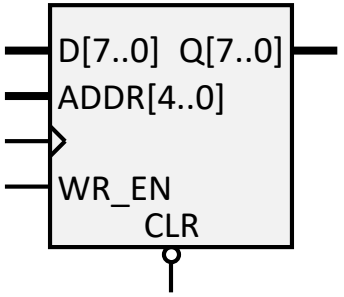



```
    ---------- Escrita:
    PROCESS(nclr, clk)
    BEGIN
        IF nclr = '0' THEN
            mem_reg <= (OTHERS => (OTHERS => '0'));
        ELSIF RISING_EDGE(clk) THEN
            IF wr_en = '1' THEN
                mem_reg(addr_int) <= d;
            END IF;
        END IF;
    END PROCESS;
    ---------- Leitura:
    q <= mem_reg(addr_int);
END arch1;
```

Outra opção na próxima página

Exemplo reg32x8
Banco de registradores

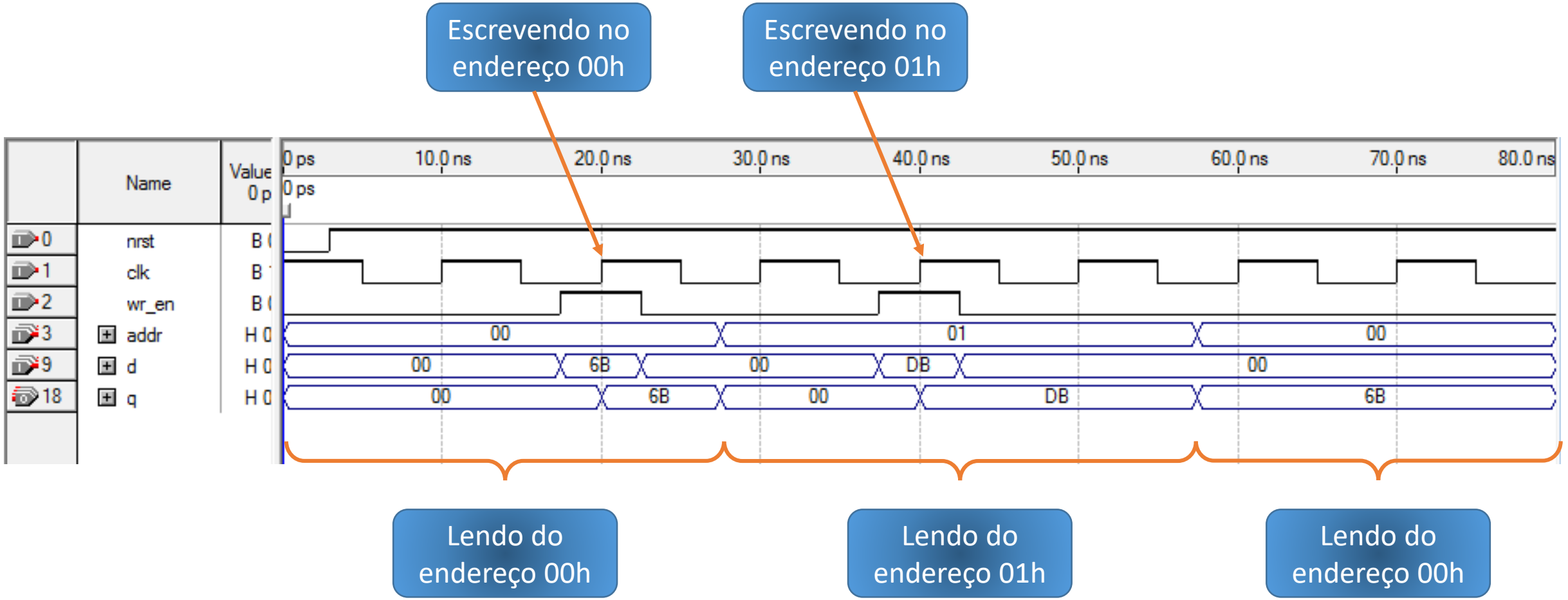
Exemplo reg_32x8: **banco de 32 registradores**, com 8 bits cada um e portas separadas para escrita e leitura. Flip-flops disparados pela borda de subida de *clock*.



Outra opção para limpar todos os ff

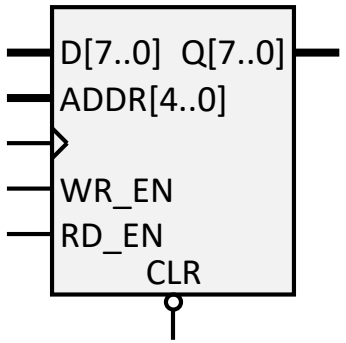
```
    ---------- Escrita:
    PROCESS(nclr, clk)
    BEGIN
        IF nclr = '0' THEN
            FOR i IN 0 TO 31 LOOP
                mem_reg(i)  <= (OTHERS => '0');
            END LOOP;
        ELSIF RISING_EDGE(clk) THEN
            IF wr_en = '1' THEN
                mem_reg(addr_int) <= d;
            END IF;
        END IF;
    END PROCESS;
    ---------- Leitura:
    q <= mem_reg(addr_int);
END arch1;
```

