Síntese de Hardware 1

Edson Midorikawa

PCS3225

2º Semestre de 2023

Projeto do Contador 1

- Descrição Textual

> Resolva o seguinte problema:
>
> Há uma entrada de *n* bits. Ao receber um sinal de **iniciar**, seu módulo conta quantos 1s existem na entrada de *n* bits e produz na saída um inteiro correspondente a esta contagem.
>
> Exemplo:
> entrada: 1010111100001011 (15 *bits*),
> saída: 8

Projeto do Contador 1

Resolva o seguinte problema:

Há uma entrada de *n* bits. Ao receber um sinal de **iniciar**, seu módulo conta quantos 1s existem na entrada de *n* bits e produz na saída um inteiro correspondente a esta contagem.

Exemplo:
entrada: 101011100001011 (15 *bits*),
saída: 8

- Especificação

Descrição RTL:

```
1  while(1) {
2      while(Start == 0);
3      Done = 0;
4      Data = Input;
5      Ocount = 0;
6      Mask = 1;
7      while(Data > 0) {
8          Temp = Data & Mask;
9          Ocount = Ocount + Temp;
10         Data >>= 1;
11     }
12     Output = Ocount;
13     Done = 1;
14 }
```

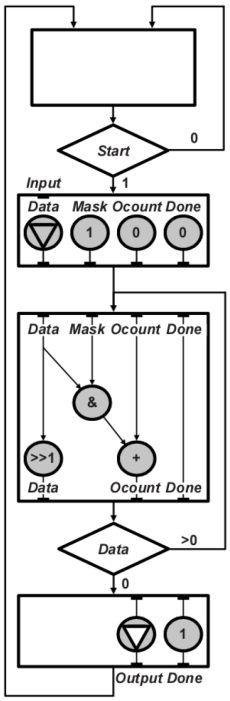
Projeto do Contador 1

Resolva o seguinte problema:

Há uma entrada de *n* bits. Ao receber um sinal de **iniciar**, seu módulo conta quantos 1s existem na entrada de *n* bits e produz na saída um inteiro correspondente a esta contagem.

Exemplo:
entrada: 101011100001011 (15 *bits*),
saída: 8

- Especificação (CDFG)



Descrição RTL:

```
1  while(1) {
2      while(Start == 0);
3      Done = 0;
4      Data = Input;
5      Ocount = 0;
6      Mask = 1;
7      while(Data > 0) {
8          Temp = Data & Mask;
9          Ocount = Ocount + Temp;
10         Data >>= 1;
11     }
12     Output = Ocount;
13     Done = 1;
14 }
```

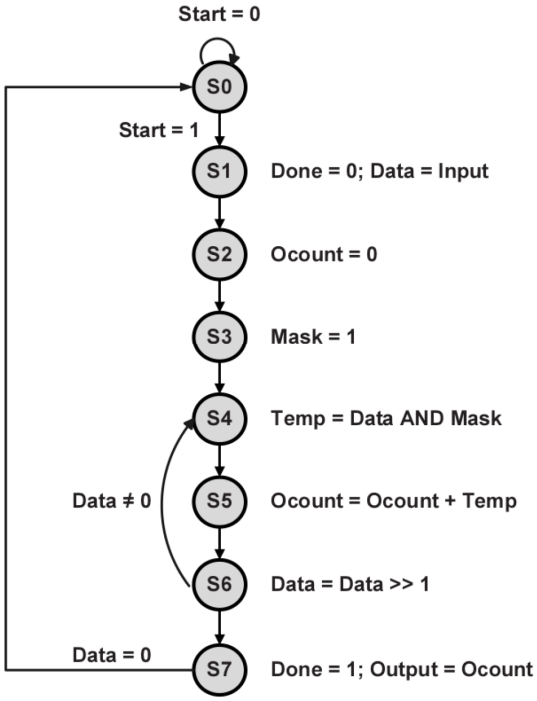
Projeto do Contador 1

Resolva o seguinte problema:

Há uma entrada de *n* bits. Ao receber um sinal de **iniciar**, seu módulo conta quantos 1s existem na entrada de *n* bits e produz na saída um inteiro correspondente a esta contagem.

