UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS ARARANGUÁ

RELATÓRIO DO TRABALHO 2
CRONÔMETRO DECRESCENTE

IGOR FIEDLER DE BASTOS E VINÍCIUS DOMINGOS GABRIEL

2025

INTRODUÇÃO

Este relatório tem como objetivo apresentar a metodologia adotada para a descrição do hardware de um cronômetro decrescente de basquete, bem como relatar os principais desafios enfrentados durante o desenvolvimento e a implementação do sistema na placa Nexys 2.

O projeto foi desenvolvido utilizando a linguagem de descrição de hardware VHDL, com foco na criação de um sistema capaz de realizar a contagem regressiva de tempo em minutos e segundos, com controle manual por botões e visualização em displays de sete segmentos.

A seguir, o relatório detalha todas as etapas do projeto, desde o circuito e código até os testes finais em software e na placa.

CIRCUITO

O circuito no qual nos baseamos é composto por diversos processos isolados que, quando combinados, formam o funcionamento completo do cronômetro decrescente. Entre esses componentes, destacam-se um divisor de clock, dois contadores decrescentes e uma máquina de estados.

O divisor de clock foi descrito com o objetivo de gerar um pulso de 1 segundo a partir do clock base de 25 MHz da placa Nexys 2. Esse sinal de 1 Hz serve como base temporal para os dois contadores decrescentes: um responsável pelos segundos, e o outro, pelos minutos.

A lógica de controle do cronômetro é governada por uma máquina de estados com três estados distintos: REP, LOAD e COUNT.

Estado REP

- Representa o estado inicial e também o estado final do cronômetro.
- Quando o botão de reset é pressionado, os displays de sete segmentos são zerados, mostrando 00.00.
- Neste estado, o circuito aguarda uma nova instrução.
- Quando o botão de carga é pressionado, o sistema vai para o estado LOAD.
- Nesse momento, o valor das chaves de entrada é lido e armazenado no contador de minutos.
- O valor aceito pelo sistema deve estar entre 01 e 99. Valores fora dessa faixa são considerados don't-care, ou seja, o comportamento não está definido

para essas condições.

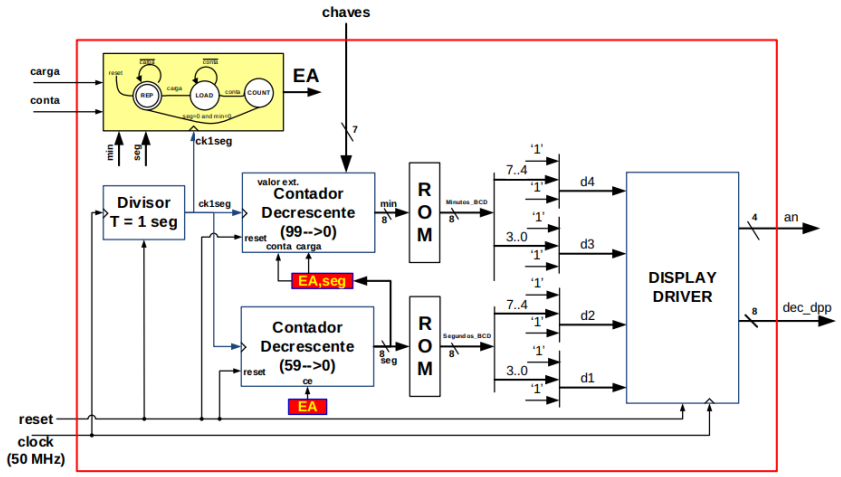
Estado LOAD

- Neste estado, o valor definido nas chaves é carregado apenas no contador de minutos.
- O contador de segundos é automaticamente inicializado com 00.
- Quando o botão de contagem é pressionado, o sistema passa para o estado COUNT.

Estado COUNT

- O cronômetro inicia a contagem regressiva, decrementando os segundos a cada pulso de 1 Hz.
- Quando os segundos chegam a 00, e o contador de minutos ainda é maior que zero, os minutos são decrementados e os segundos reiniciados em 59.
- A contagem continua até atingir exatamente 00.00, momento em que o processo é encerrado.
- Ao atingir esse ponto, o sistema retorna automaticamente ao estado REP, aguardando uma nova instrução de carga ou reset.

O circuito descrito pode ser visto na imagem a seguir:



CÓDIGO VHDL

Utilizamos a linguagem VHDL para descrever o presente hardware, no software ISE da Xilinx.

