ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES I

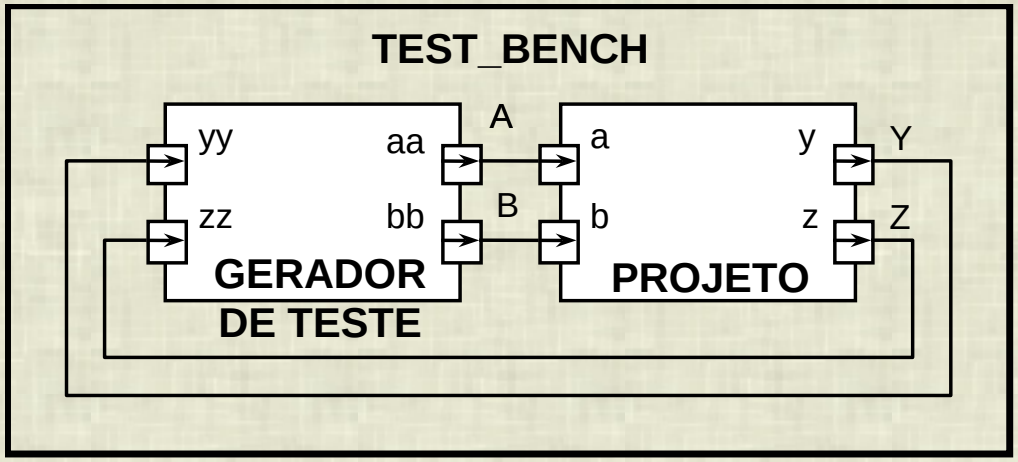
Verificação por Simulação

Circuitos Descritos em VHDL

prof. Dr. César Augusto M. Marcon
prof. Dr. Edson Ifarraguirre Moreno

Validação por Simulação

- **Utilizar um circuito de teste*: test-bench***
  - Contém um circuito **gerador de teste** e uma instância do projeto
  - O **gerador de teste** deve gerar um conjunto de estímulos para o circuito que permitam que o comportamento do circuito seja verificado
  - O **gerador de teste** normalmente tem um descrição comportamental não sintetizável, enquanto o projeto tem uma descrição sintetizável
  - O test_bench **não** contém portas de entrada/saída



Validação por Simulação

```
library IEEE;
use IEEE.std_logic_1164.all;
```

Bibliotecas básicas

```
entity tb is
end tb;
```

O test_bench não contém portas na interface externa

Seleção da entidade

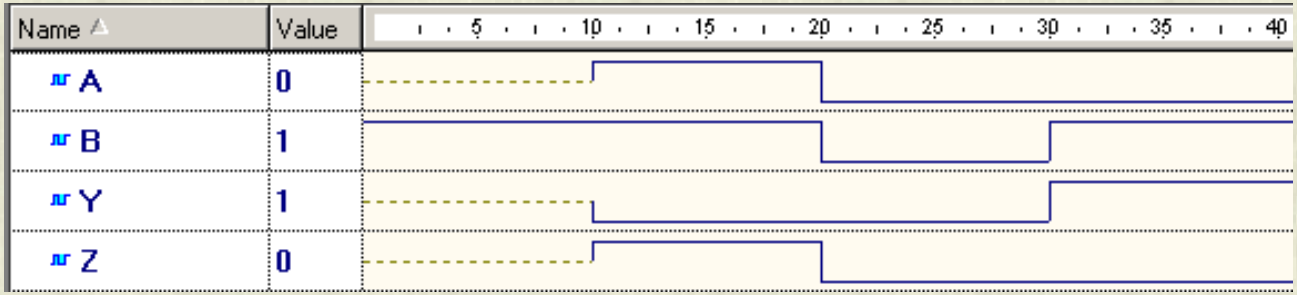
```
architecture Tb of tb is
    signal A, B, Y, Z: std_logic;
begin
    id1: entity work.Entidade port map(a=>A, b=>B, y=>Y, z=>Z);
```

Sinais internos ao test-bench

Instanciação do projeto (circuito) a ser testado

```
    A <= '1' after 10ns, '0' after 20ns;
    B <= '1', '0' after 20ns, '1' after 30ns;
end Tb;
```

