Lab. 03 - Verilog

Latches, Flip-flops e Registradores

Projetos de Sistemas Integrados

Prof. Roberto Ribeiro Neli

Nestes experimentos vamos aprofundar os circuitos latches, flip-flops e registradores.

Parte I

FPGAs Altera incluem flip-flops que estão disponíveis para os usuários utilizem na implementação de seus circuitos. Você vai aprender a criar elementos de armazenamento utilizando o Verilog como ferramenta.

A figura 1 ilustra uma latch RS. Dois tipos de código Verilog podem ser utilizados para descrever estes circuitos, conforme ilustrado na figura 2. A figura 2ª ilustra uma latch utilizando lógica instanciada e a 2b utilizando expressões lógicas. Se esta latch for projetada em FPGA utilizando "*lookup tables*" de 4 entradas (LUTs – blocos construídos a partir de uma tabela verdade), então somente um LUT é necessário, conforme ilustrado na Figura 3a.

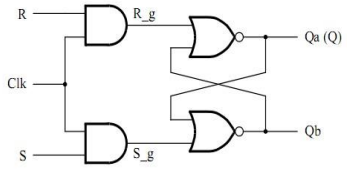


Figure 1. Circuito da latch RS.

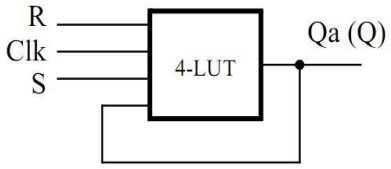
```
// A gated RS latch
module part1 (Clk, R, S, Q);
   input Clk, R, S;
   output Q;

   wire R_g, S_g, Qa, Qb /* synthesis keep */ ;

   and (R_g, R, Clk);
   and (S_g, S, Clk);
   nor (Qa, R_g, Qb);
   nor (Qb, S_g, Qa);

   assign Q = Qa;

endmodule
```

Figura 2a. RS latch com lógica instanciada.

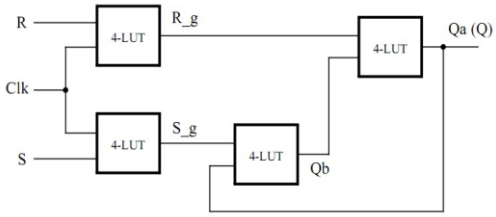
```
// A gated RS latch
module part1 (Clk, R, S, Q);
   input Clk, R, S;
   output Q;

   wire R_g, S_g, Qa, Qb /* synthesis keep */ ;

   assign R_g = R & Clk;
   assign S_g = S & Clk;
   assign Qa = ~(R_g | Qb);
   assign Qb = ~(S_g | Qa);

   assign Q = Qa;

endmodule
```

Figura 2b. RS latch usando expressões lógicas.

Embora a latch possa ser projeta corretamente utilizando a LUT de 4 entradas, esta implementação não permite a criação de sinais internos, como R_g e S_g, porque eles não são saídas da LUT. Para manter estes sinais internos no circuito implementado, é necessário incluir um direcionador de compilação no código. Na Figura 2 o diretivo **/* synthesis keep */** foi incluído para instruir o Quartus II a usar elementos lógicos separados para cada sinal R_g, S_g, Qa, e Qb.



(a) Latch utilizando um LUT de 4 entradas.



(b) Latch utilizando 4 LUTs de 4 entradas.

Figura 3. Latch RS da Figura 1 implementada.

Crie uma Latch RS no Quartus II utilizando o código Verilog mostrado na figura 2 (Pode ser qualquer um deles). Siga as instruções:

1. Crie o primeiro código sem o diretivo **/* synthesis keep */;**
2. Clique em: **Tools → NETLIST VIEWERS → TECHNOLOGY MAP VIEWER (POST-MAPPING).** Veja que para este caso é criado somente um LUT, onde os *wires* estão codificados dentro desta LUT.
3. Coloque agora o comando **/* synthesis keep */**;
4. Clique em: **Tools → NETLIST VIEWERS → TECHNOLOGY MAP VIEWER (POST-MAPPING).** Veja que neste caso o circuito criado é parecido com o ilustrado na Figura 3b, formado por 4 LUTs e com os *wires* criados fisicamente.

Parte II

A Figura 4 ilustra uma Latch do tipo D.