Cada componente foi dividido em diferentes processos:

P1: Divisor de clock

Este processo tem como função gerar um pulso de 1 segundo a partir do clock de 25 MHz da placa Nexys 2.

```
P1: process(clock, reset)
begin
     if reset = '1' then
          count_clk <= 0;
          clk_1s <= '0';
     elsif rising_edge(clock) then
          if count_clk = CLOCK_FREQ - 1 then
               count_clk <= 0;
               clk_1s <= not clk_1s;
          else
               count_clk <= count_clk + 1;
          end if;
     end if;
end process;
```

P2: Transição de estados

Responsável por armazenar o estado atual da máquina de estados.

```
P2: process(clk_1s, reset)

begin

    if reset = '1' then

        EA <= IDL;

    elsif rising_edge(clk_1s) then

        EA <= PE;

    end if;

end process;
```

P3: Lógica de transição de estados

Define a lógica da máquina de estados, que controla o funcionamento do cronômetro.

```
P3: process(EA, carga, conta, min_int, seg_int)

begin

    case EA is
        when IDL =>
            if carga = '1' then
                PE <= LOAD;
            else
                PE <= IDL;
            end if;

        when LOAD =>
            if conta = '1' then
                PE <= COUNT;
            else
                PE <= LOAD;
            end if;

        when COUNT =>
            if min_int = 0 and seg_int = 0 then
                PE <= IDL;
            else
                PE <= COUNT;
            end if;
    end case;

end process;
```

P4: Contador de segundos

Controla a contagem decrescente dos segundos

```
P4: process(clk_1s, reset)
 begin
     if reset = '1' then
          seg_int <= 0;
     elsif rising_edge(clk_1s) then
          if EA = IDL or EA = LOAD then
               seg_int <= 0;
          elsif EA = COUNT then
               if seg_int > 0 then
                    seg_int <= seg_int - 1;
               elsif min_int > 0 then
                    seg_int <= 59;
               else
                    seg_int <= 0;
               end if;
          end if;
     end if;
 end process;
```

P5: Contador de minutos

Responsável pela manipulação dos minutos.

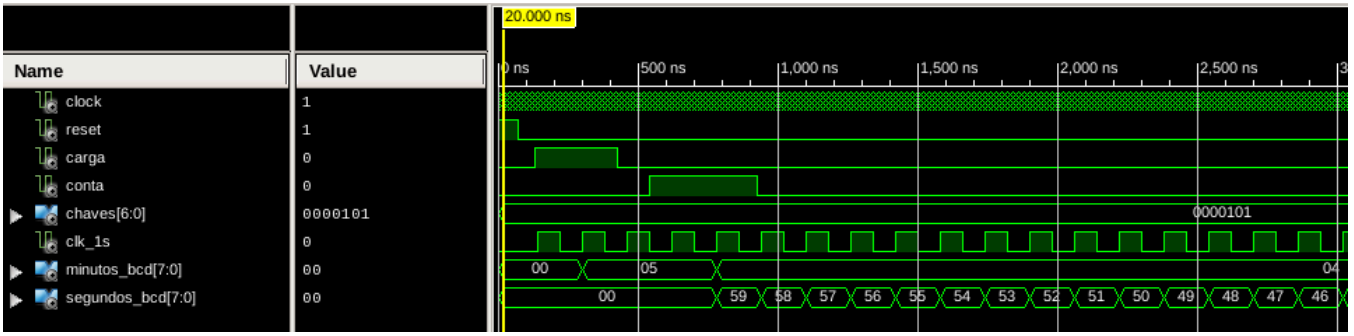
```
P5: process(clk_1s, reset)
 begin
     if reset = '1' then
          min_int <= 0;
     elsif rising_edge(clk_1s) then
          if EA = IDL then
               min_int <= 0;
          elsif EA = LOAD then
               min_int <= to_integer(unsigned(chaves));
          elsif EA = COUNT and seg_int = 0 and min_int > 0 then
               min_int <= min_int - 1;
          end if;
     end if;
 end process;
```

TESTES REALIZADOS

Para os testes em software, utilizamos um testbench disponibilizado pelo professor Ney.