Gerador de teste

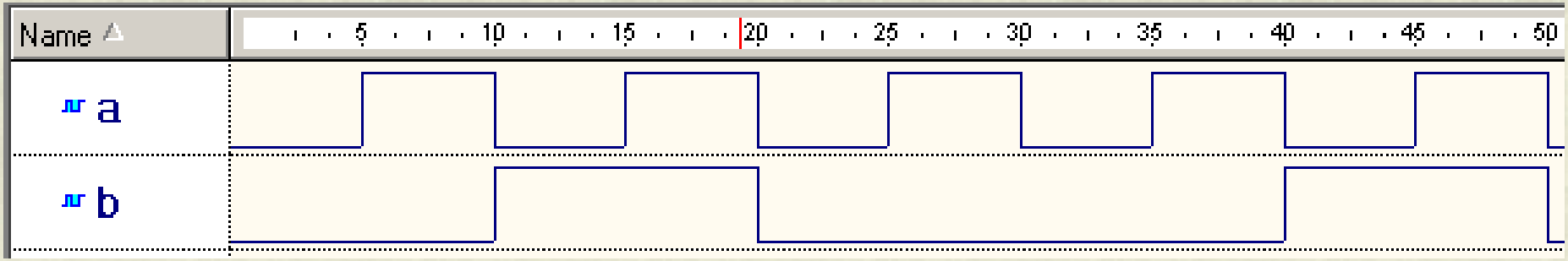


Ferramenta de Simulação/Síntese

- **Disponível no ambiente Linux da FACIN**
- **Xilinx ISE**
  - Permite a síntese e a simulação para um dispositivo programável
- **ModelSim**
  - Permite a simulação de uma descrição VHDL
- **Lançamento da ferramenta**
  - Abrir Terminal
    - Application >> Accessories >> Terminal
  - Carregar parâmetros gerais
    - source /soft64/source_gaph
  - Para carregar parâmetros e lançar o ISE
    - module load ise
    - ise
  - Para carregar parâmetros e lançar o Modelsim
    - module load modelsim
    - vsim

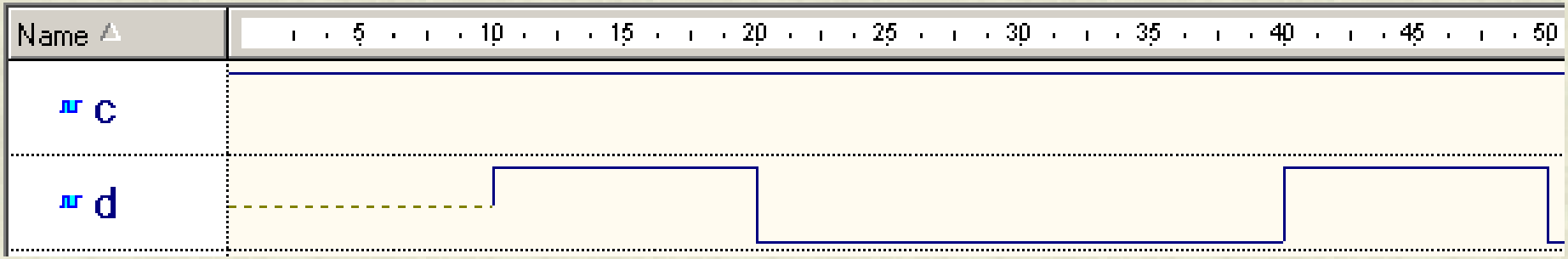
Uso de Processos em Test-Benchs

```
architecture Processos of Processos is
    signal a, b: std_logic;
begin
    process
    begin
        a <= '0', '1' after 5ns;
        wait for 10ns;
    end process;
    process
    begin
        b <= '0', '1' after 10ns, '0' after 20ns;
        wait for 30ns;
    end process;
end Processos;
```



