Laboratório de Sistemas Digitais

Trabalho Prático nº 10

Modelação em VHDL de Memórias ROM e RAM de um Porto e Multi-porto

Objetivos

- Modelação em VHDL de memórias ROM e RAM com escrita síncrona e leitura (as)síncrona.
- Simulação e implementação em FPGA de memórias de um ou mais portos (multi-porto).
- Parametrização de memórias.

Sumário

Este trabalho pretende introduzir vários tipos de memórias e a sua descrição em VHDL para simulação e implementação em FPGA. Na primeira parte vai ser implementada uma memória ROM (*Read Only Memory*), que tem armazenado um conjunto de valores predefinidos, lidos inicialmente de forma manual e seguidamente de forma sequencial utilizando um contador. Na segunda parte vai ser modelada uma memória RAM (*Random Access Memory*), de leitura/escrita, em que estas operações vão ser controladas manualmente através de *switches* e o valor lido da memória visualizado em LEDs. Na terceira parte é implementada uma RAM de dois portos, um dos quais será utilizado para escrita, controlada por *switches*, e um de leitura assíncrona controlada por um contador. A quarta e quinta partes são dedicadas à construção de modelos de memórias parametrizáveis, em que o número de palavras da memória (profundidade da memória) e o número de bits de cada palavra (largura da memória) são definidos estaticamente aquando da instanciação. Finalmente a sexta parte é dedicada à implementação de uma memória com leitura síncrona.

Parte I

1. Abra a aplicação *"Quartus Prime"* e crie um novo projeto para a FPGA Cyclone IV EP4CE115F29C7. Poderá designar o projeto e a entidade *top-level* como "ROM_Demo".

2. Descreva em VHDL uma memória ROM de 16 posições e com um comprimento de palavra de 8 bits. Designe a entidade por "ROM_16_8". A informação armazenada na ROM deverá ser definida através de uma constante inicializada com os valores que entender adequados.

3. Crie um símbolo para poder usar o módulo "ROM_16_8" em diagramas lógicos.

4. Crie um novo ficheiro, chamado "ROM_Demo.bdf", onde deverá instanciar graficamente o módulo "ROM_16_8" e realizar a sua ligação a dispositivos do *kit*, para que seja possível estabelecer o endereço de leitura através de *switches*, assim como a visualização em LEDs do valor lido da memória (valor armazenado no endereço de memória definido pelos *switches*).

5. Compile o projeto e teste-o no *kit* (não se esqueça de importar o ficheiro "master.qsf").

6. Repita os pontos anteriores, mas utilizando um contador incrementado à frequência de 1Hz para endereçar a memória. Desenhe previamente no seu *log book* o diagrama de blocos do circuito (**sugestão**: faça uso dos componentes que considerar convenientes, implementados em trabalhos práticos anteriores).

[TPC] Repita o ponto 6 para a frequência de 10Hz.

Parte II

1. Crie um novo projeto para a FPGA Cyclone IV EP4CE115F29C7. Poderá designar o projeto e a entidade *top-level* como "RAM_1P_Demo".

2. Implemente em VHDL uma memória RAM de 16 posições, com um comprimento de palavra de 8 bits e com um porto de acesso para escrita síncrona e leitura assíncrona (i.e. um único endereço para leitura/escrita). Designe e entidade por "RAM_1P_16_8.vhd". Os portos de entrada e saída da entidade poderão seguir a nomenclatura adotada nas aulas teórico-práticas, ou seja **`clk, address, writeEnable, writeData e readData.`**

3. Verifique, por simulação, o comportamento da RAM, criando para o efeito um ficheiro VHDL com uma *testbench* adequada. O processo para aplicação dos vetores de simulação deverá permitir a escrita e leitura do conteúdo de vários endereços de memória (idealmente todos) – ver slides da aula teórico-prática para um exemplo de construção de uma *testbench* para uma memória.

4. Crie um novo ficheiro, chamado "RAM_1P_Demo.vhd", para instanciar o módulo "RAM_1P_16_8", permitindo que a escrita/leitura seja controlada por *switches* e visualizada em LEDs do *kit*, isto é, utilizando os *switches* para definir o endereço da memória a aceder (**`address`**), o valor a escrever (**`writeData`**) e o sinal que habilita a escrita (**`writeEnable`**). Para o sinal de relógio deve usar um botão de pressão (e.g. KEY(0)). O valor da posição de memória endereçada (**`readData`**) deverá ser mostrado em LEDs do *kit*.

