Universidade Federal de Viçosa
Campus Florestal
CCF 251 – Introdução aos Sistemas Lógicos Digitais
Prof. José Augusto Miranda Nacif

Trabalho Prático 01 - Circuitos Combinacionais - Implementação em FPGA

Gabriel Rodrigues Marques - 5097
Alan Araújo dos Reis - 5096
Rodrigo dos Santos Miranda - 5090
Edgar Alves de Jesus - 5087

09 de novembro de 2022

1. OBJETIVOS

Para essa segunda parte do Trabalho Prático, foi pedido a implementação do código desenvolvido em Verilog anteriormente em uma FPGA, um dispositivo lógico programável que suporta a implementação de circuitos digitais.

2. IMPLEMENTAÇÃO EM FPGA

2.1 DESENVOLVIMENTO

Primeiramente, foi preciso realizar a configuração do ambiente para trabalhar com a FPGA. Foi necessário realizar o *download* do *software* **Intel Quartus** e **System Builder**. Durante o processo de instalação, em Linux, vários problemas começaram a surgir já de início. Desse modo, migramos para o Windows para dar prosseguimento ao desenvolvimento.

Em seguida, em posse da **FPGA Cyclone IV Altera DE2-115** de modelo **EP4CE115F29C7**, disponibilizada pela universidade, demos início ao projeto. Foi utilizado o System Builder para gerar os arquivos de trabalho e o Intel Quartus para implementação e execução do código.

Com tudo configurado, o código foi compilado (Compile Design) e executado na FPGA (Program Device). A partir disso, com a análise do que estava ocorrendo na FPGA, foram feitas algumas modificações necessárias para o seu funcionamento correto.

2.2 MODIFICAÇÕES

Inicialmente, para facilitar o trabalho no Intel Quartus, juntamos o módulo ***echo_codficador.v*** e ***echo_display.v*** em um único módulo.

Em seguida, quando executado na FPGA, as entradas de ***ready*** e ***reset*** estavam funcionando normalmente. Entretanto, por algum motivo que não identificamos, a entrada para o número binário de 4 bits estava invertida, lia-se os bits ***DCAB*** primeiro, ao invés de ***ABCD*** como esperado. Desse modo, a forma como a atribuição era feita foi invertida, o que solucionou o problema, funcionando conforme o esperado.

```
echo_codificador codificador(.RE(SW[0]),
                             .RS(SW[1]),
                             .A(SW[4]),
                             .B(SW[5]),
                             .C(SW[6]),
                             .D(SW[7]),
                             .S1(LEDR[4]),
                             .S2(LEDR[3]),
                             .S3(LEDR[2]),
                             .S4(LEDR[1]),
                             .S5(LEDR[0])
                            );
```

```
echo_codificador codificador(.RE(SW[0]),
                             .RS(SW[1]),
                             .A(SW[7]),
                             .B(SW[6]),
                             .C(SW[5]),
                             .D(SW[4]),
                             .S1(LEDR[4]),
                             .S2(LEDR[3]),
                             .S3(LEDR[2]),
                             .S4(LEDR[1]),
                             .S5(LEDR[0])
                            );
```

Alteração de atribuição *echo_codificador* no arquivo DE2_115.v

Um outro empecilho deu-se na parte do *display*. Após solucionar o problema da entrada, reparamos que os números não estavam aparecendo no visor, pelo menos não completamente. Os segmentos que deveriam ser acesos, estavam apagados e, os que deveriam estar apagados, estavam acesos. Deduzimos que, talvez, os segmentos a serem acesos do *display* seriam aqueles que estivessem recebendo um nível lógico baixo e não alto. Com isso em mente, as saídas para o *display*, ***abcdefg***, foram negadas e obtivemos o resultado desejado.