Exemplo:
entrada: 101011100001011 (15 *bits*),
saída: 8

- Especificação (FSMD)



Descrição RTL:

```
while(1) {
    while(Start == 0);
    Done = 0;
    Data = Input;
    Ocount = 0;
    Mask = 1;
    while(Data > 0) {
        Temp = Data & Mask;
        Ocount = Ocount + Temp;
        Data >>= 1;
    }
    Output = Ocount;
    Done = 1;
}
```

Projeto do Contador 1

Resolva o seguinte problema:

Há uma entrada de *n* bits. Ao receber um sinal de **iniciar**, seu módulo conta quantos 1s existem na entrada de *n* bits e produz na saída um inteiro correspondente a esta contagem.

Exemplo:
entrada: 101011100001011 (15 *bits*),
saída: 8

- Projeto Customizado (*custom design*)

Descrição RTL simplificada:

```
while(1) {
    while(Start == 0);
    Done = 0;
    Data = Input;
    Ocount = 0;
    while(Data > 0) {
        if(bit menos significativo de Data == 1)
            Ocount = Ocount + 1;
        Data >>= 1;
    }
    Output = Ocount;
    Done = 1;
}
```

Usamos apenas 2 variáveis!

Descrição RTL:

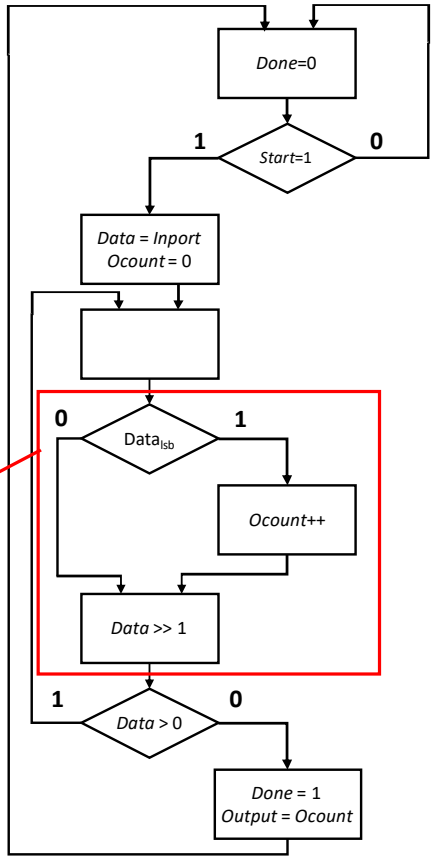
```
while(1) {
    while(Start == 0);
    Done = 0;
    Data = Input;
    Ocount = 0;
    Mask = 1;
    while(Data > 0) {
        Temp = Data & Mask;
        Ocount = Ocount + Temp;
        Data >>= 1;
    }
    Output = Ocount;
    Done = 1;
}
```

Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - diagrama ASM

Descrição RTL simplificada:

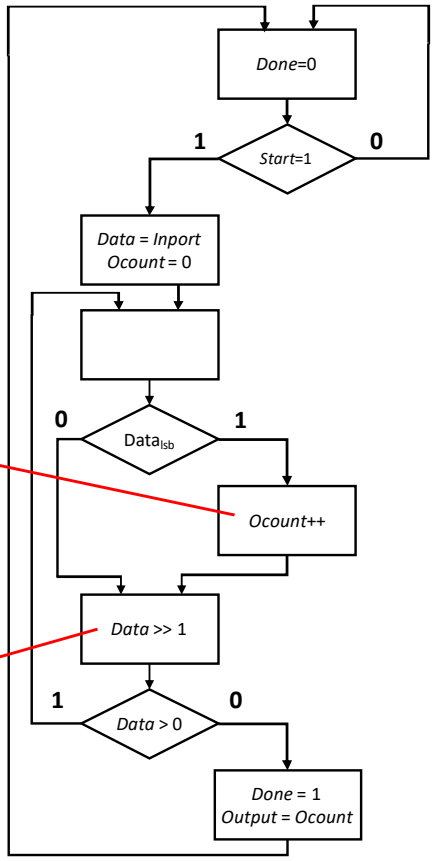
```
1  while(1) {
2      while(Start == 0);
3      Done = 0;
4      Data = Input;
5      Ocount = 0;
6      while(Data > 0) {
7          if(bit menos significativo de Data == 1)
8              Ocount = Ocount + 1;
9          Data >>= 1;
10     }
11     Output = Ocount;
12     Done = 1;
13 }
```



Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - componentes do Fluxo de Dados

