UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Relatório – Máquina de estados em VHDL I

Disciplina: ELE0518 – Laboratório de Sistemas Digitais

Alunos: Bruno Matias de Sousa
Levy Gabriel da Silva Galvão
Pedro Henrique de Souza Fonsêca dos Santos

Data: 24/05/2019

1. Introdução

Diante das dificuldades enfrentadas em projetos anteriores, bem como as limitações devido à complexidade dos problemas, uma solução para uma máquina de estados de mais alto nível é implementá-la com o uso de VHDL, uma linguagem de descrição de hardware que nos auxilia na simplificação dos circuitos mais complexos na sua montagem em protoboard, utilizando uma placa FPGA ou similares para se utilizar com a linguagem.

O objetivo dessa prática é implementar uma máquina de estados básica em VHDL. Assim, permitindo fixar o conhecimento obtido acerca do assunto de circuitos sequenciais e relacionado ao uso da sintaxe em VHDL para formalizar o código.

2. Referencial teórico

VHDL é uma linguagem de descrição de *hardware* e significa *Very High Speed Integrated Circuit Hardware Description Language*. Existem várias como ela, que se encaixam como *Hardware Description Language* (HDL), à título de exemplo: Verilog; AHDL (Altera Hardware Description Language); SystemC.

Com esse tipo de linguagem, a arquitetura de um projeto pode ser implementada mais facilmente por meio da escrita de um código em um nível mais alto de abstração. Isso reduz o tempo de projeto e de testes, uma vez que a placa na qual o código será carregado se encarrega de definir os caminhos e as portas lógicas a serem utilizadas. Assim, evita-se toda uma cultura de testar o projeto em protoboard por meio de circuitos integrados e com uma grande quantidade de fios. O problema desse tipo de linguagem é que a placa se encarregará da destinação do uso das portas lógicas e caminhos, implicando que o hardware obtido nem sempre será o mais otimizado possível.

Como forma de aplicar o conhecimento acerca da linguagem, foi proposto nessa prática o projeto de uma simples MdE. Ela deverá possuir uma entrada H, o *reset* (R) e cinco saídas, cada um de 1 bit. A sequência a ser produzida será:

$$00000 \rightarrow 10000 \rightarrow 01000 \rightarrow 00100 \rightarrow 00010 \rightarrow 00001$$

Deverá repetir a sequência enquanto H=1. Quando H=0 deve-se voltar imediatamente para o estado com saída 00000. A máquina deve possuir um contador que vai contar um total de 3 ciclos completo. Quando isso acontecer a máquina deverá ir para um estado onde a saída é 11111 e só sairá quando H=0. Vale destacar que quando H=0 o contador não será zerado, mas quando o *reset* for ativado, a máquina de estados deverá ir para o estado inicial e com o contador de ciclos zerado.

3. Metodologia

O presente projeto fez uso dos seguintes equipamentos:

- Placa do Kit DE2 da Altera;
- Quartus.

Inicialmente fora concebida a máquina de estados. Não sem antes de determinar o que significa cada estado. Os estados S de subscrito 0 a 5 representarão os estados do ciclo normal com suas respectivas saídas elencadas na figura abaixo. O sétimo estado será o novo estado que só será acessado após 3 ciclos. O estado R foi denominado na estirpe desse projeto como um pseudo estado, pois ele não será implementado no código com o mesmo tipo que os estados anteriores, mas ele representará a operação de atribuição do zero ao contador após acionar o reset.

ESTADOS	SAÍDAS
S_0	00000
S_1	10000
S_2	01000
S_3	00100
S_4	00010
S_5	00001
S_6	11111
R	-----

Figura 1 - Tabela de com as respectivas saídas para cada estado.

Com todos os estados definido, fora construída a máquina de estados de alto nível, de acordo com a figura abaixo. Sendo a variável *count* representada como o contador de ciclos.