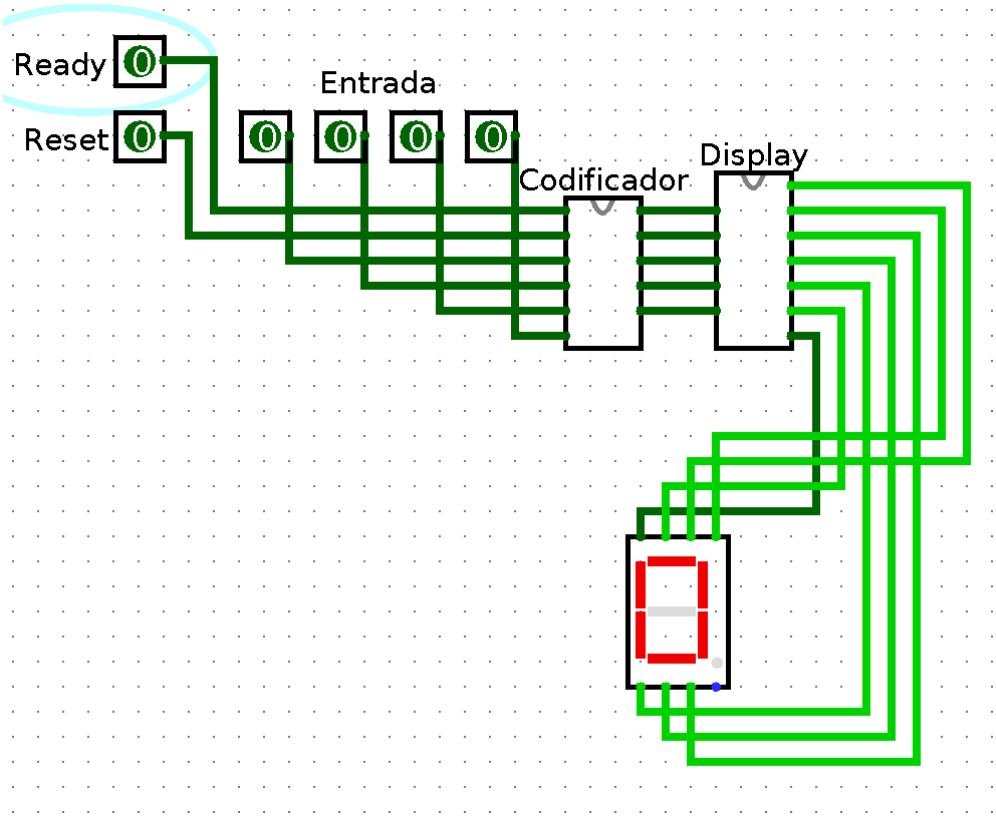
```
assign a = (~(~S1 | S5 | (S2 & ~S4)));
assign b = (~(~S2 | ~S5));
assign c = (~(~S2 | S3));
assign d = (~((~S1 & ~S3) | (S2 & ~S4) | (S2 & S5)));
assign e = (~((~S1 & ~S5) | (~S3 & S4) | (~S1 & S2) | (S2 & ~S3)));
assign f = (~((~S1 & ~S3) | (S2 & S4)));
assign g = (~((~S3 & S5) | S2));
```

