Relatório da atividade de maquina de estados

Ferdando Farias Dimas[1], **Frank B. Ferreira Boa Morte**[1], **Mercedes M. B. Diniz**[1]

[1]Instituto de Tecnologia - ITEC - Universidade Federal do Pará (UFPA)
Caixa Postal 479 - 66075-110 - Belém - PA - Brazil

fernando.dimas@itec.ufpa.br, frank.morte@itec.ufpa.br

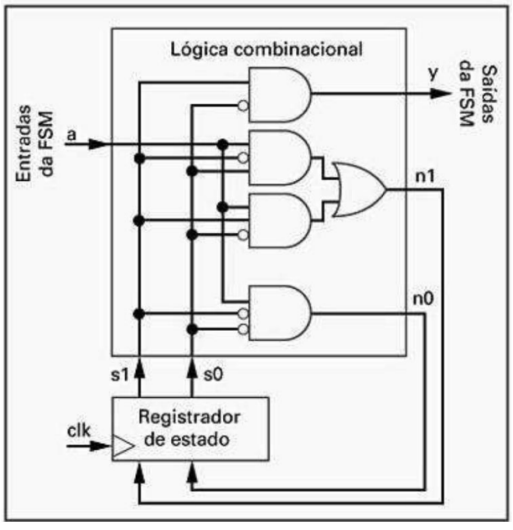
mercedes.diniz@itec.ufpa.br

Abstract. *This report has evaluative purposes for the curricular component Hardware design and interfacing and corresponds to activity on sequential logic circuits and state machines, with the objective of implementing them in the Quartus IDE*

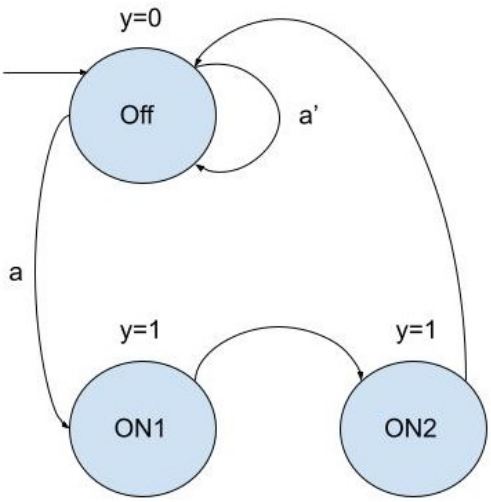
Resumo. *Este relatório tem fins avaliativos para a componente curricular Projeto de hardware e interfaceamento e corresponde a atividade sobre circuitos lógicos sequenciais e máquinas de estados, tendo como objetivo a implementação dos mesmos na IDE do Quartus. [Pedroni 2010][AB]*

1. Engenharia reversa do circuito

O objetivo da atividade é fazer a engenharia reversa do comportamento do circuito sequencial abaixo (a).



(a) Circuito Sequencial



(b) Maquina de Estados

A o circuito representa uma máquina de estados com circuitos lógicos combinacionais e sequenciais. Tendo uma entrada 'a' e uma saída 'y', registrador de 2 bits para formar 4 estados e os sinais n0 e n1 para o próximo estado. Cada estado tem uma sequência binária única que foi atribuída. Serão utilizado apenas 3 dos 4 estados disponíveis, ou seja, um estado será inutilizado.

	Imput			Output		
	s1	s0	a	y	n1	n0
OFF	0	0	0	0	0	0
	0	0	1	0	0	1
ON1	0	1	0	0	0	0
	0	1	1	0	1	0
ON2	1	0	0	1	0	0
	1	0	1	1	1	0
ON3	1	1	0	0	0	0
	1	1	1	0	0	0

Figura 2. Tabela de estado

O circuito possui a seguinte lógica:

$$y = s1 + s0'$$
n1= as1's0 + as1s0'
n0= as1's0'