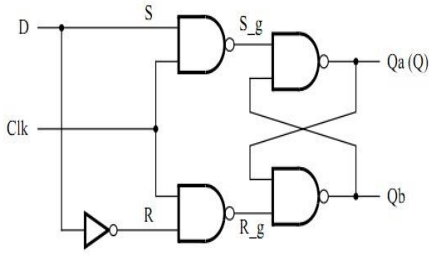


Figura 4. Circuito Latch D.

Crie um código em Verilog, seguindo os passos a seguir:

1. Copie o código da Figura 2b, adaptando para que ele faça a função de uma Latch do tipo D. Use o diretivo **/* synthesis keep */** para garantir que os sinais **R**, **S_g**, **R_g**, **Qa**, e **Qb** sejam criados em elementos lógicos separados.
2. Crie um "**University Programa VWF**", simule o seu projeto e veja se está funcionado adequadamente.
3. Implemente seu projeto no Kit DE2-115. Utiliza a chave SW_0 para inserir o sinal D, a SW_1 como Clk. Conecte a saída Q no $LEDR_0$.

Parte III

A Figura 5 ilustra um flip-flop D *master-slave*.

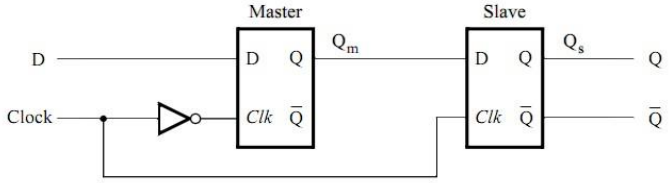


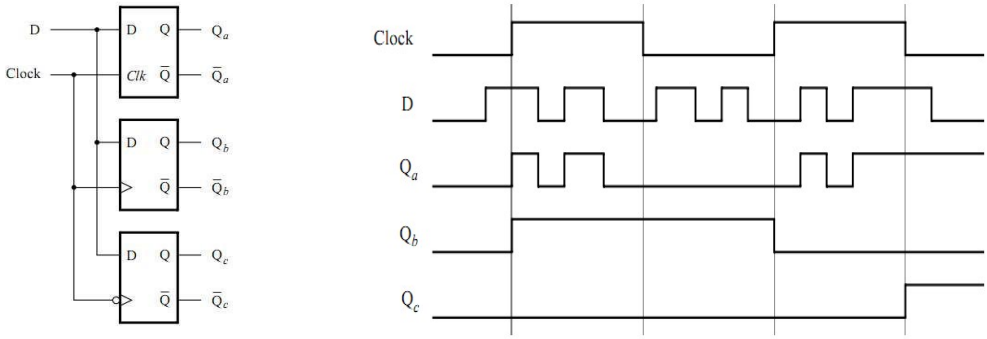
Figura 5. Circuito de um flip-flop D *master-slave*.

Desenvolva um código em Verilog que execute a função do Flip-Flop da Figura 5.

1. Crie um "**University Programa VWF**", simule o seu projeto e veja se está funcionado adequadamente.
2. Implemente seu projeto no Kit DE2-115. Utiliza a chave SW_0 para inserir o sinal D, a SW_1 como Clk. Conecte a saída Q no $LEDR_0$.

Parte IV

A Figura 6 ilustra um circuito com 3 elementos de memória: uma latch D, um flip-flop D em flanco positivo, e um flip-flop D em flanco negativo.



(a) Circuito (b) Diagrama de tempo

Figure 6. Circuito e *waveforms*.

Projete um circuito em Verilog que execute a função dos 3 elementos de memória.

Implemente seu projeto no Kit DE2-115. Utiliza a chave SW_0 para inserir o sinal D, a SW_1 como Clk. Conecte as saídas Qa, Qa/, Qb, Qb/, Qc e Qc/ nos $LEDR_0$ ~ $LEDR_5$.

Parte V

```
module D_latch  (D, Clk, Q);
   input D, Clk;
   output reg Q;

   always @ (D, Clk)
      if (Clk)
         Q = D;
endmodule
```

Figura 7. Código Verilog de uma Latch D.

Projete um algoritmo em Verilog que receba 2 números de 16 bits, A e B. Os números devem ser informados utilizando as chaves $SW_{15\sim0}$. Os números A e B devem ser mostrados nos displays $HEX_{7\sim4}$. Primeiro deve ser informado o número A, que deve ser armazenado na memória e depois deve ser informado o número B, utilizando as mesmas chaves $SW_{15\sim0}$. Utilize KEY_0 como reset assíncrono e KEY_1 como clock.

= Chamar o professor para validar o exercício.

Adaptado dos exemplos que acompanham o KIT DE2-115.