Alteração das operações lógicas do *echo_display* no arquivo Verilog1.v

3. RESULTADOS

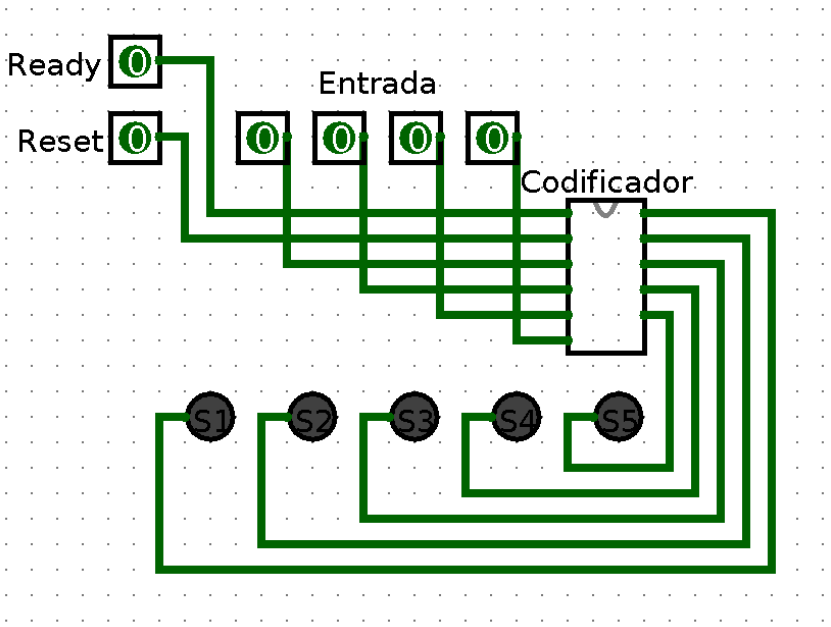
3.1 RESULTADO IDEALIZADO

Inicialmente, a implementação só mostraria no *display* o número em decimal do binário codificado, como mostrado no *gif* abaixo.



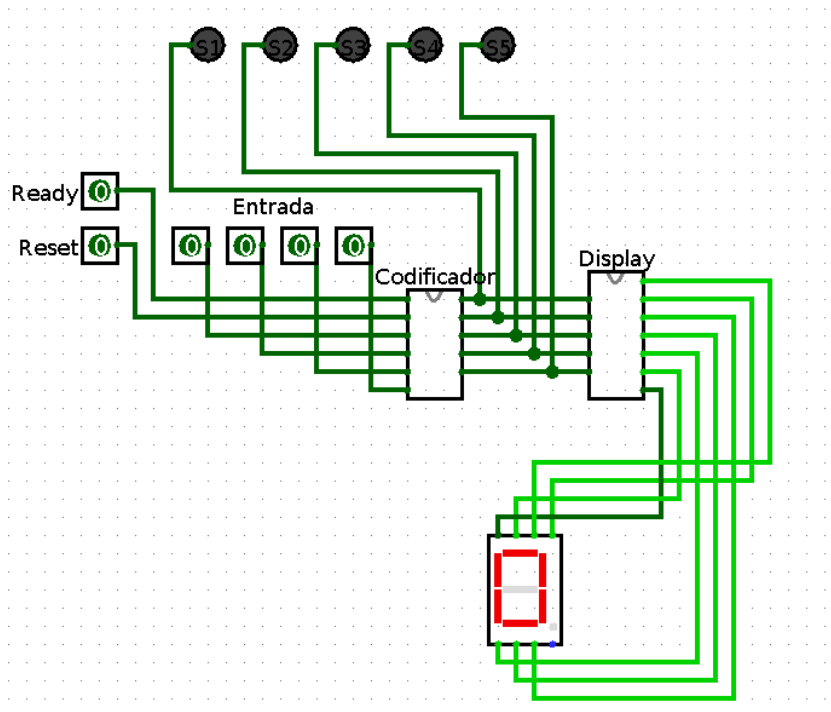
Circuito com Display - gif_circuitos_display.gif

Posteriormente, por orientação do monitor, foi solicitado que a implementação mostrasse também a cadeia de bits do número binário codificado, como mostrado no *gif* abaixo.



