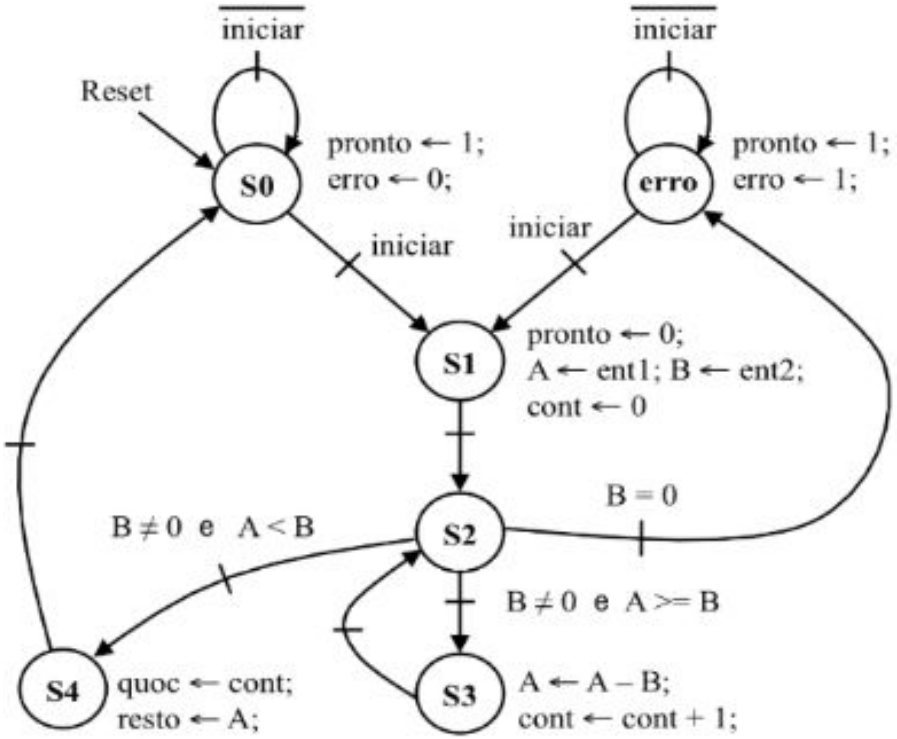
Projeto de Sistemas Digitais

- Arthur Alexandre Nascimento
- Eduardo Vinicius Betim
- Fábio Pereira dos Santos
- Rafael Francisco Réus

