

# Relatório da Prática V - Diretório

Alunos: Bruno Rocha Ribeiro e Cristhian Sala Minoves

## 1. Introdução

Em contrapartida ao protocolo Snooping - MSI, visando auxiliar a reduzir os problemas por ele trazidos, foi elaborado um outro protocolo, o Diretório. A sua principal diferença é a existência de pequenas tabelas, tanto anexas às caches de cada processador, como nas suas memórias externas. Com isso, é possível ter um maior controle dos dados, como em qual processador ele se encontra, quais outras caches locais determinado dado está compartilhado ou se ele se encontra em um processador externo.

Apesar da grande diferença, o diretório e o snooping guardam uma semelhança: as suas transições de estados também são geridas por uso de uma máquina de estados — os quais são os mesmos na máquina de emissão e há uma troca na máquina receptora do estado inválido para o estado uncached. No entanto, suas transições são diferentes, possuindo, também, mensagens diferentes tanto de entrada como do barramento.

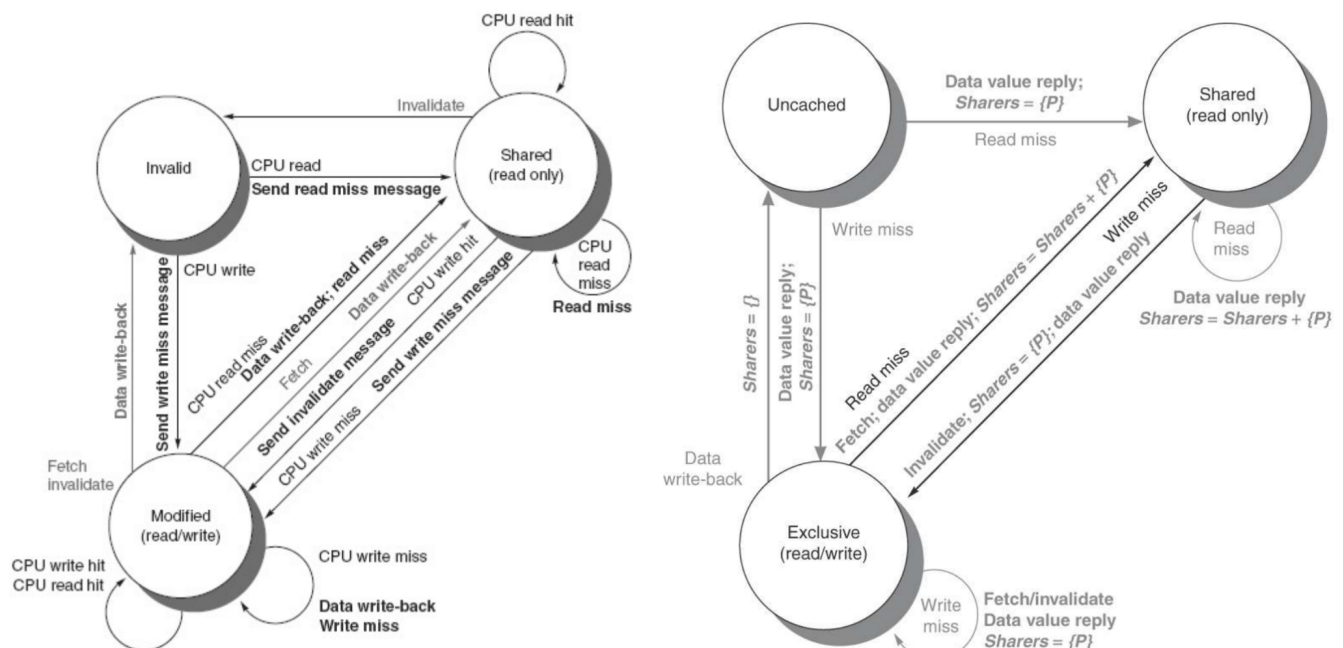


Figura 1: Funcionamento do protocolo Diretório com as máquinas emissora(esquerda) e receptora(direita)

## 2. Projeto e seus testes

Em nosso trabalho, confeccionamos apenas a máquina de estados do protocolo Diretório, sem a implementação de caches, memórias, processadores, etc. Abaixo será listado como foi feita essa parte.

