UNIVERSIDADE FEDERAL DE VIÇOSA
CIÊNCIA DA COMPUTAÇÃO
CFC 251-Introdução aos Sistemas Lógicos Digitais

Segundo Trabalho Prático

Urna Eletrônica

Cláudio Barbosa – 3492
Filipe Fonseca – 3502
Guilherme Corrêa – 3509
Isabella Ramos - 3474

Professor: José Augusto Miranda Nacif
Monitor: Lucas Ferreira S. Duarte

FLORESTAL – MG
NOVEMBRO – 2018

1. INTRODUÇÃO

Uma Máquina de Estados Finitos (Finite State Machine) é um modelo matemático amplamente utilizado na representação do comportamento de sistemas computacionais e circuitos lógicos em geral. O modelo pode ser genericamente descrito como uma máquina abstrata que sempre deve estar em algum de seus finitos estados, um por vez. O estado no qual a máquina se encontra em dado instante é conhecido como estado atual. Todos os estados armazenam informações sobre o passado, isto é, as mudanças ocorridas desde o início do sistema até o presente momento.

Uma transição indica uma mudança de estado e é descrita através de uma condição que precisa ser realizada para que tal transição ocorra. Uma ação é a descrição de uma atividade que deve ser realizada em um determinado momento.

2. DESENVOLVIMENTO

2.1. Objetivos

O objetivo principal deste trabalho prático consiste no projeto e implementação de uma Máquina de Estados Finitos que simule o comportamento de uma Urna Eletrônica simplificada. A FSM deverá ser responsável por receber o número de um candidato e concluir se o voto é válido ou não, de acordo com as especificações abaixo:

1. Receber os números de 4 canditados, que equivalerão aos números de matricula dos integrantes do grupo: 3474,3492,3502 e 3509;
2. Contabilizar os votos individualmente
3. Qualquer voto destinado a um numero invalido é contabilizado como nulo;
4. O eleitor votara somente uma vez;
5. Os votos devem ser lidos, processados e contabilizados até que o sinal de controle *finish* seja ativado;

Para tal serão utilizados software de simulação e implementação, sendo eles: Icarus Verilog para elaboração e simulação dos módulos em *Verilog*; *GTKWave* para visualização das formas de onda resultantes; *Jflap* para elaboração do diagrama de estados de transição; Kit de FPGA DE2-115 da Altera.

Sendo que ao final, seu design deverá ser capaz de receber um número, dígito por dígito, concluir se o número corresponde a um voto válido ou nulo, contabilizar a informação e apresentar os resultados finais.

.

2.2. Elaboração

Para a elaboração deste trabalho foi considerado o diagrama básico do modulo da urna apresentado na Figura 1 e o funcionamento de seus sinais descrito na Tabela 1.

Figura 1: Diagrama de entradas e saídas do módulo urna.

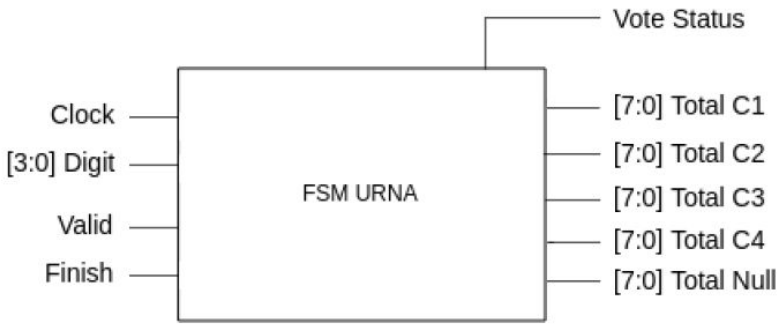
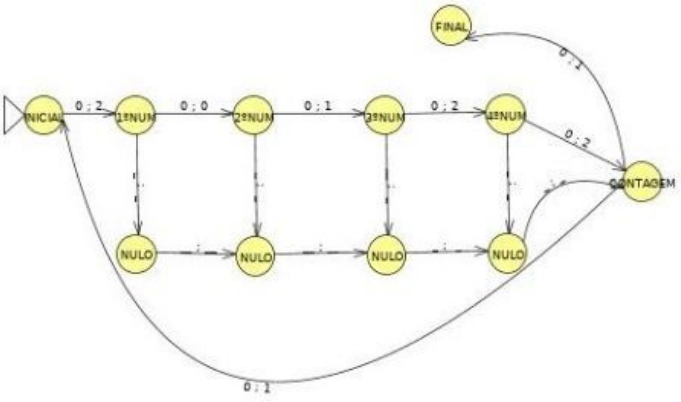


Tabela 1: Descrição dos sinais do módulo urna.