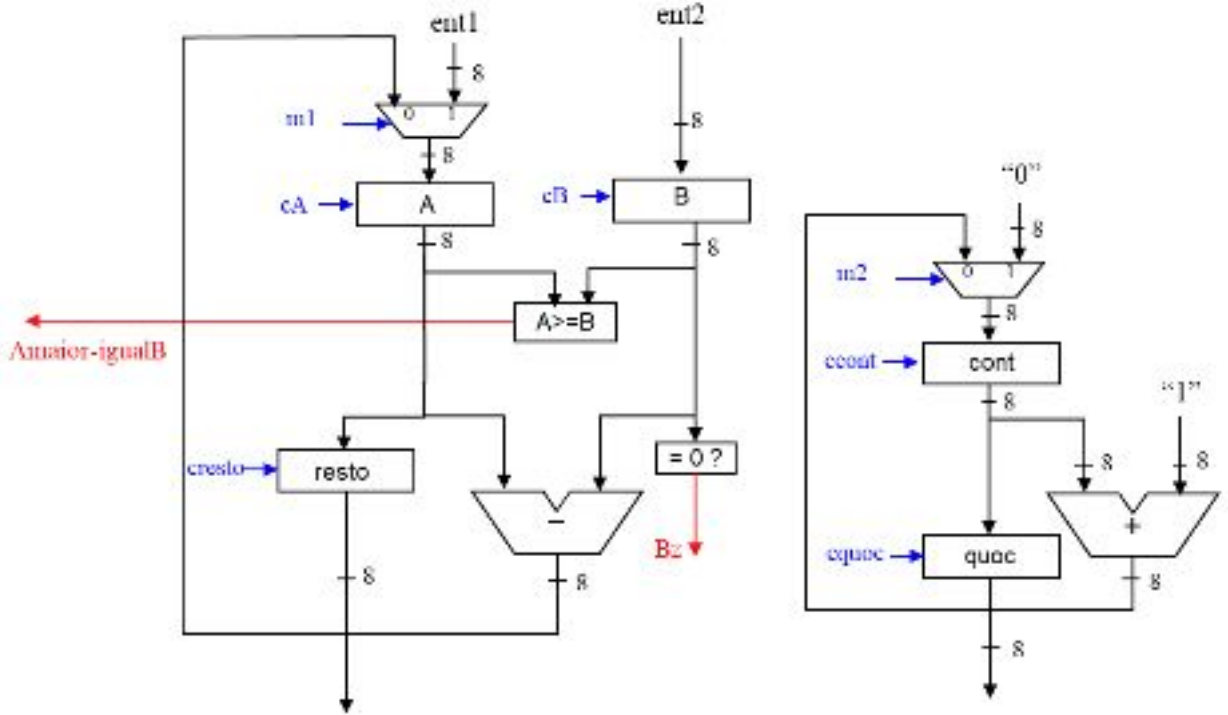
Algoritmo de divisão

```
   Algoritmo divseq
1  Início{
2    A ← ent1; B ← ent2; cont ← 0; pronto←0;
3    if B!=0 then {
4      while A >= B {
5        cont ← cont + 1;
6        A ← A - B; }
7      quoc ← cont; resto ← A;
8      pronto←1; erro←0;
9    }
10   else
11     pronto←1; erro←1;
12 }
```

