Trabalho Prático 3 – TP3 (EM DUPLAS)

Giovana Raupp e Maria Dlugokinski

O módulo a ser implementado é o ***traffic_filter***. O objetivo deste módulo é detectar determinados padrões em um fluxo de entrada de dados serial. Estes padrões correspondem a "ataques", devendo-se **notificar** o mesmo. O ***traffic_filter*** bloqueia a saída de dados em caso de detecção de igualdade entre o fluxo de entrada e um determinado padrão.

- **[2,0 pontos]** Apresentar a descrição da FSM, com o seu desenho, e explicação do funcionamento do circuito.

Descrição da FMS:

Nossa máquina de estados possui oito estados diferentes: S0, BUSCANDO, BLOQUEIO, ZERAR, COMP0, COMP1, COMP2, COMP3. Sendo os 4 ultimos comandos de ação e responsaveis pela configuração dos padrões. Dentro destes oito estados, a máquina verifica o valor do sinal "prog" e transita para um novo estado de acordo com esse valor.

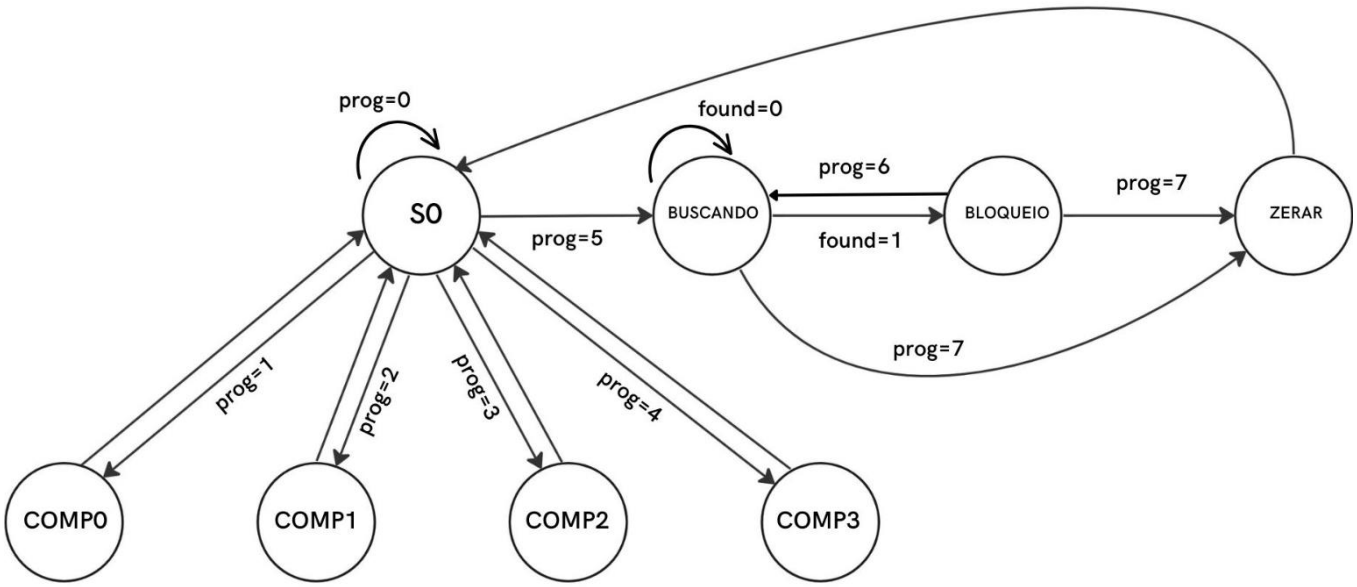
No estado S0: Se prog =1 vai para o estado COMP0, em que é armazenado o padrão e ativado o sinal seletor (0), que habilita a comparação no compara_dado. A mesma coisa acontece para os 3 outros estados, mudando apenas o valor de prog e consequentemente o valor do seletor. Ou seja, se prog=2 vai para o estado COMP1. Se prog=3 vai pra o estado COMP2. E se prog=4 vai para o estado COMP3. Já com a seleção do prog=5 a maquina vai para o estado BUSCANDO. No entento, se prog=0, a máquina se mantém no estado S0, não realizando nenhuma função.