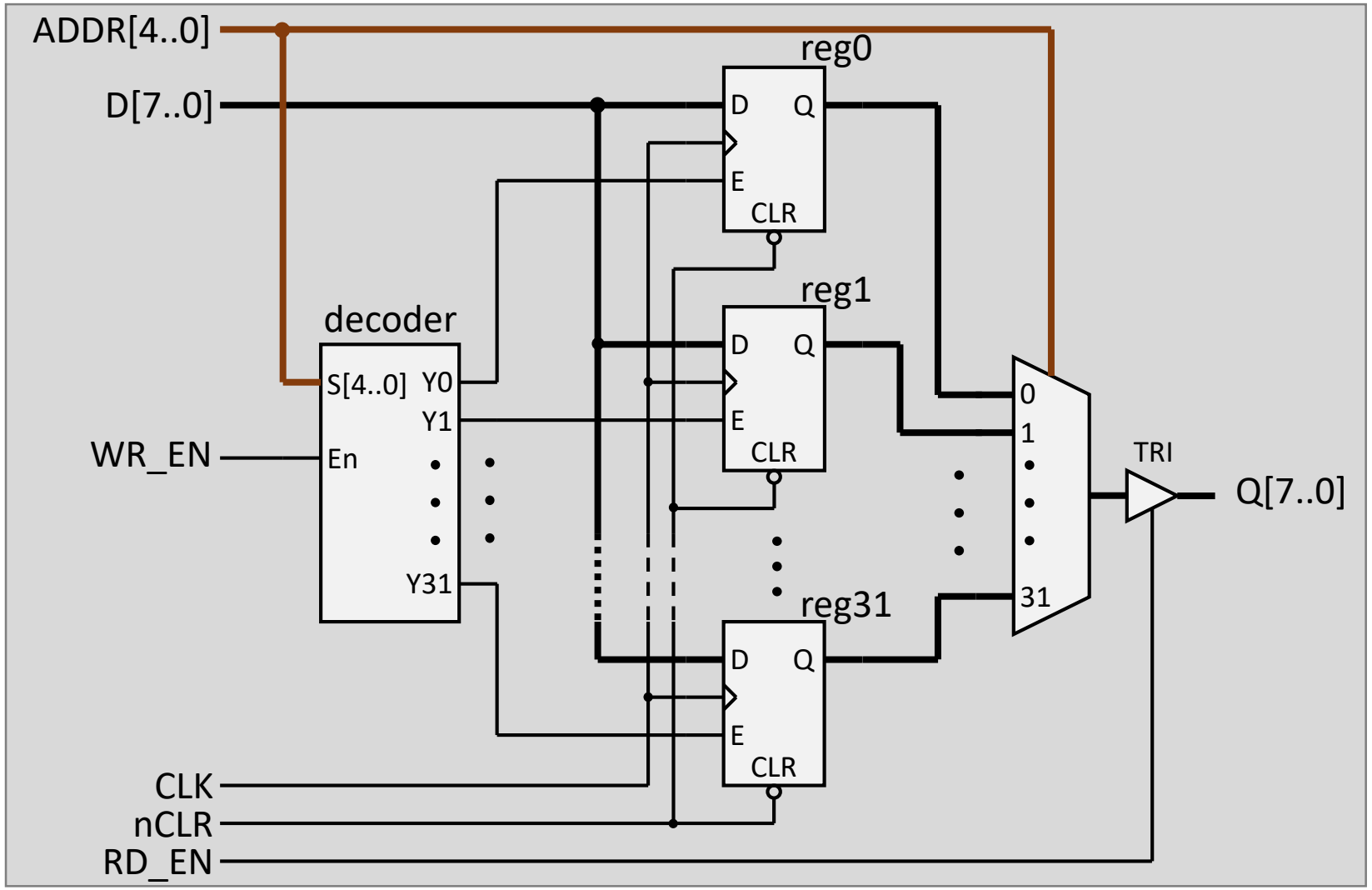
Exemplo reg32x8
Banco de registradores



Exemplo reg32x8z
Banco de registradores

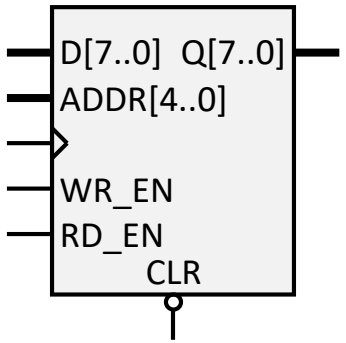
Exemplo reg_32x8z: **banco de 32 registradores**, igual ao anterior, mas com saída ***tri-state***, controlada pelo sinal rd_en, ativo em nível alto.





Exemplo reg32x8z
Banco de registradores

Exemplo reg_32x8z: **banco de 32 registradores**, igual ao anterior, mas com saída ***tri-state***, controlada pelo sinal rd_en, ativo em nível alto.



O código é o mesmo do exemplo anterior, exceto pelo acréscimo da entrada **rd_en** na ENTITY e pela lógica de leitura, descrita abaixo:

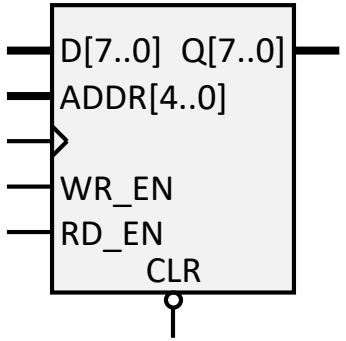
```
    ----------- Leitura:
    PROCESS(rd_en, mem_reg, addr_int)
    BEGIN
        IF rd_en = '1' THEN
            q <= mem_reg(addr_int);
        ELSE
            q <= (OTHERS => 'Z');
        END IF;
    END PROCESS;

END arch1;
```

Código sequencial

Na próxima página, exemplo de leitura usando código concorrente

Exemplo reg32x8z
Banco de registradores

Exemplo reg_32x8z: **banco de 32 registradores**, igual ao anterior, mas com saída ***tri-state***, controlada pelo sinal rd_en, ativo em nível alto.

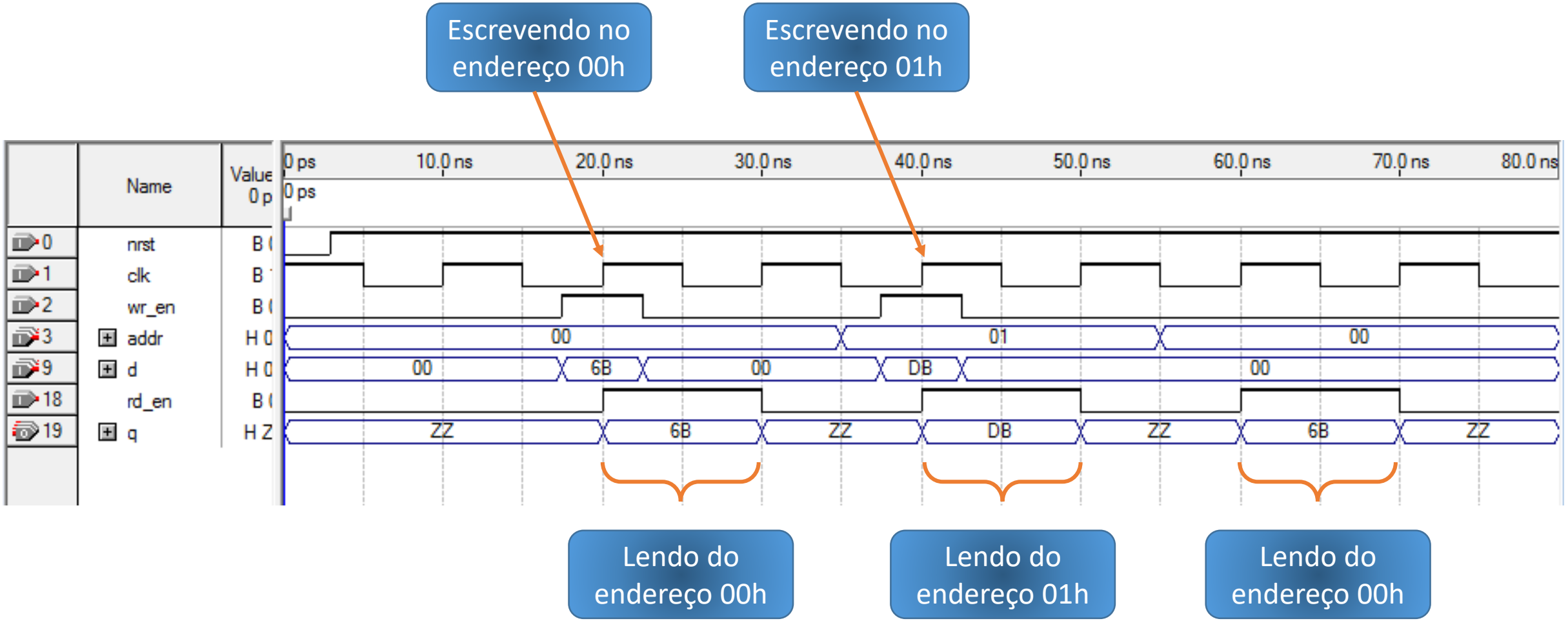


O código é o mesmo do exemplo anterior, exceto pelo acréscimo da entrada rd_en na ENTITY e pela lógica de leitura, descrita abaixo:

```
    ----------- Leitura:
    q <= mem_reg(addr_int) WHEN rd_en = '1' ELSE
         (OTHERS => 'Z');

END arch1;
```