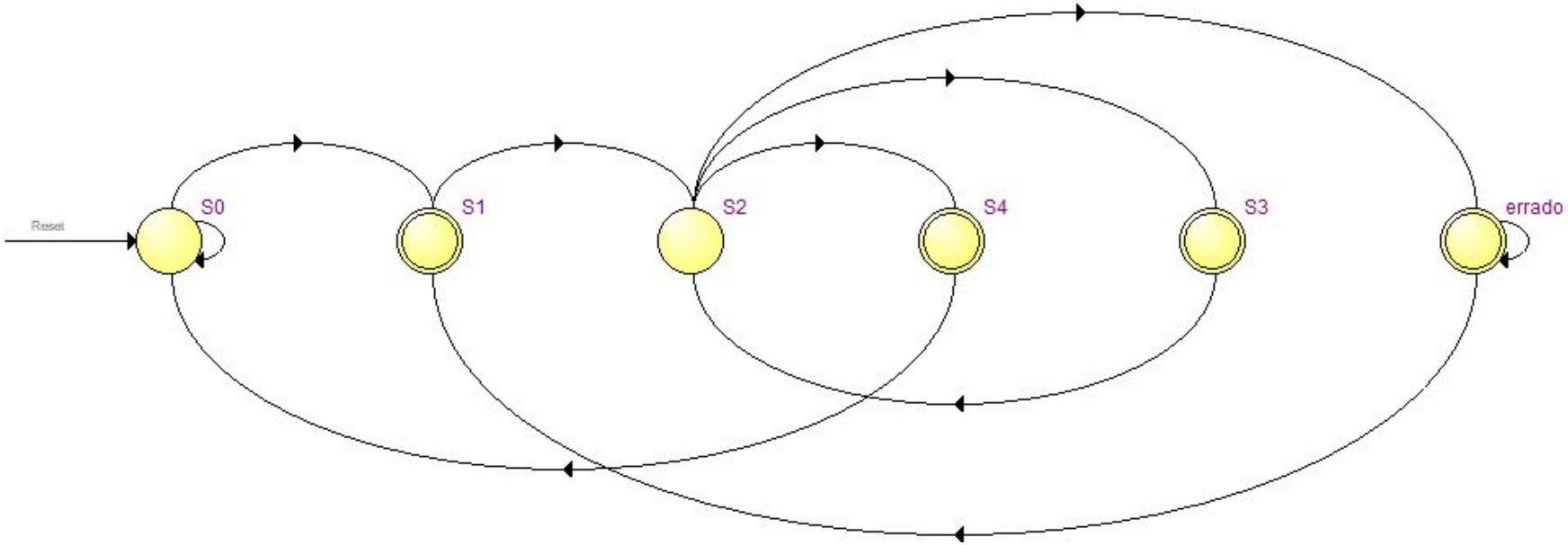
FSMD


iniciar
iniciar
Reset
pronto ← 1;
erro ← 0;
S0
erro
pronto ← 1;
erro ← 1;
iniciar
iniciar
S1
pronto ← 0;
A ← ent1; B ← ent2;
cont ← 0
B = 0
B ≠ 0 e A < B
S2
B ≠ 0 e A >= B
S4
quoc ← cont;
resto ← A;
S3
A ← A – B;
cont ← cont + 1;

