Sistemas Reconfiguráveis
Eng. de Computação

Profs. Francisco Garcia e Antônio Hamilton Magalhães

Aula 11 – Introdução às máquina de estados finitos (FSM)

rev. C – 31/10/23

Máquina de estados finitos

- Uma **máquina de estados finitos** (em inglês: *finite state machine* - FSM) é um modelo usado para representar um sistema lógico. Como o nome já diz, essa máquina (às vezes chamada de autômato) pode estar em um número finito de estados.
- A máquina está em apenas um estado por vez, este estado é chamado de **estado atual**.
- Uma transição indica uma mudança de estado e é descrita por uma condição que precisa ser realizada para que a transição ocorra.
- Existem FSMs síncronas e assíncronas. Estudaremos apenas as FSMs **síncronas**, onde as transições de estado ocorrem sincronizadas com as transições de um sinal, chamado comumente de *clock*.

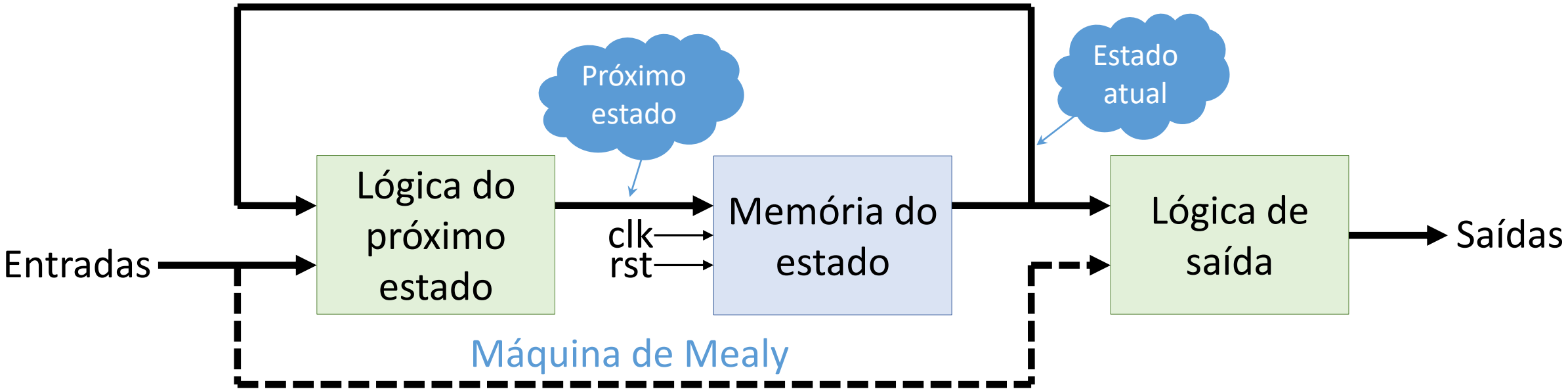
Máquina de estados finitos

- Existem dois tipos de máquinas de estados:
- **Máquina de Moore**: As saídas dependem apenas do estado atual
- **Máquina de Mealy**: As saídas dependem do estado atual e também das entradas

Máquina de estados finitos

- A implementação de uma FSM pode ser dividida em três blocos:
- **Memória do estado**: Circuito de lógica sequencial (flip-flops) que memoriza o estado atual. As mudanças de estado são disparadas por um sinal de *clock*. Pode ter uma entrada que estabelece um estado inicial de forma assíncrona (*reset*)
- **Lógica do próximo estado**: Lógica combinacional que, para cada estado atual e condição de entrada, determina qual o próximo estado. (Em uma FSM síncrona, o estado só vai ser atualizado no próximo pulso de *clock*)
- **Lógica de saída**: Combinacional. Determina as saídas, com base no estado atual (máquina de Moore) ou no estado atual e entradas (máquina de Mealy)