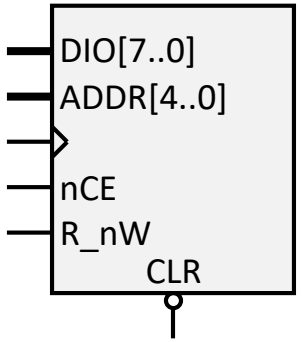
Código concorrente

Exemplo reg32x8z
Banco de registradores

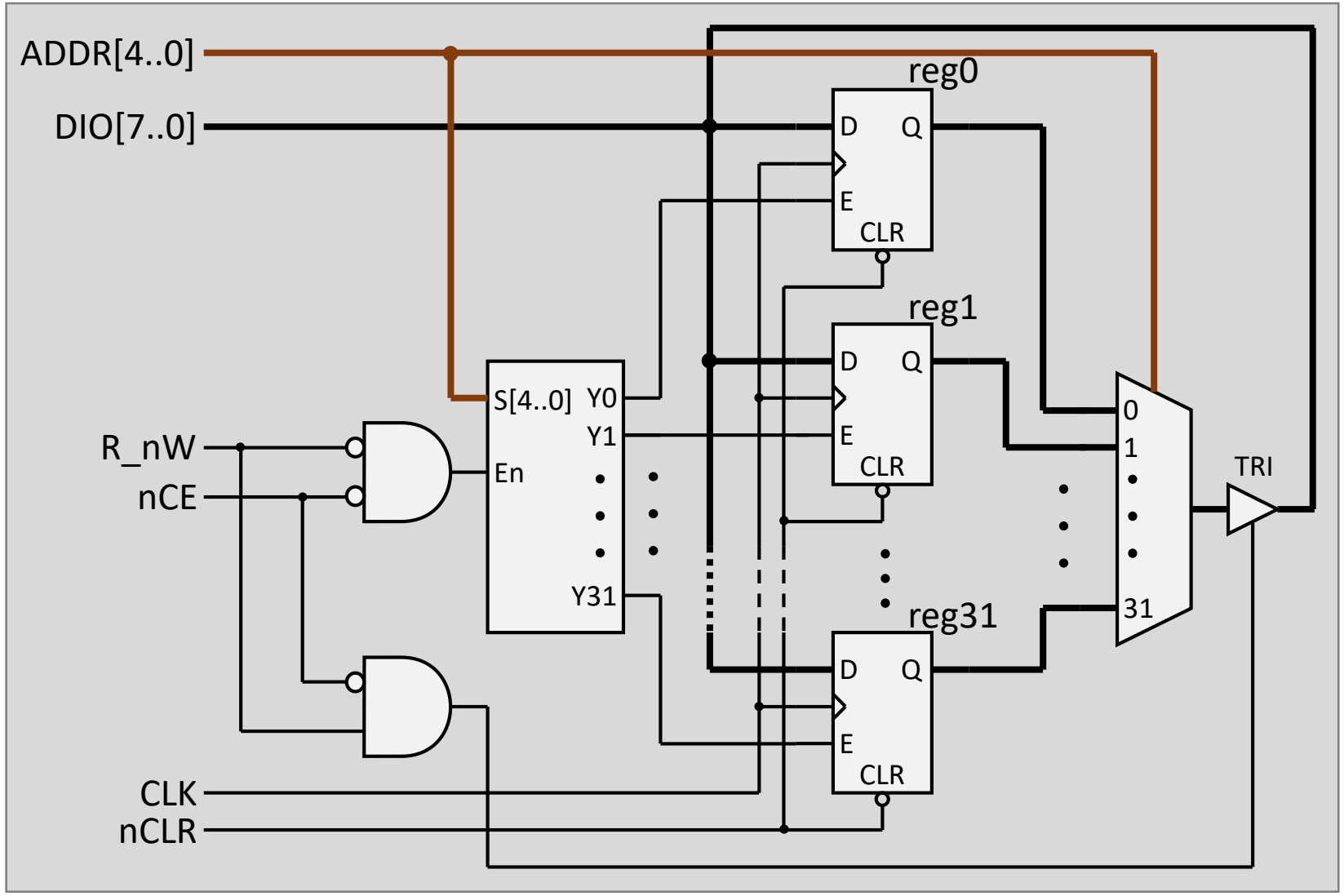


Exemplo reg32x8b
Banco de registradores

Exemplo reg_32x8b: **banco de 32 registradores**, igual ao anterior, mas com **porta bidirecional**

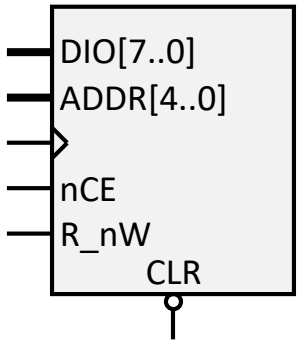


nCE	R_nW	Operação	DIO[7..0]
1	X	Nenhuma	Hi-Z
0	0	Escrita	Entrada
0	1	Leitura	Saída



Exemplo reg32x8b
Banco de registradores

Exemplo reg_32x8b: **banco de 32 registradores**, igual ao anterior, mas com **porta bidirecional**



Modo INOUT

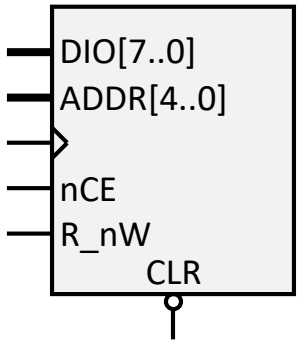
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;
-----------------------------------------------
ENTITY reg32x8b IS
    PORT (
        nclr  : IN STD_LOGIC;
        clk   : IN STD_LOGIC;
        nce   : IN STD_LOGIC;
        r_nw  : IN STD_LOGIC;
        addr  : IN STD_LOGIC_VECTOR(4 DOWNTO 0);
        dio   : INOUT STD_LOGIC_VECTOR(7 DOWNTO 0)
    );
END ENTITY;
-----------------------------------------------
```

nCE	R_nW	Operação	DIO[7..0]
1	X	Nenhuma	Hi-Z
0	0	Escrita	Entrada
0	1	Leitura	Saída

Continua na próxima página

Exemplo reg32x8b
Banco de registradores

Exemplo reg_32x8b: **banco de 32 registradores**, igual ao anterior, mas com **porta bidirecional**



nCE	R_nW	Operação	DIO[7..0]
1	X	Nenhuma	Hi-Z
0	0	Escrita	Entrada
0	1	Leitura	Saída

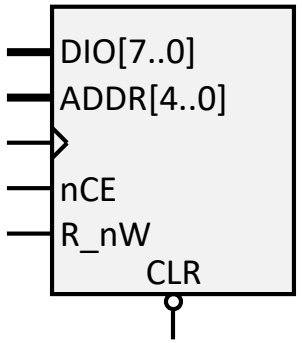
Essa parte continua igual (exceto pelo nome da ENTITY):

