UNIVERSIDADE FEDERAL DA FRONTEIRA SUL
CAMPUS CHAPECÓ
CURSO DE CIÊNCIA DA COMPUTAÇÃO

ISADORA LAÍS RUSCHEL

TRABALHO MÁQUINAS DE ESTADO

CONTROLADOR DE BARRAMENTO

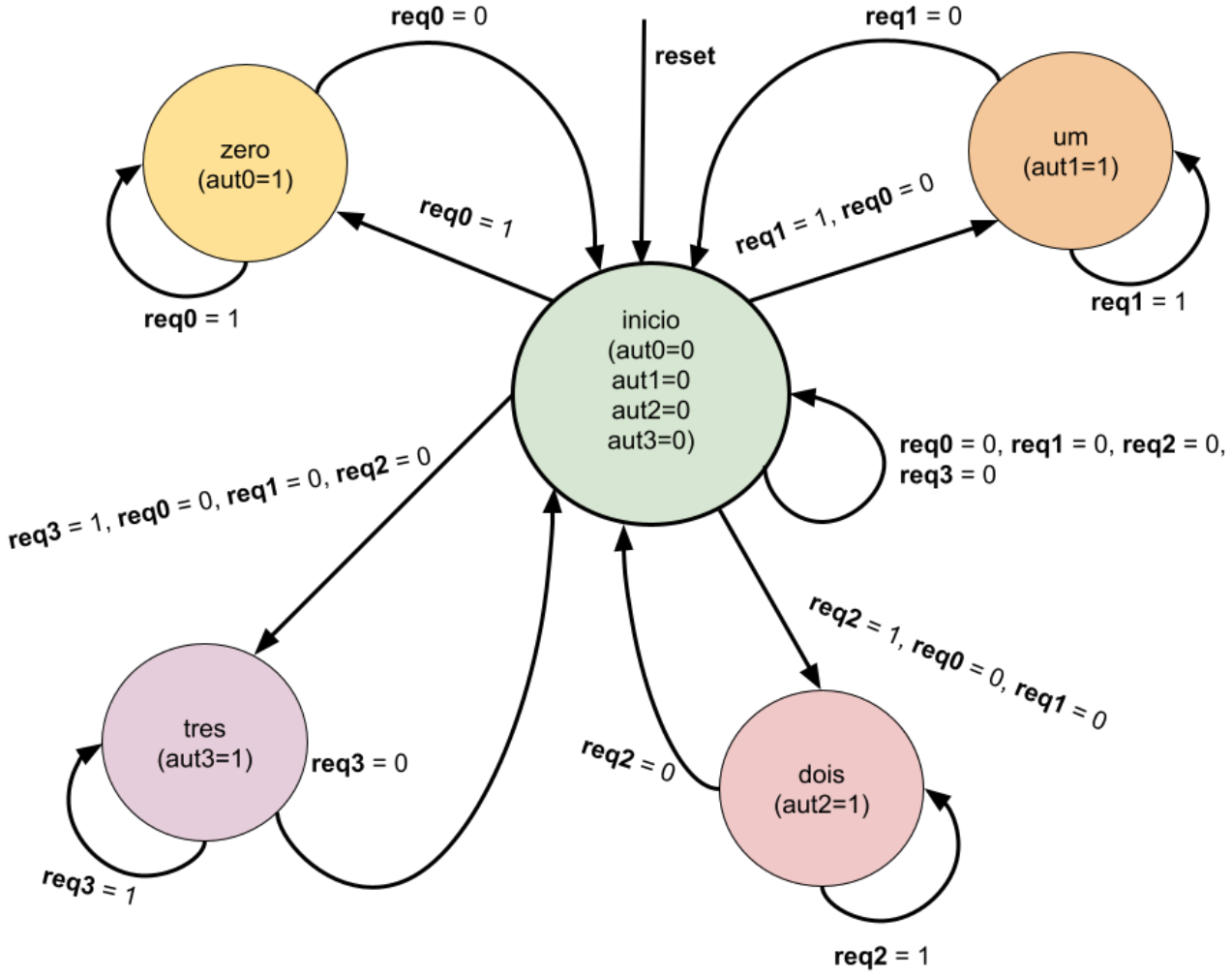
CHAPECÓ
2024

SUMÁRIO

1 INTRODUÇÃO

Para fazer o controlador de barramento, criei quatro estados, cada um indica qual requisição está sendo atendida pelo árbitro. O estado “zero” indica que o disp0 está acessando o barramento, o estado “um” indica que o disp1 está acessando o barramento, o estado “dois” indica que o disp2 está acessando o barramento, já o estado “tres” indica que o disp3 está acessando o barramento. O estado “inicio” é o estado inicial, o qual atribui sempre 0 às saídas “aut” do barramento e é de passagem obrigatória para mudar de estados.

Então construí o diagrama de estados seguinte:



Após isso, construí a tabela de estados:

Estado atual	req0	req1	req2	req3	Próximo estado
inicio	0	0	0	0	inicio
inicio	0	0	0	1	tres
inicio	0	0	1	X	dois
inicio	0	1	X	X	um
inicio	1	X	X	X	zero
zero	0	X	X	X	inicio
zero	1	X	X	X	zero
um	X	0	X	X	inicio
um	X	1	X	X	um
dois	X	X	0	X	inicio
dois	X	X	1	X	dois
tres	X	X	X	0	inicio
tres	X	X	X	1	tres

E a tabela de saída:

Estado	aut0	aut1	aut2	aut3
inicio	0	0	0	0
zero	1	0	0	0
um	0	1	0	0
dois	0	0	1	0
tres	0	0	0	1

Assim, desenvolvi o sistema em VHDL, criei a entidade “controlador” com entrada para o clock e para o reset, porém, ao invés de utilizar quatro entradas e saídas separadas para cada aut e req, eu utilizei dois vetores de 4 posições, sendo assim, aut0 se tornou aut(0), req0 se tornou req(0), e o mesmo se repete para o restante dos “aut’s” e “req’s”. Após isso, segui para a criação da arquitetura do controlador utilizando máquina de estados.

2 ARQUITETURA DO CONTROLADOR

Nomeei a arquitetura do controlador como “comport_controlador”, criei os estados “inicio”, “zero”, “um”, “dois” e “tres”, e então um signal para armazenar o estado atual.

