Centro de Informática
Universidade Federal da Paraíba

Relatório Final

Relatório - CI SmartLift 2.0

Lucas Moreira e Silva Alves - 20170027336
Renan Almeida Goes Vieira de Melo - 20170021539
Sanny Alves de Sousa - 11324088

João Pessoa, 2018

Centro de Informática
Universidade Federal da Paraíba

Relatório - CI SmartLift 2.0

Relatório elaborado para o projeto final da disciplina Circuito Lógicos II, ministrada pelo Professor Eudisley Gomes dos Anjos do Centro de Informática da Universidade Federal da Paraíba.

João Pessoa, 2018

Resumo

O presente relatório condiz com às metodologias utilizadas para a realização do projeto SmartLift 2.0, o qual encontrou-se compreendido com uma modelagem de máquina de estados elaborado para o mesmo. Foi utilizado um display de 7 segmentos para expor seus movimentos, e para esse evento foi lidado também com um LCD, LEDs e algumas chaves, os quais fazem parte da composição do kit utilizado para a realização do projeto, sendo ele: kit DE2-115 da Altera.

Quanto aos estados, o próprio possui três, são eles: parado, subir e descer, em que o usuário teve a possibilidade de escolher uma das chaves, entre SW0 e SW8, onde as mesmas representaram um andar, e posteriormente ao apertar um determinado botão sua escolha era autenticada, fazendo com que o elevador pudesse realizar o movimento solicitado. Ao passo que no display apresentava o andar pelo qual está passando e ao lado o andar pretendido.

Palavras-chave: Verilog, Elevador, FPGA, Chaves, LCD, LEDs e Botão.

Lista de siglas

VHDL - Hardware Description Language (Linguagem de descrição de hardware);

FPGA - Field Programmable Gate Array (Arranjo de portas programáveis em campo);

IEEE - O Instituto de Engenheiros Eletricistas e Eletrônicos;

IDE - Integrated Development Environment (Ambiente de desenvolvimento integrado).

Sumário

1. Introdução

Este relatório tem como objetivo a explanação de etapas para a construção de um SmartLift, em alto nível, bem como seus resultados ao longo de sua elaboração. Onde o mesmo funcione de acordo com a máquina de estados criada. Para seu funcionamento foi utilizado um display de 7 segmentos simulando sua movimentação, seu processo foi feito valendo-se da linguagem Verilog HDL e kits de FPGA, o qual: denominado DE2-115, sendo o mesmo modelo da Altera.

A linguagem empregada foi criada em 1985 pela Gateway Design Automation, essa empresa foi comprada em 1990 e posteriormente passou-se para domínio público, e sua regulamentação é realizada pelo IEEE. Ela possui certa aparência condizente com outras linguagens, tais como: C e C++. A placa utilizada foi criada em 1984, criada por Ross Freeman. Diz respeito a um circuito integrado baseado em matriz de blocos lógicos e de conexões programáveis.

O referido projeto sucedeu-se por intermédio de assuntos abordados em sala de aula ao longo do período que se faz presente, fazendo o uso da placa, já mencionada, juntamente com Verilog VHDL. Sua efetiva realização foi composta por um grupo constituído por três graduandos de Engenharia de Computação. O próprio foi projetado partindo dos procedimentos acordados para seu funcionamento, os quais: deveria possuir os estados - parado, partida, subindo e descendo. Buscou-se a utilização de variadas ferramentas disponíveis no kit de hardware, tais como: push-buttons, LEDs, LCDs, Switches, como também exercitou-se a implementação de máquina de estados, fixando assim os assuntos dados em sala.

Por fim, o elevador teve sua eficiência, no que diz respeito ao requerido do próprio, sendo ele capaz de ser movido pelos andares, de acordo com a escolha do usuário, enquanto que o display apresentava o atual andar e ao lado o andar pretendido\selecionado.