Nos estados COMP0, COMP1, COMP2 e COMP3: Se prog = 0 retorna ao estado S0, fazendo o caminho de volta.

No estado BUSCANDO ocorre a comparação do dado com o padrão: A máquina verifica o valor do sinal "found", que é ativado se der o "match" entre o dado e o padrão, ou seja, são iguais. Se for '1', vai para o estado BLOQUEIO. Se for '0', permanece no estado BUSCANDO. Além disso, se prog=7, vai para o estado ZERAR. (O alarme é habilitado nesse estado).

No estado BLOQUEIO suspende-se o fluxo de dados: Se prog=6, a máquina retorna para o estado BUSCANDO. Se prog=7, vai para o estado ZERAR.

No estado ZERAR: A máquina zera os registradores, que haviam memorizado os padrões, e volta para o estado S0, reiniciando tudo.



- **[1,0 pontos]** Modelar corretamente o módulo *compara_dado*. Este módulo é um componente do *traffic_filter*.

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

entity compara_dado is
port (clock    : in std_logic;
reset    : in std_logic;
dado     : in std_logic_vector(7 downto 0);
pattern  : in std_logic_vector(7 downto 0);
prog     : in std_logic;
habilita : in std_logic;
match    : out std_logic
);
end entity;

architecture a1 of compara_dado is

signal padrao : std_logic_vector(7 downto 0) := "00000000";
signal igual : std_logic := '0';

begin

process(clock, reset)
begin
if reset = '1' then
padrao <= "00000000";
elsif rising_edge(clock) then
if prog = '1' then
padrao <= pattern;
end if;
end if;
end process;

igual <= '1' when dado = padrao else '0';

match <= habilita and igual;

end a1;
```

- **[4,0 pontos]** Desenvolver a modelagem do circuito proposto em linguagem VHDL.
    - [0.5 ponto] correta modelagem da entidade
    - [0.5 ponto] correta modelagem dos sinais internos à arquitetura
    - [0.5 ponto] instanciação correta dos 4 módulos *compara_dado*
    - [1.0 ponto] correta modelagem da máquina de estados finita (FSM)
    - [1.0 ponto] correta modelagem dos registradores dependentes da FSM, e registrador de deslocamento
    - [0.5 ponto] correta modelagem dos circuitos combinacionais e atribuição das saídas

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

```vhdl
use IEEE.std_logic_unsigned.all;

---------------------------------------
-- Entidade
---------------------------------------
entity tp3 is
  port (clock   : in std_logic;
        reset   : in std_logic;
        din     : in std_logic;
        padrao  : in std_logic_vector(7 downto 0);
        prog    : in std_logic_vector(2 downto 0);
        dout    : out std_logic;
        alarme  : out std_logic;
        numero  : out std_logic_vector(1 downto 0);
        );
end entity;

---------------------------------------
-- Arquitetura
---------------------------------------
architecture tp3 of tp3 is
  type state is (S0, COMP0, COMP1, COMP2, COMP3, BUSCANDO, BLOQUEIO, ZERAR); -- estados da
maquina
  signal EA, PE: state;

  -- Aqui escreve todos os signals do circuito

  signal data: std_logic_vector(7 downto 0) := "00000000";

  signal prog0: std_logic := '0';
  signal prog1: std_logic := '0';
  signal prog2: std_logic := '0';
  signal prog3: std_logic := '0';

  signal sel0: std_logic := '0';
  signal sel1: std_logic := '0';
  signal sel2: std_logic := '0';
  signal sel3: std_logic := '0';

  signal match0: std_logic := '0';
  signal match1: std_logic := '0';
  signal match2: std_logic := '0';
  signal match3: std_logic := '0';

  signal found: std_logic := '0';

  signal alarme_int: std_logic := '0';



begin

  -- REGISTRADOR DE DESLOCAMENTO QUE RECEBE O FLUXO DE ENTRADA
  process(reset, clock)
  begin
```