Máquina de estados finitos

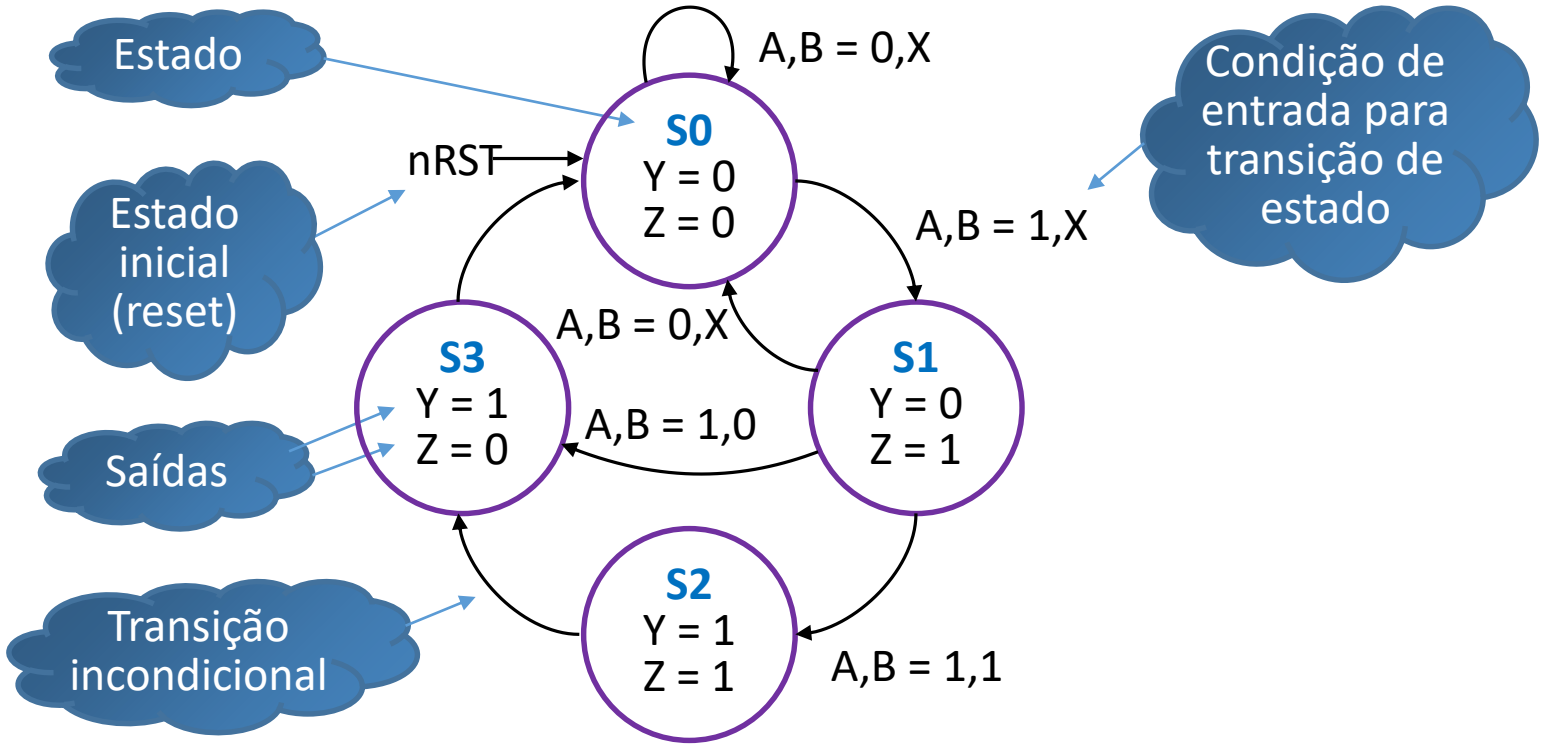


Máquina de estados finitos

- Existem alguns métodos de documentar o funcionamento de uma máquina de estados. Abaixo é mostrado um exemplo de um **diagrama de transição de estados** de uma máquina de Moore

A máquina de estados desse exemplo tem quatro estados (S0, S1, S2 e S3), duas entradas (A e B) e duas saídas (Y e Z)