Nome	Tamanho	E/S	Descrição
Clock	1 bit	Entrada	Pulso de clock do sistema.
Digit	4 bits	Entrada	Dígito entre 0 e 9 que compõe o número do candidato desejado.
Valid	1 bit	Entrada	Sinal de controle que sincroniza a entrada de um dígito na máquina de estados. Deve ser acionado mediante a inserção de cada novo dígito.
Finish	1 bit	Entrada	Sinal de controle que finaliza a eleição, apresenta os votos contabilizados e então zera os contadores (reset).
Vote Status	1 bit	Saída	O valor 1 indica que o voto foi válido, enquanto 0 indica um voto nulo.
Total C1, C2, ..., Null	8 bits cada	Saídas	5 saídas que apresentam o total de votos para cada um dos 4 candidatos, bem como o total de votos nulos.

Com base nas instruções fornecidas, foi elaborada utilizando o *software JFlap* uma máquina de estados que permitisse a contabilização do voto de todos candidatos e a quantidade de votos nulos. Além de permitir que a contagem seja exibida ao final da eleição – ou seja, a chave referente ao *finish* receba o valor 1.

Figura 2: Máquina de estados referente a urna.



O código para implementação e simulação foi elaborado com a utilização do *software Atom* . Os comentários (em negrito) explicam o funcionamento e a lógica

aplicada ao código:

- Módulo para implementação:

```
module
urnaeletronica(control,digit,tisabella,tclaudio,tfilipe,tguilherme
,tnulo,clock,valid,finish,votestatus);
input wire clock,valid,finish;
input wire [3:0]digit;
input wire [2:0]control;
output reg [7:0]tisabella = 8'b00000000;
output reg [7:0]tclaudio = 8'b00000000;
output reg [7:0]tfilipe = 8'b00000000;
output reg [7:0]tguilherme = 8'b00000000;
output reg [7:0]tnulo = 8'b00000000;
output reg votestatus;
reg [3:0]estado = 4'b0000;
reg [7:0]isabella = 8'b00000000;
reg [7:0]claudio = 8'b00000000;
reg [7:0]filipe = 8'b00000000;
reg [7:0]guilherme = 8'b00000000;
reg [7:0]nulo = 8'b00000000;

     /* Isabella - 3474
          Cláudio - 3492
     Filipe - 3502
    Guilherme - 3509
*/

always @(posedge clock) begin
if(finish) begin
      case (control) //inicio case
            3'b000: begin //zera os resultados obtidos na eleicao
                  isabella <= 8'b00000000;
                  claudio <= 8'b00000000;
                  filipe <= 8'b00000000;
                  guilherme <= 8'b00000000;
                  nulo <= 8'b00000000;
                  tisabella <= 8'b00000000;
                  tclaudio <= 8'b00000000;
                  tfilipe <= 8'b00000000;
                  tguilherme <= 8'b00000000;
                  tnulo <= 8'b00000000;
            end
            3'b001: begin //atribui quantidade de votos finais
para isabella
                  tisabella <= isabella;
            end
```

```
            3'b010: begin //atribui quantidade de votos finais
para claudio
                  tclaudio <= claudio;
            end
            3'b011: begin //atribui quantidade de votos finais
para filipe
                  tfilipe <= filipe;
            end
            3'b100: begin //atribui quantidade de votos finais
para guilherme
                  tguilherme <= guilherme;
            end
            3'b101: begin //atribui quantidade de votos nulo
                  tnulo <= nulo;
            end
      endcase //fim case
end
      else if(~finish) begin
            case(estado) //inicio case
                  4'b0000: begin //primeiro estado
                        if((digit == 4'b0011) & (valid)) begin
//comparando 3 em binário
                              estado <= 4'b0001;
                        end
                        else begin //voto nulo
                              estado <= 4'b0101; //vai para o
segundo estado nulo
                        end
                  end
                  //fim primeiro estado

                  //inicio 2°numvoto
                  4'b0001: begin //segundo estado
                        if((digit == 4'b0100) & (valid)) begin
//comparando 4 em binário
                              estado <= 4'b0010;
                        end
                        else if((digit == 4'b0101) & (valid)) begin
//comparando 5 em binário
                              estado <= 4'b0010;
                        end
                        else begin //voto nulo
                              estado <= 4'b0110; //vai para o
terceiro estado nulo
                        end
                  end
                  //fim segundo estado

                  4'b0010: begin //terceiro estado
                        if((digit == 4'b1001) & (valid)) begin
//comparando 9 em binário
                              estado <= 4'b1000;
                        end
```

```
                        else if((digit == 4'b0111) & (valid)) begin
//comparando 7 em binário
                              estado <= 4'b0011;
                        end
                        else if((digit == 4'b0000) & (valid)) begin
//comparando 0 em binário
                              estado <= 4'b0011;
                        end
                        else begin //voto nulo
                              estado <= 4'b0100; //vai para o
quarto estado nulo
                        end
                  end
                  //fim terceiro estado

                  4'b1000: begin //quarto estado
                        if((digit == 4'b0010) & (valid)) begin
//comparando 2 em binário
                              claudio <= claudio + 8'b00000001;
//contagem de votos para claudio
                              votestatus <= 1;
                              estado <= 4'b0000;
                        end
                        else begin //voto nulo
                              nulo <= nulo + 8'b00000001;
//contagem de votos nulos
                              votestatus <= 0;
                              estado <= 4'b0000;
                        end
                  end
                  //fim quarto estado

                  4'b0011: begin //quarto estado
                        if((digit == 4'b1001) & (valid)) begin
//comparando 9 em binário
                              guilherme <= guilherme + 8'b00000001;
//contagem de votos para guilherme
                              votestatus <= 1;
                              estado <= 4'b0000;
                        end
                        else if((digit == 4'b0100) & (valid)) begin
//comparando 4 em binário
                              isabella <= isabella + 8'b00000001;
//contagem de votos para isabella
                              votestatus <= 1;
                              estado <= 4'b0000;
                        end
                        else if((digit == 4'b0010) & (valid)) begin
//comparando 2 em binário
                              filipe <= filipe + 8'b00000001;
//contagem de votos para filipe
                              votestatus <= 1;
                              estado <= 4'b0000;
                        end
```

```
                        else begin //voto nulo
                              nulo <= nulo + 8'b00000001;
//contagem de votos nulos
                              votestatus <= 0;
                              estado <= 4'b0000;
                        end
                  end
                  //fim quarto estado

                  //segundo estado nulo
                  4'b0101: begin
                        if(valid) begin
                              estado <= 4'b0110;
                        end
                  end
                  // fim segundo estado nulo

                  //terceiro estado nulo
                  4'b0110: begin
                        if(valid) begin
                              estado <= 4'b0100;
                        end
                  end
                  //fim terceiro estado nulo

                  //quarto estado nulo
                  4'b0100: begin
                        if(valid) begin
                              nulo <= nulo + 8'b00000001;
//contagem de votos nulos
                              votestatus <= 0;
                              estado <= 4'b0000;
                        end
                  //fim quarto estado nulo

                  end
            endcase //fim case
      end

end
endmodule
```