2. Metodologia

Para a composição do circuito que fará os movimentos estabelecidos do elevador, foi primeiramente realizado a criação da máquina de estados da solução **figura-1**. Programação do respectivo código em Verilog bem como o detalhamento dos componentes do programa, como também todas as ações realizadas em todo o trajeto do presente projeto.

figura-1

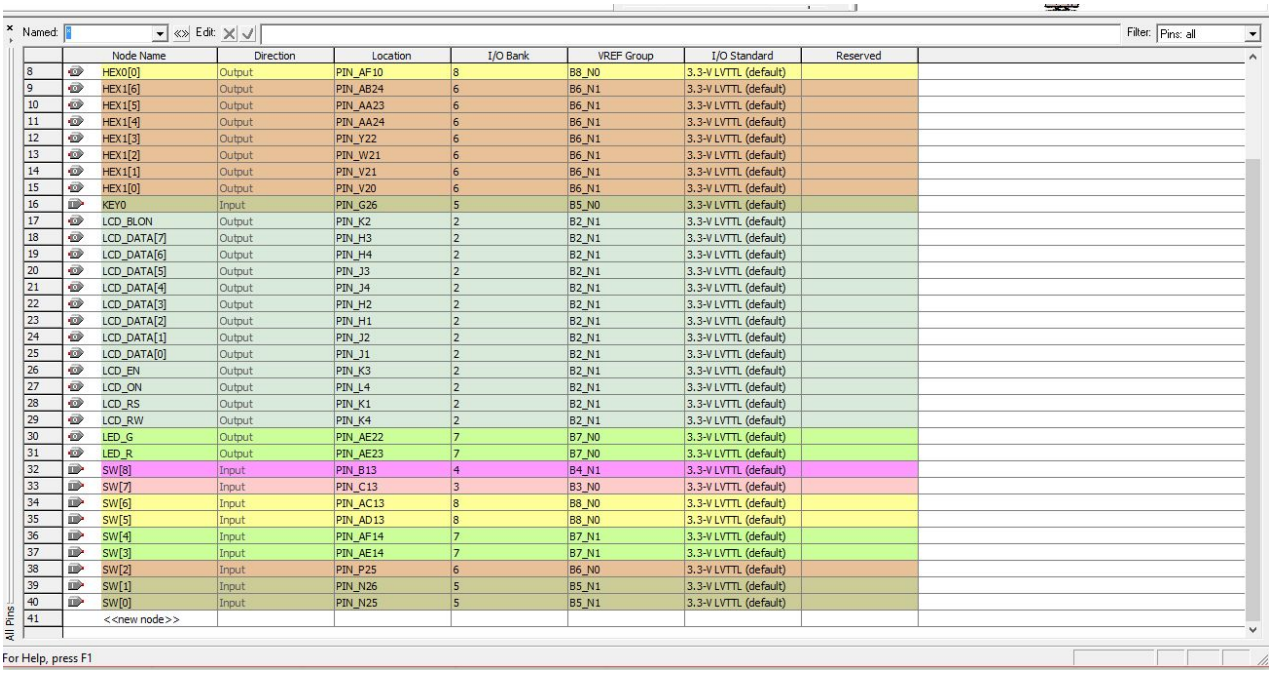
Tabela de transição de estados:

	E N T R A D A S				**SAÍDAS**
Estado	**S = 1, A=0**	**S = 1, A = 1**	**S = 1**	**S =0**	
Parado	**Descendo**	**Subindo**	**Parado**	**Parado**	**0**
Descendo	**Descendo**	**Descendo**	**Descendo**	**Parado**	**1**
Subindo	**Subindo**	**Subindo**	**Parado**	**Subindo**	**1**

tabela-1

Foi necessário a utilização de oito interruptores (ou chaves), um botão de pressão, que foi empregado no momento da escolha do andar, por esse motivo, o elevador faz seu movimento após pressionado o referido botão; o LCD para expor o movimento (subindo +, descendo -) do elevador; dois LEDs, que indica respectivamente quando o elevador se encontra fechado (vermelho) e aberto (verde); um display de sete segmentos que mostra qual o andar solicitado e o andar presente, em nosso projeto foi utilizado apenas dois; e também a criação de um módulo que realizasse o debouncing dos botões, por conta dos ruídos dos mesmos nas mudanças de estados.