1. Contador de 4 *bits*
2. Registrador deslocador de 15 *bits*



Projeto do Contador 1

PCS3225 - Sistemas Digitais II
Exercício da Aula 9 - Componentes RTL
Edson Midorikawa
Data: 18/09/2023

- Projeto Customizado (*custom design*)
  - componentes do Fluxo de Dados

1. Contador de 4 *bits*

```
entity contador4 is
    port (
        clock : in  bit;
        zera  : in  bit;
        conta : in  bit;
        Q     : out bit_vector(3 downto 0);
        fim   : out bit
    );
end entity contador4;
```

2. Registrador deslocador de 15 *bits*

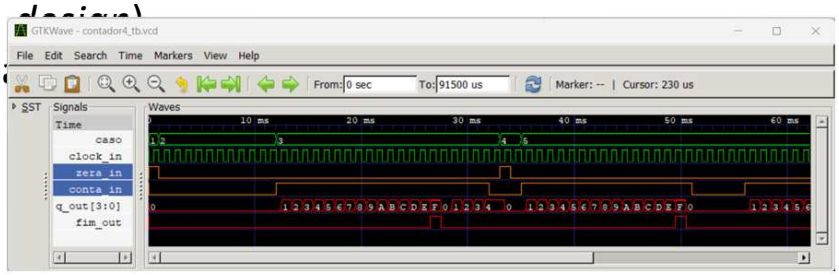
```
entity deslocador15 is
    port (
        clock   : in  bit;
        limpa   : in  bit;
        carrega : in  bit;
        desloca : in  bit;
        entrada : in  bit;
        dados   : in  bit_vector (14 downto 0);
        saida   : out bit_vector (14 downto 0)
    );
end entity deslocador15;
```

Projeto do Contador 1

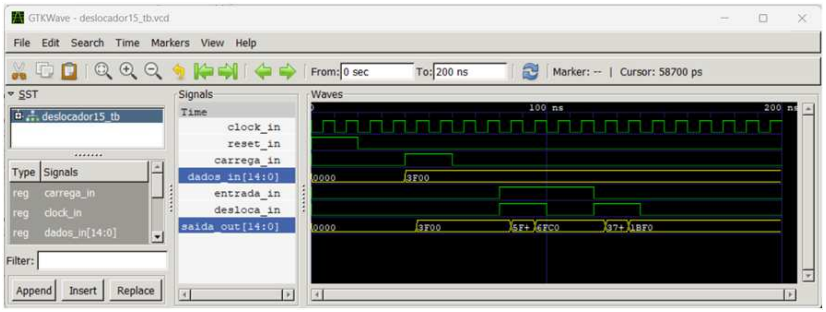
PCS3225 - Sistemas Digitais II
Exercício da Aula 9 - Componentes RTL
Edson Midorikawa
Data: 18/09/2023

- Projeto Customizado (*custom design*)
  - componentes do Fluxo de Dados

1. Contador de 4 *bits*



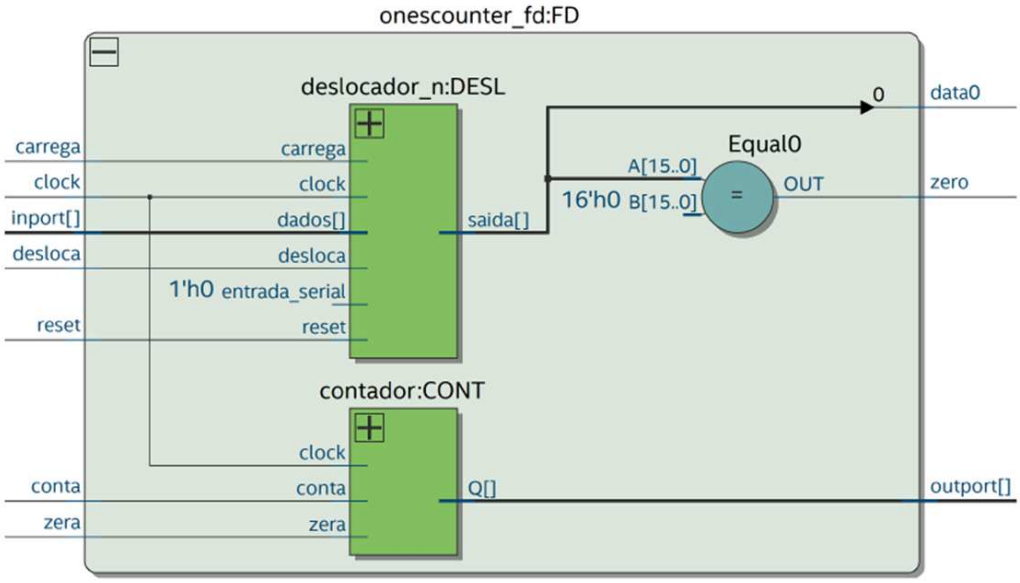
2. Registrador deslocador de 15 *bits*



Projeto do Contador 1

DICA: descrição estrutural no Capítulo 9 do "Free Range VHDL"