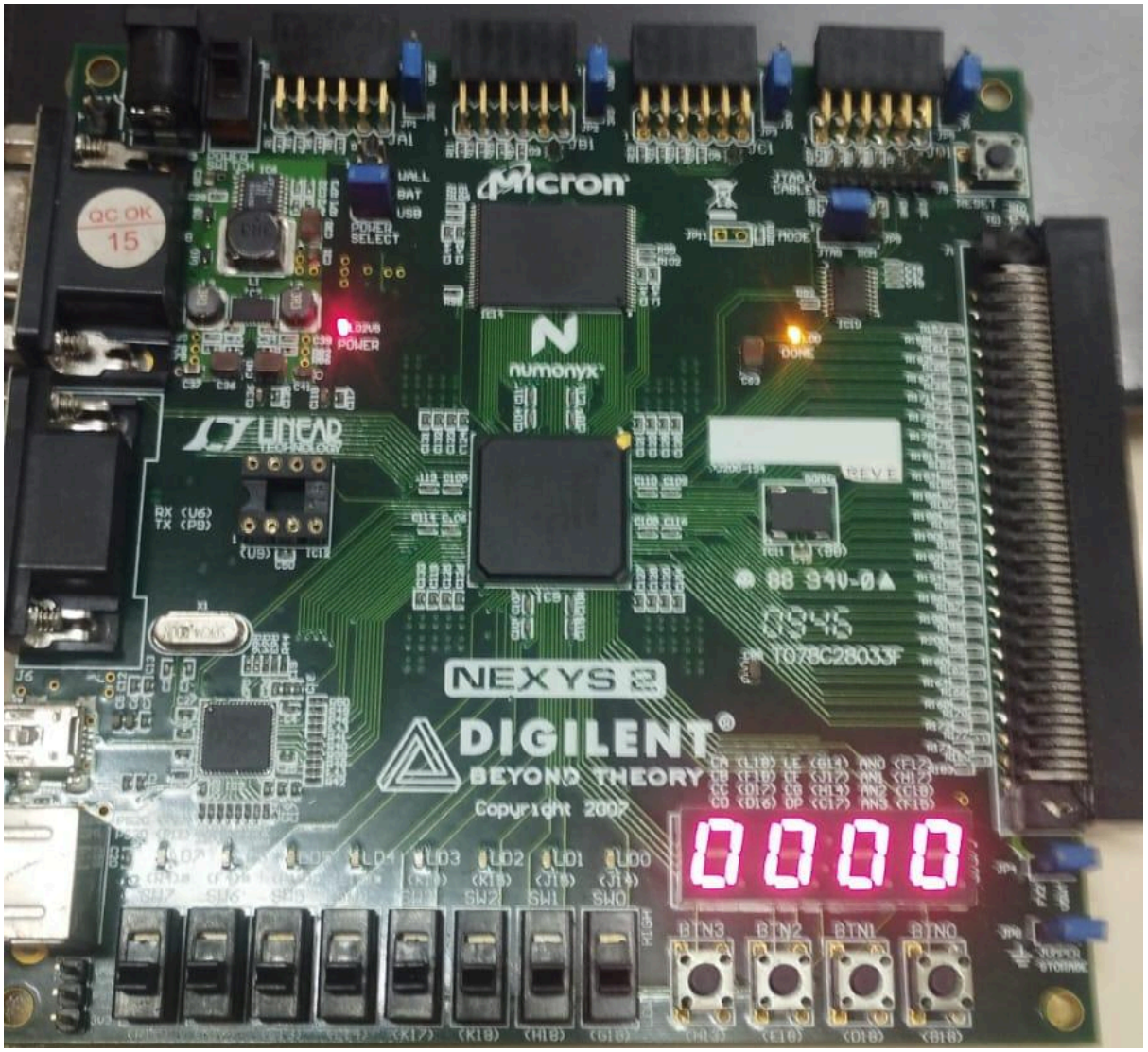
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity tb_cron_dec is
end tb_cron_dec;

architecture sim of tb_cron_dec is

    signal clock  : std_logic := '1';
    signal reset  : std_logic;
    signal carga  : std_logic;
    signal conta  : std_logic;
    signal chaves : std_logic_vector(6 downto 0);

begin

    clock <= not clock after 10 ns;

    reset  <= '1', '0' after 73 ns;
    carga  <= '0', '1' after 133 ns, '0' after 425 ns;
    conta  <= '0', '1' after 543 ns, '0' after 925 ns;
    chaves <= "0000101";

    uut : entity work.cron_dec
        generic map (
            CLOCK_FREQ => 4
        )
        port map (
            clock   => clock,
            reset   => reset,
            carga   => carga,
            conta   => conta,
            chaves  => chaves,
            an      => open,
            dec_dpp => open

