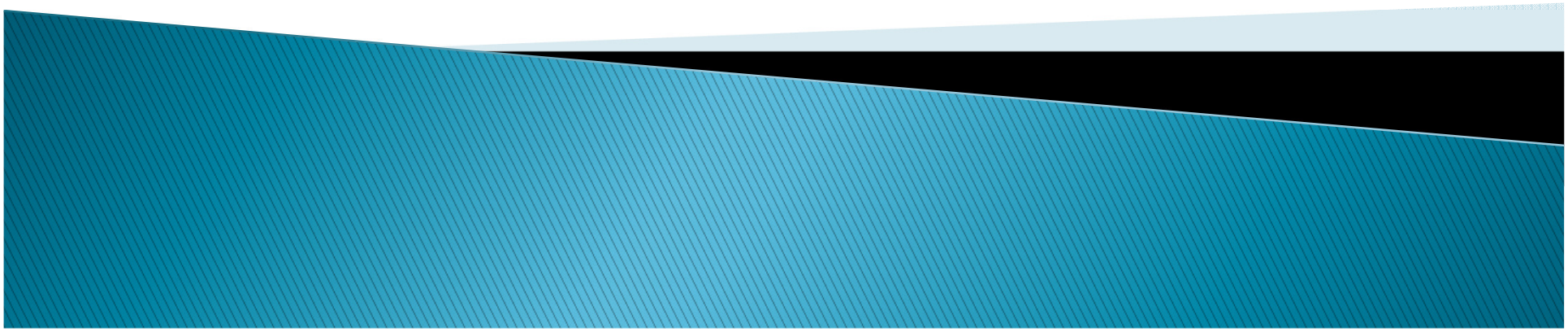
Laboratório de Eletrônica Digital
SEL-EESC-USP

EESC · USP

Introdução a VHDL

Aula 4

Professora Luiza Maria Romeiro Codá



Comandos em VHDL – Sequenciais
NULL

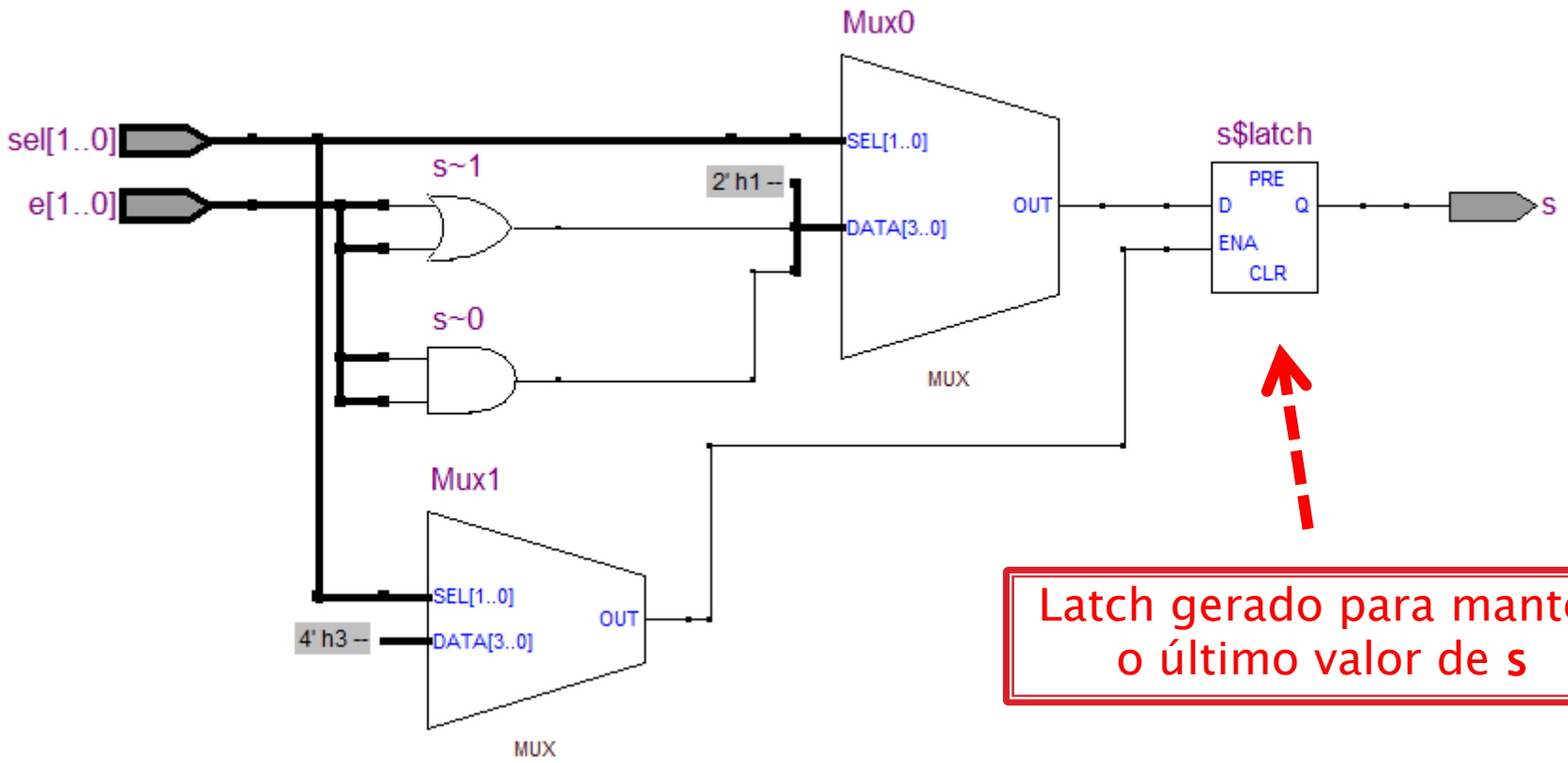
O comando "NULL" não realiza nenhuma operação, a execução é passada para o próximo comando.
Serve para indicar que nenhuma operação deve ser realizada em uma condição de CASE-WHEN ou IF-ELSE.