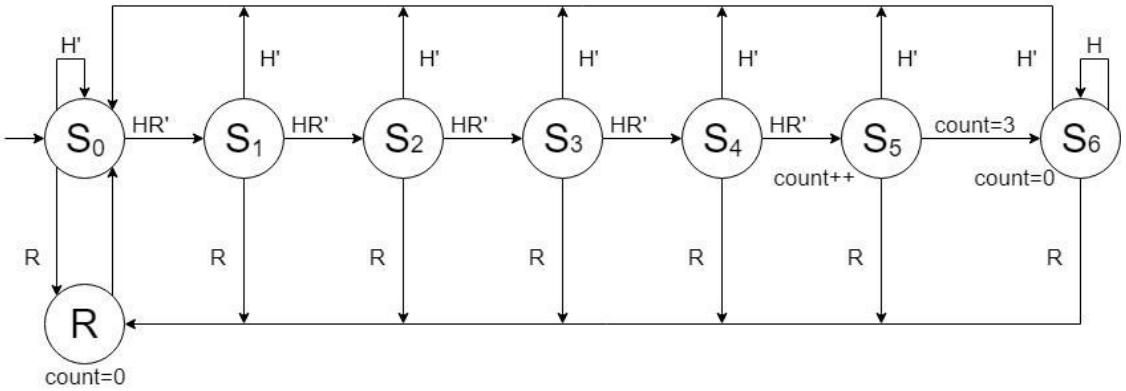

Figura 2 - Diagrama da máquina de estados para o problema. (R é um estado que não existe na implementação, ele só ilustra que quando o *reset* for dado o contador será reiniciado).

Agora será explicado o desenvolvimento do código fonte do projeto geral, destinado à placa. No que diz respeito à entidade escolhida, seu nome foi `lab7` e possui 3 entrada e 2 saídas. A entrada `h` e `rst` referem-se, respectivamente às entradas H e R da MdE. A entrada `clk_in` vai receber o clock de 50Mhz da placa DE2 da altera e vai retornar um clock de 1Hz no buffer `clk` para uso interno no código. A saída `a` representa um vetor de bits de 5 bits e representa as saídas de cada estado.

```
entity lab7 is
    port(
        h           : in std_logic;
        clk_in, rst : in std_logic;
        clk         : buffer std_logic;
        a           : out std_logic_vector (4 downto 0)
    );
end lab7;
```

Para que seja realizada a divisão do clock de 50MHz para 1Hz foi usado um componente, como descrito abaixo. A entrada `clkIn` receberá o `clk_in` do código principal, que por sua vez receberá o clock de 50Mhz e retornará o clock de 1Hz na saída `clkOut` que passará para o código principal como `clk`.

Para acessar esse novo clock foi usado um port map. Assim, logo após ao `begin` da `architecture`, fora colocado o `port map`.

```
 component ClockDiv is
   port(
       clkIn  : in std_logic;
       clkOut : out std_logic
    );
 end component;

 begin
    clock : ClockDiv port map(clk_in, clk);
```

Para facilitar o projeto foi criado o tipo `state` que representa os estados válidos da MdE e mais dois sinais que armazenarão o estado atual (`actual_s`) e o próximo estado (`next_s`), como pertencentes ao tipo `state`. O trecho de código pode ser observado abaixo.

```
type state is (s0, s1, s2, s3, s4, s5, s6);
signal actual_s, next_s: state;
```

Para o projeto foi utilizado apenas um processo. De acordo com o trecho de código abaixo, ele é sensível apenas ao clock e ao reset.

```
state_register: process (clk, rst)
```

Esse processo possui uma estrutura condicional principal na forma de um `if` que para o caso do `rst` ser acionado ele executa as devidas ações e para quando for detectado a borda de subida do clock por meio do `rising_edge(clk)` ele executar as devidas ações por meio de uma estrutura condicional do tipo `case` para avaliar qual o estado atual e decidir qual o próximo estado.

Após a devida compilação e simulações do código, suas entradas e saídas foram mapeadas segundo a pinagem da placa DE2 da Altera. Assim, foi também simulado na placa.

4. Resultados práticos

Após concluir o código fonte do VHDL, o código final foi compilado e foram feitas algumas simulações para comprovar se a lógica do projeto condiz com a desejada.