A tabela representa todos os estados da máquina, incluindo estado não utilizado, sendo ele o estado ON3. Cada estado tem uma codificação única, sendo eles:

$$Off = 00$$
$$On1 = 01$$
$$On2 = 10$$

2. Projeto de máquina de estados - Relógio de Pulso.

Contexto: O mostrador de um relógio de pulso pode fornecer uma de quatro informações: hora atual, alarme, cronometro e data, que são controladas por duas saídas do sistema digital, $s0$ e $s1$. Assume que $s0$ e $s1$ controlam um multiplexador que seleciona uma das quatro informações que vêm de um fonte externa ao sistema digital projetado. Quando um botão B é pressionado, o próximo item da sequência é exibido. Crie um diagrama de estados que descreva o comportamento. Assegure-se que ocorrerá o avanço de apenas um item, independente de quanto tempo o botão permanece pressionado, ou seja, após avançar para o próximo item da sequência, deve-se aguardar o botão ser solto. Use nomes curtos mas descritivos para representar cada estado. Faça com que a exibição da hora atual seja o estado inicial.

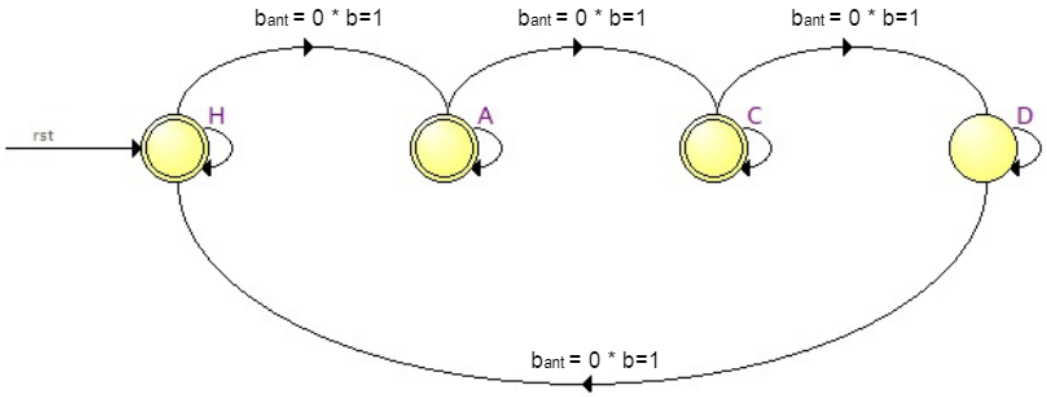


Figura 3. Maquina de Estados Gerada

```
logica_proximo_estado : process(estado_atual, b, b_anterior)
begin
case estado_atual is
  when H =>
    if b_anterior = '0' and b = '1'then
      prox_estado <= A;
    else
      prox_estado <= H;
    end if;
  When A =>
    if b_anterior = '0' and b = '1' then
      prox_estado <= C;
    else
      prox_estado <= A;
    end if;
  When C =>
    if b_anterior = '0' and b = '1' then
      prox_estado <= D;
    else
      prox_estado <= C;
    end if;
  When D =>
    if b_anterior = '0' and b = '1' then
      prox_estado <= H;
    else
      prox_estado <= D;
    end if;
end case;
end process
```

Listing 1. Logica da transição de estados

A maquina de estados tem como estado inicial H (hora) e permanece nele enquanto o botão $b = 0$. Se $b = 1$ e $bant = 0$ (estado anterior de b) então passa para o estado A (Alarme) e permanece enquanto $b = 0$. Se $b = 1$ e $bant = 0$ então passa para o estado C (Cromômetro), permanecendo enquanto $b = 0$. Se $b = 1$ e $bant = 0$ então passa para o estado D (Data), permanecendo enquanto $b = 0$. Se Se $b = 1$ e $bant = 0$ retorna para o estado inicial.

