Relatório 7 - Fluxo de Projeto Baseado em *Standard Cells*

Patrik Loff Peres (20103830)
Universidade Federal de Santa Catarina (UFSC)
Departamento de Engenharia Elétrica e Eletrônica (DEEL)

I. Introdução

Neste laboratório foi feita as sínteses lógica e física de uma unidade aritmética e lógica (UAL) com um registrador com *reset* assíncrono, ambos de 32 bits, a partir do código VHDL.

II. VHDL

Inicialmente foi implementado os código VHDL dos componentes necessários (UAL e registrador) e o código topo, juntando os dois. Também, foi feito o *testbench* para simulação. Na UAL foram implementadas 4 operações: **A+B**, **A-B**, deslocamento de 1 *bit* para a esquerda de **B** e deslocamento de 1 *bit* para a direta de **B**, que são selecionadas pela entrada **sel**. O registrador implementado foi do tipo D, que guarda a entrada a cada ciclo de relógio. No VHDL topo, a entrada da **B** da UAL recebe a saída do registrador (que também é a saída do circuito).

Registrador

```
--
    ------------------------------------------------

-- Registrador de 32 bits com reset assincrono

--
    ------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;
use work.all;

--
    ------------------------------------------------


entity reg is

port( D:  in std_logic_vector(31 downto 0);
  clk,rst:  in std_logic;
  Q:  out std_logic_vector(31 downto 0)
);
end reg;

--
    ------------------------------------------------


architecture behavior of reg is
begin
P1:process(clk,rst)
  begin
  if rst = '1' then
    Q <= "00000000000000000000000000000000";
  elsif clk'event and clk = '1' then
    Q <= D;
  end if;
end process;


end behavior;

--
    ------------------------------------------------
```

UAL

```
--
    ------------------------------------------------

-- UAL de 32 bits, sel = 00 -> soma,  sel = 01
    -> subtra  o, sel = 10 -> SHL, sel = 11
    -> SHR
--
    ------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use work.all;

--
    ------------------------------------------------


entity UAL is

port( A:  in std_logic_vector(31 downto 0);
  B:  in std_logic_vector(31 downto 0);
  sel:  in std_logic_vector(1 downto 0);
  S:  out std_logic_vector(31 downto 0)
);
end UAL;

--
    ------------------------------------------------


architecture behavior of UAL is
begin
P1:process(sel,A,B)
  begin
  case sel is
```

```
    when "00" =>
      S <= A + B;
    when "01" =>
      S <= A - B;
    when "10" =>
      S <= B(30 downto 0) & '0';
    when others =>
      S <= '0' & B(31 downto 1);
  end case;
end process;


end behavior;

--
    ------------------------------------------------
```

UAL + registrador

```
--
    ------------------------------------------------

-- UAL mais registrador de 32 bits
--
    ------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use work.all;

--
    ------------------------------------------------


entity UAL_reg is

port( A:    in std_logic_vector(31 downto 0);
  sel:    in std_logic_vector(1 downto 0);
  rst,clk:  in std_logic;
  S:    out std_logic_vector(31 downto 0)
);
end UAL_reg;

--
    ------------------------------------------------


architecture behavior of UAL_reg is
signal aux: std_logic_vector(31 downto 0);
signal B: std_logic_vector(31 downto 0);
component UAL is
port(   A:  in std_logic_vector(31 downto 0);
  B:  in std_logic_vector(31 downto 0);
  sel:  in std_logic_vector(1 downto 0);
  S:  out std_logic_vector(31 downto 0));
end component;

component reg is
port(   D:  in std_logic_vector(31 downto 0);
  clk,rst:  in std_logic;
  Q:  out std_logic_vector(31 downto 0));
end component;

begin
UAL1: UAL port map (A, B, sel, aux);
REGISTRADOR: reg port map (aux, clk, rst, B);
S <= B;

end behavior;

--
    ------------------------------------------------
```

Testbench

```
----------------------------------------
-- Testbench para o UAL_reg-32
----------------------------------------

Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
use work.all;

entity tb_UAL_reg is  -- entity declaration
end tb_UAL_reg;

----------------------------------------

architecture arq_tb of tb_UAL_reg is

component UAL_reg is

port( A:    in std_logic_vector(31 downto 0);
  sel:    in std_logic_vector(1 downto 0);
  rst,clk:  in std_logic;
  S:    out std_logic_vector(31 downto 0)
);
end component;

signal T_clk,T_rst: std_logic:= '1';
signal T_sel: std_logic_vector(1 downto 0):= "
    00";
signal T_A: std_logic_vector(31 downto 0):="
    00000000000000000000000000000001";
signal T_S: std_logic_vector(31 downto 0);
signal fim : boolean :=false;         -- para
    terminar a simula  o

begin

  U1: UAL_reg port map(T_A,T_sel,T_rst,T_clk,
      T_S);

  T_clk <= not T_clk after 5 ns;    -- gera o
      sinall de clock
  T_rst <= '1','0' after 17 ns;   -- gera o
      sinal de reset

  process
  begin
    T_sel <= "00";
    T_A <= "00000000000000000000000000000001";

    wait for 55 ns;
    T_sel <= "01";
    T_A <= "10000000000000000000000000000000";
    wait for 50 ns;
    T_sel <= "10";
    T_A <= "11000000000000000000000000000011";
    wait for 55 ns;
```

```
    T_sel <="11";
    T_A <= "11000000000000000000000000000011";
    wait for 50 ns;
  end process;

end arq_tb;
```