Circuito com LED's - gif_circuitos_led.gif

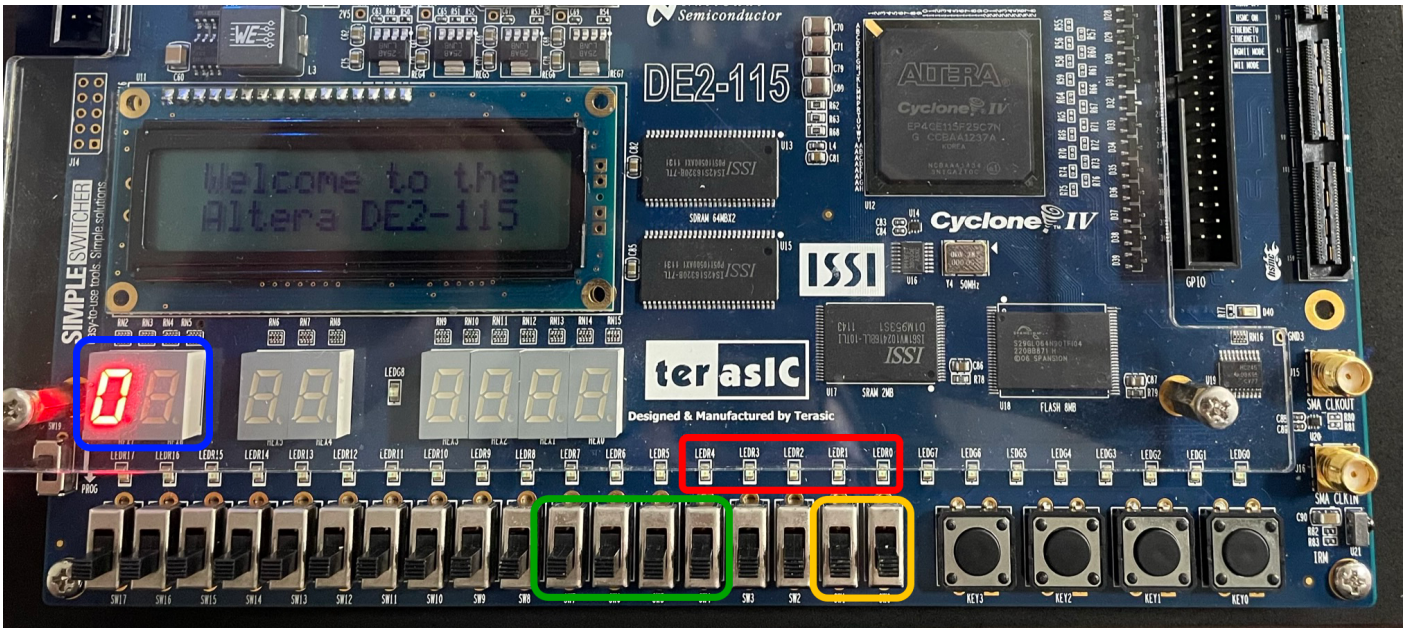
Ao final, juntamos as duas ideias que foram idealizadas em uma única solução. Dessa forma, temos os 5 LED's que identificam a cadeia de bits do número binário codificado e um *display* que exibe o número em decimal do binário codificado, como é possível ver no *gif* abaixo.



Circuito Completo - gif_circuito_completo.gif

3.2 RESULTADO EM FPGA

Na FPGA tem-se um resultado exatamente igual ao que foi mostrado inicialmente nas simulações. No retângulo em amarelo, tem-se o *ready e o reset*, em verde a entrada para número binário de 4 bits, em vermelho a saída da cadeia de bits do número binário codificado, que é mostrado por meio de 5 LED's. E por fim, em azul, o *display* de 7 segmentos que exibe o número em decimal do binário codificado.



Observações: mais detalhes do funcionamento da FPGA podem ser vistos no vídeo que foi enviado junto com esta documentação.

4. CONCLUSÃO

O desenvolvimento do Trabalho Prático, principalmente dessa segunda parte, foi bastante interessante, aumentando nossa proximidade e afinidade com a área de *hardware*. Por fim, com a utilização de ferramentas, como System Builder e Intel Quartus, juntamente com a FPGA Cyclone IV Altera DE2-115, obtivemos um resultado, que ao nosso ver, atende às especificações solicitadas anteriormente.

5. REFERÊNCIAS

[1] R. Katz, G. Borriello, Contemporary Logic Design, 2ª edição, Prentice Hall, 2004.

[2] TANENBAUM, A.S. Organização Estruturada de Computadores. 5. ed. Editora Pearson Prentice Hall, 2007.;

[3] Github. Disponível em: </https://github.com/>;

[4] Stack Overflow. Disponível em <https://stackoverflow.com/>;

[5] Geeks for Geeks. Disponível em <https://www.geeksforgeeks.org/>;

[6] How to Program Your First FPGA Device / Intel. Disponível em <How to Program Your First FPGA Device>

[7] Terasic Inc.. Disponível em <https://www.terasic.com.tw/en/>

[8] Icarus Verilog;

[9] GTKWave;

[10] Logisim;

[11] Visual Studio Code;

[12] Intel Quartus Prime;