```
  if reset = '1' then
    reg_din <= (others => '0');
  elsif rising_edge(clock) then
    reg_din <= din & reg_din(7 downto 1);
  end if;
end process;

-- 4 PORT MAPS PARA OS compara_dado
cd0_inst: entity work.compara_dado
  port map (
      clock => clock,
      reset => reset
      dado => data,
      padrao => pattern,
      prog => program(0),
      habilita => sel(0),
      match => match(0)
     );

 cd1_inst: entity work.compara_dado
  port map (
      clock => clock,
      reset => reset
      dado => data,
      padrao => pattern,
      prog => program(1),
      habilita => sel(1),
      match => match(1)
  );

  cd2_inst: entity work.compara_dado
  port map (
      clock => clock,
      reset => reset
      dado => data,
      padrao => pattern,
      prog => program(2),
      habilita => sel(2),
      match => match(2)
  );

  cd3_inst: entity work.compara_dado
  port map (
      clock => clock,
      reset => reset
      dado => data,
      padrao => pattern,
      prog => program(3),
      habilita => sel(3),
      match => match(3)
  );

  found  <=  match(0) or match(1) or match(2) or match(3);

program(0) <= '1' when EA = COMP0 else '0';
```

```
process(clock, reset)
begin
if reset = '1' then
sel(0) <= '0';
elsif rising_edge(clock) then    --  registradores para ativar as comparações(4)
if EA=COMP0 then
sel(0) <= '1';
elsif EA= COMP1 then
sel(0) <= '0';
end if;
end if;

program(1) <= '1' when EA= COMP1 else '0';

process(clock, reset)
begin
if reset = '1' then
sel(1) <= '0';
elsif rising_edge(clock) then
if EA=COMP1 then
sel(1) <= '1';
elsif EA= COMP2 then
sel(1) <= '0';
end if;
end if;

program(2) <= '1' when EA= COMP2 else '0';

process(clock, reset)
begin
if reset = '1' then
sel(2) <= '0';
elsif rising_edge(clock) then
if EA=COMP2 then
sel(2) <= '1';
elsif EA= COMP3 then
sel(2) <= '0';
end if;
end if;

program(3) <= '1' when EA= COMP3 else '0';

process(clock, reset)
begin
if reset = '1' then
sel(3) <= '0';
elsif rising_edge(clock) then
if EA=COMP3 then
sel(3) <= '1';
elsif EA= S0 then
sel(3) <= '0';
end if;
end if;
```

```
  --  registrador para o alarme interno

  process(clock, reset)
  begin
  if reset = '1' then
  alarme_int <= '0';
  elsif rising_edge(clock) then
  if EA = BUSCANDO then
  alarme_int <= found;
  elsif EA = ZERAR then
  alarme_in <= '0';
  end if;
  end if;
  end process;

  -- MAQUINA DE ESTADOS (FSM)
process (EA, prog, found)
begin
  case EA is
    when S0 =>
      if prog = "000" then
        PE <= S0;
      elsif prog = "001" then
        PE <= COMP0;
      elsif prog = "010" then
        PE <= COMP1;
      elsif prog = "011" then
        PE <= COMP2;
      elsif prog = "100" then
        PE <= COMP3;
      elsif prog = "101" then
        PE <= BUSCANDO;
      end if;
    when COMP0 =>
      if prog = "000" then
        PE <= S0;
      end if;
    when COMP1 =>
      if prog = "000" then
        PE <= S0;
      end if;
    when COMP2 =>
      if prog = "000" then
        PE <= S0;
      end if;
    when COMP3 =>
      if prog = "000" then
        PE <= S0;
      end if;
    when BUSCANDO =>
      if found = '1' then
        PE <= BLOQUEIO;
      elsif found = '0' then
        PE <= BUSCANDO;
      elsif prog = "111" then
```

```
        PE <= ZERAR;
      end if;
    when BLOQUEIO =>
      if prog = "110" then
        PE <= BUSCANDO;
      elsif prog = "111" then
        PE <= ZERAR;
      end if;
    when ZERAR =>
        PE <= S0;
  end case;
end process;