Lembrando que na simulação as entradas H e R da MdE são, respectivamente, h e rst. O clock de operação atribuído para a simulação está na variável clk. As saídas estão armazenados no vetor de bits a.

A primeira simulação consistiu em manter a chave H=1 durante toda a simulação. Isso permitiu que os três ciclos fossem completados e quando alcançado o estado de saída 11111, permaneceu nele, pois H=1. O reset permaneceu no 0, R=0.

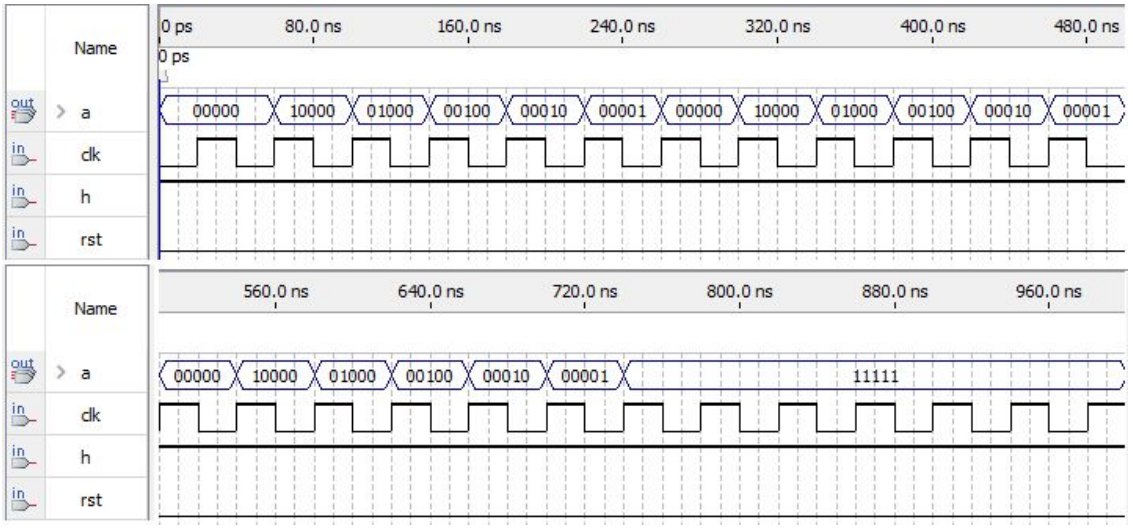

Figura 3 - Simulação para a entrada H=1 e reset=0 (a parte inferior é a sequência da parte superior).

Já na simulação seguinte, observada na figura abaixo, após os três ciclos e ao chegar no estado de saída 11111, a chave H foi para 0 por um ciclo de clock e voltou a 1, retornando a contar do estado inicial. Uma vez que alcançou o estado 11111 o contador estava zerado.