Os círculos representam os estados e as setas as transições de estado



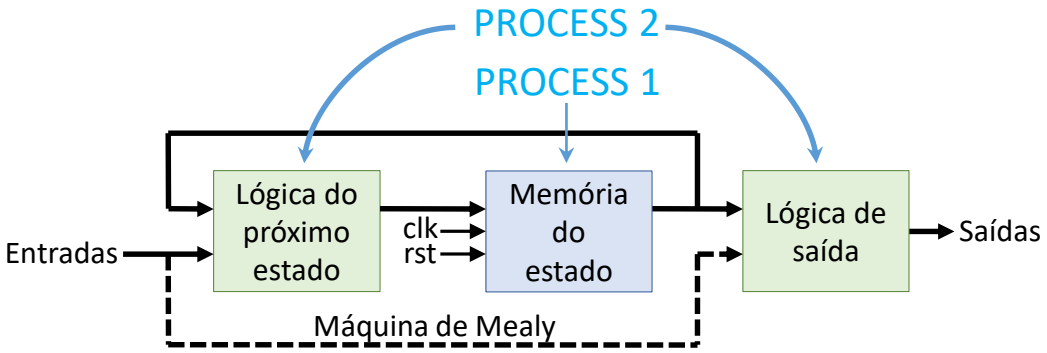
Máquina de estados finitos

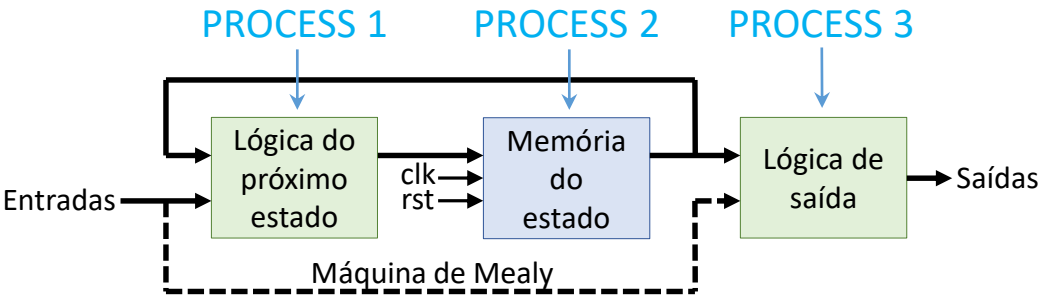
A codificação do comportamento de uma máquina de estados finitos em VHDL deve ser feita usando código sequencial e permite três opções:

- Um único PROCESS, com toda a lógica
- não recomendado

- Dois PROCESS, um para a memória do estado (flip-flops) e outro para as lógicas do próximo estado e de saída (combinacionais)
- recomendado para a maior parte das aplicações

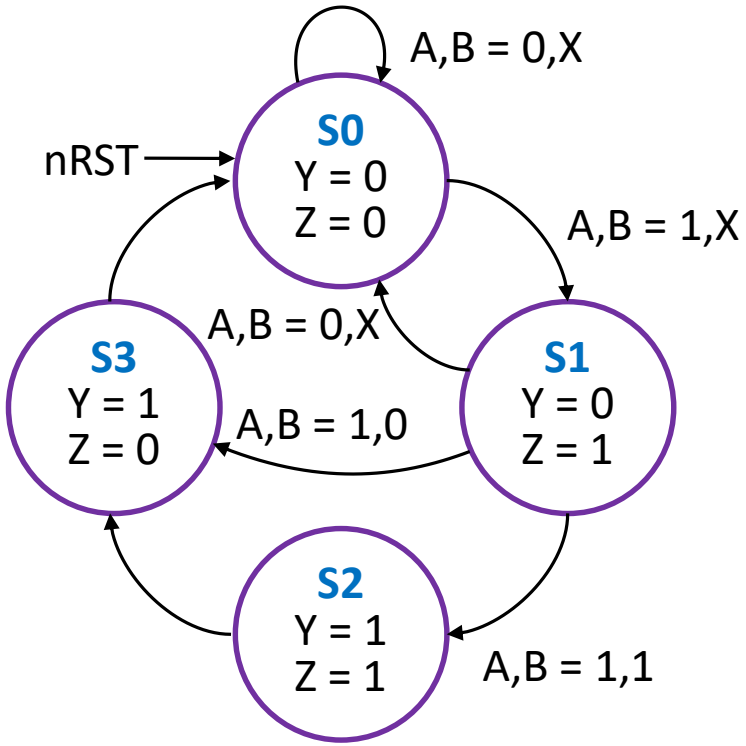
- Três PROCESS, um para a memória do estado (flip-flops), outro para a lógica do próximo estado e um terceiro para a lógica de saída (combinacionais)
- recomendado para aplicações muito complexas