```
ARCHITECTURE arch1 OF reg32x8b IS
    TYPE mem_type IS ARRAY(0 TO 31) OF
        STD_LOGIC_VECTOR(7 DOWNTO 0);
    SIGNAL mem_reg : mem_type;
    SIGNAL addr_int : INTEGER RANGE 0 TO 31;
BEGIN
    ----------- Conversão:
    addr_int <= TO_INTEGER(UNSIGNED(addr));
```

Continua na próxima página

Exemplo reg32x8b
Banco de registradores

Exemplo reg_32x8b: **banco de 32 registradores**, igual ao anterior, mas com **porta bidirecional**



Modificado em relação aos exemplos anteriores

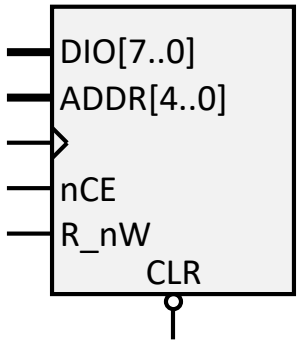
```
---------- Escrita:
PROCESS(nclr, clk)
BEGIN
    IF nclr = '0' THEN
        mem_reg <= (OTHERS => (OTHERS => '0'));
    ELSIF RISING_EDGE(clk) THEN
        IF nce = '0' AND r_nw = '0' THEN
            mem_reg(addr_int) <= dio;
        END IF;
    END IF;
END PROCESS;
```

nCE	R_nW	Operação	DIO[7..0]
1	X	Nenhuma	Hi-Z
0	0	Escrita	Entrada
0	1	Leitura	Saída

Continua na próxima página

Exemplo reg32x8b
Banco de registradores

Exemplo reg_32x8b: **banco de 32 registradores**, igual ao anterior, mas com **porta bidirecional**



Modificado em relação aos exemplos anteriores

```
    ----------- Leitura:
    PROCESS(nce, r_nw, mem_reg, addr_int)
    BEGIN
        IF nce = '0' AND r_nw = '1' THEN
            dio <= mem_reg(addr_int);
        ELSE
            dio <= (OTHERS => 'Z');
        END IF;
    END PROCESS;

END arch1;
```

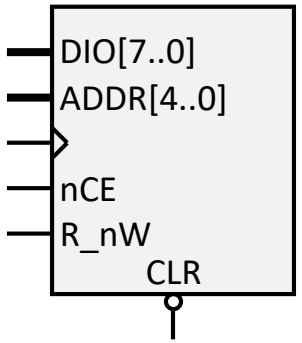
Código sequencial

nCE	R_nW	Operação	DIO[7..0]
1	X	Nenhuma	Hi-Z
0	0	Escrita	Entrada
0	1	Leitura	Saída

Na próxima página, exemplo de leitura usando código concorrente

Exemplo reg32x8b
Banco de registradores

Exemplo reg_32x8b: **banco de 32 registradores**, igual ao anterior, mas com **porta bidirecional**



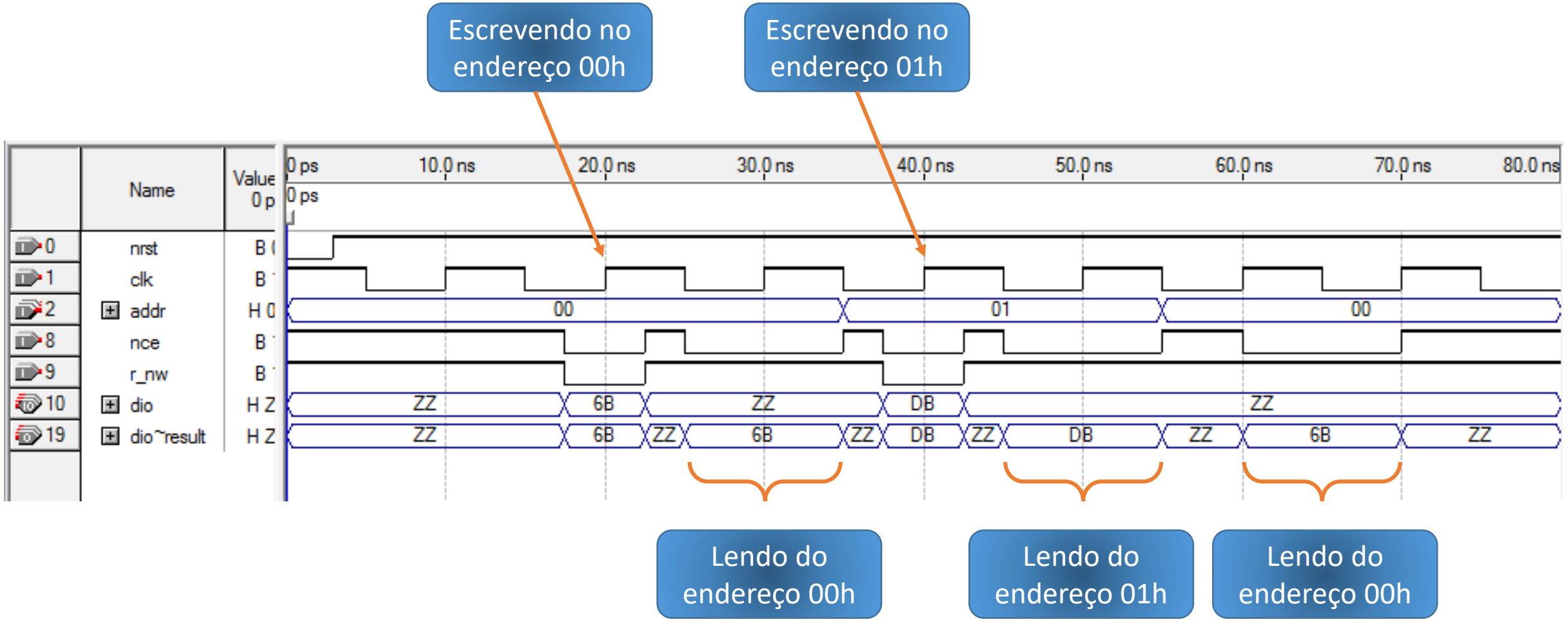
Modificado em relação aos exemplos anteriores

```
    ----------- Leitura:
    dio <= mem_reg(addr_int)
             WHEN nce = '0' AND r_nw = '1' ELSE
           (OTHERS => 'Z');