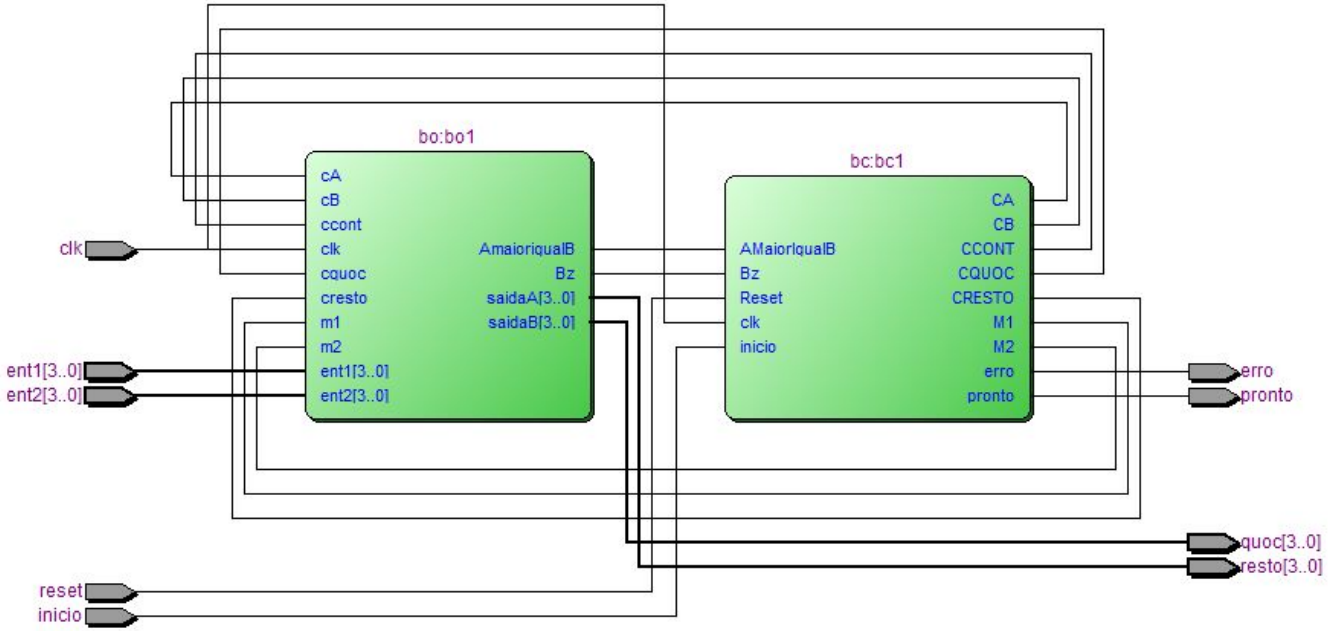
Bloco Operativo



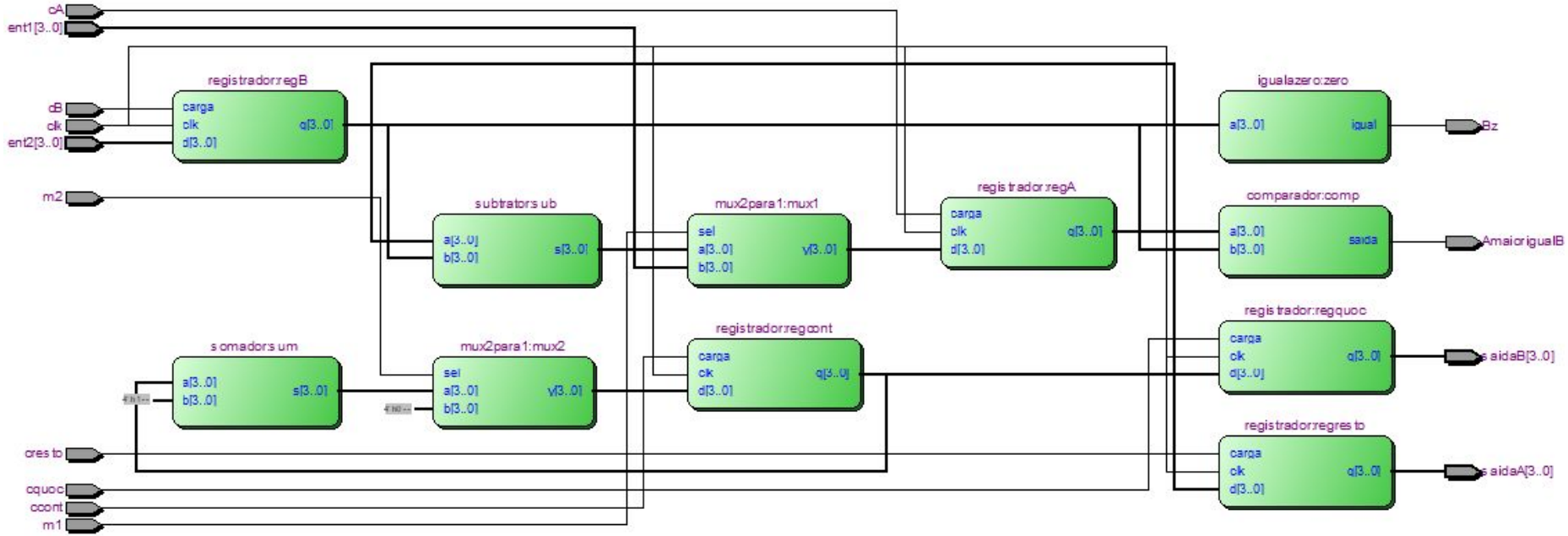
State Machine Viewer



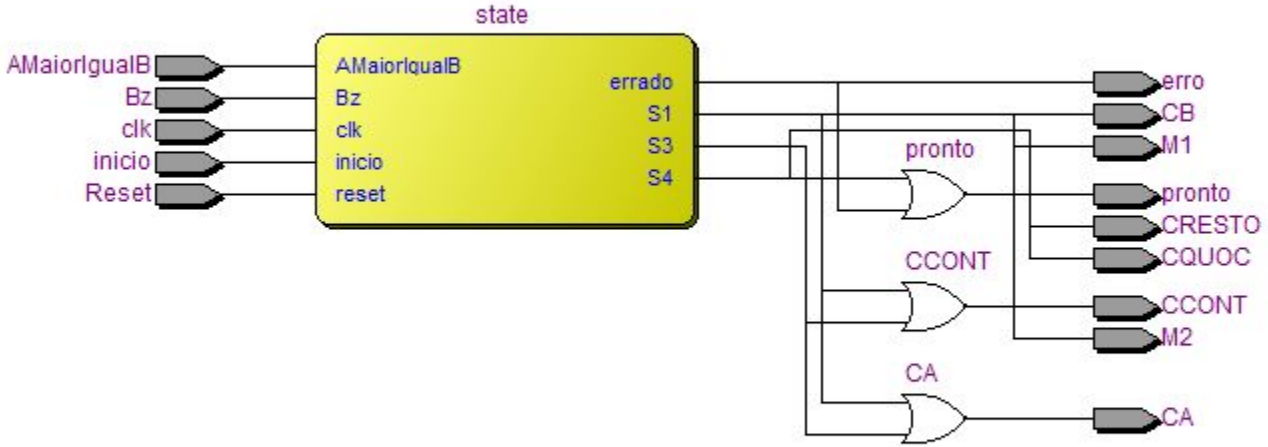
RTL Viewer - Topo



RTL Viewer - BO



RTL Viewer - BC



Comparação N = 4 e N = 8

Flow Status	Successful - Wed Nov 25 16:51:03 2020
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	divisor
Top-level Entity Name	divisor
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	33 / 33,216 (< 1 %)
Total combinational functions	21 / 33,216 (< 1 %)
Dedicated logic registers	26 / 33,216 (< 1 %)
Total registers	26
Total pins	21 / 475 (4 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

Flow Status	Successful - Wed Nov 25 16:55:59 2020
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	divisor
Top-level Entity Name	divisor
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	50 / 33,216 (< 1 %)
Total combinational functions	34 / 33,216 (< 1 %)
Dedicated logic registers	46 / 33,216 (< 1 %)
Total registers	46
Total pins	37 / 475 (8 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

Bc alternativo

Flow Status	Successful - Wed Dec 09 18:00:24 2020