# Máquina emissora

```
1 module state_machine(scao, escrita, clock, dataWriteBack)://dir
2 input clock;
3 input [2:0] scao //simboliza a entrada
4 /*
5 000 - read miss
6 001 - read hit
7 010 - write hit
8 011 - invalidate
9 100 - write miss
10 101 - fetch
11 110 - fetch invalidate
12 */
13 output reg [2:0] escrita://simboliza a saída
14 /*
15 000 - read miss
16 001 - read hit
17 010 - write hit
18 011 - invalidate
19 100 - write miss
20 101 - fetch
21 110 - fetch invalidate
22 111 - mensagem valida
23 */
24 reg [1:0] estado;
25 /*
26 00 - invalid
27 01 - shared
28 10 - modified
29 */
30
31 output reg dataWriteBack;
32 /*
33 0 = sem writeback
34 1 = com writeback
35 */
36
37 always @(posedge clock) begin
38     dataWriteBack = 1'b0;
39     escrita = 3'b111;
40     case(estado)
41         2'b00: begin//invalid
42             case (scao)
43                 3'b000://read miss
44                     begin
45                         estado = 1'b01;
46                         escrita = 3'b000;
47                     end
48                 3'b001://write miss
49                     begin
50                         estado = 1'b10;
51                         escrita = 3'b100;
52                     end
53             endcase
54         end
55         2'b01: begin //shared
56             case(scao)
57                 3'b000://read miss
58                     escrita = 3'b000;
59                 3'b010://write hit
60                     begin
61                         estado = 1'b10;
62                         escrita = 3'b011;
63                     end
64                 3'b001://read hit
65                     estado = 1'b01;
66                 3'b010://write miss
67                     begin
68                         estado = 2'b10;
69                         escrita = 3'b100;
70                     end
71                 3'b011;
72                     estado = 1'b00;
73             endcase
74         end
75         2'b10: begin//modified
76             case(scao)
77                 3'b100://fetch
78                     begin
79                         estado = 2'b01;
80                         dataWriteBack = 1'b1;
81                     end
82                 3'b000://read miss
83                     begin
84                         estado = 2'b01;
85                         dataWriteBack = 1'b1;
86                         escrita = 3'b000;
87                     end
88                 3'b110://fetch invalidate
89                     begin
90                         estado = 2'b00;
91                         dataWriteBack = 1'b1;
92                     end
93                 3'b100://read miss
94                     begin
95                         estado = 2'b01;
96                         dataWriteBack = 1'b1;
97                         escrita = 3'b000;
98                     end
99                 3'b001://read hit
100                     estado = 1'b10;
101                 3'b010://write hit
102                     estado = 1'b10;
103             endcase
104         end
105         default
106             estado = 2'b10;//modified
107     endcase
108 end
109 endmodule
110
```

Figura 2: Implementação da máquina emissora

Seguindo a máquina de estados exposta anteriormente, foi feita a codificação dela. Foi observado os estados, as transições e as mensagens colocadas em cada uma das transições

