PUC-Minas - Ciência da Computação

ARQ1 - Guia 07

Período: 17-21/09/2012

Tema: Introdução à linguagem Verilog

Atividade: Circuitos sequenciais - Máquinas de estado finito (FSM)

O1.) Projetar e descrever em Verilog um módulo para implementar uma máquina de estados finitos segundo a abordagem de Mealy.
O nome do arquivo deverá ser Mealy1101.v, e poderá seguir o modelo descrito abaixo.

```
// --- Mealy FSM
// -----
               Mealy FSM Diagram
                                   1 | // found
        1 v
                             0
  [start] ---> [id1] ---> [id11] ---> [id110]
                      1/ ^
                                   0 | // not found
   ^ \0
            0 |
                                    // constant definitions
`define found
`define notfound 0
// FSM by Mealy
module mealy1101 (y, x, clk, reset);
output y;
input x;
input clk;
input reset;
 reg
       y;
                // state identifiers
 parameter
 start = 2'b00,
 id1 = 2'b01,
 id11 = 2'b11,
 id110 = 2'b10;
  reg [1:0] E1;// current state variables
  reg [1:0] E2;// next state logic output
```

```
// next state logic
  always @(x or E1)
  begin
   y = notfound;
   case (E1)
   start:
    if (x)
     E2 = id1;
    else
     E2 = start;
   id1:
    if (x)
     E2 = id11;
    else
     E2 = start;
   id11:
    if (x)
     E2 = id11;
    else
     E2 = id110;
   id110:
    if (x)
     begin
      E2 = id1;
      y = found;
     end
     else
     begin
      E2 = start;
      y = \infty
     end
   default: // undefined state
      E2 = 2'bxx;
   endcase
  end // always at signal or state changing
// state variables
  always @( posedge clk or negedge reset )
  begin
   if (reset)
   E1 = E2; // updates current state
   else
   E1 = 0; // reset
  end // always at signal changing
```

endmodule // mealy1101

02.) Projetar e descrever em Verilog um módulo para implementar uma máquina de estados finitos segundo a abordagem de Moore.
O nome do arquivo deverá ser Moore.v, e poderá seguir o modelo descrito abaixo.

```
// -----
// --- Moore FSM
                         Moore FSM Diagram
                 1
                            0
                                               1 | // found
                      ٧
                                        1
  [start] ---> [id1] ---> [id11] ---> [id110] ---> [id1101]
   ^ \0
                     1/ ^
            0 |
                                  0 |
*/
// constant definition
`define found
`define notfound 0
// FSM by Moore
module moore1101 (y, x, clk, reset);
output y;
input x;
input clk;
input reset;
reg
       у;
 parameter
               // state identifiers
 start = 3'b000,
         = 3'b001,
 id1
 id11 = 3'b011,
 id110 = 3'b010,
 id1101 = 3'b110; // signal found
  reg [2:0] E1;// current state variables
```

reg [2:0] E2;// next state logic output

```
// next state logic
  always @(x or E1)
  begin
   case(E1)
   start:
    if (x)
     E2 = id1;
    else
     E2 = start;
   id1:
    if (x)
     E2 = id11;
    else
     E2 = start;
   id11:
    if (x)
     E2 = id11;
    else
     E2 = id110;
   id110:
    if (x)
     E2 = id1101;
    else
     E2 = start;
   id1101:
    if (x)
     E2 = id11;
    else
     E2 = start;
   default: // undefined statee
     E2 = 3bxxx;
   endcase
  end // always at signal or state changing
// state variables
  always @( posedge clk or negedge reset )
  begin
   if (reset)
   E1 = E2; // updates current state
   else
   E1 = 0; // reset
  end // always at signal changing
// output logic
  always @(E1)
  begin
  y = E1[2]; // first bit of state value (MOORE indicator)
  end // always at state changing
endmodule // moore1101
```

03.) Projetar e descrever em Verilog um módulo para testar as máquinas de estados finitos segundo as abordagens de Mealy e de Moore. O nome do arquivo deverá ser Exemplo0051.v, e poderá seguir o modelo descrito abaixo.

```
// --- Mealy-Moore FSM
'include "mealy1101.v"
'include "moore1101.v"
module Exemplo0051;
reg clk, reset, x;
wire m1, m2;
mealy1101 mealy1 (m1, x, clk, reset);
moore1101 moore1 ( m2, x, clk, reset );
initial
 begin
  $display ("Time X Mealy Moore");
// initial values
    clk = 1;
    reset = 0;
    x = 0;
// input signal changing
 #5 reset = 1;
 #10 x = 1;
 #10 x = 0;
 #10 x = 1;
 #20 x = 0;
 #10 x = 1;
 #10 x = 1;
 #10 x = 0;
 #10 x = 1;
 #30 $finish;
 end // initial
always
 #5 clk = ~clk;
always @( posedge clk )
 begin
 $display ( "%4d %4b %4b %5b", $time, x, m1, m2 );
 end // always at positive edge clocking changing
endmodule // Exemplo0051
```

- 04.) Projetar e descrever em Verilog um módulo para implementar uma máquina de estados finitos, segundo a abordagem de Mealy, capaz de reconhecer uma sequência (1010) com interseção (1001001 deverá ser reconhecida também). O nome do arquivo deverá ser Exemplo0052.v. Incluir previsão de testes e verificação do circuito pelo Logisim.
- 05.) Projetar e descrever em Verilog um módulo para implementar uma máquina de estados finitos, segundo a abordagem de Moore, capaz de reconhecer uma sequência (1010) sem interseção (1001001 não deverá ser reconhecida).
 O nome do arquivo deverá ser Exemplo0053.v. Incluir previsão de testes e verificação do circuito pelo Logisim.
- 06.) Projetar e descrever em Verilog um módulo para implementar uma máquina de estados finitos, capaz de reconhecer a primeira sequência (1100) que aparecer.
 O nome do arquivo deverá ser Exemplo0054.v. Incluir previsão de testes e verificação do circuito pelo Logisim.

Extra

- 07.) Projetar e descrever em Verilog um módulo para implementar uma máquina de estados finitos, capaz de reconhecer uma sequência de quatro dígitos binários que termine com três valores iguais a 0 (x000, por exemplo). O nome do arquivo deverá ser Exemplo0055.v. Incluir previsão de testes e verificação do circuito pelo Logisim.
- 08.) Projetar e descrever em Verilog um módulo para implementar uma máquina de estados finitos, capaz de reconhecer uma sequência de três dígitos binários iguais (000 ou 111).
 O nome do arquivo deverá ser Exemplo0056.v. Incluir previsão de testes e verificação do circuito pelo Logisim.

Instruções para ver as cartas de tempo no GTKWave:

- 01.) Abrir o módulo de visualização (GTKWave)
- 02.) Selecionar a pasta de trabalho:

File

Open

Exemplo0041 (.vcd) (por exemplo)

03.) Selecionar os sinais desejados:

clk (sinal a ser visto)

clock (outro sinal a ser visto)

(selecionar, arrastar e soltar na coluna à direita)