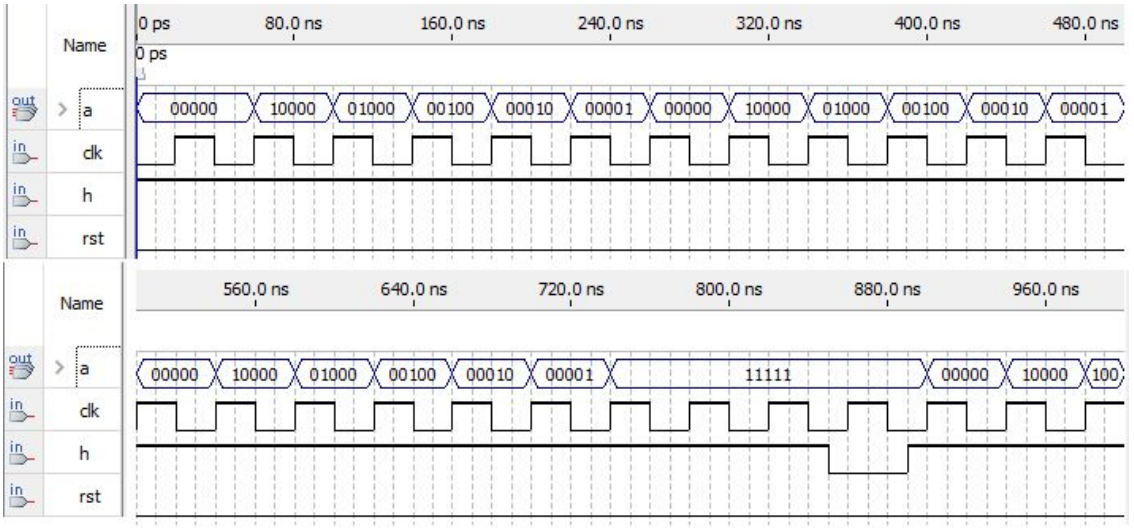

Figura 4 - Simulação para a entrada H=1, indo para H=0 após os três ciclos e voltando para H=1 e reset=0 (a parte inferior é a sequência da parte superior).

Na seguinte simulação, observada na figura abaixo, o H permaneceu em 1 durante todo o tempo. Porém, durante uma parte do segundo ciclo o reset foi acionado, R=1, permitindo que voltasse imediatamente para o estado inicial e o contador foi zerado. O reset permaneceu acionado apenas por um período de clock. A prova de que o contador foi zerado é que mais dois ciclos foram simulados e não entrou no estado de saída 11111.

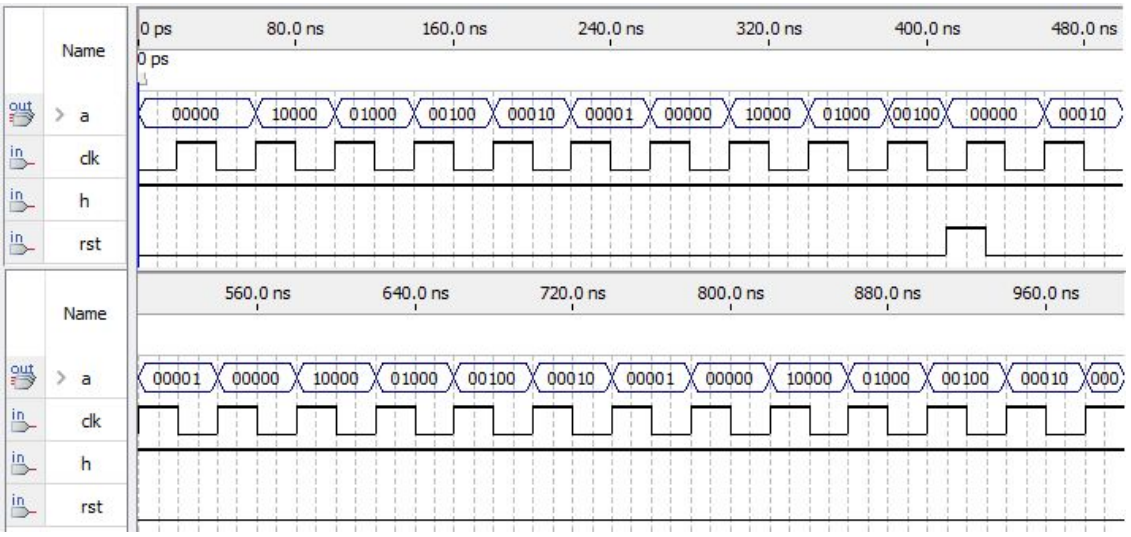

Figura 5 - Simulação para a entrada H=1 e reset=1 em algum ponto do segundo ciclo, retornando a zero permitindo a continuidade do ciclo, porém com o contador zerado (a parte inferior é a sequência da parte superior).

Nessa outra simulação, da figura abaixo, o raciocínio foi semelhante à da simulação anterior, porém ao invés do reset ser acionado, foi o H que voltou para 0 durante a metade do segundo ciclo. Após um período de clock ele voltou a ser 1, H=1, permitindo que a máquina retornasse a partir do estado inicial, mas com o contador não zerado. A prova foi que após os dois ciclos que o H voltou a ser 1, o contador marcou três ciclos e foi para o estado de saída 11111.