III. Simulação *Zero-Delay*

Para verificar o funcionamento adequado do circuito foi realizada um simulação com atraso zero (figura 1).

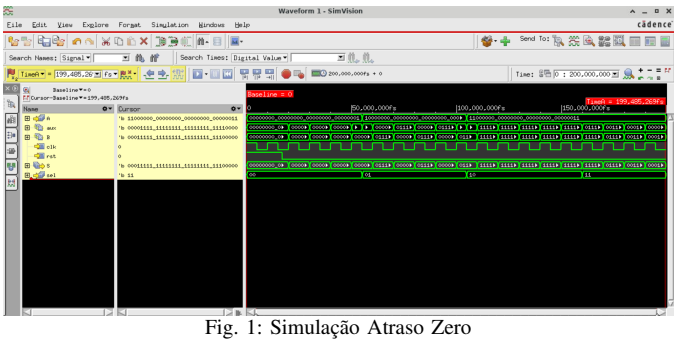

Fig. 1: Simulação Atraso Zero

IV. Síntese Lógica

Com isso, foi realizada a síntese lógica do circuito, que resultou em 134 *standard cells*, cujo informações de área e *timing* estão nas figuras 2 e 3 respectivamente. Destaca-se nesses dados, a área total ocupada de $10279,006\mu m^2$ e atraso critico de $6304ps$cujo caminho se dá pelo sinal seletor da UAL através dos *carry-in's* e *carry-out's* do somador, que foi sintetizado como um *carry-ripple* até a entrada do registrador do ultimo bit do resultado da operação.

```
legacy_genus:/> report area
============================================================
  Generated by:           Genus(TM) Synthesis Solution 21.17-s066_1
  Generated on:           Jun 18 2024  04:21:24 pm
  Module:                 UAL_reg
  Technology libraries:   PnomV180T025_STD_CELL_7RF
                          physical_cells
  Operating conditions:   _nominal_
  Interconnect mode:      global
  Area mode:              physical library
============================================================

Instance Module  Cell Count  Cell Area  Net Area   Total Area
------------------------------------------------------------
UAL_reg                134   7251.686  3027.319    10279.006
```

Fig. 2: Informações sobre área ocupada

V. Simulação Atraso Unitário

Em seguida foi realizada a simulação de atraso unitário para verificar que o circuito funciona adequadamente mesmo se entradas e saídas não mudarem instantaneamente após cada transição. O resultado encontra-se na figura 4, e mostra que o circuito continua operando como esperado.

VI. Síntese Física

Com a simulação de atraso unitário concluída validando os resultado até agora, foi realizada a síntese física do circuito, que resultou no leiaute da figura 5. O leiaute passou nas verificações DRC e *Process Antenna* como mostram as figura 6 e 7.

```
File  Edit  View  Terminal  Tabs  Help
UAL_sub_32_11_Y_add_30_11_g964__1881/COUT   ADDF_E          1  19.8  109  +152    4754 R
UAL_sub_32_11_Y_add_30_11_g963__6131/CIN                                +0    4754
UAL_sub_32_11_Y_add_30_11_g963__6131/COUT   ADDF_E          1  19.8  109  +152    4906 R
UAL_sub_32_11_Y_add_30_11_g962__7098/CIN                                +0    4906
UAL_sub_32_11_Y_add_30_11_g962__7098/COUT   ADDF_E          1  19.8  109  +152    5058 R
UAL_sub_32_11_Y_add_30_11_g961__8246/CIN                                +0    5058
UAL_sub_32_11_Y_add_30_11_g961__8246/COUT   ADDF_E          1  19.8  109  +152    5210 R
UAL_sub_32_11_Y_add_30_11_g960__5122/CIN                                +0    5210
UAL_sub_32_11_Y_add_30_11_g960__5122/COUT   ADDF_E          1  19.8  109  +152    5363 R
UAL_sub_32_11_Y_add_30_11_g959__1705/CIN                                +0    5363
UAL_sub_32_11_Y_add_30_11_g959__1705/COUT   ADDF_E          1  19.8  109  +152    5515 R
UAL_sub_32_11_Y_add_30_11_g958__2802/CIN                                +0    5515
UAL_sub_32_11_Y_add_30_11_g958__2802/COUT   ADDF_E          1  19.8  109  +152    5667 R
UAL_sub_32_11_Y_add_30_11_g957__1617/CIN                                +0    5667
UAL_sub_32_11_Y_add_30_11_g957__1617/COUT   ADDF_E          1  18.5  104  +150    5817 R
UAL_sub_32_11_Y_add_30_11_g956__3680/B                                  +0    5817
UAL_sub_32_11_Y_add_30_11_g956__3680/Z      XOR2_C          1  16.7  181  +101    5918 R
g1777__4319/B1                                                          +0    5918
g1777__4319/Z                               AO22_C          1  13.7  156  +172    6090 R
REGISTRADOR_Q_reg[31]/D                     DFFR_E                       +0    6090
REGISTRADOR_Q_reg[31]/CLK                   setup                    0  +215    6304 R
------------------------------------------------------------------------------------------
(clock clk)                                 capture                         10000 R
------------------------------------------------------------------------------------------
Cost Group   : 'clk' (path_group 'clk')
Timing slack :    3696ps
Start-point  : sel[1]
End-point    : REGISTRADOR_Q_reg[31]/D
```