- Módulo para simulação:

```
Módulo para simulação:
`include "urnaeletronica.v"
module testbench();
reg clock,valid,finish;
reg [3:0]digit;
reg [2:0]control;
wire [7:0]tisabella;
wire [7:0]tclaudio;
wire [7:0]tfilipe;
```

```
wire [7:0]tguilherme;
wire [7:0]tnulo;
wire votestatus;

urnaeletronica instancial(.votestatus(votestatus),
.control(control), .clock(clock), .valid(valid), .finish(finish),
.digit(digit), .tisabella(tisabella), .tclaudio(tclaudio),
.tfilipe(tfilipe), .tguilherme(tguilherme), .tnulo(tnulo));

initial begin
  $dumpfile("urnaeletronica.vcd");
  $dumpvars(0, testbench);
  $display("-----Urna Eletronica-----");
  $monitor("- Votos para Isabella: %d\n- Votos para Claudio: %d\n-
Votos para Filipe: %d\n- Votos para Guilherme: %d\n- Votos
nulos:%d\n", tisabella, tclaudio, tfilipe, tguilherme, tnulo);
end

initial begin

// Inserindo dígitos para Isabella
  #1; clock = 0;
  #1; clock = 1; finish = 0; valid = 0;digit = 4'b0011; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0100; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0111; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0100; valid = 1;



 // Inserindo dígitos para Cláudio
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0011; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0100; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b1001; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0010; valid = 1;

// Inserindo dígitos para Filipe
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0011; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0101; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0000; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0010; valid = 1;

// Inserindo dígitos para Guilherme
  #1; clock = 0;
```

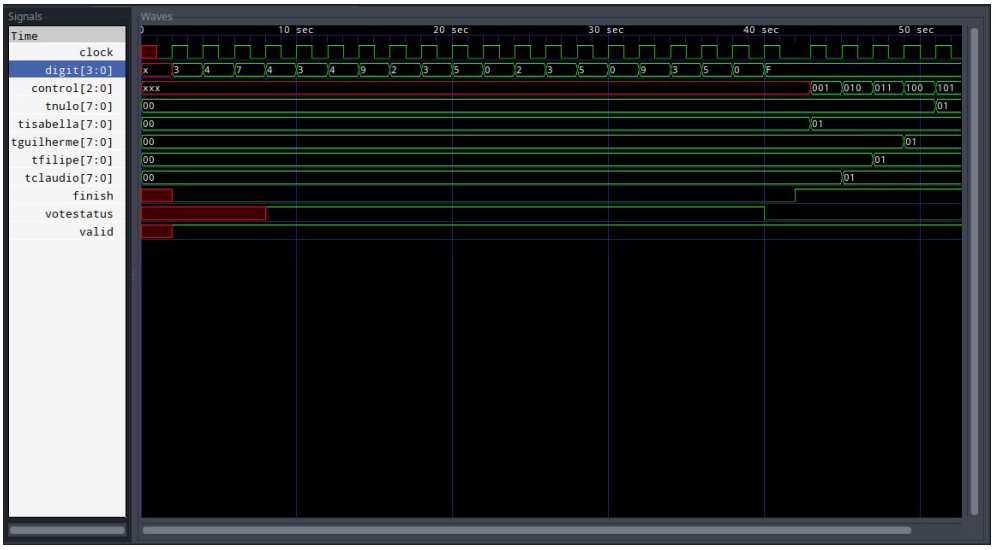
```
  #1; clock = 1; valid = 0; digit = 4'b0011; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0101; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0000; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b1001; valid = 1;

// Inserindo dígitos para Nulo
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0011; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0101; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b0000; valid = 1;
  #1; clock = 0;
  #1; clock = 1; valid = 0; digit = 4'b1111; valid = 1;


  #1; clock = 0;
  #1; clock = 1; finish = 1; clock = 0;//Finalizando a votação
  #1; clock = 1 ;control = 3'b001; //mostrando votos de Isabella
  #1; clock = 0;
  #1; clock = 1; control = 3'b010; //mostrando votos de Cláudio
  #1; clock = 0;
  #1; clock = 1; control = 3'b011; //mostrando votos de Filipe
  #1; clock = 0;
  #1; clock = 1; control = 3'b100; //mostrando votos de Guilherme
  #1; clock = 0;
  #1; clock = 1; control = 3'b101; //mostrando votos nulos
  #1; clock = 0;
  #1; clock = 1; control = 3'b000; //zerando os votos
  #1;
  $finish;
end
endmodule
```

