Vitor Amadeu Souza

FPGA

Aplicações em VHDL
Volume V

© 2023 by Cerne Tecnologia e Treinamento Ltda.

© 2023 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

Maio de 2023

Direitos reservados por:

Cerne Tecnologia e Treinamento Ltda

Produção: *Cerne Tecnologia e Treinamento*

E-mail da Empresa: *cerne@cerne-tec.com.br*

Home Page: *www.cerne-tec.com.br.com.br*

Atendimento ao Consumidor: *sac@cerne-tec.com.br*

Contato com o Autor: *vitor@cerne-tec.com.br*



FEITO NO BRASIL

“Melhor é repelir dez verdades do que admitir uma única falsidade.”

Allan Kardec

Kits Didáticos e Gravadores da Cerne Tecnologia

A Cerne tecnologia têm uma linha completa de aprendizado para os microcontroladores da família PIC, 8051, Holtek, dsPIC, ARM e etc. Veja os detalhes de cada um nas figuras abaixo:



Kit Cerne CPLD

- CPLD ALTERA
- Botões
- Leds
- Gravação ICSP
- Display 7 Segmentos
- E muito mais!

Uma linha completa de componentes para o desenvolvimento de seus projetos eletrônicos como displays, PICs, botões, leds, cristais e etc.

Visite a nossa página na Internet, no endereço **www.cerne-tec.com.br** e conheça melhor nossos serviços e produtos.

Cerne
Conhecimento para o Desenvolvimento

www.cerne-tec.com.br

Sumário

Capítulo I – Metodologia de desenvolvimento.....6

Capítulo II – Exemplos no VHDL.....7

1. Flip-flop assíncrono RS com porta NAND.....7
2. Flip-flop assíncrono RS com porta NOR.....11
3. Flip-flop assíncrono RS com PRESET e RESET.....15
4. Flip-flop JK.....28

Capítulo I
Metodologia de desenvolvimento

A proposta desta obra é continuar apresentando exemplos em VHDL para FPGA. Exemplos com flip-flops RS assíncrono e síncrono e JK são temas abordados neste volume.

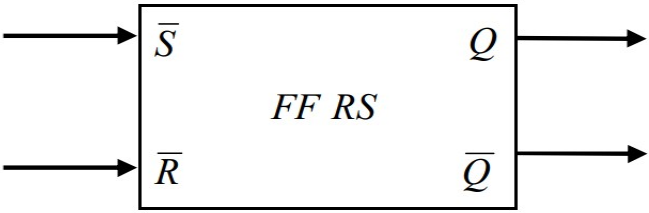
Para isso, recomenda-se que o leitor leia as obras dos volumes passados como *FPGA Aplicações em VHDL (2014), FPGA Aplicações em VHDL Volume II (2023), FPGA Aplicações em VHDL Volume III (2023) e FPGA Aplicações em VHDL Volume IV (2023)* do mesmo autor e editora de modo a continuar seus estudos com eficiência.

Capítulo II
Exemplos no VHDL

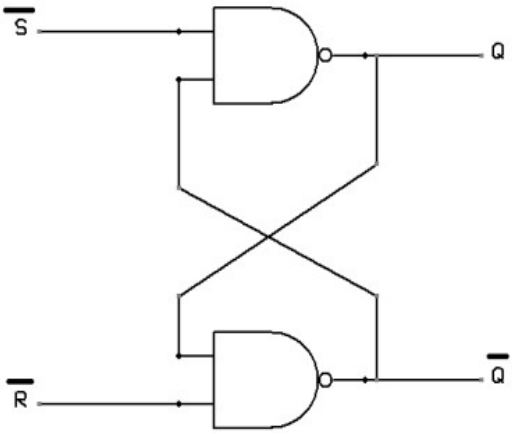
1. Flip-flop assíncrono RS com porta NAND

Podemos construir um elemento básico de memória chamado de flip-flop do tipo RS (Reset e Set) ou SR (Set e Reset) usando portas lógicas NAND. Este flip-flop possui duas entradas, uma chamada R (Reset) e outra S (Set) e duas saídas chamadas de Q e Q barrado.