5. Compile o projeto e teste-o no *kit* (não se esqueça de importar o ficheiro "master.qsf").

Parte III

1. Crie um novo projeto para a FPGA Cyclone IV EP4CE115F29C7. Poderá designar o projeto e a entidade *top-level* como "RAM_2P_Demo".

2. Implemente em VHDL uma memória RAM de 16 posições com um comprimento de palavra de 8 bits e com dois portos de acesso (um para escrita síncrona e outro para leitura assíncrona, existindo entradas de endereçamento separadas para escrita e para leitura). Designe a entidade por "RAM_2P_16_8.vhd". Os portos de entrada e saída da entidade VHDL poderão seguir a nomenclatura adotada nas aulas teórico-práticas, ou seja **`clk`**, **`writeEnable`**, **`writeAddress`**, **`writeData`**, **`readAddress`** e **`readData`**.

3. Crie um símbolo para poder usar o módulo "RAM_2P_16_8" em diagramas lógicos.

4. Crie um novo ficheiro, chamado "RAM_2P_Demo.bdf" para instanciar graficamente o módulo "RAM_2P_16_8". A escrita na RAM deve ser controlada por *switches* (tal como descrito na parte II deste trabalho). Para a leitura deverá utilizar um contador incrementado à frequência de 1Hz, que vai "percorrer" (endereçar) toda a memória de forma cíclica. A saída **`readData`** deverá ser ligada a LEDs para visualização do valor lido da memória.

5. Compile o projeto e teste-o no *kit* (não se esqueça de importar o ficheiro "master.qsf").

Parte IV

Repita a parte II fazendo a parametrização da memória, isto é, permitindo através de *generics* em VHDL especificar a profundidade (número de palavras de armazenamento) e a largura (número de bits por palavra). De seguida instancie no ficheiro *top-level* (em VHDL) a memória parametrizável, configurando-a com a mesma dimensão que a "RAM_1P_16_8". Compile o projeto e teste-o no *kit* (não se esqueça de importar o ficheiro "master.qsf").

Parte V

Repita a parte III fazendo a parametrização da memória, isto é, permitindo através de *generics* em VHDL especificar a profundidade (número de palavras de armazenamento) e a largura (número de bits por palavra). De seguida instancie no ficheiro *top-level* (em diagrama lógico) a memória parametrizável, configurando-a com a mesma dimensão que a "RAM_2P_16_8". Compile o projeto e teste-o no *kit* (não se esqueça de importar o ficheiro "master.qsf").

Parte VI

1. Crie um novo projeto para a FPGA Cyclone IV EP4CE115F29C7. Poderá designar o projeto e a entidade *top-level* como "RAM_2PSync_Demo".

2. Implemente em VHDL uma memória RAM de 16 posições com um comprimento de palavra de 8 bits e com dois portos de acesso (um para escrita síncrona e outro para leitura síncrona, existindo entradas de endereçamento separadas para escrita e leitura). Designe e entidade por "RAM_2PSync_16_8.vhd". Os portos de entrada e de saída da entidade VHDL poderão seguir a nomenclatura adotada nas aulas teórico-práticas, ou seja **`writeClk`**, **`writeEnable`**, **`writeAddress`**, **`writeData`**, **`readClk`** , **`readAddress`** e **`readData`**.

3. Crie um símbolo para poder usar o módulo "RAM_2PSync_16_8" em diagramas lógicos.

4. Crie um novo ficheiro, chamado "RAM_2PSync_Demo.bdf" para instanciar graficamente o módulo "RAM_2PSync_16_8". A escrita e a leitura da RAM deverão ser ambas controladas por *switches*. A saída **`readData`** deverá ser ligada a LEDs para visualização do valor lido da memória.

5. Compile o projeto e teste-o no *kit* (não se esqueça de importar o ficheiro "master.qsf").

PDF criado em 04/04/2018 às 16:43:09