Comandos em VHDL – Sequenciais
NULL
Exemplo de Aplicação com CASE-WHEN

Na descrição a seguir, a saída **s** recebe o resultado de uma entre duas operações lógicas, selecionadas pelo sinal **sel** ("00" indica AND e "01" indica OR). Caso **sel** seja diferente destes valores, a saída não deve ser alterada.

```
ENTITY ex_mux1 IS
    PORT(e, sel : IN  BIT_VECTOR (1 DOWNTO 0);
         s      : OUT BIT);
END ex_mux1;

ARCHITECTURE a OF ex_mux1 IS
BEGIN
    PROCESS (e, sel)
    BEGIN
        CASE sel IS
            WHEN "00" => s <= e(0) AND e(1);
            WHEN "01" => s <= e(0) OR  e(1);
            WHEN OTHERS => NULL; -- saída s não se altera
        END CASE;
    END PROCESS;
END a;
```

Comandos em VHDL – Sequenciais
NULL
Exemplo de Aplicação com CASE-WHEN



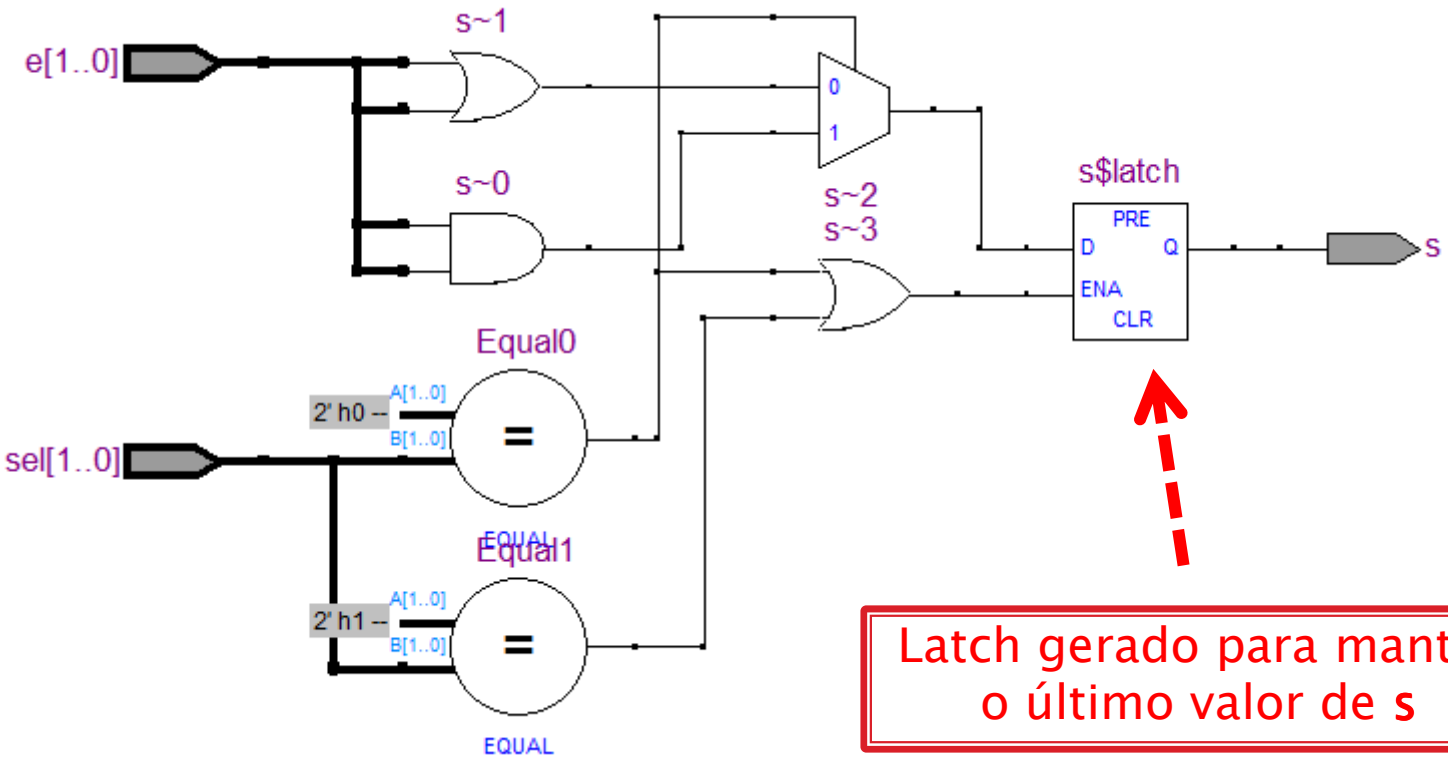
Latch gerado para manter o último valor de **s**

Comandos em VHDL – Sequenciais
NULL
Exemplo de Aplicação com IF-THEN-ELSE-END IF

Na descrição a seguir, a saída **s** recebe o resultado de uma entre duas operações lógicas, selecionadas pelo sinal **sel** ("00" indica AND e "01" indica OR). Caso **sel** seja diferente destes valores, a saída não deve ser alterada.