END arch1;
```

Código concorrente

nCE	R_nW	Operação	DIO[7..0]
1	X	Nenhuma	Hi-Z
0	0	Escrita	Entrada
0	1	Leitura	Saída

Exemplo reg32x8b
Banco de registradores



Exercícios
Banco de registradores

1) Implementar e testar no simulador (simulação funcional) um banco de registradores semelhante ao exemplo reg32x8b, porém com 64 registros de 16 bits cada.

   Obs.: Para a simulação de sinais bidirecionais, são necessários dois sinais no diagrama temporal. Um representa o que vem do ambiente externo para o projeto em teste. Outro representa o resultado na porta do projeto e é criado automaticamente pelo simulador, recebendo um sufixo "**~result**". Nesse sinal é que irão aparecer os dados de saída, quando a leitura for ativada.

   Veja as instruções para a simulação a seguir.

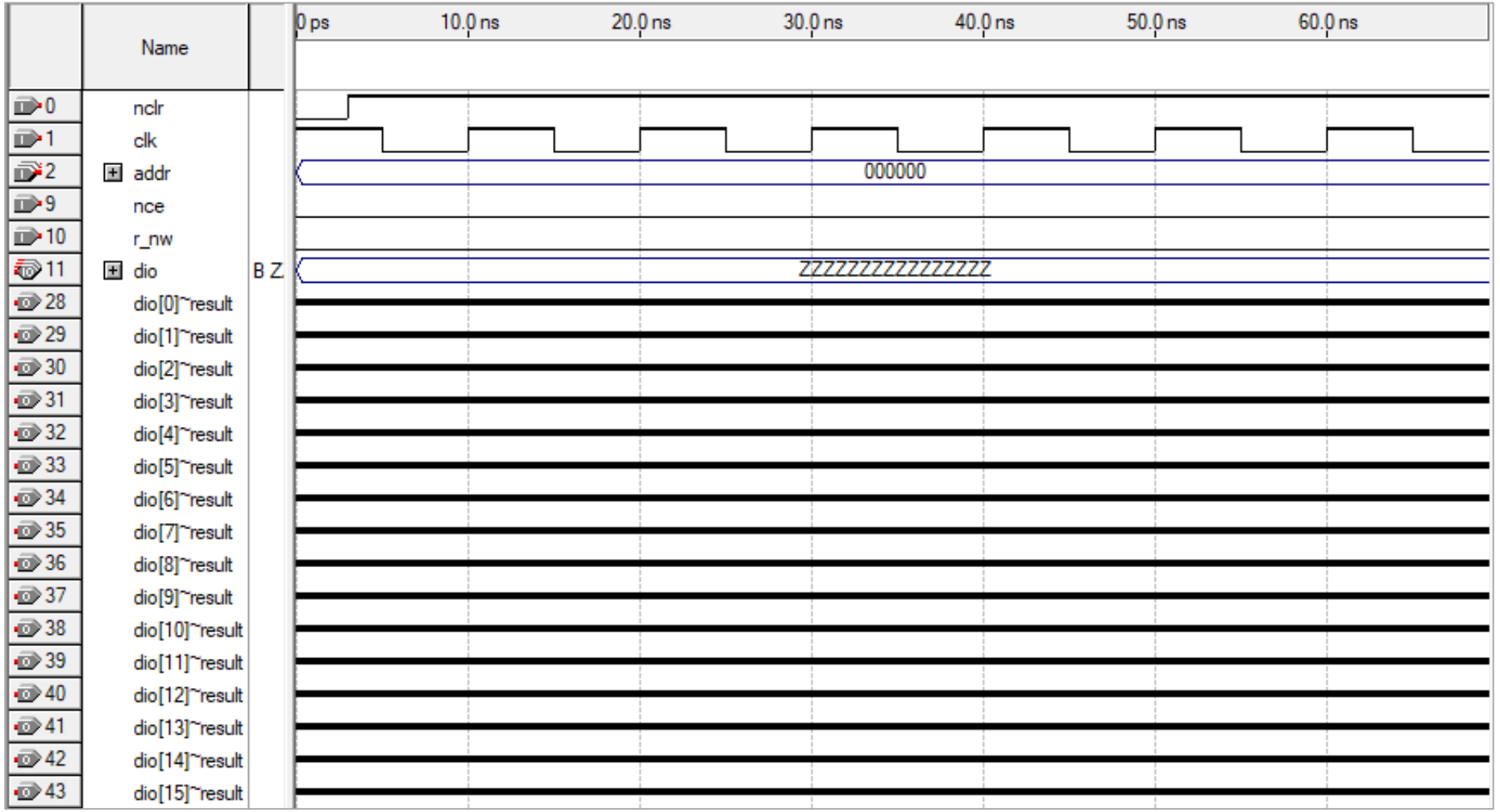
2) Implementar e testar no módulo DE-2 o exemplo reg32x8, usando *pushbuttons* para os sinais **nrst** e **clk**, chaves para as demais entradas e LEDs para as saídas.

Exercícios
Banco de registradores

Quando o sinal “**~result**” é criado, ele não está agrupado, aparecendo como uma série de sinais individuais. Eles devem ser agrupados. Para isso:

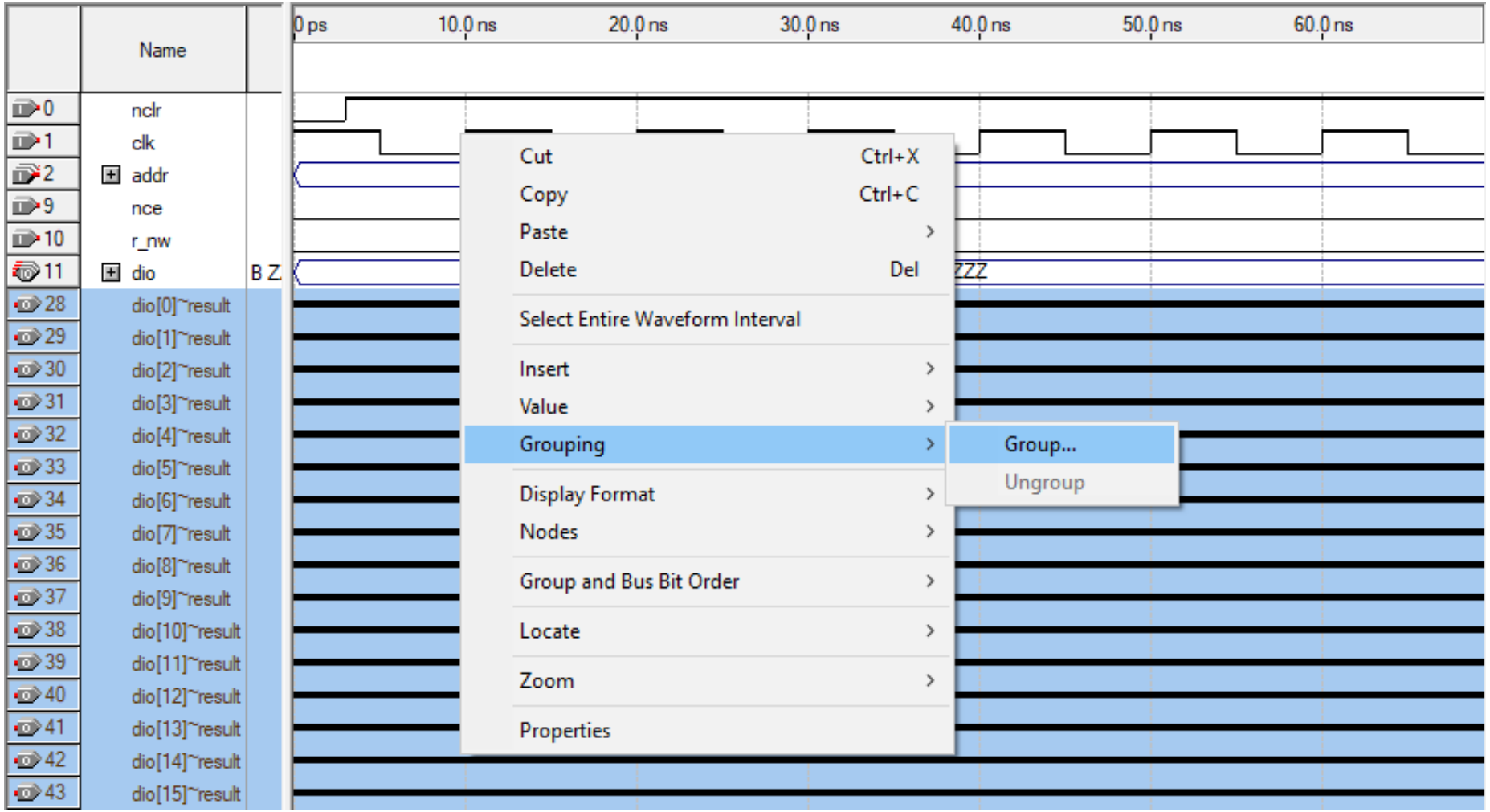
Selecionar o primeiro desses sinais, clicando com o botão esquerdo do *mouse* em cima de seu nome.

Selecionar todos os sinais, pressionando “Ctrl” enquanto clica no último sinal.



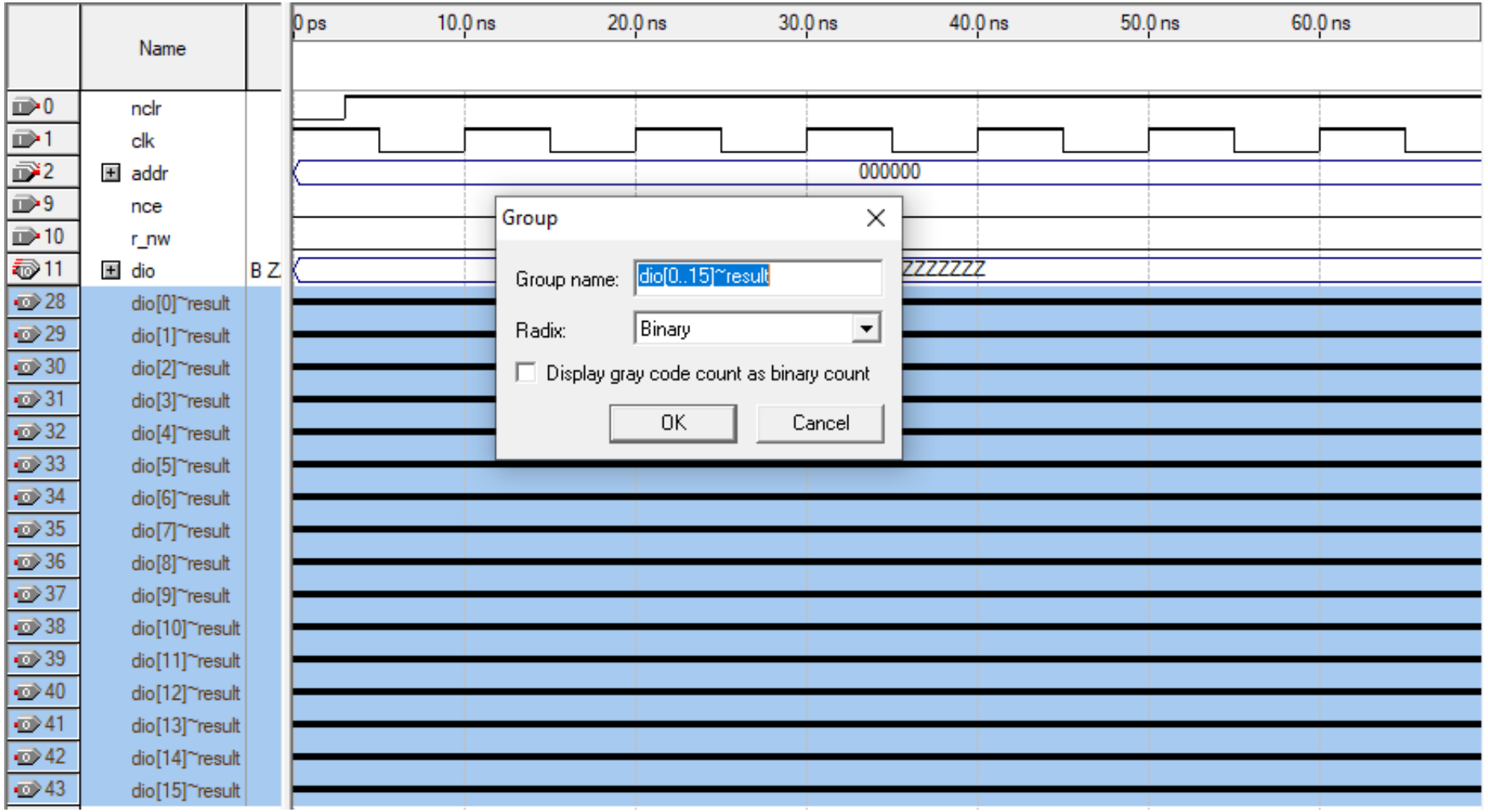
Exercícios
Banco de registradores