- Projeto Customizado (*custom design*)
  - componentes do Fluxo de Dados



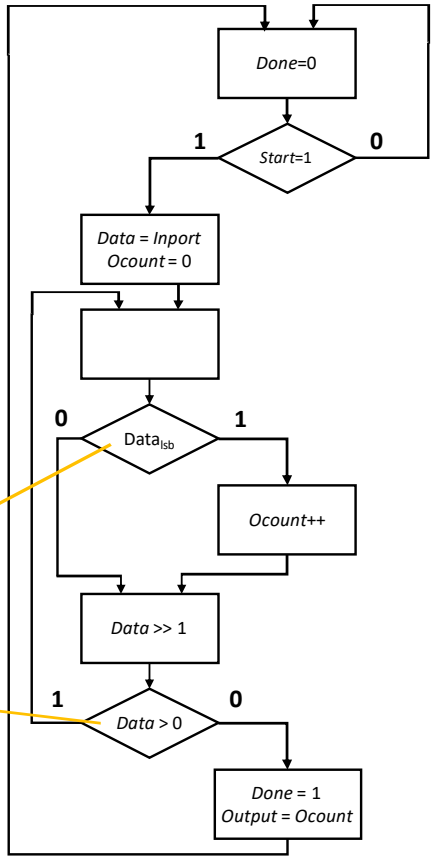
Projeto do Contador 1

- Projeto Customizado
  - entidade do Fluxo de Dados

```
entity onescounter_fd is
  port (
    clock     : in  bit;
    reset     : in  bit;
    inport    : in  bit_vector(14 downto 0);
    zera      : in  bit;
    conta     : in  bit;
    carrega   : in  bit;
    desloca   : in  bit;
    outport   : out bit_vector(3 downto 0);
    data0     : out bit;
    zero      : out bit
  );
end entity;
```

Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - testes das variáveis de condição

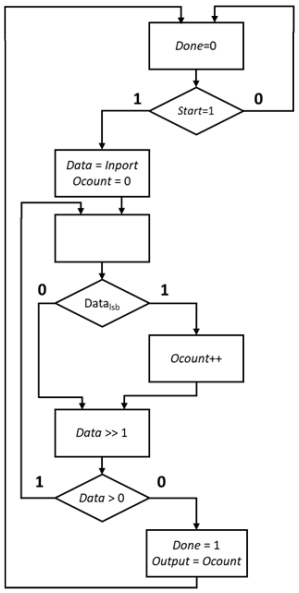


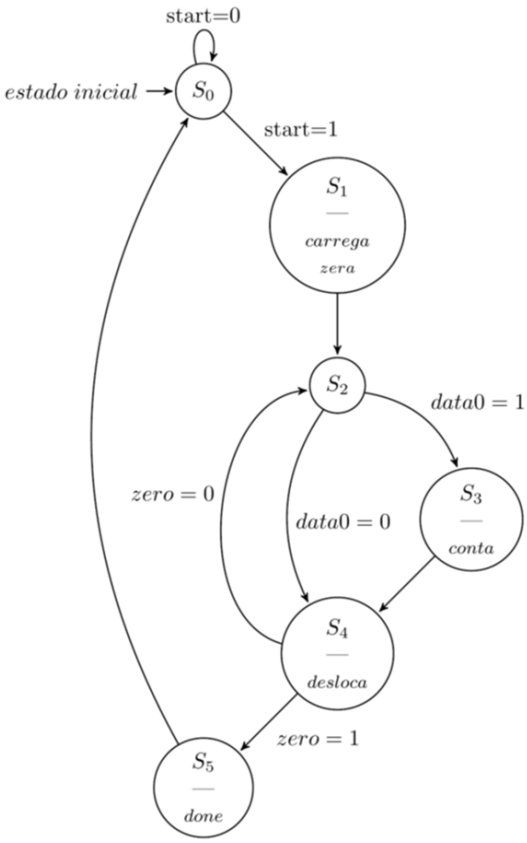
Data$_{lsb}$: bit menos significativo de Data