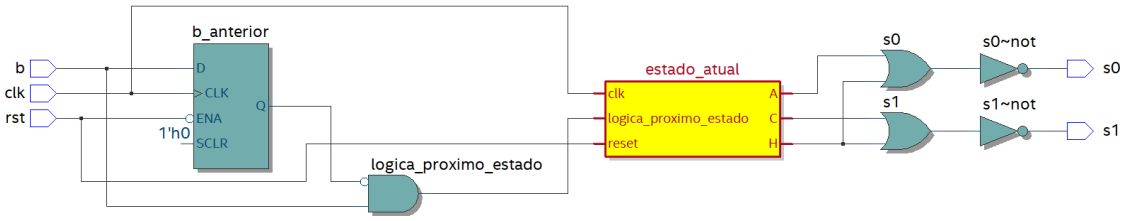


Figura 4. RTL

2.1. Testbench

A simulação dos estados ocorre em função da borda de subida do clock, sendo feita a leitura das entradas e definindo estados para a saída. Entretanto como a atribuição dos valores de entrada também é feita a partir do clock, a leitura considera os valores antes do pulso causando um pequeno atraso na mudança de estado, como é exibido nas Figuras 5 e 6.

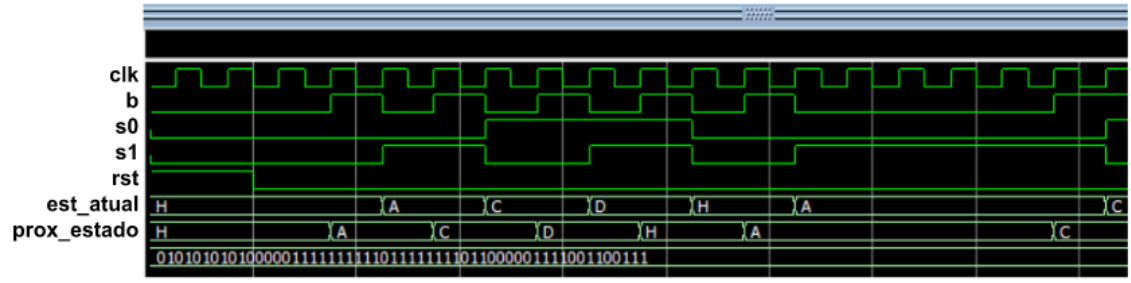


Figura 5. Acionamento do botão em intervalos curtos

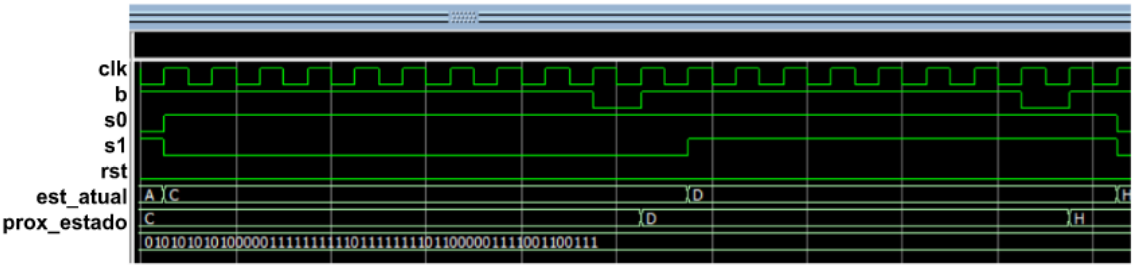


Figura 6. Acionamento do botão em intervalos longos

3. Projeto de máquina de estados - Portão Automático.

Contexto: Deseja-se projetar uma máquina de estados que controla os processos de abertura e fechamento de um portão. A máquina obedece a um único comando de um botão B. Se o botão é pressionado uma vez, o portão fecha, se é pressionado novamente, o portão abre. Nos processos de abertura ou fechamento, se houver algo no percurso do portão, indicado pelo sensor de presença P, o portão para o seu movimento, e retoma quando não houver mais nada no seu percurso. Há ainda dois sinais que indicam os fins de curso, ou fim do processo de abertura e de fechamento, FC1 e FC2, respectivamente, que indicam

à máquina de estados que é possível finalizar o processo. A máquina de estados gera dois sinais de saída S0 e S1 que controlam os processos de abertura e de fechamento, respectivamente. A inicialização do sistema é feita considerando que o portão está fechado. Entretanto, pode ocorrer por exemplo uma situação de exceção, como uma falta de energia, logo, não deve-se assumir que o portão está fechado, ou seja, deve-se, após a inicialização fechar o portão para deixar o sistema em um estado conhecido.

