Biblioteca LPM - Memória RAM

Guimarães, João Guilherme M.
joaog95@live.com

7 de setembro de 2019

1 Introdução

A biblioteca LPM é uma padronização da industria, para fornecer módulos eficientes e de grande abstração para os projetistas de hardware, e devido sua integração com o Quartus, iremos utilizar o módulo de memória RAM com a configuração 32 posições x 8 bits.

2 Objetivos

Com esta prática, espera-se um melhor entendimento do módulo de memória da biblioteca LPM e uma boa assimilação do conteúdo visto em sala de aula sobre o processo de escrita e leitura de dados.

3 Material

Para realização desta prática, foi utilizado os seguintes equipamentos e softwares:

- Logisim 2.7.1;
- ModelSim 10.1d;
- Quartus 13.0sp1;
- FPGA EP2C35F672C6 e
- Kernel Linux / SO Deepin 15.11.

4 Desenvolvimento

A primeira aula prática da disciplina de Laboratório de Arquitetura de Computadores II, é dividida em 2 partes, a primeira consiste em simular o módulo de memória RAM da biblioteca LPM e realizar os processos de escrita e leitura. Já a segunda parte da prática, consiste em inicializar a memória utilizando um arquivo MIF (*Memory Inicialization File*).

De acordo com o roteiro de aula, os dados obtidos nas duas práticas deveram ser apresentados em um display de 7-segmentos, sendo assim, foi desenvolvido um módulo decodificador BCD com o auxílio do software Logisim para criação da Tabela Verdade e para a simplificação das equações.

A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0
0	0	1	1	0	0	0	0	1	1	0
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	1	0	0	1	0	0
0	1	1	0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	1	1	0	0
1	0	1	0	0	0	0	1	0	0	0
1	0	1	1	1	1	0	0	0	0	0
1	1	0	0	1	1	1	0	0	1	0
1	1	0	1	1	0	0	0	0	1	0
1	1	1	0	0	1	1	0	0	0	0
1	1	1	1	0	1	1	1	0	0	0

Figura 1: Tabela Verdade para o decodificador BCD

Obs.: os displays de 7-segmentos da FPGA EP2C35F672C6, possuem a configuração de ânodo comum, sendo assim, para acender os LEDs é necessário a aplicação do sinal 0, e para apagá-los, o sinal 1.

A partir da Tabela Verdade apresentada acima, foi obtida as seguintes equações para cada segmento do display:

```
a = (~D && ~C && ~B && A) || (C && ~B && ~A) || (D && ~C && B && A) || (D && C && ~B);
b = (~D && C && ~B && A) || (C && B && ~A) || (D && B && A) || (D && C && ~A);
c = (~D && ~C && B && ~A) || (D && C && ~A) || (D && C && B);
d = (~C && ~B && A) || (~D && C && ~B && ~A) || (C && B && A) || (D && ~C && B && ~A);
e = (~D && A) || (~C && ~B && A) || (~D && C && ~B);
f = (~D && ~C && A) || (~D && ~C && B) || (~D && B && A) || (D && C && ~B);
g = (~D && ~C && ~B) || (~D && C && B && A);
```

Figura 2: Equações simplificadas para cada segmento do display

Com o módulo decodificador finalizado, foi dado prosseguimento a criação da memória RAM utilizando a biblioteca LPM e seu respectivo arquivo de inicialização. Para tal, foi seguido o tutorial fornecido em sala e obtivemos o seguinte resultado:

```
module ramlpm (
    input [4:0] address,
    input clock,
    input [7:0] data,
    input wren,
    output [7:0] q
);
```

Figura 3: Assinatura do módulo da memória RAM

Addr	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10	+11	+12	+13	+14	+15	+16	ASCII
0	5	2	1	0	5	0	0	0	0	0	0	130	0	122	222	0	0	.....z...
17	0	0	23	111	0	6	0	12	11	0	120	0	123	0	0			...o.....x.{..