A plataforma de desenvolvimento utilizada, integrada à um LCD de duas linhas, cada uma com dezesseis caracteres visíveis e memória para um total de quarenta caracteres por linha. A interface entre o FPGA e o LCD é constituída por quatro linhas de dados, três linhas de controle e duas de alimentação. A **figura-2** e **figura-3** revelam os pinos que foram setados dos elementos que foram utilizados.

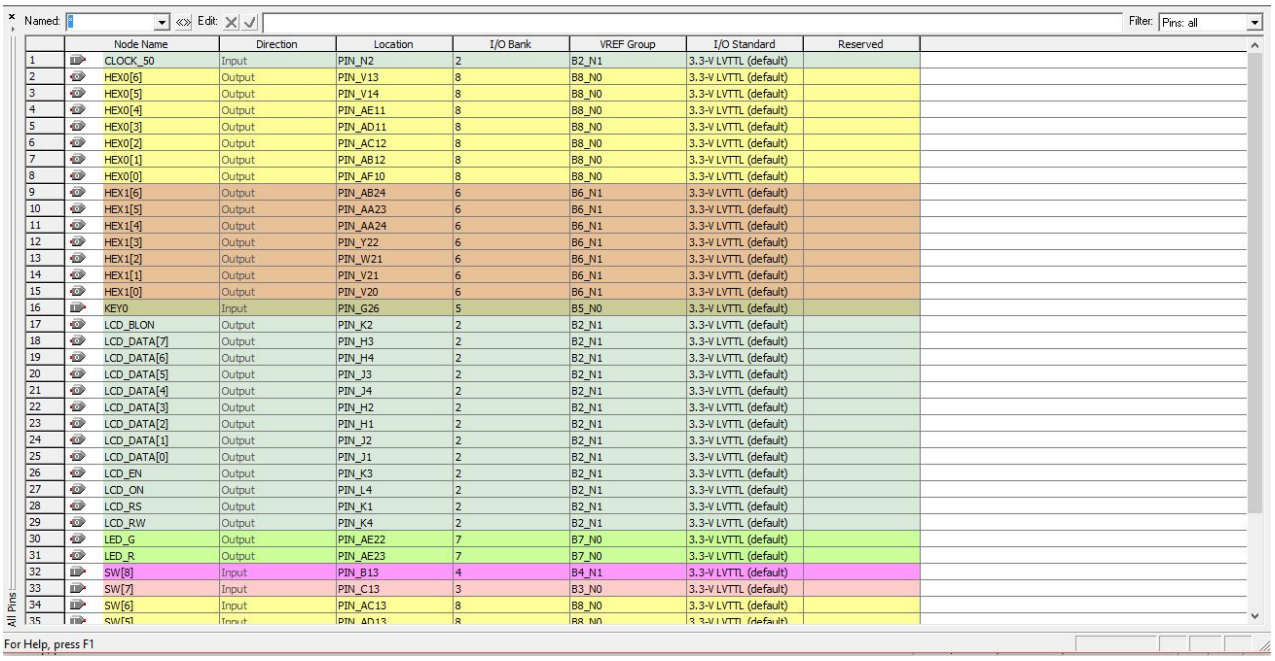

Named: Edit: Filter: Pins: all

	Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
8	HEX0[0]	Output	PIN_AF10	8	B8_N0	3.3-V LVTTL (default)	
9	HEX1[6]	Output	PIN_AB24	6	B6_N1	3.3-V LVTTL (default)	
10	HEX1[5]	Output	PIN_AA23	6	B6_N1	3.3-V LVTTL (default)	
11	HEX1[4]	Output	PIN_AA24	6	B6_N1	3.3-V LVTTL (default)	
12	HEX1[3]	Output	PIN_Y22	6	B6_N1	3.3-V LVTTL (default)	
13	HEX1[2]	Output	PIN_W21	6	B6_N1	3.3-V LVTTL (default)	
14	HEX1[1]	Output	PIN_V21	6	B6_N1	3.3-V LVTTL (default)	
15	HEX1[0]	Output	PIN_V20	6	B6_N1	3.3-V LVTTL (default)	
16	KEY0	Input	PIN_G26	5	B5_N0	3.3-V LVTTL (default)	
17	LCD_BLON	Output	PIN_K2	2	B2_N1	3.3-V LVTTL (default)	
18	LCD_DATA[7]	Output	PIN_H3	2	B2_N1	3.3-V LVTTL (default)	
19	LCD_DATA[6]	Output	PIN_H4	2	B2_N1	3.3-V LVTTL (default)	