```
ENTITY ex_mux2 IS
    PORT(e, sel : IN  BIT_VECTOR(1 DOWNTO 0);
         s      : OUT BIT);
END ex_mux2;

ARCHITECTURE a OF ex_mux2 IS
BEGIN
    PROCESS (e, sel)
    BEGIN
        IF    sel = "00" THEN s <= e(0) AND e(1);
        ELSIF sel = "01" THEN s <= e(0) OR e(1);
        ELSE  NULL; -- Não é necessário colocar o ELSE
        END IF;
END PROCESS;
END a;
```

Comandos em VHDL – Sequenciais
NULL
Exemplo de Aplicação com IF-THEN-ELSE-END IF



Latch gerado para manter o último valor de **s**

Atributos – Conceito

Atributos são informações adicionais associadas a objetos (como por exemplo, a sinais) que possibilitam a verificação de transições de sinal e o modelamento de atrasos.

Biblioteca WORK:

ATRIBUTO	FUNÇÃO
`nome_do_sinal'EVENT`	Verdadeiro se ocorreu uma troca de valor do sinal no ciclo corrente de simulação, falso caso contrário.

Atributos – Aplicações
Verificação de transição de um sinal

Biblioteca WORK:

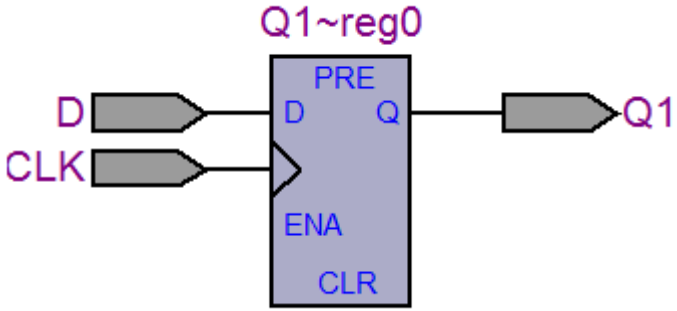
DESCRIÇÃO VHDL	FUNÇÃO
`clk'EVENT AND clk ='1'`	Seleção da transição positiva do sinal clk
`clk'EVENT AND clk = '0'`	Seleção da transição negativa do sinal clk