Figura 4: Exibição do arquivo MIF

5 Simulação

5.1 Parte I

Para simplificar o processo de simulação, o módulo de memória e decodificador foram testados separadamente, e seguem abaixo:

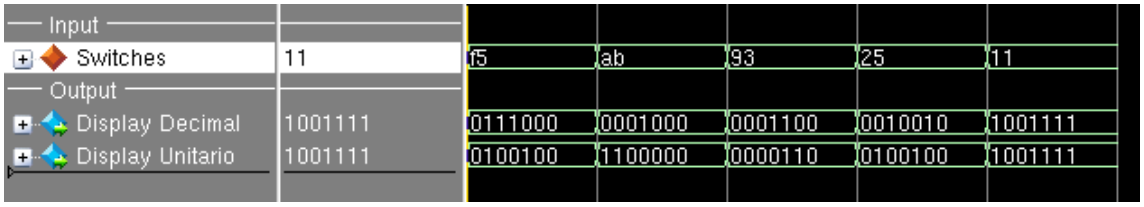

Figura 5: Simulação do Decodificador BCD

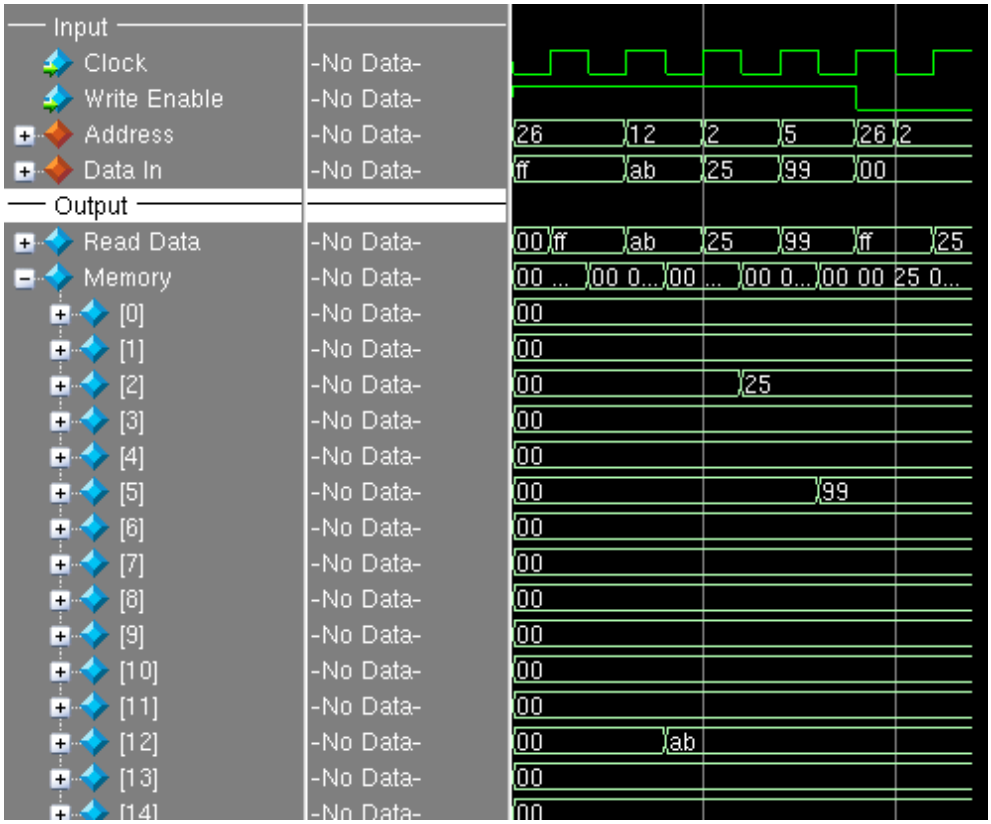

Figura 6: Simulação da Memória

Analisando a figura 6, é possível perceber que a implementação da memória RAM pela biblioteca LPM, realiza o processo de leitura na borda de subida do *Clock*, e a escrita na borda de descida. Esta implementação se assemelha com que acontece normalmente em processadores que aproveitam do Pipeline, como foi visto na disciplina de Arquitetura de Computadores I.