Máquina de estados finitos
Exemplo

- **Exemplo fsm_1**: Descrever em VHDL a FSM cujo diagrama de transição de estados é mostrado abaixo:



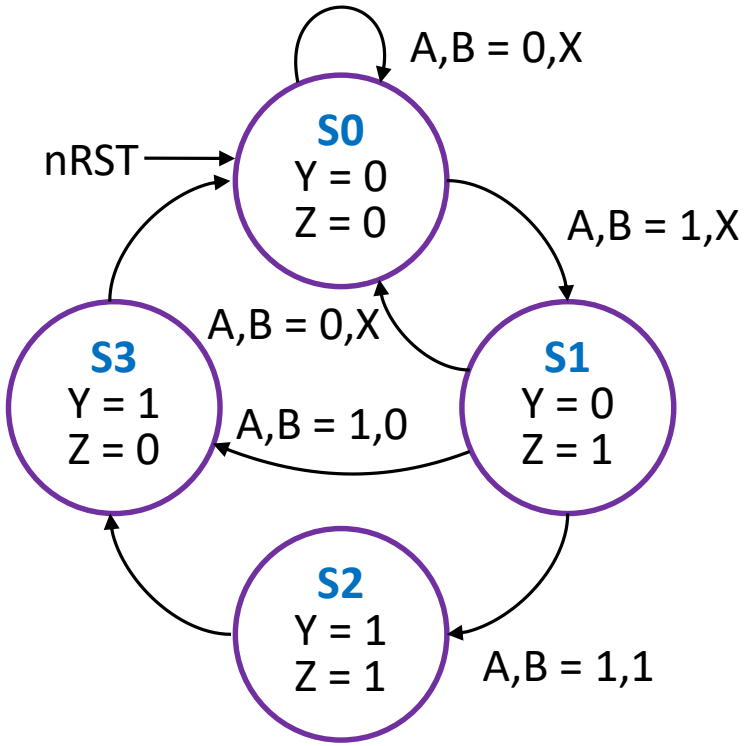

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY fsm_1 IS
PORT (
        clk, nrst, a, b : IN STD_LOGIC;
        y, z : OUT STD_LOGIC
);
END ENTITY;
.......
```

Continua na próxima página

Máquina de estados finitos
Exemplo

- **Exemplo fsm_1**: Descrever em VHDL a FSM cujo diagrama de transição de estados é mostrado abaixo:

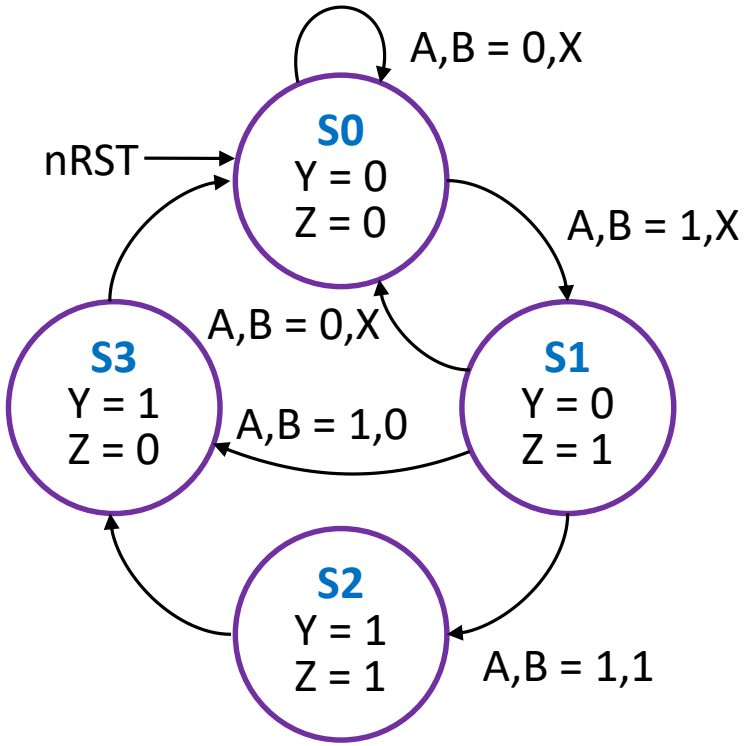



