RELATÓRIO LABORATÓRIO 2: MÁQUINA DE ESTADOS FINITOS

Arthur Faria Campos 16/0024242

Programa de Engenharia Eletrônica
Faculdade Gama - Universidade de Brasília
St. Leste Projeção A - Gama Leste, Brasília - DF, 72444-240
email: arthur-fc@hotmail.com

Felipe Lima Alcântara 16/0027918

Programa de Engenharia Eletrônica
Faculdade Gama - Universidade de Brasília
St. Leste Projeção A - Gama Leste, Brasília - DF, 72444-240
email: lipelima0327@gmail.com

RESUMO

O documento apresenta o relatório técnico do primeiro experimento, da matéria Prática de Eletrônica Digital 2. Este experimento denominado "Máquina de Estados finitos". Esta atividade teve o objetivo de realizar implementações em VHDL de máquinas de estados finitos, para suprir as necessidades de cada exercício.

1. INTRODUÇÃO

A máquina de estados é um circuito sequencial, que combina um bloco de memória com blocos combinacionais. O estado representa uma situação onde o circuito se encontra ao longo do tempo, o estado então é uma combinação de entradas e saídas que recebe um nome.

Onde o bloco de memória terá o papel de armazenar os estados e os blocos combinacionais o papel de cálculo do próximo estado e das saídas do circuito. Porém a saída dos blocos combinacionais também dependerão do estado atual do circuito.

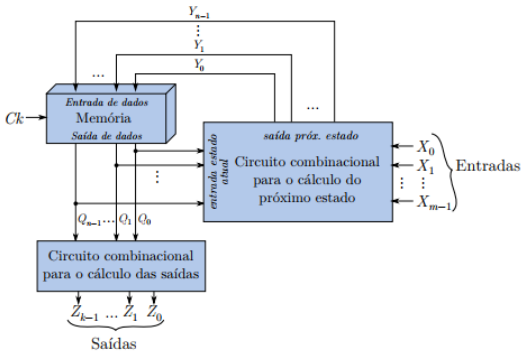


Figura 1: Modelo geral de uma máquina de estados com entrada de Clock.

Este modelo é geral, pois nem todas as máquinas de estados possuem variáveis de entradas e de saídas, entretanto todas têm variáveis de ativação e de estado.

Existem dois tipos de máquinas de estado a máquina de Moore e a de Mealy.

Máquina de Moore: A saída dependerá exclusivamente do estado atual e a entrada só interfere na transição de estado.

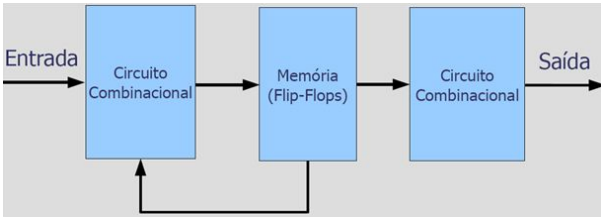


Figura 2: FSM de Moore

Máquina de Mealy: A saída depende do estado e da entrada, portanto a entrada interfere tanto na transição de estado quanto na saída.

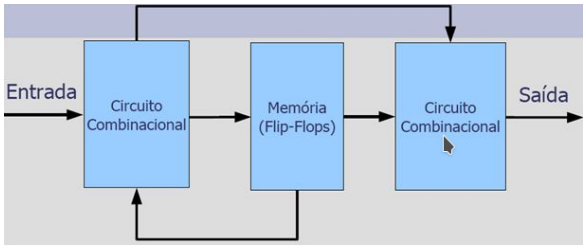


Figura 3: FSM Mealy

A máquina de Mealy, tem a possibilidade de se ter um número de estados menor que a de Moore. Porém ela é sensível às mudanças da entrando dentro do ciclo de clock, afetando a saída do circuito.

2. EXPERIMENTO

Para o Laboratório 2 tivemos que resolver 2 exercícios, o primeiro era projetar uma máquina de estados para um robô seguidor de linhas e o segundo projetar uma fechadura digital.

2.1. Exercício 01

Para realizar o projeto do robô seguidor de linhas foi necessário, 5 chaves de entrada e 6 leds de saída.