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	divisor
Top-level Entity Name	divisor
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	55 / 33,216 (< 1 %)
Total combinational functions	35 / 33,216 (< 1 %)
Dedicated logic registers	47 / 33,216 (< 1 %)
Total registers	47
Total pins	37 / 475 (8 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

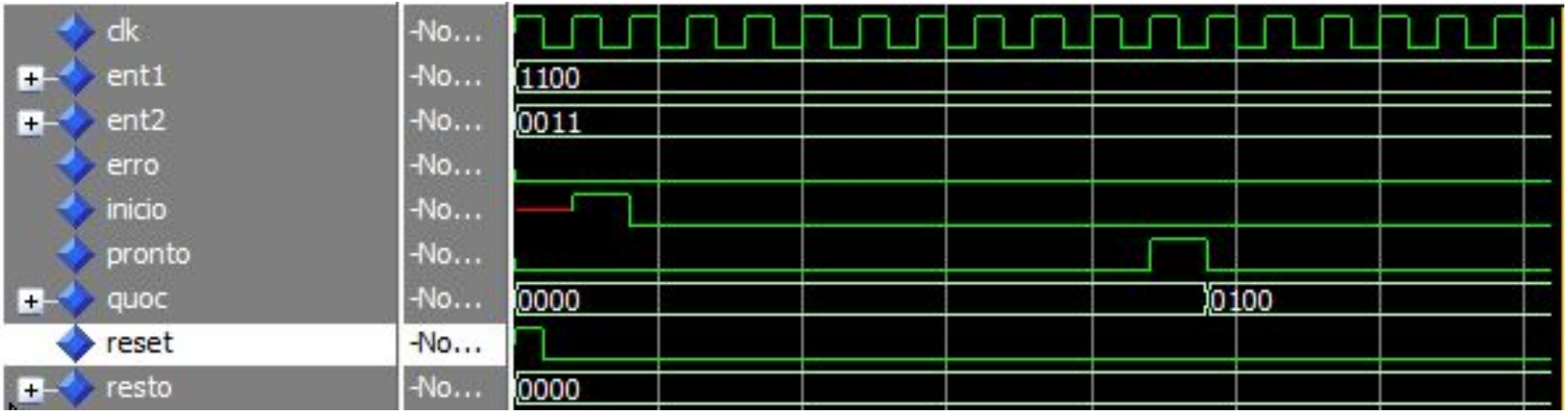
	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	erro	clk	6.438	6.438	Rise	clk
2	pronto	clk	6.843	6.843	Rise	clk
3	⌄ quoc[*]	clk	6.420	6.420	Rise	clk
1	quoc[0]	clk	6.421	6.421	Rise	clk
2	quoc[1]	clk	6.421	6.421	Rise	clk
3	quoc[2]	clk	6.421	6.421	Rise	clk
4	quoc[3]	clk	6.642	6.642	Rise	clk
5	quoc[4]	clk	6.664	6.664	Rise	clk
6	quoc[5]	clk	6.420	6.420	Rise	clk
7	quoc[6]	clk	6.616	6.616	Rise	clk
8	quoc[7]	clk	6.661	6.661	Rise	clk
4	⌄ resto[*]	clk	6.074	6.074	Rise	clk
1	resto[0]	clk	6.100	6.100	Rise	clk
2	resto[1]	clk	6.400	6.400	Rise	clk
3	resto[2]	clk	6.365	6.365	Rise	clk
4	resto[3]	clk	6.481	6.481	Rise	clk
5	resto[4]	clk	6.346	6.346	Rise	clk
6	resto[5]	clk	6.361	6.361	Rise	clk
7	resto[6]	clk	6.074	6.074	Rise	clk
8	resto[7]	clk	6.358	6.358	Rise	clk