Fig. 3: Informações sobre *timing*

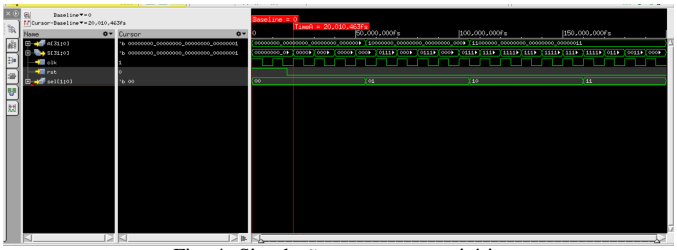

Fig. 4: Simulação com atraso unitário

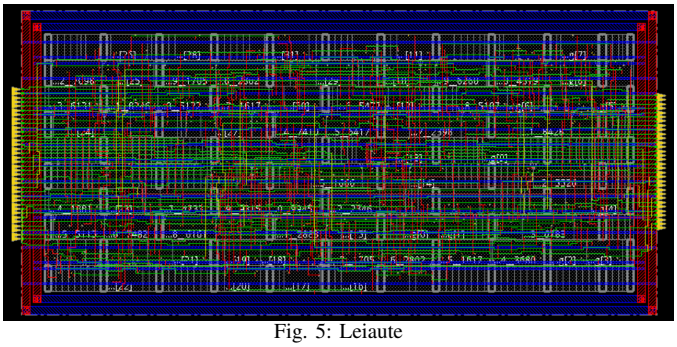

Fig. 5: Leiaute

```
*** Starting Verify DRC (MEM: 2023.3) ***

  VERIFY DRC ..... Starting Verification
  VERIFY DRC ..... Initializing
  VERIFY DRC ..... Deleting Existing Violations
  VERIFY DRC ..... Creating Sub-Areas
  VERIFY DRC ..... Using new threading
  VERIFY DRC ..... Sub-Area: {0.000 0.000 166.880 79.520} 1 of 1
  VERIFY DRC ..... Sub-Area : 1 complete 0 Viols.

  Verification Complete : 0 Viols.

 *** End Verify DRC (CPU: 0:00:00.0  ELAPSED TIME: 0.00  MEM: 256.1M) ***
```

Fig. 6: Verificação DRC

```
innovus 7>
******* START VERIFY ANTENNA ********
Report File: UAL_reg.antenna.rpt
LEF Macro File: UAL_reg.antenna.lef
Verification Complete: 0 Violations
******* DONE VERIFY ANTENNA ********
(CPU Time: 0:00:00.0  MEM: 0.000M)
```

Fig. 7: Simulação com atraso unitário

VII. Simulação com atraso preciso

Com todas as informações do circuito é possível fazer uma simulação considerando o atraso mais próximo do real para

o circuito sintetizado, considerando os atrasos das portas e do roteamento. A figura 8 mostra que o circuito continua funcionando como especificado.

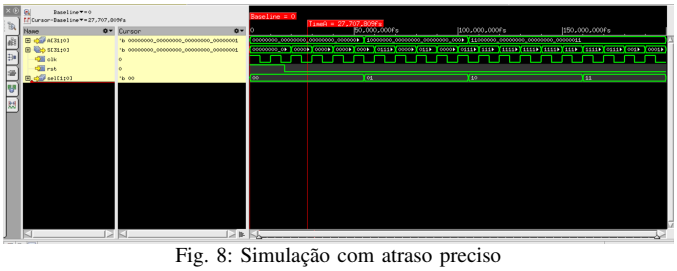

Fig. 8: Simulação com atraso preciso

Considerando os resultados obtemos a FoM = Área ocupada / frequência do relógio, portanto:

$$FoM = \frac{10279,006\mu m^2}{100MHz} = 1,0279\mu m^2/Hz \qquad (1)$$