## Testes

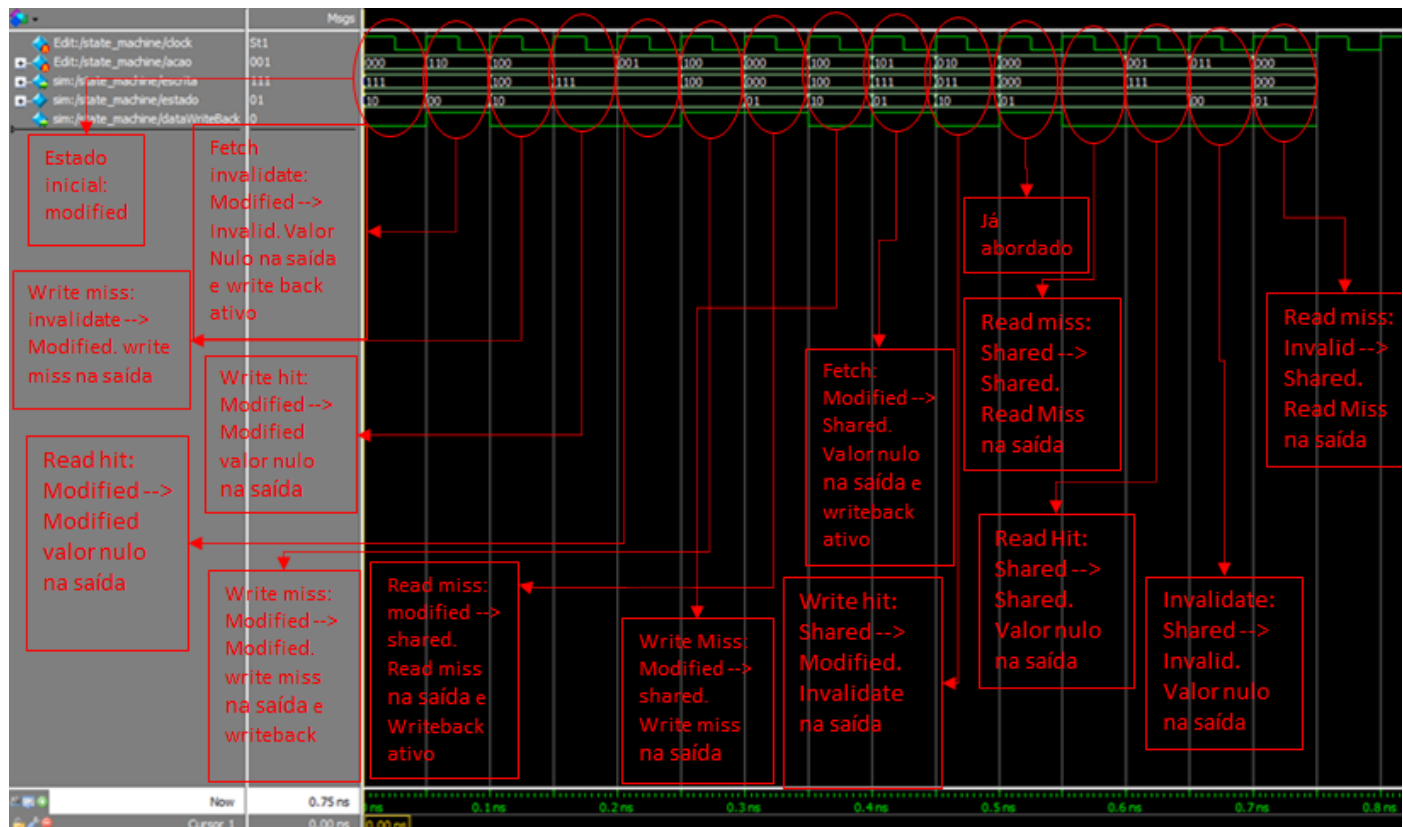


Figura 3: Testes da máquina emissora

Verifica-se nessa imagem o funcionamento correto de todas as transições da máquina emissora.

## Máquina receptora

```
1  module state_machine2(acao, escrita, clock, dataWriteBack, dataValueReply, sharers); //dir
2  input clock;
3  input [1:0] acao; //simboliza a entrada
4  /*
5  00 - read miss
6  01 - read hit
7  10 - Data Writeback
8  11 - write miss
9  */
10 output reg [1:0] escrita; //simboliza a saida
11 /*
12 00 - valor nulo
13 01 - invalidate
14 10 - fetch
15 11 - fetch invalidate
16 */
17 output reg [1:0] sharers;
18 /*
19 00 - Nulo
20 01 - sharers = {}
21 10 - sharers = {p}
22 11 - Sharers = sharers + {p}
23 */
24 reg [1:0] estado;
25 /*
26 00 - uncached
27 01 - shared
28 10 - exclusive
29 */
30 input dataWriteBack;
31 /*
32 0 - sem writeback
33 1 - com writeback
34 */
35 output reg dataValueReply;
36
37 always @(posedge clock) begin
38     escrita = 2'b00;
39     sharers = 2'b00;
40     dataValueReply = 1'b0;
41     case(estado)
42     2'b00: begin //uncached
43         case (acao)
44         2'b00: //read miss
45             begin
46                 estado = 2'b01;
47                 dataValueReply = 1'b1;
48                 sharers = 2'b10;
49             end
50         2'b11: //write miss
51             begin
52                 estado = 2'b10;
53                 dataValueReply = 1'b1;
54                 sharers = 2'b10;
55             end
56         endcase
57     end
58     2'b01: begin //shared
59         case(acao)
60         2'b00: //read miss
61             begin
62                 estado = 2'b01;
63                 sharers = 2'b11;
64                 dataValueReply = 1'b1;
65             end
66         2'b11: //write miss
67             begin
68                 estado = 2'b10;
69                 escrita = 2'b01;
70                 dataValueReply = 1'b1;
71                 sharers = 2'b10;
72             end
73         endcase
74     end
75     2'b10: begin //exclusive
76         case(acao)
77         2'b11: //write miss
78             begin
79                 estado = 2'b10;
80                 dataValueReply = 1'b1;
81                 sharers = 2'b10;
82                 escrita = 2'b11;
83             end
84         2'b00: //read miss
85             begin
86                 estado = 2'b01;
87                 dataValueReply = 1'b1;
88                 sharers = 2'b11;
89                 escrita = 2'b10;
90             end
91         2'b10: //data writeback
92             begin
93                 estado = 2'b00;
94                 sharers = 2'b01;
95             end
96         endcase
97     end
98     default
99     estado = 2'b00;
100 endcase
101 end
102 endmodule
103
```