20	LCD_DATA[5]	Output	PIN_J3	2	B2_N1	3.3-V LVTTL (default)	
21	LCD_DATA[4]	Output	PIN_J4	2	B2_N1	3.3-V LVTTL (default)	
22	LCD_DATA[3]	Output	PIN_H2	2	B2_N1	3.3-V LVTTL (default)	
23	LCD_DATA[2]	Output	PIN_H1	2	B2_N1	3.3-V LVTTL (default)	
24	LCD_DATA[1]	Output	PIN_J2	2	B2_N1	3.3-V LVTTL (default)	
25	LCD_DATA[0]	Output	PIN_J1	2	B2_N1	3.3-V LVTTL (default)	
26	LCD_EN	Output	PIN_K3	2	B2_N1	3.3-V LVTTL (default)	
27	LCD_ON	Output	PIN_L4	2	B2_N1	3.3-V LVTTL (default)	
28	LCD_RS	Output	PIN_K1	2	B2_N1	3.3-V LVTTL (default)	
29	LCD_RW	Output	PIN_K4	2	B2_N1	3.3-V LVTTL (default)	
30	LED_G	Output	PIN_AE22	7	B7_N0	3.3-V LVTTL (default)	
31	LED_R	Output	PIN_AE23	7	B7_N0	3.3-V LVTTL (default)	
32	SW[8]	Input	PIN_B13	4	B4_N1	3.3-V LVTTL (default)	
33	SW[7]	Input	PIN_C13	3	B3_N0	3.3-V LVTTL (default)	
34	SW[6]	Input	PIN_AC13	8	B8_N0	3.3-V LVTTL (default)	
35	SW[5]	Input	PIN_AD13	8	B8_N0	3.3-V LVTTL (default)	
36	SW[4]	Input	PIN_AF14	7	B7_N1	3.3-V LVTTL (default)	
37	SW[3]	Input	PIN_AE14	7	B7_N1	3.3-V LVTTL (default)	
38	SW[2]	Input	PIN_P25	6	B6_N0	3.3-V LVTTL (default)	
39	SW[1]	Input	PIN_N26	5	B5_N1	3.3-V LVTTL (default)	
40	SW[0]	Input	PIN_N25	5	B5_N1	3.3-V LVTTL (default)	
41	<<new node>>						