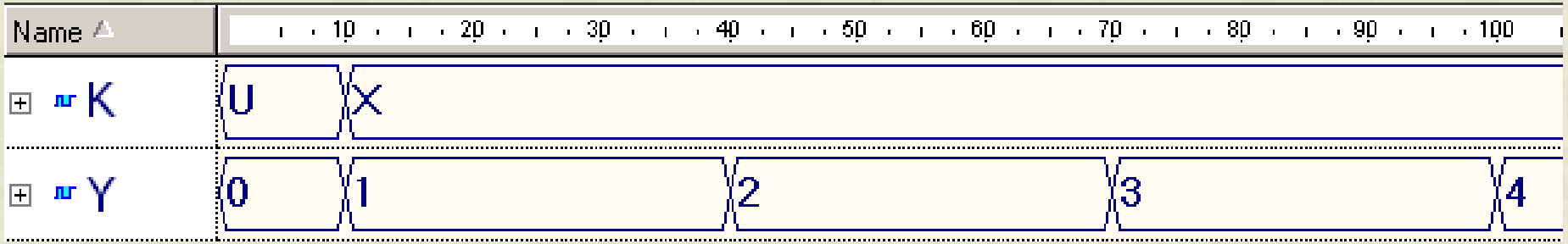
Uso de Processos em Test-Benchs

```
architecture Processos of Processos is
    signal c, d: std_logic;
begin
    process
    begin
        c <= '1', '0' after 20ns;
        wait for 20ns;
    end process;
    process
    begin
        d <= '1' after 10ns, '0' after 20ns;
        wait for 30ns;
    end process;
end Processos;
```



Uso de Processos em Test-Benchs

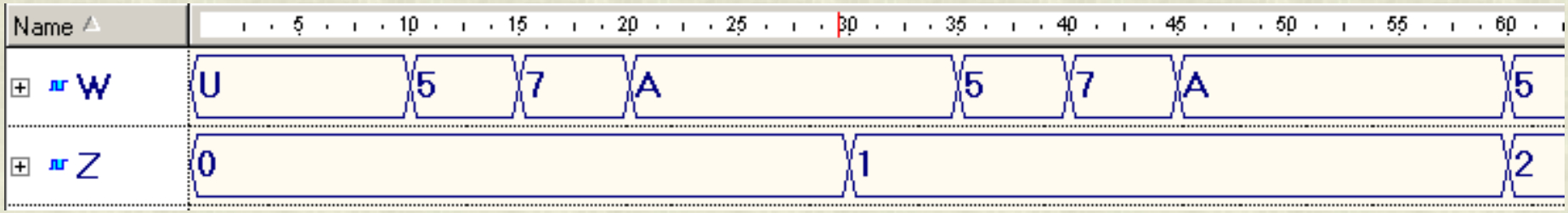
```
architecture Processos of Processos is
    signal K: std_logic_vector(3 downto 0);
    signal Y: std_logic_vector(3 downto 0) := (others => '0');
begin
   process
    begin
        K <= K + '1' after 10ns;
        wait for 30ns;
    end process;
    process
    begin
        Y <= Y + '1' after 10ns;
        wait for 30ns;
    end process;
end Processos;
```



Uso de Processos em Test-Benchs

```
architecture Processos of Processos is
    signal Z: std_logic_vector(3 downto 0) := (others => '0');
    signal W: std_logic_vector(3 downto 0);
begin
    process
    begin
        Z <= Z + '1' after 30ns;
        wait for 30ns;
    end process;

    process
    begin
        W <= "0101" after 10ns, x"7" after 15ns, x"A" after 20ns;
        wait for 25ns;
    end process;
end Processos;
```