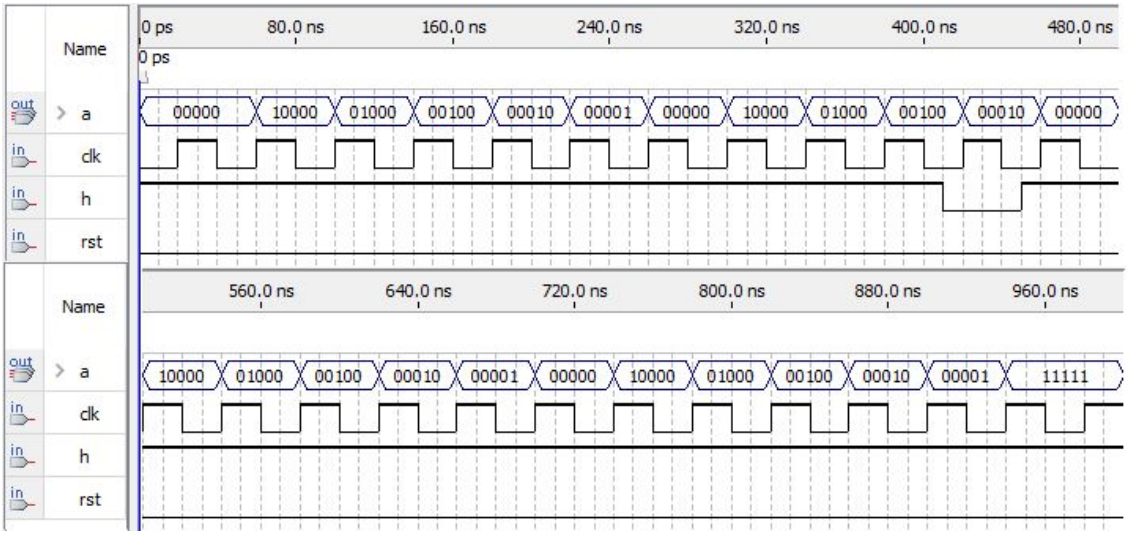

Figura 6 - Simulação para a entrada H=1 e H=0 em algum ponto do segundo ciclo, retornando a zero permitindo a continuidade do ciclo, porém com o contador não zerado, permitindo que após a conclusão de mais dois ciclos seja alcançada a saída 11111 (a parte inferior é a sequência da parte superior).

Assim, essa gama de simulações contempla o comportamento da máquina de estados que se desejou projetar. O clock utilizado nas simulações foi escolhido arbitrariamente. Porém o clock da placa DE2 da Altera foi reduzido para 1Hz para ser utilizado na simulação do projeto na FPGA.

Os resultados após carregar o código na placa condisseram com as simulações realizadas no Quartus.

5. Conclusão

O resultado do trabalho foi satisfatório e correspondeu a todas as expectativas delimitadas no escopo do projeto, além disso, foi possível perceber que a facilidade de se trabalhar com código em vez da montagem em si do circuito facilita possíveis projetos de alta complexidade de circuitos digitais.

Esse projeto se mostrou essencial para introduzir o discente ao uso de VHDL, aplicar a sintaxe apreendida na teoria. Também permitiu aprimorar a experiência em combate a erros mais comuns de código.

Uma das dificuldades da resolução do problema nos apresentadas no projeto se dar no fato da comunicação do código com a placa FPGA, que precisou se entender um pouco mais de como realizar tal operação.

Outro ponto essencial foi permitir a introdução da aplicação do projeto na FPGA. Seja no que consta a posição de cada pino e a função de cada um e como eles podem contribuir para representar as entradas e saídas do projeto.

5. Referências Bibliográficas

ABNT, Associação Brasileira de Normas Técnicas. **NBR 10719 – Apresentação de relatórios técnico-científicos**. Rio de Janeiro: ABNT, Copyright © 1989.

MARCONI, Marina de A. & LAKATOS, Eva M. **Fundamentos de metodologia científica**. 5 ed. Editora Atlas. São Paulo, 2003.

TOCCI, Ronald J. **Digital Systems**: principles and applications. 11 ed. Pearson Education India, 1991.

Slide e anotações da Aula. **Linguagem de Descrição de Hardware**. Professor: Carlos Yuri Ferreira Silva.