```
ARCHITECTURE arch1 OF fsm_1 IS
TYPE state_type IS (s0, s1, s2, s3);
SIGNAL next_state, pres_state : state_type;
BEGIN
-- Parte sequencial da máquina de estados:
PROCESS(nrst, clk)
BEGIN
        IF nrst = '0' THEN
                pres_state <= s0;
        ELSIF RISING_EDGE(clk) THEN
                pres_state <= next_state;
        END IF;
END PROCESS;
```

Continua na próxima página

Máquina de estados finitos
Exemplo

- **Exemplo fsm_1**: Descrever em VHDL a FSM cujo diagrama de transição de estados é mostrado abaixo:

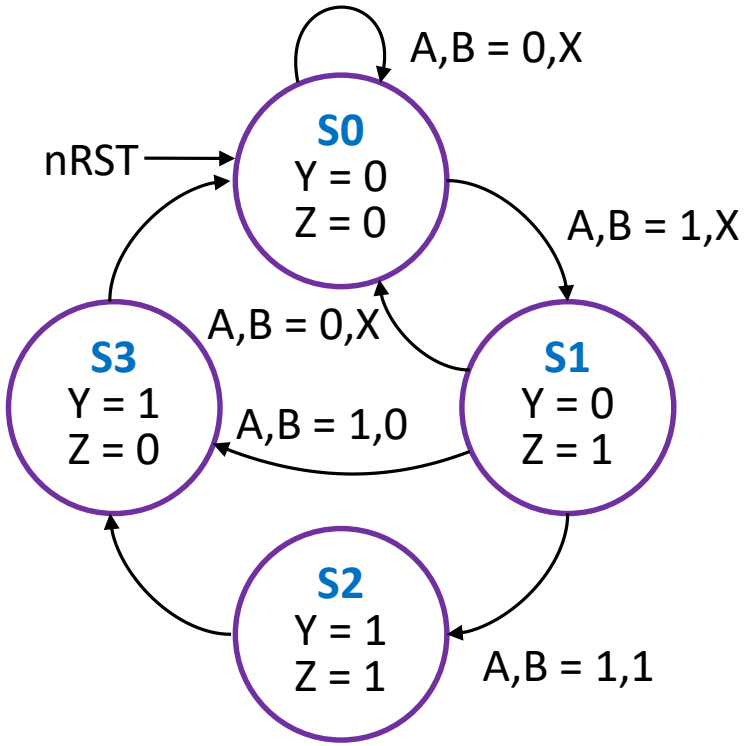