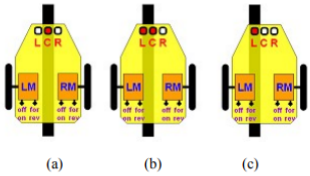
Duas chaves foram designadas para o controle da carga da bateria, denominado BL, uma chave para o sensor da linha na esquerda, denominado L, uma chave para o sensor da linha no centro, denominado C e, por fim, uma para o sensor da linha na direita, denominado R.

Para o sentido de movimento de cada motor foram necessárias duas leds, LM e RM, e para a representação da potência da bateria foi usada uma para cada motor no caso duas.

O funcionamento do robô é realizado da seguinte forma:

- Se o robô detecta a linha com algum sensor, a saída do sensor é '1'.
- Se as leds de movimento estiverem desligadas o motor estará parado, se "01" estarão girando a esquerda, se "10" a direita e "11" é inválido.
- A velocidade de movimento dos motores dependerá da saída dos sensores, se os sensores da esquerda ou da direita, estiverem ligados sozinhos o robô deverá fazer o giro rápido para o lado inverso, caso esses sensores estejam ligados junto ao sensor centro, o giro será suave.
- O controle de bateria é, se BL = "00", a potência do motor estará entre as porcentagens de (75% à 100%), caso BL= "01" a potência será entre (50% à 75%), caso BL= "10" será (25% à 50%) e por fim BL= "11" será entre (0% à 25%).
- A representação da velocidade dos motores dependerá do nível da bateria e será representado pelo ciclo ativo do PWM.

A figura a seguir resume o funcionamento do robô.



Funcionamento do robô seguidor de linha. (a) Segue em frente; (b) Gira suavemente à direita; (c) Gira rapidamente à direita.

Figura 4: Funcionamento Robô

Antes da implementação em VHDL, foi necessário fazer uma tabela para entender qual seria o movimento de giro do robô para cada LM e RM.

Entradas			Saida	Estado Motor			
L	C	R	LM	RM	MotorE	MotorD	Movimento
0	0	0	"00"	"00"	parado	parado	parado
0	0	1	"10"	"01"	esquerda	direita	gira esquerda rapido
0	1	0	"01"	"01"	direita	direita	frente
0	1	1	"01"	"00"	direita	parado	gira esquerda suave
1	0	0	"01"	"10"	direita	esquerda	gira direita rapido
1	0	1	N/A	N/A	N/A	N/A	N/A
1	1	0	"00"	"01"	parado	direita	gira direita suave
1	1	1	N/A	N/A	N/A	N/A	N/A

Tabela 1: Movimento Robô

Com essa tabela, agora foi só a implementação da máquina de Estados em VHDL. Inicialmente é necessário a criação dos estados, para isso é utilizado a função type onde será criado uma faixa de valores definida pelo usuário.

Para realizar a escolha dos estados, será necessário a utilização da sentença sequencial case-when que olhará o estado atual e o valor das entradas, definindo qual será o caminho a seguir.

O PWM utilizado foi o do laboratório anterior, sendo ele um component do circuito, porém anteriormente o DUTY e o TIMER eram entradas, nesta implementação o TIMER é fixo em "1111" e o DUTY um sinal que receberá o valor dependendo do valor da entrada BL.

2.2. Exercício 02

O exercício 02 é a implementação de uma FSM que controle a fechadura digital da Figura 5. Uma de suas características e opção de gravar uma senha de bloqueio para a fechadura.

Para isso foram utilizados as seguintes especificações de entrada e saida:

- (4) switches de ENTRADA para a senha.
- (1) switch para Ativar a Gravação;
- (2) pushbuttons (APLICA e Reset).
- Um LED de saída chamado ABRE que indica se a fechadura abre ('1' lógico) ou fecha ('0' lógico).
- Um LED de saída chamado ALARME que indica se a combinação da fechadura é incorreta.
- E um LED de saída para o Clock Auxiliar.

A utilização da Fechadura digital é bem simples, quando ingressada a combinação correta de três números nos switches de entrada, o sinal ABRE é acionado e a fechadura é aberta. Cada número da combinação é de 4 bits (números de 0 a 9).