All Pins

For Help, press F1

figura-2



Named: Edit: Filter: Pins: all

	Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
1	CLOCK_50	Input	PIN_N2	2	B2_N1	3.3-V LVTTL (default)	
2	HEX0[6]	Output	PIN_V13	8	B8_N0	3.3-V LVTTL (default)	
3	HEX0[5]	Output	PIN_V14	8	B8_N0	3.3-V LVTTL (default)	
4	HEX0[4]	Output	PIN_AE11	8	B8_N0	3.3-V LVTTL (default)	
5	HEX0[3]	Output	PIN_AD11	8	B8_N0	3.3-V LVTTL (default)	
6	HEX0[2]	Output	PIN_AC12	8	B8_N0	3.3-V LVTTL (default)	
7	HEX0[1]	Output	PIN_AB12	8	B8_N0	3.3-V LVTTL (default)	
8	HEX0[0]	Output	PIN_AF10	8	B8_N0	3.3-V LVTTL (default)	
9	HEX1[6]	Output	PIN_AB24	6	B6_N1	3.3-V LVTTL (default)	
10	HEX1[5]	Output	PIN_AA23	6	B6_N1	3.3-V LVTTL (default)	
11	HEX1[4]	Output	PIN_AA24	6	B6_N1	3.3-V LVTTL (default)	
12	HEX1[3]	Output	PIN_Y22	6	B6_N1	3.3-V LVTTL (default)	
13	HEX1[2]	Output	PIN_W21	6	B6_N1	3.3-V LVTTL (default)	
14	HEX1[1]	Output	PIN_V21	6	B6_N1	3.3-V LVTTL (default)	
15	HEX1[0]	Output	PIN_V20	6	B6_N1	3.3-V LVTTL (default)	
16	KEY0	Input	PIN_G26	5	B5_N0	3.3-V LVTTL (default)	
17	LCD_BLON	Output	PIN_K2	2	B2_N1	3.3-V LVTTL (default)	
18	LCD_DATA[7]	Output	PIN_H3	2	B2_N1	3.3-V LVTTL (default)	
19	LCD_DATA[6]	Output	PIN_H4	2	B2_N1	3.3-V LVTTL (default)	
20	LCD_DATA[5]	Output	PIN_J3	2	B2_N1	3.3-V LVTTL (default)	
21	LCD_DATA[4]	Output	PIN_J4	2	B2_N1	3.3-V LVTTL (default)	
22	LCD_DATA[3]	Output	PIN_H2	2	B2_N1	3.3-V LVTTL (default)	
23	LCD_DATA[2]	Output	PIN_H1	2	B2_N1	3.3-V LVTTL (default)	
24	LCD_DATA[1]	Output	PIN_J2	2	B2_N1	3.3-V LVTTL (default)	
25	LCD_DATA[0]	Output	PIN_J1	2	B2_N1	3.3-V LVTTL (default)	
26	LCD_EN	Output	PIN_K3	2	B2_N1	3.3-V LVTTL (default)	
27	LCD_ON	Output	PIN_L4	2	B2_N1	3.3-V LVTTL (default)	
28	LCD_RS	Output	PIN_K1	2	B2_N1	3.3-V LVTTL (default)	
29	LCD_RW	Output	PIN_K4	2	B2_N1	3.3-V LVTTL (default)	
30	LED_G	Output	PIN_AE22	7	B7_N0	3.3-V LVTTL (default)	
31	LED_R	Output	PIN_AE23	7	B7_N0	3.3-V LVTTL (default)	
32	SW[8]	Input	PIN_B13	4	B4_N1	3.3-V LVTTL (default)	
33	SW[7]	Input	PIN_C13	3	B3_N0	3.3-V LVTTL (default)	
34	SW[6]	Input	PIN_AC13	8	B8_N0	3.3-V LVTTL (default)	
35	SW[5]	Input	PIN_AD13	8	B8_N0	3.3-V LVTTL (default)	

All Pins

For Help, press F1

figura-3

Após a realização da pinagem, foi desabilitado os demais pinos em tri-stage. Com a utilização da linguagem de hardware foi efetuado a conversão dos módulos para máquinas de estados finitos, os quais fazem o controle do andamento do elevador.

Para efetivação da compilação dos códigos em verilog foi utilizada a IDE quartus II 9.1 web edition, esse software permite que seja programado um design de dispositivos lógicos, produzidos pela Altera, através dele nos é permitido a realização e sínteses de linguagens HDL, como o verilog além de examinar diagramas RTL e fazer a simulação da reação de dispositivos em diferentes estímulos, tal como pulsos de clock. Foi utilizada a versão 9.1 por haver suporte á família da placa usada.

A efetivação do projeto teve a duração de quatro dias, para sua implementação contando com a assistência efetiva do monitor da disciplina.

3. Descrição do Projeto

A Lógica de funcionamento do elevador obedece uma máquina de estados que possui três estados, que pode ser visualizado na **figura-1** e na **tabela-1** pode-se observar as transições dos estados.

São eles:

1. **Partida**: Reseta o sistema e volta para o estado zero, andar 0 (terreo).