Caso a entrada R seja ativada, a saída Q ficará em nível baixo e caso a entrada S seja ativa, a saída Q ficará em nível alto. A saída Q barrado é o complemento de Q. Observe que as saídas permanecem neste estado, pois o flip-flop é uma memória. Podemos representar de forma genérica um flip-flop RS (Reset-Set) ou SC (Set-Clear) da seguinte forma:



Observe que tanto a entrada R quanto S são ativas em nível baixo, pois são do tipo barrado. A construção deste flip-flop está apresentado a seguir, fazendo uso de portas lógicas NAND.

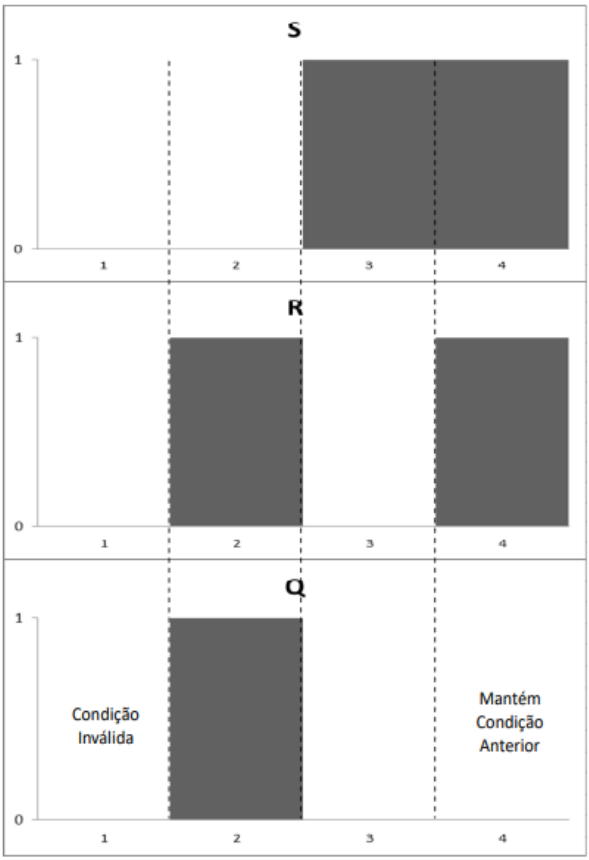


A seguir a tabela verdade deste flip-flop.

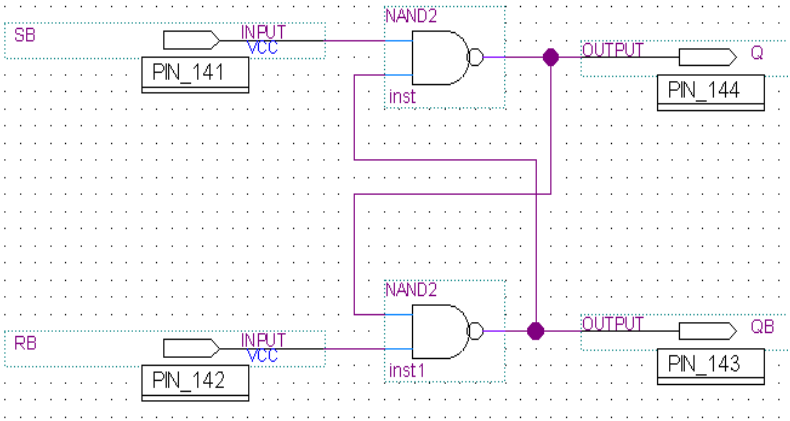
S	R	Q
0	0	Inválido
0	1	1
1	0	0
1	1	Q_0

Observe que não podemos utilizar este flip-flop com as suas entradas em nível 0 simultaneamente. Quando S=0 e R=1, a saída Q=1. De outra forma, quando S=1 e R=0, a saída Q=0. No estado em que S e R estão em nível alto, a saída Q permanece sem alteração de seu estado.

A próxima carta de tempos mostra o comportamento (behavior) deste flip-flop.



É possível simular o funcionamento deste flip-flop no esquemático (schematic), fazendo o circuito conforme apresentado abaixo.



Acompanhe a seguir o código VHDL que implementa tal solução.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.ALL;

entity ios is
port(clk : in std_logic;
          q  : inout std_logic;
          qb : inout std_logic;
          sb : in std_logic;
          rb : in std_logic);
end ios;

architecture flipflop of ios is
begin
   process(clk)
```