        );

end sim;
```

Os seguintes resultados foram obtidos:

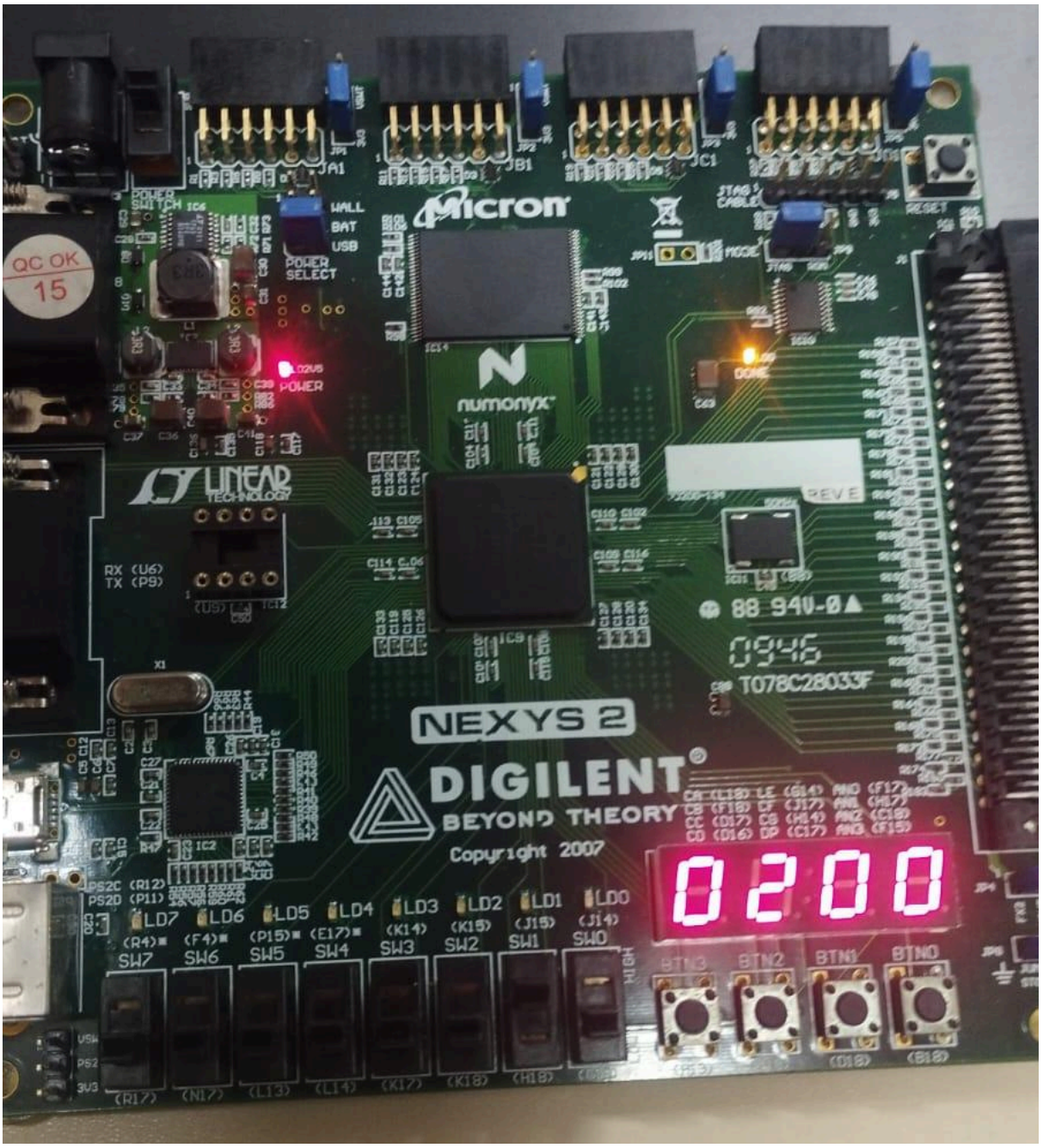


Realizamos também testes práticos, onde enviamos o hardware para a placa Nexus 2, e testamos seu funcionamento.

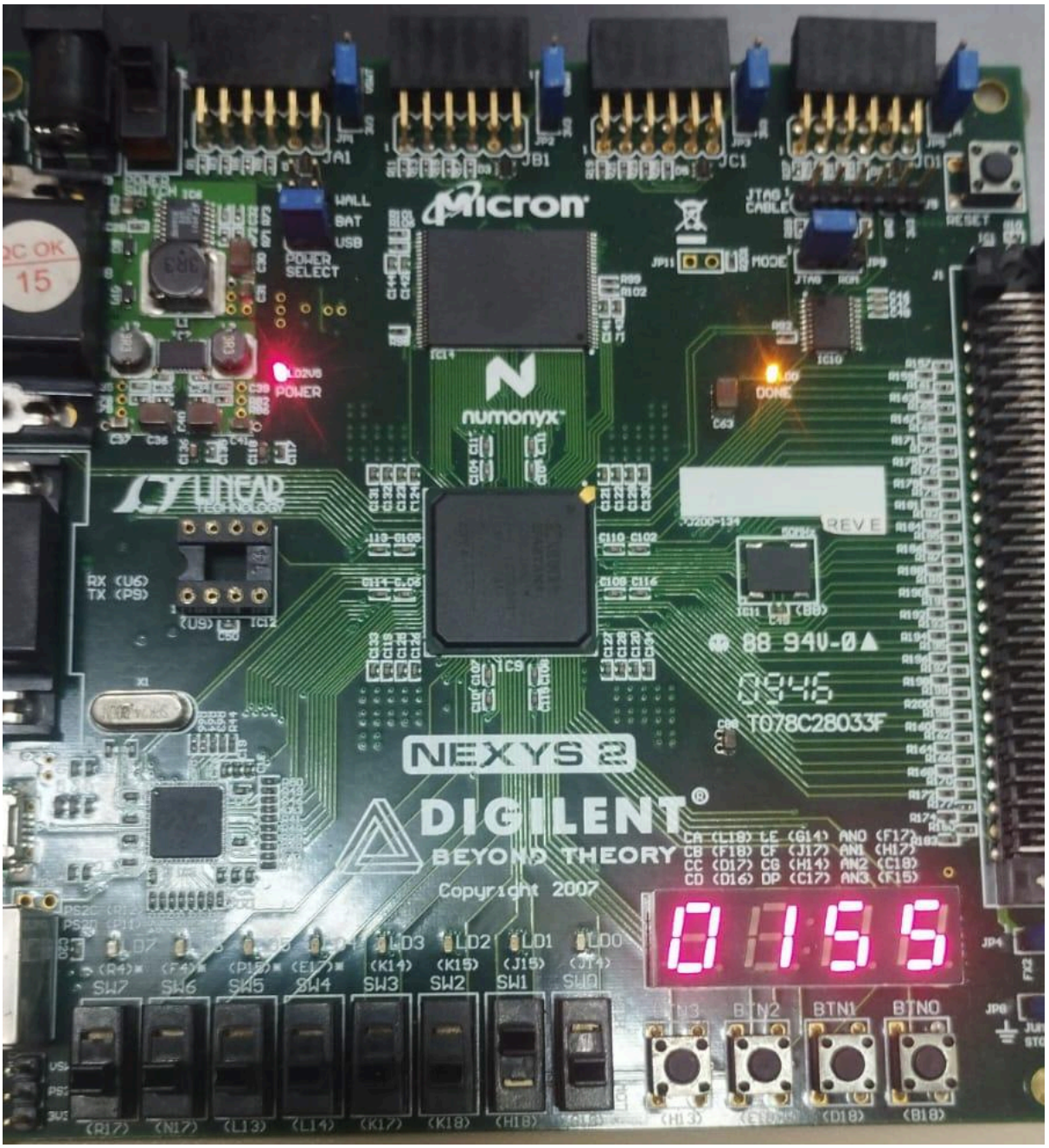
Testando no estado inicial:



Teste carregando 2 minutos:



Teste rodando a partir de 2 minutos:



CONCLUSÃO

O desenvolvimento do cronômetro decrescente nos permitiu aprender a aplicar mais profundamente os conceitos da linguagem VHDL e nos permitiu uma interação prática com a placa Nexys 2. Apesar dos desafios encontrados, foi possível superá-los por meio da análise do comportamento do sistema no testbench, ajustes na lógica e observação dos sinais gerados.

A experiência reforçou o entendimento das boas práticas em projetos digitais e destacou a importância da simulação como ferramenta essencial no desenvolvimento com VHDL.