Clicar com o botão direito do mouse, selecionar “Grouping” e depois “Group...”



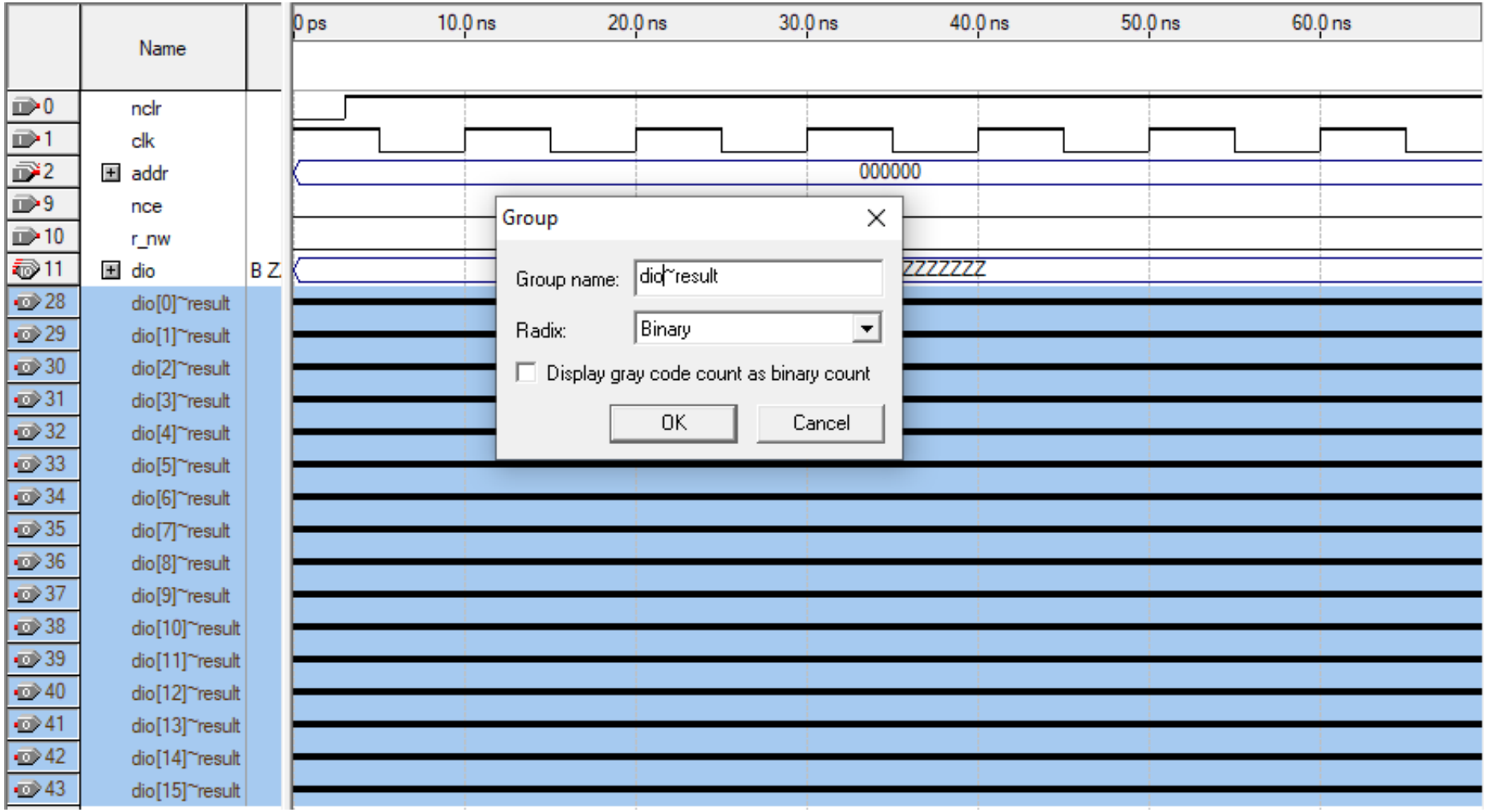
Exercícios
Banco de registradores

Na janela aberta, editar o nome do grupo, deletando os colchetes e tudo que está dentro deles.



Exercícios
Banco de registradores

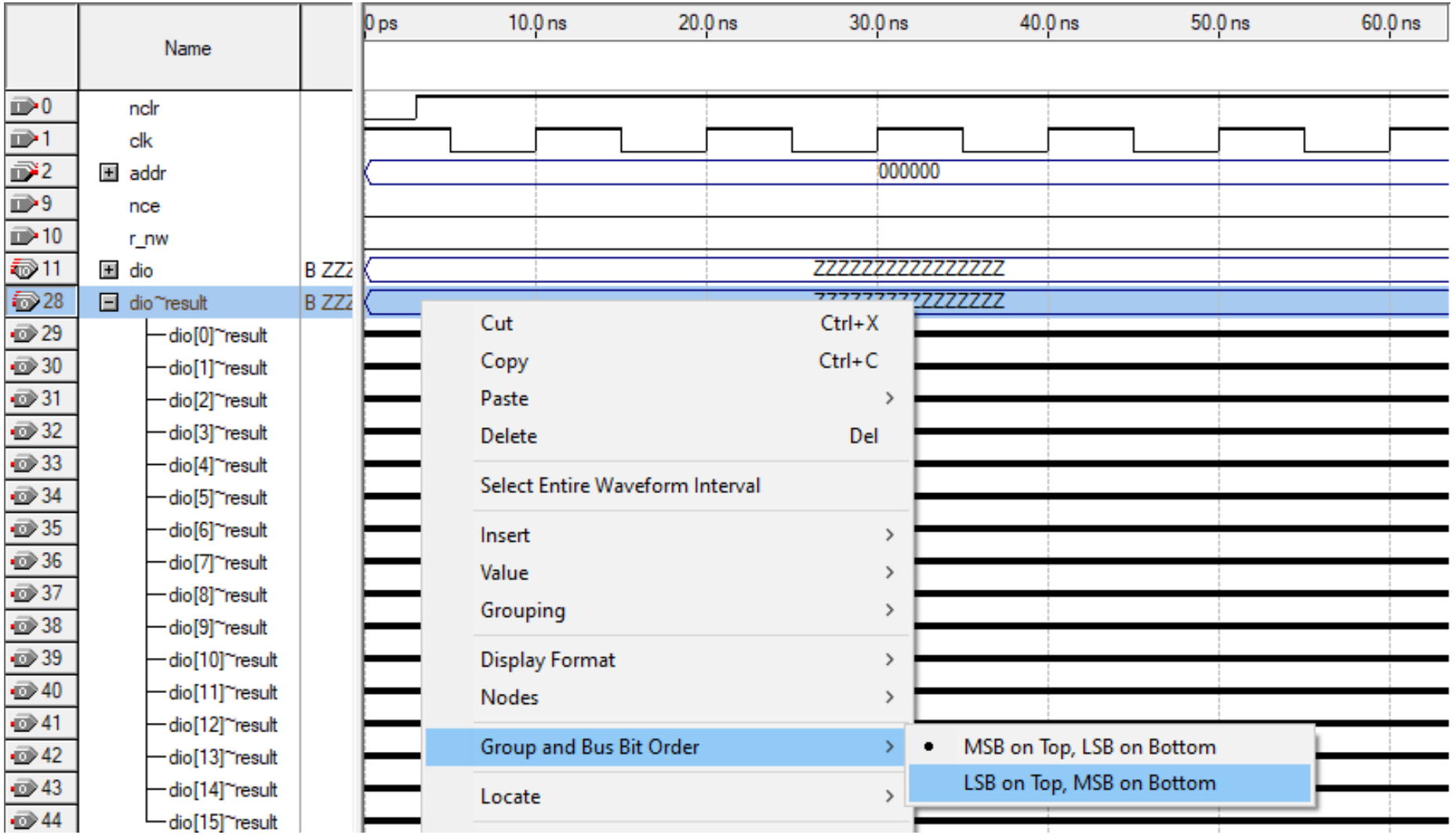
Selecionar “OK” para criar o grupo



Exercícios
Banco de registradores

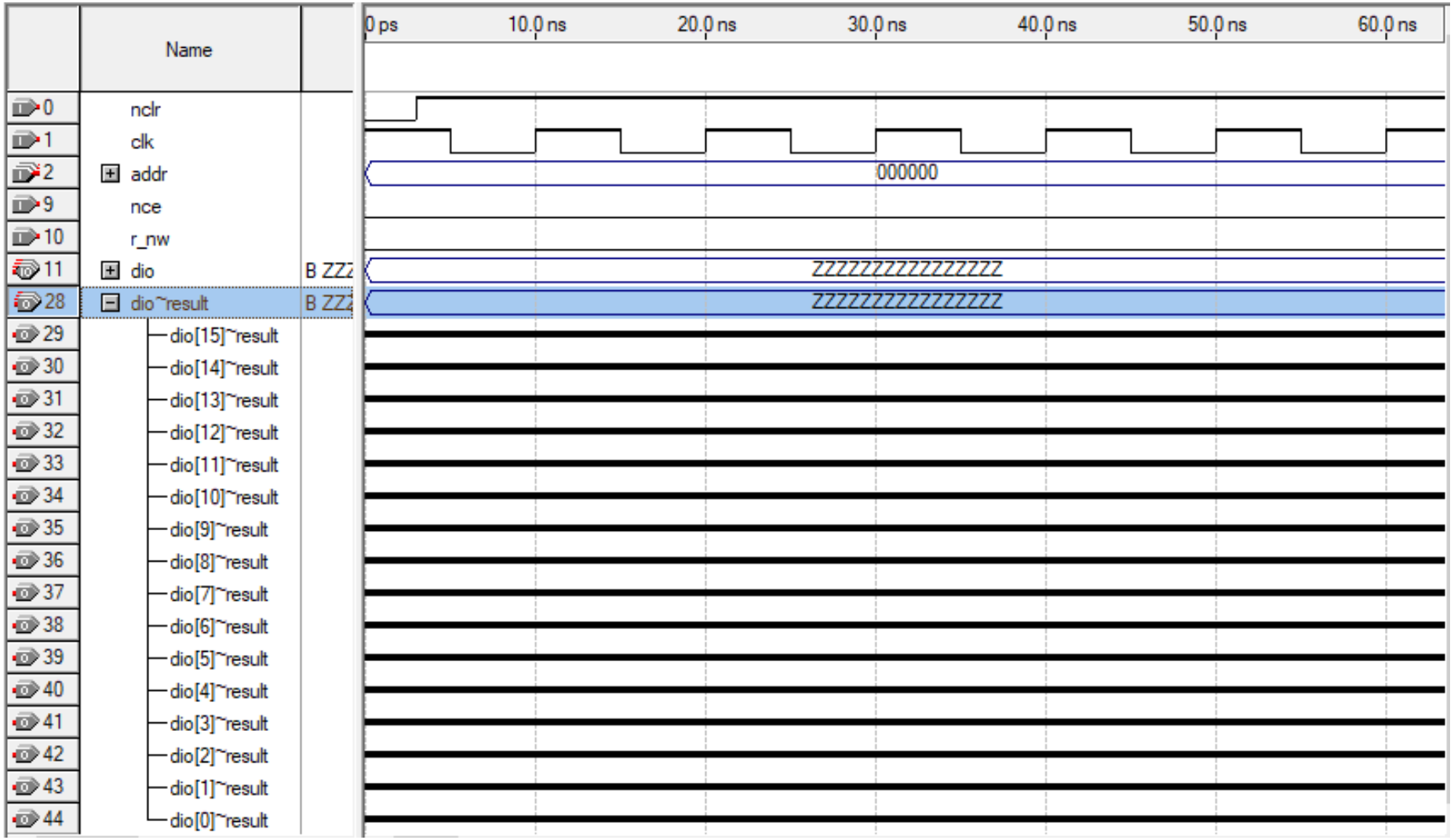
No entanto, a ordem dos bits dentro do grupo ficou com o bit 0 em cima, enquanto o sinal **dio** tem o bit 15 em cima. Para inverter a ordem:

Selecionar o grupo **dio~result**. Clicar com o botão direito do mouse, selecionar "Group and Bus Bit Order" e depois inverter a seleção na segunda janela.



Exercícios
Banco de registradores