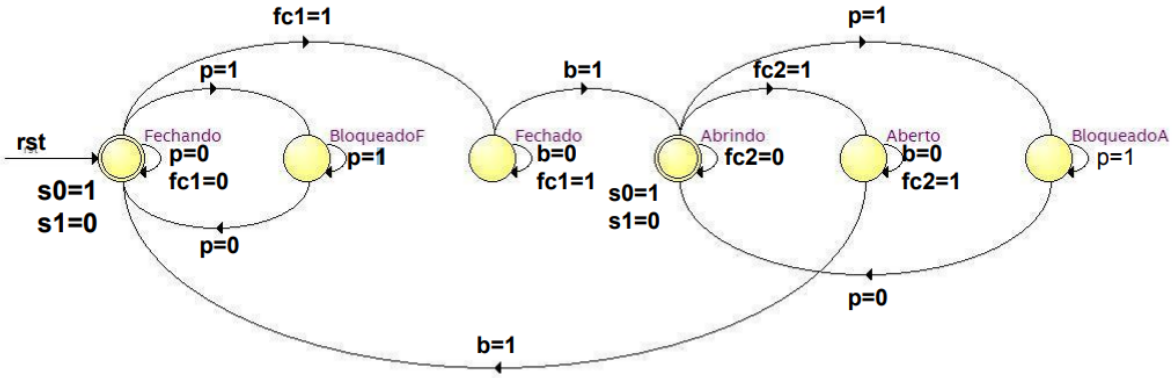


Figura 7. Maquina de Estados Gerada

A máquina de estados se inicia com o estado $Fechando$, caso p seja 1 então passa para o estado $BloqueadoF$, que significa que existe algo impedindo o fechamento do portão, permanece-se nesse estado enquanto p for igual a 1, caso p volte a ser zero então retorna-se ao estado de $Fechando$. Quando $fc1$ for igual a 1 a máquina segue para o próximo estado, sendo ele $fechado$, se o botão não for pressionado e $fc1$ continuar sendo 1 a máquina permanece no estado $fechado$. Quando pressionarmos o botão e b for igual a 1 segue-se para o estado $Abrindo$ onde a máquina permanecerá enquanto $fc2$ for igual a 0. Caso tenhamos algo impedindo o fechamento do portão, ou seja se p for igual a 1, a máquina segue para o estado $BloqueadoA$ e lá permanece enquanto p for igual a 1, caso p volte a ser 0 então retorna-se ao estado $Abrindo$. No momento que a máquina está no mesmo estado e $fc2$ for igual a 1 então a máquina segue para o estado $Aberto$ e lá permanece enquanto b for igual a 0 e $fc2$ for igual a 1. Finalizando seu ciclo, caso o estado seja o mesmo e o botão for pressionado a máquina volta ao estado $Fechando$.

```
logica_proximo_estado : process(estado_atual, b, b_anterior, p, fc1,
    fc2)
begin
  case estado_atual is
    when Fechando =>
      if  p = '1' and fc1 = '0'then
        prox_estado <= BloqueadoF;
      elsif fc1 = '1' then
        prox_estado <= Fechado;
      else
        prox_estado <= Fechando;
      end if;
```

```
    when BloqueadoF =>
      if p = '0' then
        prox_estado <= Fechando;
      else
        prox_estado <= BloqueadoF;
      end if;

    when Fechado =>
      if b = '1' and b_anterior = '0' and fc1 = '0'then
        prox_estado <= Abrindo;
      else
        prox_estado <= Fechado;
      end if;

    when Abrindo =>
       if p = '1' and fc2 = '0' then
        prox_estado <= BloqueadoA;
      elsif fc2 = '1' then
        prox_estado <= Aberto;
      else
        prox_estado <= Abrindo;
      end if;

    when BloqueadoA =>
      if p = '0' then
        prox_estado <= Abrindo;
      else
        prox_estado <= BloqueadoA;
      end if;

    when Aberto =>
      if b = '1' and b_anterior = '0' and fc2 = '0'then
        prox_estado <= Fechando;
      else
        prox_estado <= Aberto;
      end if;
  end case;
end process;
```