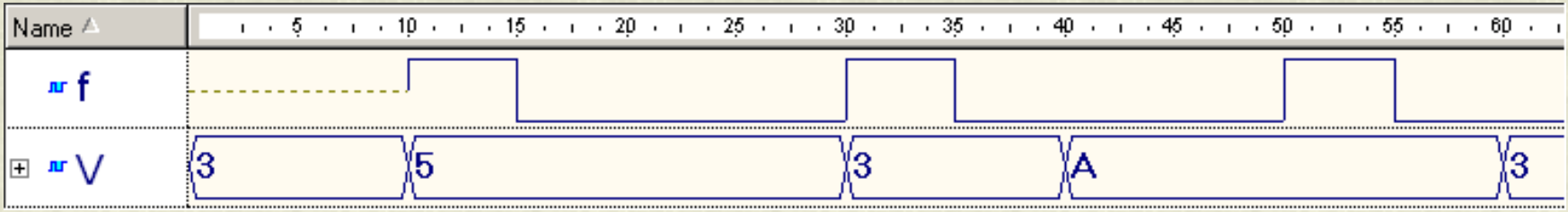


Exercícios de Uso de Processos em Test-Benchs

1. **Dado os processos abaixo, obtenha formas de onda equivalentes**

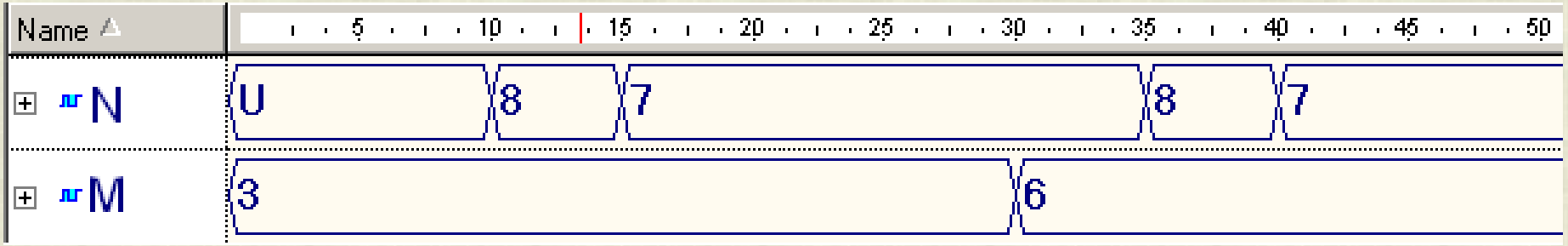
```
architecture Processos of Processos is
    signal M: std_logic_vector(3 downto 0) := "0011";
    signal N: std_logic_vector(3 downto 0);
begin
    process
    begin
        M <= M + M after 30ns;
        wait for 30ns;
    end process;
    process
    begin
        N <= "0101" + M after 10ns, "0111" after 15ns;
        wait for 25ns;
    end process;
end Processos;
```

2. **Dadas as formas de onda abaixo, obtenha processos que as gerem**



Soluções de Exercícios
Uso de Processos em Test-Benchs

1. Dado os processos abaixo, obtenha formas de onda equivalentes



2. Dadas as formas de onda abaixo, obtenha processos que as gerem

```
architecture Processos of Processos is
    signal f: std_logic;
    signal V: std_logic_vector(3 downto 0) := (others => '0');
begin
    process
    begin
        f <= '1' after 10ns, '0' after 15ns;
        wait for 20ns;
    end process;
    process
    begin
        V <= "0011", V + "0101" after 10ns;
        wait for 30ns;
    end process;
end Processos;
```

Exercício

1. **Faça test-benchs para alguns dos circuitos vistos na aula passada**

Auxiliar

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_signed.all;

entity Processos is
end Processos;

architecture Processos of Processos is
   signal a, b, c, d: std_logic;
   signal K: std_logic_vector(3 downto 0);
   signal Y: std_logic_vector(3 downto 0) := (others => '0');
   signal Z: std_logic_vector(3 downto 0) := (others => '0');
   signal W: std_logic_vector(3 downto 0);
begin
   process
   begin
      a <= '0', '1' after 5ns;
      wait for 10ns;
   end process;
   process
   begin
      b <= '0', '1' after 10ns, '0' after 20ns;
      wait for 30ns;
   end process;
   process
   begin
      c <= '1', '0' after 20ns;
      wait for 20ns;
   end process;
   process
   begin
      d <= '1' after 10ns, '0' after 20ns;
      wait for 30ns;
   end process;
   process
   begin
      K <= K + '1' after 10ns;
      wait for 30ns;
   end process;
   process
   begin
      Y <= Y + '1' after 10ns;
      wait for 30ns;
   end process;
   process
   begin
      Z <= Z + '1' after 30ns;
      wait for 30ns;
   end process;
   process
   begin
      W <= "0101" after 10ns, x"7" after 15ns, x"A" after 20ns;
      wait for 25ns;
   end process;
end Processos;
```