Comparação N = 4 e N = 8

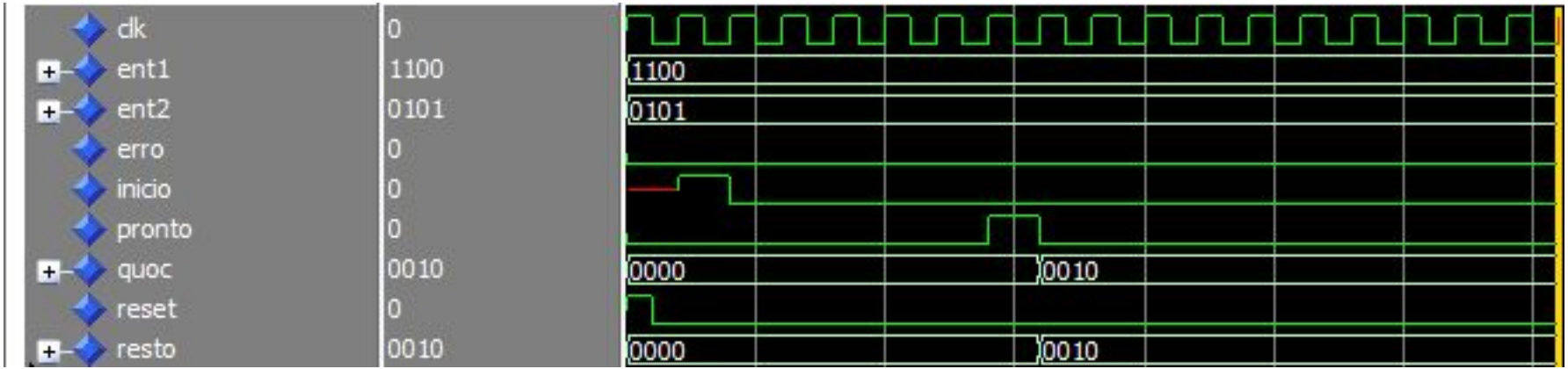
	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	erro	clk	6.750	6.750	Rise	clk
2	pronto	clk	7.227	7.227	Rise	clk
3	quoc[*]	clk	6.392	6.392	Rise	clk
1	quoc[0]	clk	6.392	6.392	Rise	clk
2	quoc[1]	clk	6.585	6.585	Rise	clk
3	quoc[2]	clk	6.603	6.603	Rise	clk
4	quoc[3]	clk	6.600	6.600	Rise	clk
4	resto[*]	clk	6.342	6.342	Rise	clk
1	resto[0]	clk	6.402	6.402	Rise	clk
2	resto[1]	clk	6.342	6.342	Rise	clk
3	resto[2]	clk	6.595	6.595	Rise	clk
4	resto[3]	clk	6.610	6.610	Rise	clk