O número é ingressado pressionando o botão APLICA ('1' lógico), que ira trocar o estado das FSM utilizadas.

E para gravar os três valores de uma nova combinação o switch GRAVAR deve ficar em '1' lógico, e em seguida cada valor da combinação é ingressado usando os switches e o botão APLICA. O botão Reset apaga os registradores que armazenam os três valores da combinação e define o Alarme =’0’ e Abre =’0’.

Um clock auxiliar com período de 4 segundos também é utilizado para que se possa manipular os switches de ENTRADA e o pushbutton APLICA antes de cada borda de subida do sinal de clock.

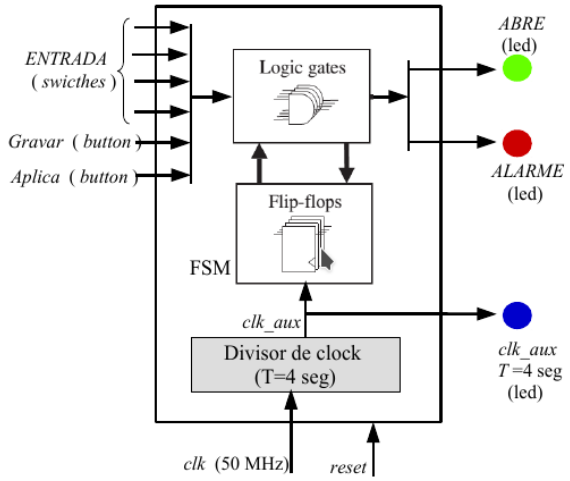


Figura 5: Fechadura Digital

O funcionamento da Fechadura digital é feito por 2 máquinas de estados (FSM). Uma das é responsável pela verificação da senha Figura 6, enquanto a outra e responsável pela gravação da Senha Figura 7. Também temos um Divisor de Clock, o mesmo utilizado nos experimentos anteriores.

As máquinas de estados são ativadas pelo Switch GRAVAR, por isso enquanto uma está ativa a outra está desativada e presa em seu estado inicial.

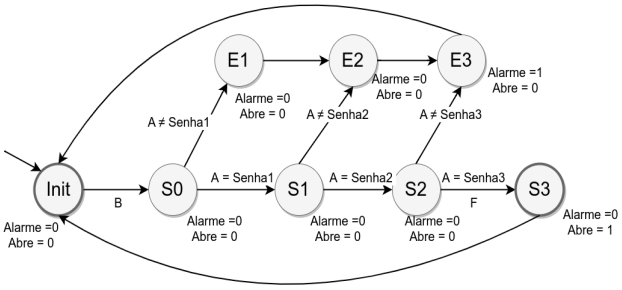


Figura 6: FSM de Verificação

Assim, para dificultar a descoberta da sequência correta, o alarme somente será acionado após a inserção de uma sequência incorreta completa, ou seja, quando pelo menos um dos três valores da combinação está errado.

Por isso se alguma senha for errada eles irão para os estados e erro e mesmo que acertem depois a sequência permanecerão no estado de erro. Sua codificação também só depende do estado atual, classificando então como uma máquina de Moore.

A FSM de Gravação (Figura 7) e ativada com o swtich GRAVAR =’1’, sendo ela responsável pela gravação das senhas nos registradores. Mas deve se tomar cuidado no código pra se evitar Latchs.

Esse problema foi resolvido utilizando uma condição de rising_edge antes da verificação dos estados, forçando a utilização de FFs ao invés do Latchs. Isso pode ser observado na linha 37 do codigo da Figura 8.