O controlador feito possui reset assíncrono, o qual funciona atribuindo ao estado atual o estado inicial, sem necessidade de mudança no clock. Quando ativo, as entradas “req” não são verificadas.

Quando há subida do clock e o reset não está ativo, as entradas são verificadas e caso haja mudança no valor da requisição correspondente ao estado atual, este mudará na descida do clock, caso contrário, o estado permanecerá o mesmo. No caso do estado “inicio”, qualquer mudança em alguma das entradas levará a outro estado, mas se as entradas forem todas 0, ele não mudará.

Se uma requisição está sendo atendida pelo árbitro, o sinal “aut” do dispositivo correspondente não mudará até que o dispositivo abaixe seu sinal de requisição, independentemente dos sinais de requisição dos outros dispositivos.

Desta forma, foram feitos dois process, um para tratar a mudança de estados e outro para atribuir valores às saídas conforme o estado atual. A prioridade de atendimento às requisições é tratada por meio de estruturas “if”, ou seja, verificando primeiro se os dispositivos com maior prioridade estão requisitando acesso ao barramento.

3 CONCLUSÃO

O controlador de barramento desenvolvido utiliza vetores para as saídas e entradas, mas funciona da maneira esperada, atendendo as requisições com prioridades diferentes, sendo o disp0 com maior prioridade e o disp3 com a menor, com os estados e os respectivos sinais “aut” mudando apenas na subida do clock.

O atendimento às requisições é feito por meio da mudança de estados, começando sempre no estado “inicio”, atendendo um dispositivo por vez, e toda vez que alguma requisição que está sendo atendida não deseja mais acessar o barramento, mudando seu sinal req para 0, o próximo estado será o “inicio” novamente, não podendo haver transição imediata de um estado que está atendendo uma requisição para atender outro.