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	erro	clk	6.613	6.613	Rise	clk
2	pronto	clk	7.186	7.186	Rise	clk
3	quoc[*]	clk	6.270	6.270	Rise	clk
1	quoc[0]	clk	6.370	6.370	Rise	clk
2	quoc[1]	clk	6.358	6.358	Rise	clk
3	quoc[2]	clk	6.270	6.270	Rise	clk
4	quoc[3]	clk	6.411	6.411	Rise	clk
5	quoc[4]	clk	6.523	6.523	Rise	clk
6	quoc[5]	clk	6.528	6.528	Rise	clk
7	quoc[6]	clk	6.357	6.357	Rise	clk
8	quoc[7]	clk	6.396	6.396	Rise	clk
4	resto[*]	clk	6.333	6.333	Rise	clk
1	resto[0]	clk	6.370	6.370	Rise	clk
2	resto[1]	clk	6.333	6.333	Rise	clk
3	resto[2]	clk	6.838	6.838	Rise	clk
4	resto[3]	clk	6.351	6.351	Rise	clk
5	resto[4]	clk	6.582	6.582	Rise	clk
6	resto[5]	clk	6.550	6.550	Rise	clk
7	resto[6]	clk	6.840	6.840	Rise	clk
8	resto[7]	clk	6.370	6.370	Rise	clk

Simulação 12 / 3 - 4 bits



Simulação 12 / 5 - 4 bits



Simulação 12 / 3 - 8 Bits

Cálculo do tempo para execução (N=8 bits)

Primeiro algoritmo

Divisor: 00000001_2 ➔ 1_{10}

Dividendo: 11111111_2 ➔ 255_{10}

Tempo de clock: aproximadamente 7,2 ns

1xS0 + 1xS1 + 255 x (S2 + S3) + S2 + S4 =

514 transições de estado ➔ 514 x 7,2 = 3700,8 ns

Segundo algoritmo

Divisor: 00000001_2 ➔ 1_{10}

Dividendo: 11111111_2 ➔ 255_{10}

Tempo de clock: aproximadamente -- ns

1xS0 + 1xS1 + 255 x (S2 + S3 + S4) + S2 + S4 =

769 transições de estado ➔ 769 x --- =

Testbench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE std.textio.ALL;
USE ieee.std_logic_textio.ALL; -- para tratamento de arquivos e texto
ENTITY divisor_tb IS
END divisor_tb;

ARCHITECTURE tb OF divisor_tb IS

    COMPONENT divisor IS
        PORT (
            reset, clk, inicio : IN STD_LOGIC;
            ent1, ent2 : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
            pronto, erro : OUT STD_LOGIC;
            quoc, resto : OUT STD_LOGIC_VECTOR(3 DOWNTO 0)
        );
    END COMPONENT;
    CONSTANT clk_period : TIME := 20 ns;
    SIGNAL clk, reset, inicio : STD_LOGIC;
    SIGNAL ent1, ent2 : STD_LOGIC_VECTOR(3 DOWNTO 0);
    SIGNAL pronto, erro : STD_LOGIC;
    SIGNAL quoc, resto : STD_LOGIC_VECTOR(3 DOWNTO 0);
BEGIN
    UUT : ENTITY work.divisor PORT MAP (reset, clk, inicio, ent1, ent2, pronto, erro, quoc, resto);

    reset <= '1', '0' AFTER 5 ns;
    inicio <= '1';


    clock_gen : PROCESS
    BEGIN
        clk <= '1';
        WAIT FOR clk_period/2;
        clk <= '0';
        WAIT FOR clk_period/2;
    END PROCESS;

    file_io : PROCESS
        VARIABLE read_col_from_input_buf : line; -- buffers de entrada e saida
        FILE input_buf : text; -- text is keyword
        VARIABLE write_col_to_output_buf : line;
        FILE output_buf : text; -- text is keyword