Biblioteca IEEE pacote 1164:

DESCRIÇÃO VHDL	FUNÇÃO
`RISING_EDGE(clk)`	Seleção da transição positiva do sinal clk
`FALLING_EDGE(clk)`	Seleção da transição negativa do sinal clk

Obs.: As funções RISING_EDGE() e FALLING_EDGE() apenas retornam TRUE se houver uma transição de 0 para 1 ou de 1 para 0, ignorando transições envolvendo outros valores (X, –, H, etc.).

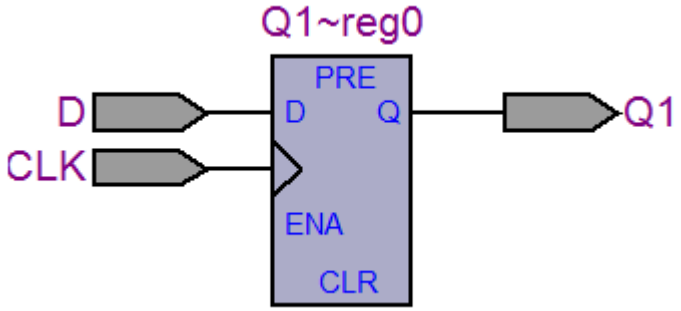
FF tipo D disparado na borda de subida do clock Usando IF-THEN




```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY FF_tipoD IS
    PORT(D, CLK : IN     STD_LOGIC;
         Q1     : BUFFER STD_LOGIC); -- Saída de
          -- Q1     : OUT STD_LOGIC); opção 2
END FF_tipoD;

ARCHITECTURE a OF FF_tipoD IS
--SIGNAL sinal_Q1: STD_LOGIC; opção 2
BEGIN
    PROCESS(CLK)
    BEGIN
        IF (CLK'EVENT AND CLK = '1') THEN -- Aguarda borda de subida do CLK
            Q1 <= D;
      --sinal_Q1 <=D; opção 2
       -- ELSE Q1 <= Q1; Desnecessário incluir
        END IF;
    END PROCESS;

      --Q1 <= sinal_Q1; opção 2
END a;
```

Obs.: Todos os sinais assíncronos entram na lista de sensibilidade do PROCESS

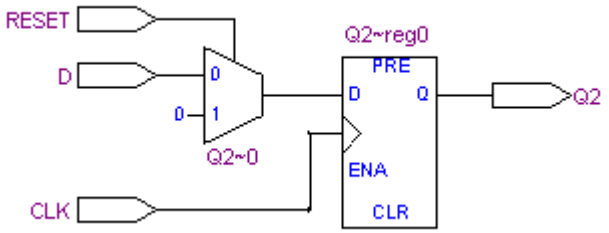
FF tipo D disparado na borda de subida do clock Usando WAIT UNTIL



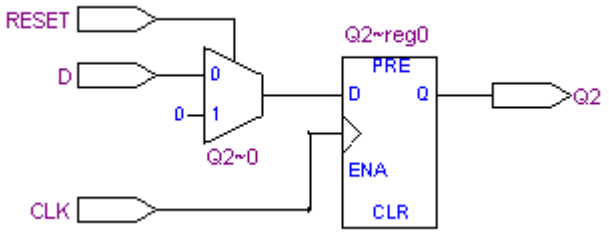

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY FF_tipoD IS
    PORT(D, CLK, : IN     STD_LOGIC;
         Q1      : BUFFER STD_LOGIC);
END FF_tipoD;

ARCHITECTURE a OF FF_tipoD IS
BEGIN
    -- Se WAIT é usado, não há lista de sensibilidade no PROCESS
    PROCESS
    BEGIN
        -- Aguarda borda de subida do clk
        WAIT UNTIL (CLK'EVENT AND CLK = '1');
            Q1 <= D;
    END PROCESS;
END a;
```

FF tipo D disparado na borda de subida do clock
RESET síncrono
Usando IF-THEN-ELSE

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY FF_tipoD_Rsinc IS
    PORT(D, CLK, RESET : IN     STD_LOGIC;
         Q2            : BUFFER STD_LOGIC);
END FF_tipoD_Rsinc;

ARCHITECTURE a OF FF_tipoD_Rsinc IS
BEGIN
    PROCESS(CLK)
    BEGIN
        -- Aguarda borda de subida do clk
        IF (CLK'EVENT AND CLK = '1') THEN
            IF RESET = '1' THEN
                Q2 <= '0';
            ELSE
                Q2 <= D;
            END IF;
        END IF;
    END PROCESS;
END a;
```