  -- SAIDAS
  alarme <= alarme_int;
  dout   <= din and not alarme_int;
  numero <= padrao;

end architecture;
```

- **[3,0 pontos]** Apresentar no relatório **um cenário** de operação contendo:
    - [1.0 ponto] programação de 4 padrões diferentes
    - [1.0 ponto] detecção de ao menos 3 padrões
    - [1.0 ponto] reprogramação de novos padrões (pode ser apenas 1) – evento não mostrado na figuraexemplo – e detecção do novo padrão programado

Sor, tentamos muito rodar o código no ModelSim, porém sempre continuava dando o mesmo erro e não formando a onda. Tentamos de tudo e realmente nos encontramos sem saída. Não conseguimos identificar que erro de sintaxe pode ter no código.

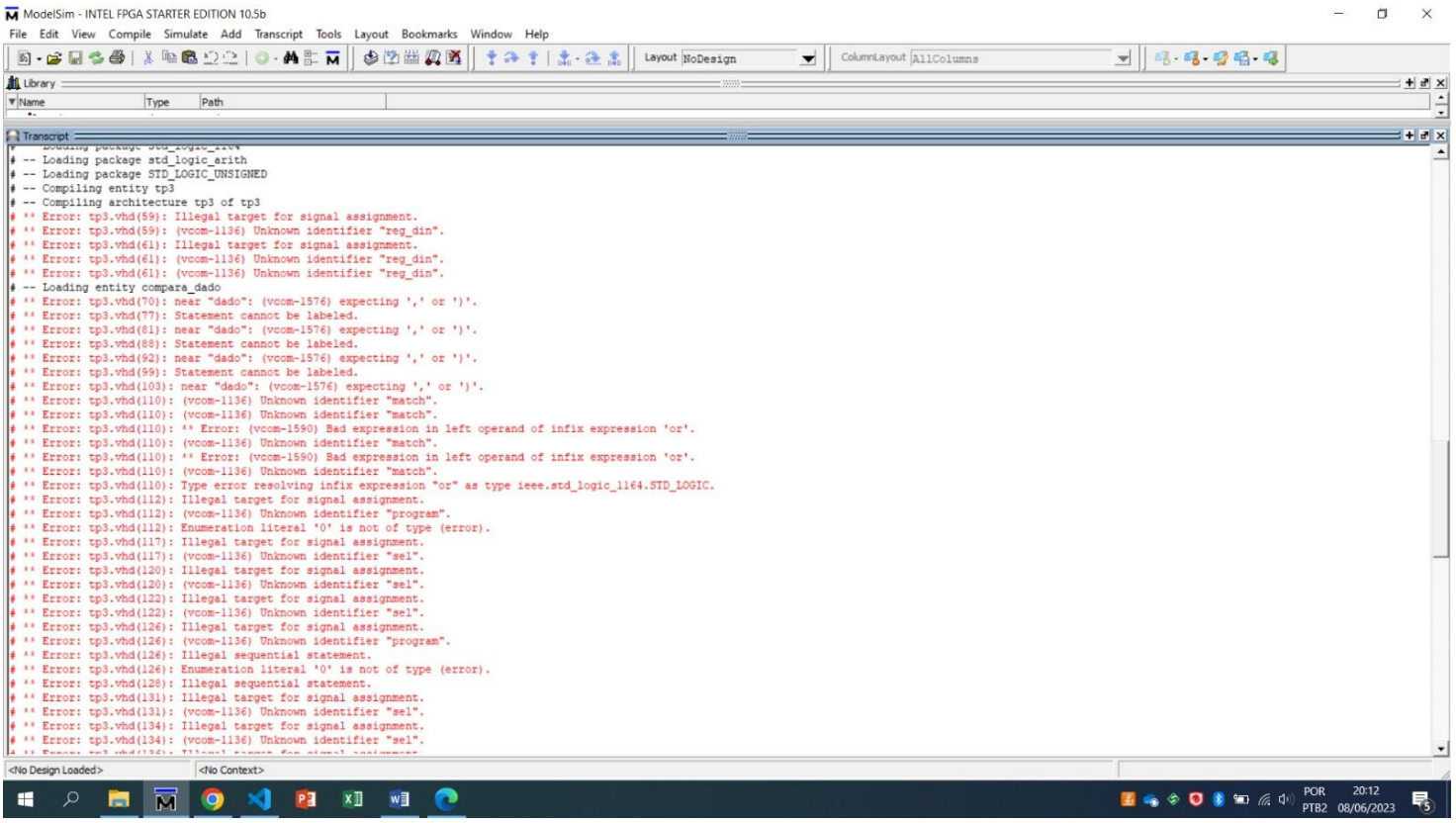
Primeiro erro encontrado:

```
ModelSim> do sim.do
# Model Technology ModelSim - Intel FPGA Edition vmap 10.5b Lib Mapping Utility 2016.10 Oct  5 2016
# vmap work work
# Modifying modelsim.ini
# Model Technology ModelSim - Intel FPGA Edition vcom 10.5b Compiler 2016.10 Oct  5 2016
# Start time: 19:28:56 on Jun 08,2023
# vcom -reportprogress 300 -work work comp.vhd
# -- Loading package STANDARD
# -- Loading package TEXTIO
# -- Loading package std_logic_1164
# -- Compiling entity compara_dado
# ** Error: comp.vhd(15): near ")": (vcom-1576) expecting IDENTIFIER.
# End time: 19:28:56 on Jun 08,2023, Elapsed time: 0:00:00
# Errors: 1, Warnings: 0
# ** Error: C:/intelFPGA/18.1/modelsim_ase/win32aloem/vcom failed.
# Error in macro ./sim.do line 5
# C:/intelFPGA/18.1/modelsim_ase/win32aloem/vcom failed.
#     while executing
# "vcom -work work comp.vhd"
```

Para tentar resolver, tiramos o ; no final da declaração da entidade na Classe compara_dado, mas entao ele reclamou da mesma coisa agora na Classe tp3:

```
# Model Technology ModelSim - Intel FPGA Edition vmap 10.5b Lib Mapping Utility 2016.10 Oct  5 2016
# vmap work work
# Modifying modelsim.ini
# Model Technology ModelSim - Intel FPGA Edition vcom 10.5b Compiler 2016.10 Oct  5 2016
# Start time: 19:53:45 on Jun 08,2023
# vcom -reportprogress 300 -work work comp.vhd
# -- Loading package STANDARD
# -- Loading package TEXTIO
# -- Loading package std_logic_1164
# -- Compiling entity compara_dado
# -- Compiling architecture a1 of compara_dado
# End time: 19:53:45 on Jun 08,2023, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
# Model Technology ModelSim - Intel FPGA Edition vcom 10.5b Compiler 2016.10 Oct  5 2016
# Start time: 19:53:45 on Jun 08,2023
# vcom -reportprogress 300 -work work tp3.vhd
# -- Loading package STANDARD
# -- Loading package TEXTIO
# -- Loading package std_logic_1164
# -- Loading package std_logic_arith
# -- Loading package STD_LOGIC_UNSIGNED
# -- Compiling entity tp3
# ** Error: tp3.vhd(20): near ")": (vcom-1576) expecting IDENTIFIER.
# End time: 19:53:45 on Jun 08,2023, Elapsed time: 0:00:00
# Errors: 1, Warnings: 0
# ** Error: C:/intelFPGA/18.1/modelsim_ase/win32aloem/vcom failed.
# Error in macro ./sim.do line 6
# C:/intelFPGA/18.1/modelsim_ase/win32aloem/vcom failed.
#     while executing
# "vcom -work work tp3.vhd"
```

Tentamos fazer a mesma coisa na Classe tp3, mas daí que deu mais erros:



Vamos continuar tentando encontrar uma solução está tarde, mas em função do tempo do prazo estamos enviando o relatório dessa forma, pois tentamos muito e não queriamos deixar de enviar.