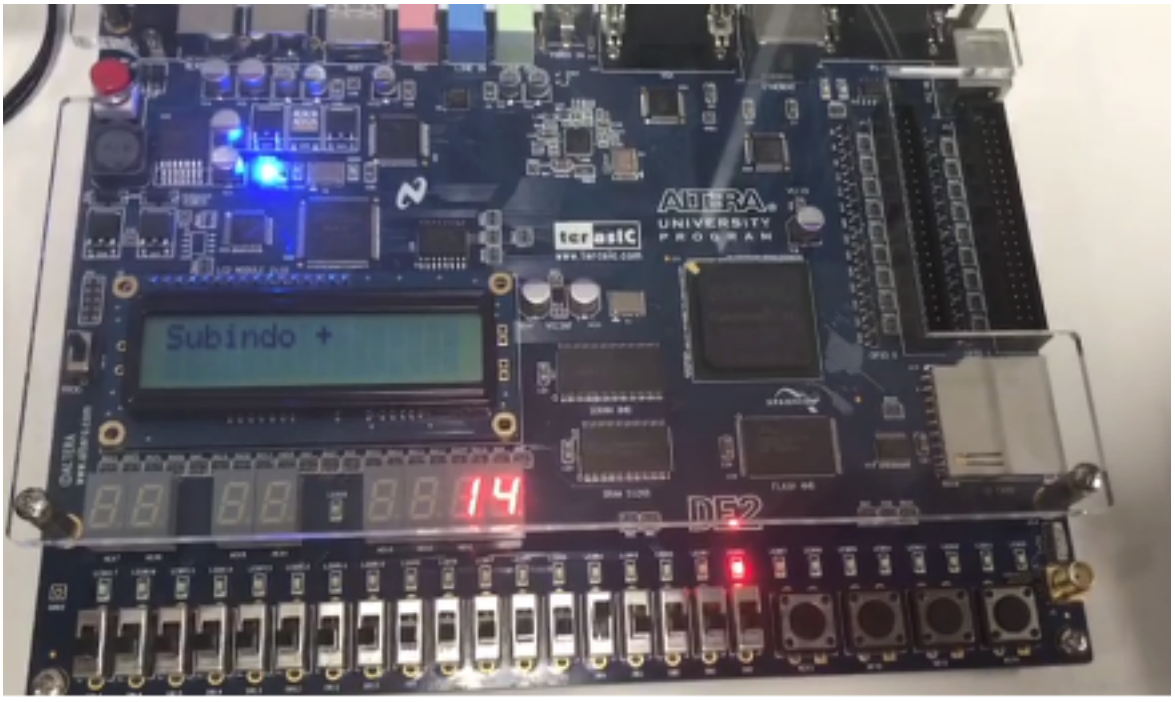
2. **Parado**: Quando o elevador, em movimento, chega a algum andar solicitado, ele para. Até que outro andar seja solicitado ou o tempo atingido, ele permanece parado no andar. Se um botão de andar é pressionado, a solicitação é registrada, e o elevador começa um movimento de subida ou de descida, em direção ao andar correspondente.

3. **Subindo e descendo**: Quando o elevador está subindo ou descendo, o elevador só poderá executar no máximo duas solicitações por vez, ou seja, só há memória para duas escolhas de andar. Após isso, não poderá chamar outros andares, acima ou abaixo de sua localização imediata. Ele permanecerá em seu movimento de subida, até o andar solicitado, em seguida ficará parado aguardando uma nova solicitação. As chaves SW0 até SW8 serão responsáveis pela seleção dos andares. Um display de 7 segmentos mostrará o andar escolhido pelas chaves. Outro display de 7 segmentos mostrará o movimento do elevador com os valores dos andares que ele está passando. Além disso, deverá ser mostrado no display LCD os estados que a máquina de estados estiver executando.