Figura 4: Implementação da máquina receptora

## Testes

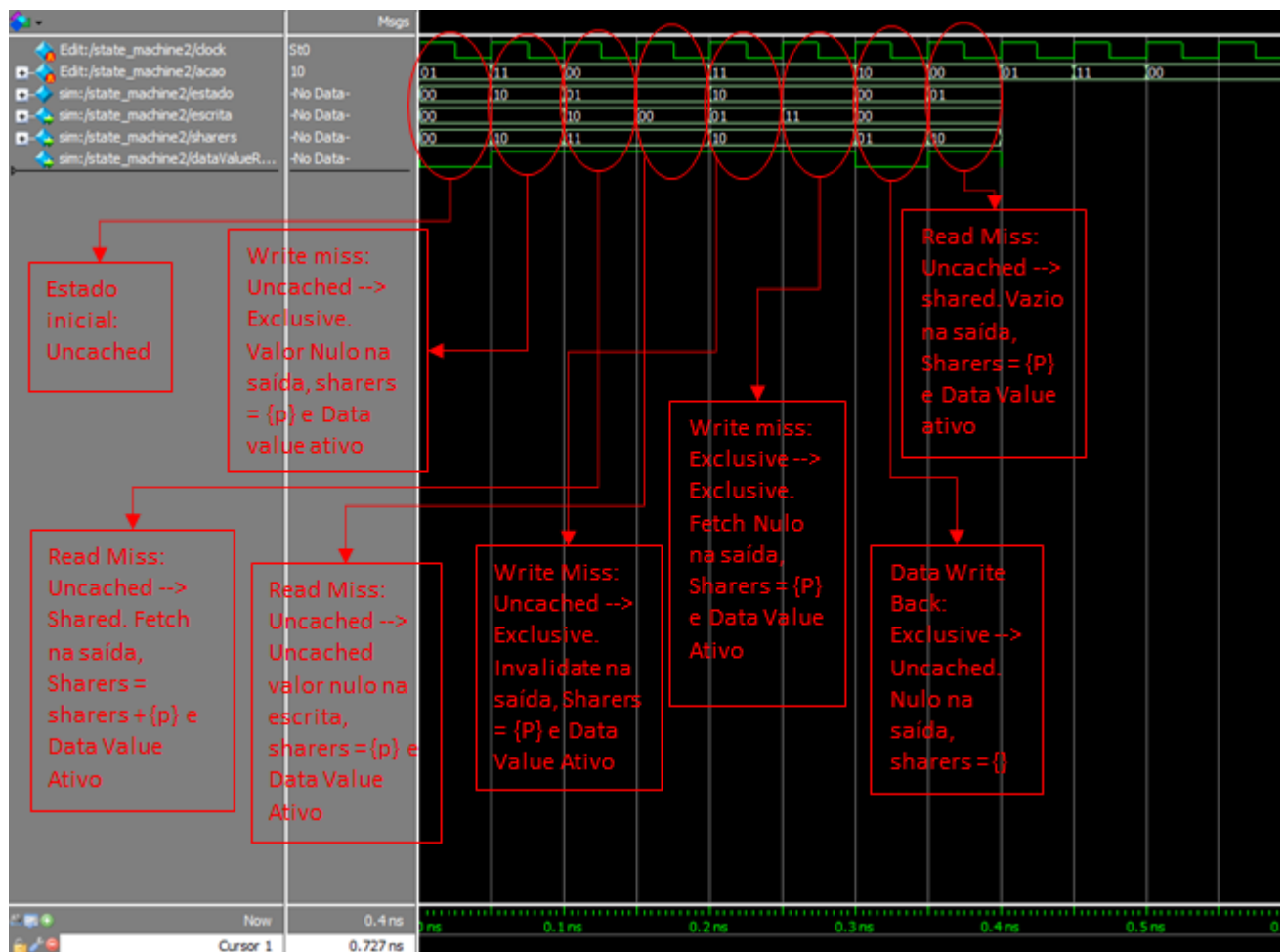


Figura 5: Testes da máquina receptora

## 3. Dificuldades encontradas

Como o projeto foi focado somente na primeira etapa, a da confecção das máquinas de estado, as dificuldades foram bem pequenas, só mesmo a confusão entre zeros e uns, situação que ocorreu também na confecção da máquina de estados do Snooping

## 4. Sugestões para melhoria da prática

Para essa parte 1, os métodos estão de acordo com o conteúdo da disciplina e aulas ministradas em sala, como sugestão apenas a questão da falta de um monitor que poderia nos auxiliar com questões mais complexas ou eventuais.

## 5.Comentários adicionais

Não é necessário comentar nada além do que já foi apresentado.