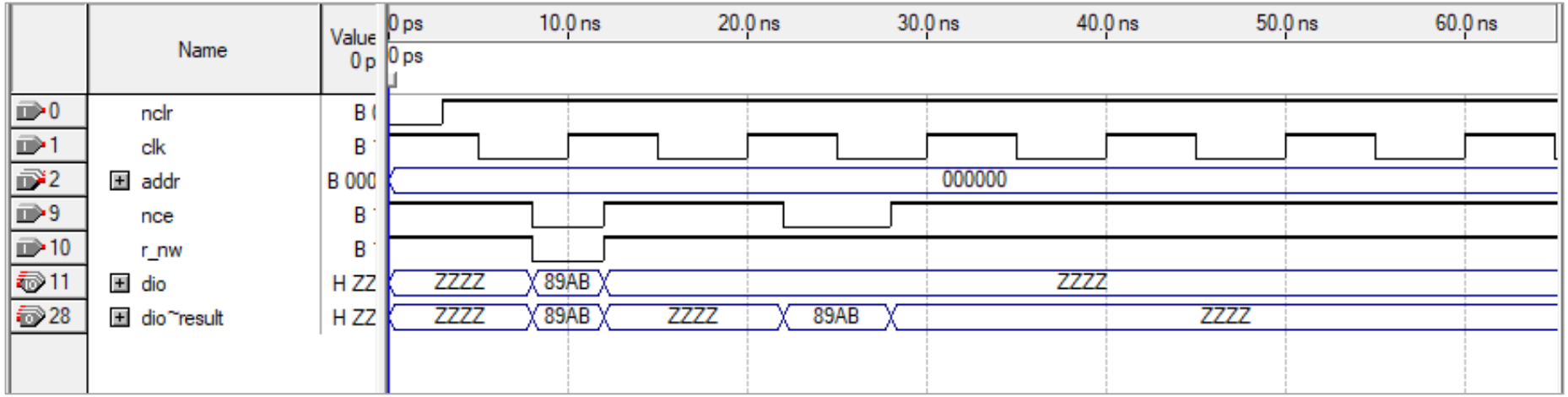
Ao lado é mostrado o grupo criado e na ordem correta.



Exercícios
Banco de registradores

O sinal **dio** deverá ser configurado para alta impedância (‘Z’) todo o tempo, exceto para as operações de escrita, quando deve ser configurado com o dado a ser escrito.

Não é necessário configurar o sinal **dio~result**. Ele mostrará automaticamente os dados nas operações de leitura.



Obs.: Para a simulação mostrada acima, os sinais **dio** e **dio~result** foram configurados para mostrar os seus valores em hexadecimal, para facilitar a interpretação dos resultados. Para isso:

Clicar no nome do sinal com o botão direito do mouse. Selecionar “Properties” e depois, no campo “Radix”, selecionar “Hexadecimal”

Fim