Listing 2. Logica da transição de estados

3.1. Testbench

Assim como relatado no projeto do Relógio de Pulso, neste projeto também há um atraso na mudança de estado da saída em função de que a leitura e a atribuição das entradas é feita na borda de subida do clock. As simulações são exibidas nas Figuras 8 e 9.

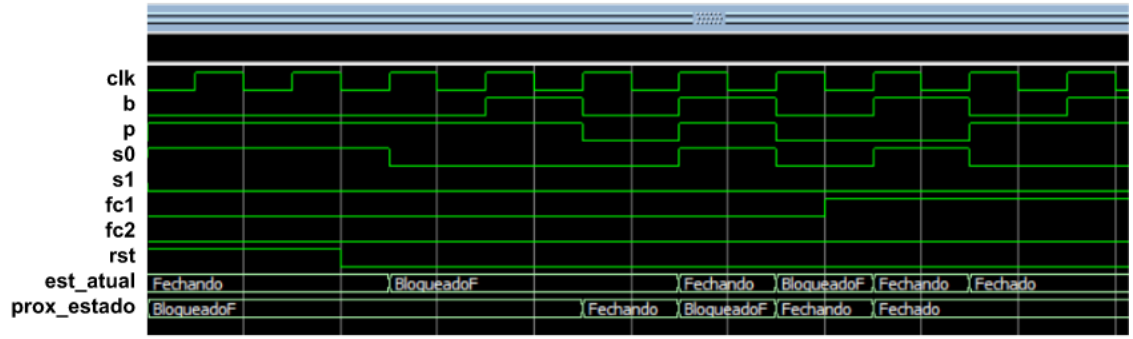


Figura 8. Fechando Portão

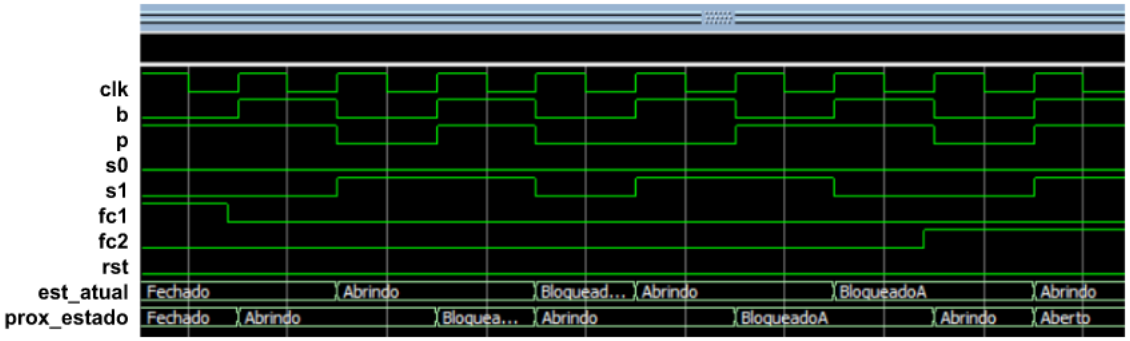


Figura 9. Abrindo Portão

Referências

AB, H. E. Vhdl handbook.

Pedroni, V. A. (2010). *Eletrônica digital moderna e VHDL*. Elsevier.