```
-- Parte combinacional da máquina de estados:
PROCESS(a, b, pres_state)
BEGIN
        CASE pres_state IS
                WHEN s0 => y <= '0'; z <= '0';
                        IF a = '0' THEN
                                next_state <= s0;
                        ELSE
                                next_state <= s1;
                        END IF;
                WHEN s1 => y <= '0'; z <= '1';
                        IF a = '0' THEN
                                next_state <= s0;
                        ELSIF b = '0' THEN
                                next_state <= s3;
                        ELSE
                                next_state <= s2;
                        END IF;
```

Continua na próxima página

Máquina de estados finitos
Exemplo

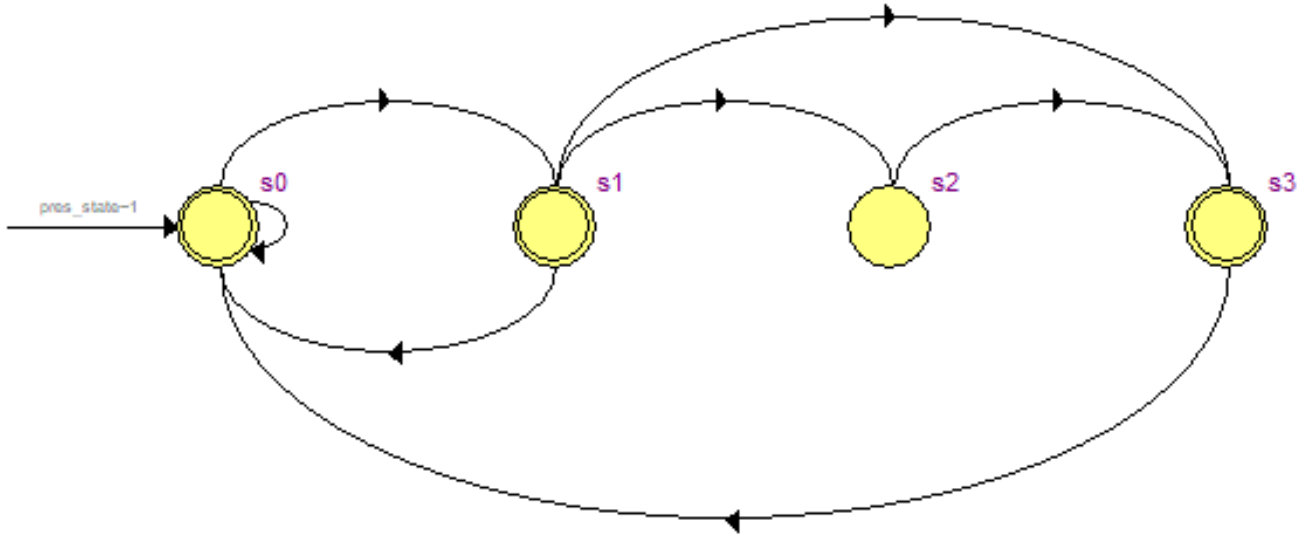
- **Exemplo fsm_1**: Descrever em VHDL a FSM cujo diagrama de transição de estados é mostrado abaixo:




```
    .....
                WHEN s2 => y <= '1'; z <= '1';
                           next_state <= s3;
                WHEN s3 => y <= '1'; z <= '0';
                           next_state <= s0;
        END CASE;
END PROCESS;
END arch1;
```

Máquina de estados finitos
Exemplo

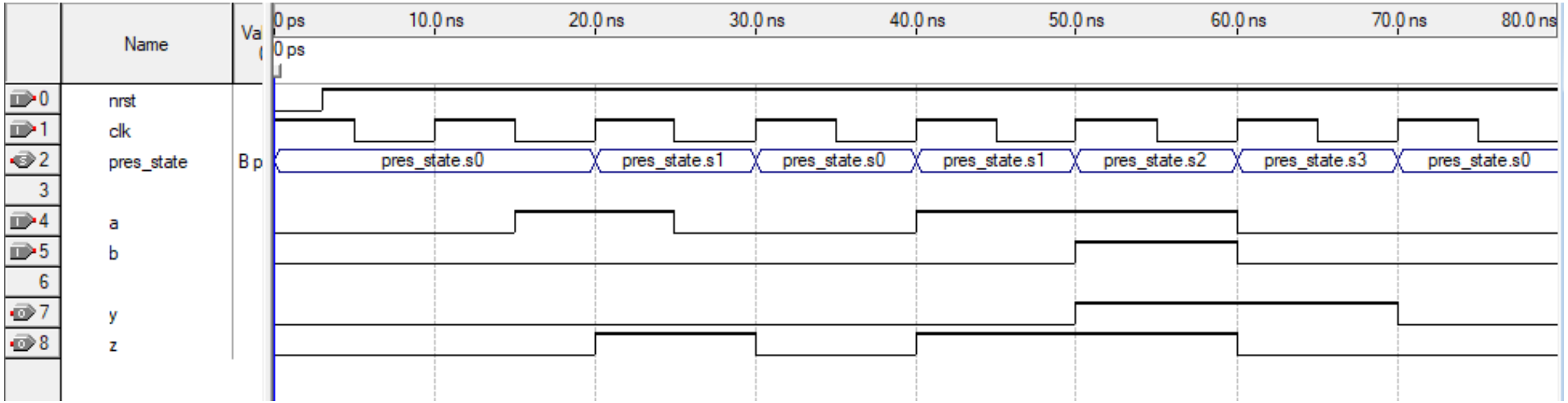
- **Quartus II** menu Tools → Netlist Viewers → State Machine Viewer:



	Source State	Destination State	Condition
1	s0	s0	(!a)
2	s0	s1	(a)
3	s1	s0	(!a)
4	s1	s2	(b).(a)
5	s1	s3	(!b).(a)
6	s2	s3	
7	s3	s0	

Máquina de estados finitos
Exemplo

- **Exemplo fsm_1** – Resultado da simulação funcional:

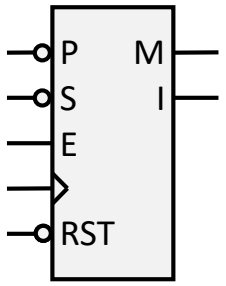


Máquina de estados finitos
Exercício

- **Exercício fsm_2:**
- **Desenhar o diagrama de transição de estados** de um máquina de estados finitos que controla um sistema de reprodução de áudio descrito a seguir.
- **Escrever o código em VHDL** para essa máquina de estados, usando dois PROCESS, um sequencial para a memória de estado e outro combinacional para as lógicas do próximo estado e de saída.
- **Testar o projeto** no módulo DE2

Máquina de estados finitos
Exercício

- **Exercício fsm_2 – especificações:**
- Esse sistema tem dois botões: PLAY e STOP. Esses botões, quando pressionados, geram um nível lógico '0' e, quando liberados, voltam para a posição original por efeito de um mola, gerando o nível lógico '1'. Existe também um sinal END, ativo em nível lógico alto, que indica que o áudio chegou ao final.
- O sistema é controlado através de duas saídas, uma para ativar a reprodução e outra para voltar a gravação para o início. Esse último não pode ser ativado durante a reprodução nem durante a pausa.



Entradas:

nrst	reset assíncrono ativo baixo
clk	clock borda de subida
p	PLAY ativo baixo
s	STOP ativo baixo
e	END ativo alto

Saídas:

m	ativa reprodução ativo alto
i	volta para início ativo alto

Máquina de estados finitos
Exercício

- **Exercício fsm_2 – especificações (continuação):**

- O estado inicial do sistema, após a energização, é parado. Estando o botão de STOP liberado, a reprodução é iniciada quando o botão PLAY é pressionado. Manter o botão PLAY pressionado depois do início da reprodução não faz efeito algum. Se o botão PLAY for liberado, a reprodução continua normalmente.
- Estando reproduzindo, pressionar o botão PLAY novamente faz a reprodução pausar. Manter o botão PLAY pressionado não faz efeito algum. Se o botão PLAY for liberado, o sistema continua pausado.
- Estando em pausa, pressionar o botão PLAY novamente faz a reprodução retornar a partir do ponto onde foi pausada. Manter o botão PLAY pressionado não faz efeito algum. Se o botão PLAY for liberado, a reprodução continua normalmente.
- Se, durante a reprodução o sinal END for ativado ou, em qualquer situação, o botão STOP for pressionado, o sistema volta para o estado inicial. Se isso acontecer com o botão PLAY pressionado, ele deve ser liberado antes de um novo comando de PLAY.

Fim