        VARIABLE val_A, val_B : STD_LOGIC_VECTOR(3 DOWNTO 0); -- entradas A e B do arquivo
        VARIABLE val_SPACE : CHARACTER; -- para espacos

    BEGIN
        file_open(input_buf, "C:\\Users\\Fabio\\Documents\\QuartusProject\\STD_LAB\\divisor\\entradas.txt", read_mode);
        file_open(output_buf, "C:\\Users\\Fabio\\Documents\\QuartusProject\\STD_LAB\\divisor\\saidas_tb.txt", write_mode);

        WAIT UNTIL reset = '0'; -- espera reset desligar

        WHILE NOT endfile(input_buf) LOOP
            readline(input_buf, read_col_from_input_buf);
            read(read_col_from_input_buf, val_A);
            read(read_col_from_input_buf, val_SPACE); -- read in the space character
            read(read_col_from_input_buf, val_B);

            -- Pass the read values to signals
            ent1 <= val_A;
            ent2 <= val_B;

            WAIT UNTIL pronto'EVENT and pronto = '0';
            WAIT FOR 2 ns;
            write(write_col_to_output_buf, quoc);
            write(write_col_to_output_buf, val_SPA--CE);
            write(write_col_to_output_buf, resto);
            writeline(output_buf, write_col_to_output_buf); --

        END LOOP;

        write(write_col_to_output_buf, STRING'("SIMULACAO CONCLUIDA"));
        writeline(output_buf, write_col_to_output_buf); --

        file_close(input_buf);
        file_close(output_buf);

        WAIT;
    END PROCESS;
END tb;
```

Conclusão

- Aumento de aprox. 50% na quantidade de elementos lógicos comparando 4 e 8 bits.
- O tempo mínimo de clock até a saída diminui levemente na versão com 8 bits (sinal pronto maior atraso).

Fim

Divisor - Relatório

O algoritmo usado funciona da seguinte maneira: “A” é o dividendo e “B” é o divisor. “B” logicamente deve ser diferente de 0. A cada ciclo, “B” é subtraído de “A” e o valor da variável de contador incrementa em 1, mostrando assim quantas vezes “B” cabe dentro de “A”. A divisão acaba quando A chegar a zero.

Grande parte do código pôde ser reaproveitado do multiplicador, e o único componente “novo” é o comparador A >= B. Sobre a máquina de estados: O estado S1 é de início. O estado S2 confere se a divisão acabou e se a entrada “B” é diferente de zero. Vai para S3 enquanto a divisão não termina, S4 quando está pronta, e “Erro” quando a entrada B é zero. S4 é responsável por liberar os valores finais de saída do circuito, quociente e resto, e por indicar que o algoritmo está pronto.

No bloco operativo, temos o registrador A, que acumula o valor de “A”, registrador B, que guarda a entrada constante “B”, e o registrador “Cont” que incrementa “1” a cada ciclo. Os registradores “Quoc” e “Resto” são responsáveis por guardar a saída até que ela esteja pronta para ser liberada, quando o algoritmo estiver completo. Temos dois comparadores, A >= B e B != 0, além de um somador e um subtrator, respectivamente responsáveis por incrementar “Cont” e decrementar “A”.

Comparando 4 e 8 bits, o número de elementos lógicos teve um aumento de aproximadamente 50%. Entretanto, o tempo mínimo de clock até a saída diminuiu levemente.