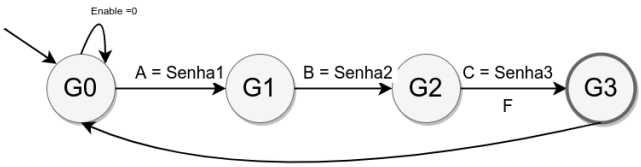


Figura 7: FSM de Gravação

```
34    Transicao : process(Estado_Atual,Aplica,Switch,Enable,Clk)
35        begin
36            if rising_edge(Clk) then
37                case Estado_Atual is
38                    when G0 =>
39                            if (Enable ='0') then
40                                Proximo_Estado <= G0;
41                            else
42                                Proximo_Estado <= G1;
43                            end if;
44                    when G1 =>
45                            if (Aplica ='1') then
46                                Password(11 downto 8) <= Switch;
47                                Proximo_Estado <= G2;
48                            else
49                                Proximo_Estado <= G1;
50                            end if;
51                    when G2 =>
52                            if (Aplica ='1') then
53                                Password(7 downto 4) <= Switch;
54                                Proximo_Estado <= G3;
55                            else
56                                Proximo_Estado <= G2;
57                            end if;
58                    when G3 =>
59                            if (Aplica ='1') then
60                                Password(3 downto 0) <= Switch;
61                                Proximo_Estado <= G1;
62                            else
63                                Proximo_Estado <= G3;
64                            end if;
65                end case;
66            end if;
67    end process;
```

Figura 8: Código da FSM Gravação

3. RESULTADOS

Antes de ser implementado na FPGA, realizamos simulações por meio de Test Bench.

3.1. Simulações exercício 01

A simulação do Exercício 1 comprova o funcionamento correto do robô com a mudança das entradas dos sensores de linha e do nível de bateria.

A figura abaixo mostra as formas de onda com a entrada L sendo a única em nível lógico alto e sendo realizado uma contagem crescente em BL para demonstrar todos os níveis de bateria. Como L é a única entrada em nível lógico alto o giro do robô deve ser rápido.

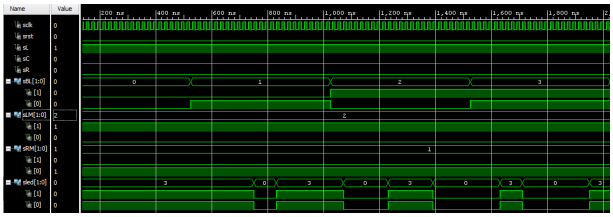


Figura 9: Simulação para LCR = "100"

Já a próxima figura mostra as formas de onda com a entrada R sendo a única e ocorrendo o mesmo processo em BL. Como já lembrado na figura anterior o giro do robô neste caso deve ser rápido.

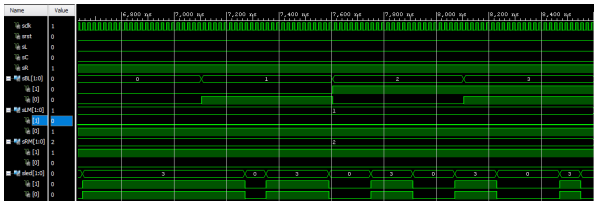


Figura 10: Simulação para LCR = "001"

Na figura 11, C e L estão em nível lógico alto, portanto o giro será suave à direita e é isso que é mostrado nas formas de onda desta figura.

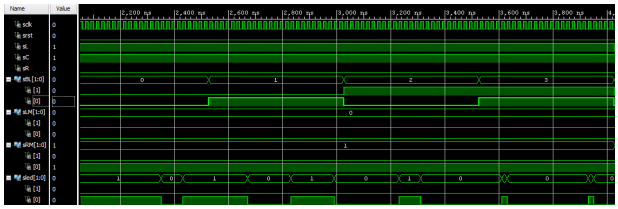


Figura 11: Simulação para LCR = "110"

Por fim, na figura 12, R e C estão em nível lógico alto, portanto o giro será suave à esquerda e é isso que é mostrado nas formas de onda desta figura.

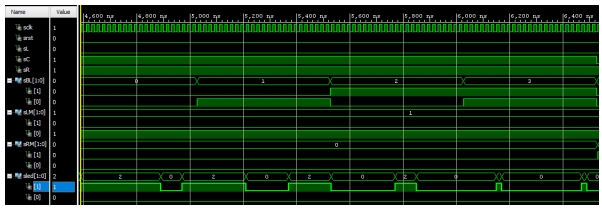


Figura 12: Simulação para LCR = "011"

Uma análise da Tabela abaixo permite inferir que muito pouco da capacidade da FPGA fora utilizada para implementar o controle do robô.