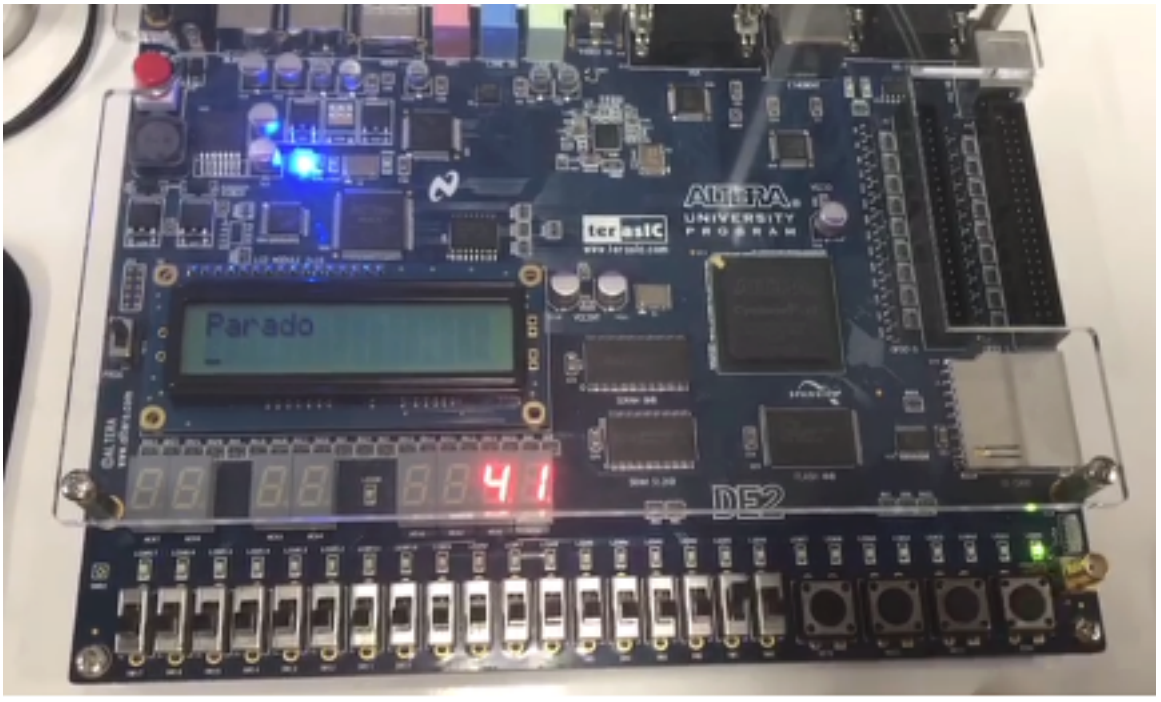
Dispositivos utilizados foram: Chaves, LEDs, Displays de 7 Segmentos, LCD.

4. Execução do Projeto, Testes e Resultados

O projeto foi feito com o kit de desenvolvimento DE2-115, o código foi feito na linguagem de descrição de hardware Verilog, os testes foram feitos no FPGA Cyclone II do tipo EP2C35F672C6, os testes foram feitos durante o projeto e, como pode ser visto na figura abaixo, o Smartlift mostra no seu LCD que está subindo, pois conforme apresentado nos displays, o andar atual é 1 (display esquerdo) e o andar que está indo é 4 (display da direita), o led vermelho está ligado, evidenciando que o Smartlift está em uso e suas portas não abrirão, quando parado, o led vermelho desliga e o verde deverá ligar.



Já na imagem abaixo, o Smartlift está parado, logo o led verde está aceso, mas por conta do delay, mesmo após o botão ter sido pressionado e o Smartlift chamado para o andar 1 o led e o LCD não alteraram (o tempo de alteração é de em torno de 1 segundo para o led e 2 segundos para o LCD).



O projeto funcionou conforme o planejado, todas as funções, incluindo os leds, display e LCD, estão funcionando como desejado, os estados alteram dependendo do andar escolhido assim como foi planejado, os andares dos displays alteram de acordo com a chave selecionado e mudam e param como devem e as chaves e o botão selecionam as opções corretas.

5. Conclusões

Por intermédio da realização do projeto aqui exposto, pode-se perceber uma visível evolução no que diz respeito aos assuntos abordados em sala de aula, visto que ao valer-nos da utilização prática do FPGA, nos foi levado a um maior reforço no entendimento do mesmo. Ademais, houve uma melhora significativa no que tange aos nossos conhecimentos de verilog e suas nuances.

6. Referências

BRAGA, R. J. G.; DAMBROS, T.; ZANETTI, V. A. (2010). *Projeto elevador (Equipamento para transporte de pessoas)*. Disponível em < http://embedded.microprocessadores.com.br/wp-content/uploads/2015/06/docs/user123_Documentacao-Elevador.pdf >. Acessado em 27 de novembro de 2017.

What is na FPGA?. Disponível em < https://www.xilinx.com/products/silicon-devices/fpga/what-is-an-fpga.html>. Acessado em 27 de novembro de 2017.