5.2 Parte II

Como dito anteriormente, a prática II consiste na inicialização da memória utilizando o arquivo MIF apresentado na figura 4.

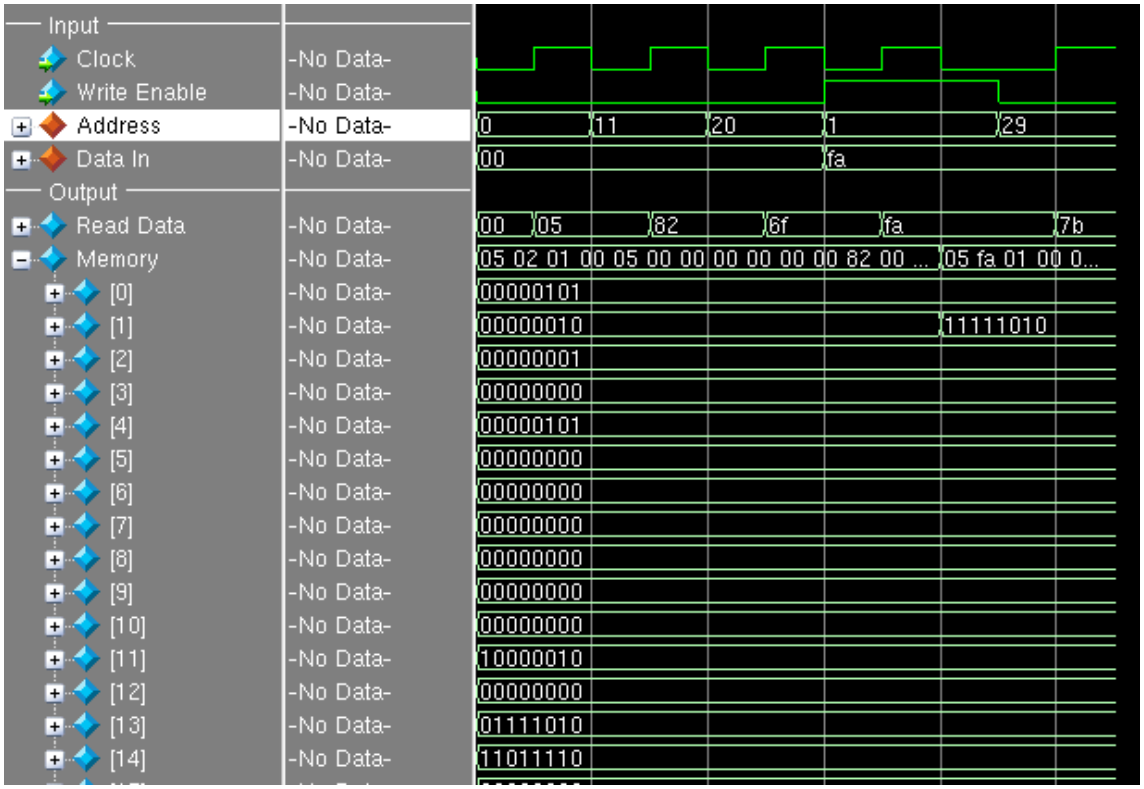

Figura 7: Simulação da Memória inicializada com o MIF

A simulação da figura 7, foi feita realizando a leitura das posições 11, 20 e 29, e a escrita na posição 1 no terceiro ciclo do *Clock*.

Obs.: como o módulo decodificador foi reaproveitado, sua respectiva simulação não foi refeita.

6 Conclusão

Com a execução desta prática, pudemos aperfeiçoar nossos conhecimentos nas bibliotecas do Quartus, na linguagem Verilog, além de assimilar melhor o conteúdo dado em sala de aula.