Resource	Utilization	Available	Utilization %
LUT	32	20800	0.15
FF	22	41600	0.05
IO	13	106	12.26
BUFG	1	32	3.13

Tabela 2: Componentes Utilizados no Exercício 1

3.2. Simulações Exercício 02

A simulação do Exercício 02 comprova o funcionamento correto das FSM com as mudanças de estados das Maquinas e a Senha Gravada.

Na simulação abaixo a máquina de estados mostra o funcionamento no caso de acerto da senha, também é possível analisar que o alarme é desativado com a abertura da chave.

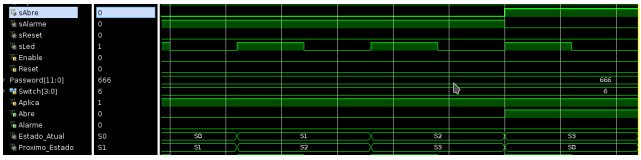


Figura 13: Simulação FSM Verificação com Abre

Já na simulação abaixo verificamos no caso de erro da sequência, a qual é ativado o alarme, e este permanece ligado ate a abertura da Chave, como na figura 13, ou caso o botão Reset seja pressionado.

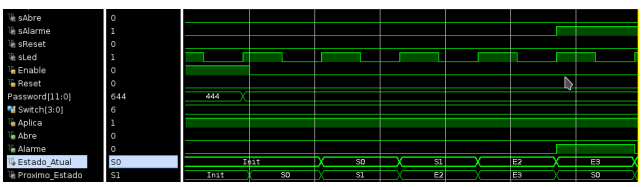


Figura 14: Simulação FSM Verificação com Alarme

A figura abaixo mostra os estados da máquina de estados de gravação, sendo que é ativada com o Enable = ‘1’, e a troca de estado é efetuada na subida do clock e o próximo estado já esta definido.

Neste caso a máquina de estados esta gravando a senha 444.

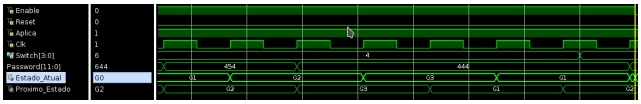


Figura 15: Simulação FSM Gravar

Com uma simples análise da Tabela abaixo percebemos que muito pouco da capacidade da FPGA fora utilizada para implementar a fechadura digital.

Resource	Utilization	Available	Utilization %
LUT	63	20800	0.30
FF	53	41600	0.13
IO	12	106	11.32
BUFG	1	32	3.13

Tabela 3: Componentes Utilizados no Exercício 02

4. DISCUSSÃO E CONCLUSÕES

A maior dificuldade do exercício 1 foi a compreensão do movimento do robô, a partir do giro do motor, após isso a implementação do projeto foi simples. A decisão de como será o giro rápido e o giro suave é muito importante, a forma mais simples de realizar essas alternâncias é manter o PWM de ambos os motores sempre iguais e alternar a velocidade com o uso de um ou dos dois motores.

Facilitando a lógica combinacional e a escrita no código, fizemos a implementação do reset, como forma de maior controle das operações do robô e para funcionar como um liga e desliga do mesmo.

Já no experimento 02, a maior dificuldade foi o controle de gravação de uma nova senha, pois caso os devidos cuidados não fossem tomados na escrita do código VHDL, o aparecimento de latchs é inevitável.

Concluímos o experimento implementando em VHDL lógicas sequências para suprir problemas reais, mostrando duas das n maneiras de se usar máquinas de estados para solucionar estes problemas.

5. REFERÊNCIAS

[1] Villanova, G. (2016). Uma arquitetura PWM em VHDL. [online] Embarcados. Disponivel em: https://www.embarcados.com.br/uma-arquitetura-pwm-em-vhdl/ [Acessado 7 Sep. 2017].

[2] Wakerly, John F., Digital Design: Principles and Practices, 4th ed., Prentice Hall, 2005.

[3] Chu, Pong P., FPGA Prototyping by VHDL Examples: Xilinx Spartan-3 Version, Wiley, 2011. [EBRARY]

[4] D’Amore, R., VHDL: Descrição e Síntese de Circuitos Digitais, 2 a . edição, LTC, 2012