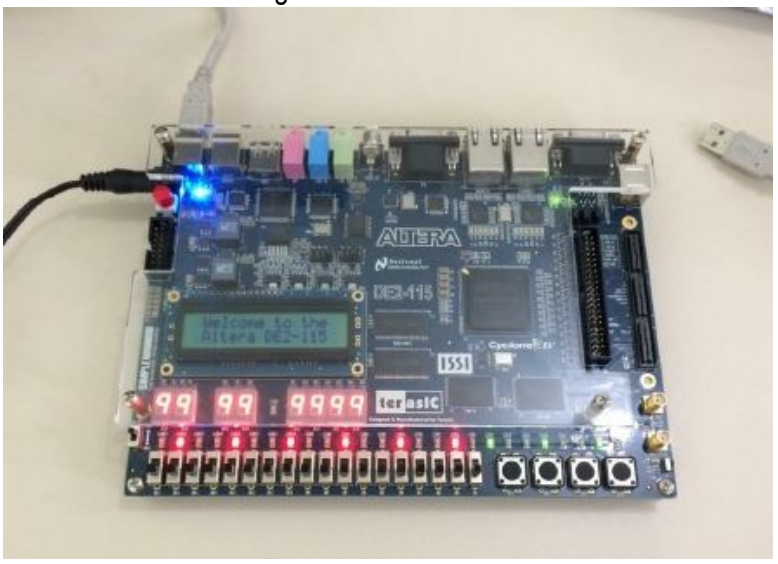
Para verificar o comportamento do módulo em função das formas de onda, foi utilizado o código para simulação no software GTKWave que apresentou o seguinte resultado das entradas (clock, valid, finish, control, digit) e das saídas (tisabella, tclaudio, tfilipe, tguilherme, tnulo, votestatus) que pode ser observado na Figura 3.

Figura 3: Representação das ondas obtidas.



Após o desenvolvimento do código em Verilog para simulação, o próximo passo foi realizar o código que será executado na FPGA DE2-115 (Figura 4).

Figura 4: Placa FPGA utilizada.



O código com as entradas e saídas estanciadas foi o seguinte:

```
`include "urnaeletronica.v"

//=======================================================
//  This code is generated by Terasic System Builder
//=======================================================

module TP_UrnaEletronica(

      ///////////// LED //////////
      LEDG,
      LEDR,
```

```
      ////////////// SW //////////
      SW,

      ////////////// SEG7 //////////
      HEX0,
      HEX1,
      HEX2,
      HEX3,
      HEX4,
      HEX5,
      HEX6,
      HEX7
);

//=======================================================
//  PARAMETER declarations
//=======================================================


//=======================================================
//  PORT declarations
//=======================================================

////////////// LED //////////
output                [8:0]       LEDG;
output               [17:0]       LEDR;

////////////// SW //////////
input                [17:0]       SW;

////////////// SEG7 //////////
output                [6:0]       HEX0;
output                [6:0]       HEX1;
output                [6:0]       HEX2;
output                [6:0]       HEX3;
output                [6:0]       HEX4;
output                [6:0]       HEX5;
output                [6:0]       HEX6;
output                [6:0]       HEX7;


//=======================================================
//  REG/WIRE declarations
//=======================================================
//=======================================================
//  Structural coding
//=======================================================


urnaeletronica instancial(.votestatus(LEDG[0]), .control(SW[17:15]),
.clock(SW[14]), .valid(SW[13]), .finish(SW[12]), .digit(SW[3:0]),
.tisabella(HEX0[6:0]), .tclaudio(HEX1[6:0]), .tfilipe(HEX2[6:0]),
.tguilherme(HEX3[6:0]), .tnulo(HEX4[6:0]));
```

3. RESULTADOS

A realização deste trabalho permitiu a utilização dos conhecimentos em máquinas de estado finitos, Verilog e apresentou softwares importantes e que são utilizados para a simulação e também implementação do código. A representação de ondas obtida através do programa GTKWave permite visualizar o funcionamento durante o tempo de execução, e o JFLAP permitiu representarmos graficamente cada estado da máquina podendo assim ver o que realmente está acontecendo.

A execução do código na FPGA serviu como demonstração das variações que um projeto, mesmo que simples como este, pode ter. O resultado obtido foi satisfatório e serviu como ferramenta didática importante, uma vez que representou a aplicação do tema abordado neste trabalho.

4. BIBLIOGRAFIA

- R. Katz, G. Borriello, Contemporary Logic Design, 2ª edição, Prentice Hall, 2004.

- F. Vahid, Digital Design with RTL Design, Verilog and VHDL, 2ª edição, Wiley, 2011.