Data>0 : indica se tem mais *bits* 1

Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - unidade de controle

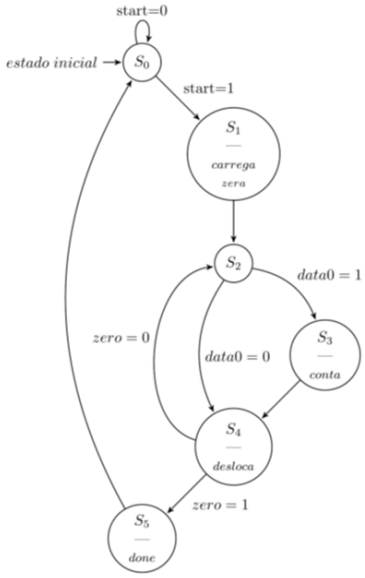




Projeto do Contador 1

DICA: Máquinas de estado em VHDL
https://balbertini.github.io/vhdl_fsm-pt_BR.html

- Projeto Customizado (*custom design*)
  - unidade de controle em VHDL

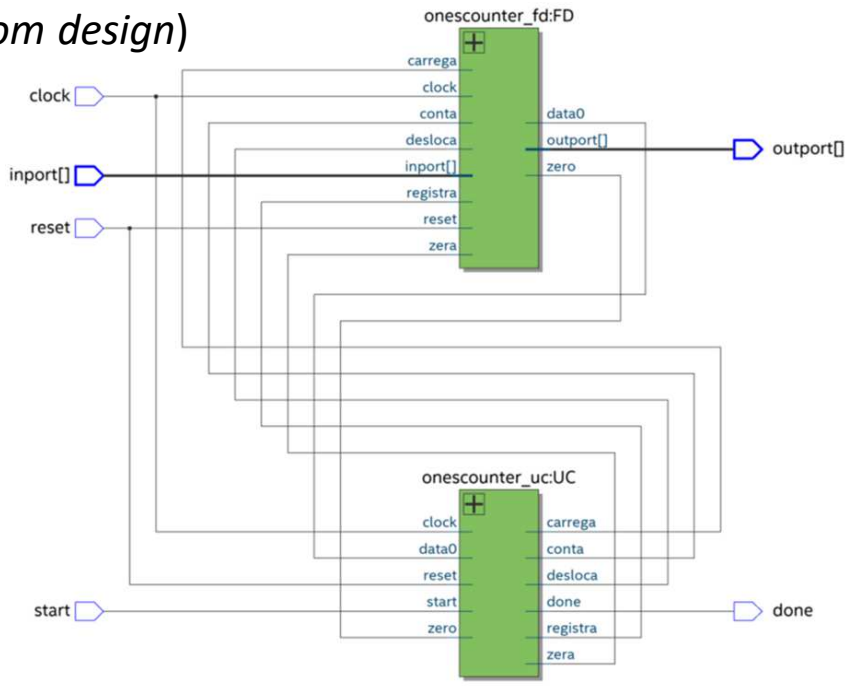



```
1  entity onescounter_uc is
2    port (
3      clock     : in  bit;
4      reset     : in  bit;
5      start     : in  bit;
6      data0     : in  bit;
7      zero      : in  bit;
8      zera      : out bit;
9      conta     : out bit;
10     carrega   : out bit;
11     desloca   : out bit;
12     done      : out bit
13   );
14 end entity;
```

Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - Sistema Digital (FD+UC)

Descrição Estrutural



Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - Sistema Digital (FD+UC)

Entidade VHDL

```vhdl
entity onescounter is
  port (
    clock   : in  bit;
    reset   : in  bit;
    start   : in  bit;
    inport  : in  bit_vector(14 downto 0);
    outport : out bit_vector(3 downto 0);
    done    : out bit
  );
end entity;
```

Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - Verificação por simulação

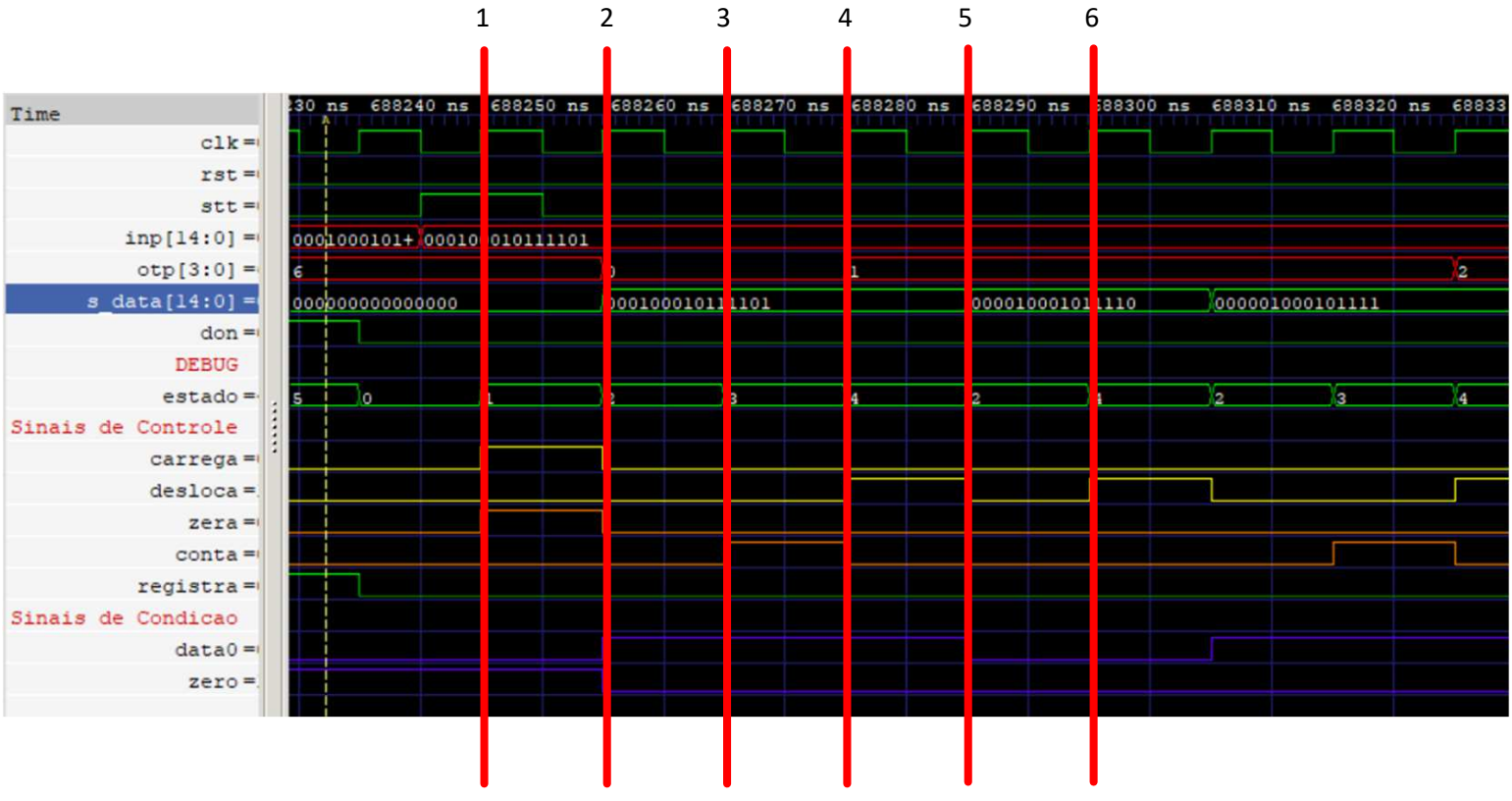
- Casos de teste (alternativas?)

- *Testbench*

```vhdl
entity onescounter_tb is
end entity;

architecture arch of onescounter_tb is

  component onescounter is port (...); end component;

  constant clocktime : time := 10 ns;
  signal clk, rst, simulando: bit := '0';
  signal st, don: bit;
  signal inp : bit_vector(14 downto 0);
  signal otp : bit_vector(3 downto 0);
  ...
```

Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - Verificação por simulação



Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - Verificação por simulação

- Casos de teste (alternativas?)
  - Subconjunto do domínio da entrada (quantos são suficientes?)
  - Teste exaustivo (qual é o tamanho?)

- Alternativas de implementação
  a) Formas de onda e/ou Assert/Report
  b) Programático, Vetor de teste ou Arquivo de dados

Projeto do Contador 1

- Projeto Customizado (*custom design*)
  - Verificação por simulação (caso 000100010111101, 7 *bits* 1)