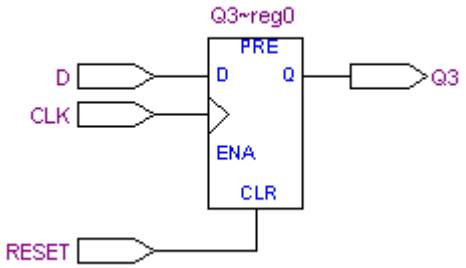


FF tipo D disparado na borda de subida do clock
RESET síncrono
Usando WAIT UNTIL




```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY FF_tipoD_Rsinc IS
    PORT(D, CLK, RESET : IN     STD_LOGIC;
         Q2            : BUFFER STD_LOGIC);
END FF_tipoD_Rsinc;

ARCHITECTURE a OF FF_tipoD_Rsinc IS
BEGIN
    PROCESS -- Se WAIT é usado não é usada lista de sensibilidade
    BEGIN
        -- Aguarda borda de subida do clk
        WAIT UNTIL (CLK'EVENT AND CLK = '1');
            IF RESET = '1' THEN
                Q2 <= '0';
            ELSE
                Q2 <= D;
            END IF;
   END PROCESS;
END a;
```

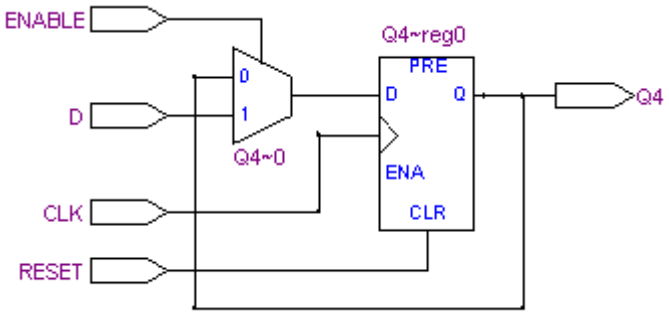
FF tipo D disparado na borda de subida do clock
RESET Assíncrono
Usando IF-ELSIF-ELSE




```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY FF_tipoD_Rassinc IS
    PORT(D, CLK, RESET : IN     STD_LOGIC;
         Q3            : BUFFER STD_LOGIC);
END FF_tipoD_Rassinc;

ARCHITECTURE a OF FF_tipoD_Rassinc IS
BEGIN
    PROCESS(RESET, CLK)
    BEGIN
        IF RESET = '1' THEN
            Q3 <= '0';
        ELSIF (CLK'EVENT AND CLK='1') THEN -- Aguarda borda de
                                           -- subida do clk
            Q3 <= D;
        ELSE
            Q3 <= Q3;
        END IF;
    END PROCESS;
END a;
```

FF tipo D disparado na borda de subida do clock
RESET e ENABLE Assíncronos
Usando IF-ELSIF-ELSE

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY FF_tipoD_ERassinc IS
    PORT(D, CLK, RESET, ENABLE : IN     STD_LOGIC;
         Q4                    : BUFFER STD_LOGIC);
END FF_tipoD_ERassinc;

ARCHITECTURE a OF FF_tipoD_ERassinc IS
BEGIN
    PROCESS (CLK, RESET)
    BEGIN
        IF RESET = '1' THEN
            Q4 <= '0';
        ELSIF (CLK'EVENT AND CLK = '1') THEN -- Aguarda borda
                                             -- de subida do CLK
            IF ENABLE ='1' THEN
                Q4 <= D;
            ELSE
                Q4 <= Q4;
            END IF;
        END IF;
    END PROCESS;
END a;
```



Prática nº10
Descrição de Flip-Flops

Implementar Flip-Flops tipo T sensíveis à borda de subida do clock, utilizando o atributo ‘EVENT e a biblioteca IEEE pacote STD_LOGIC_1164.

Mostrar exemplos com Flip-Flop tipo D

Circuitos GeradosPrática nº10
Descrição de Flip-Flops T