UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

CMP238
Projeto e Teste de Sistemas VLSI

Linguagem de Descrição de Hardware VHDL

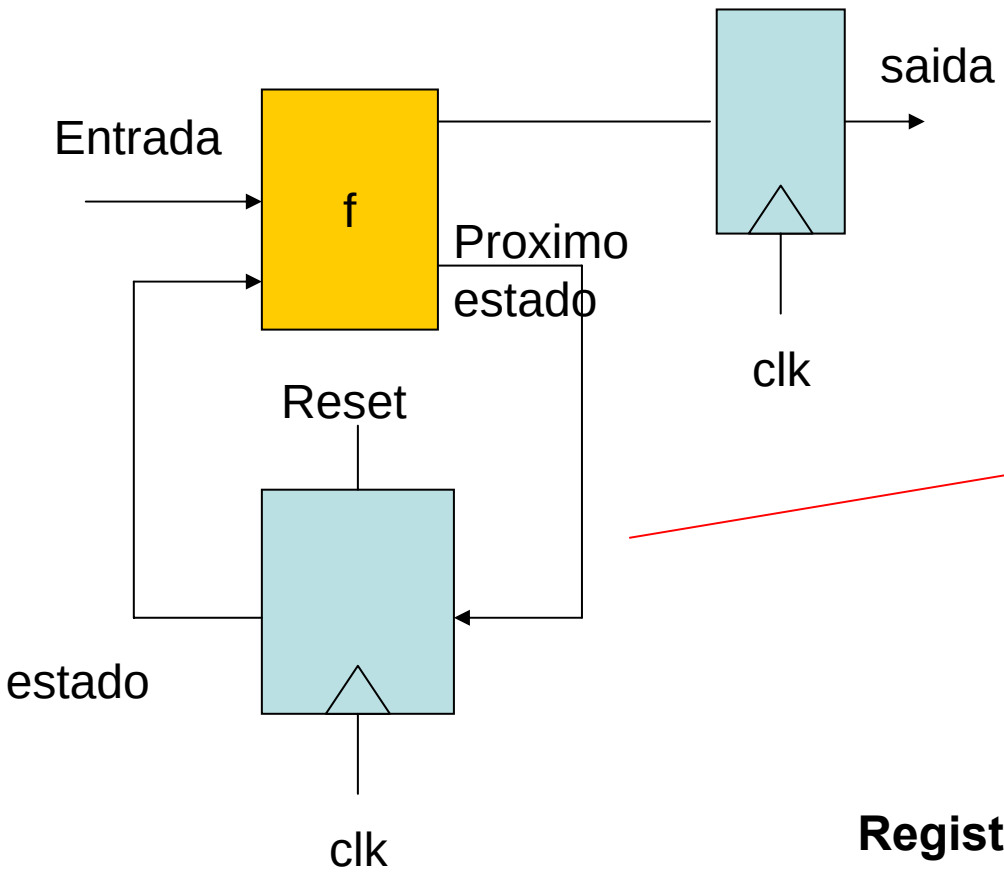
Prof. Fernanda Gusmão de Lima Kastensmidt

fglima@inf.ufrgs.br

Maquinas de Estados

Maquinas de Estado

Tipo Melay and Moore



LOGIC + Register

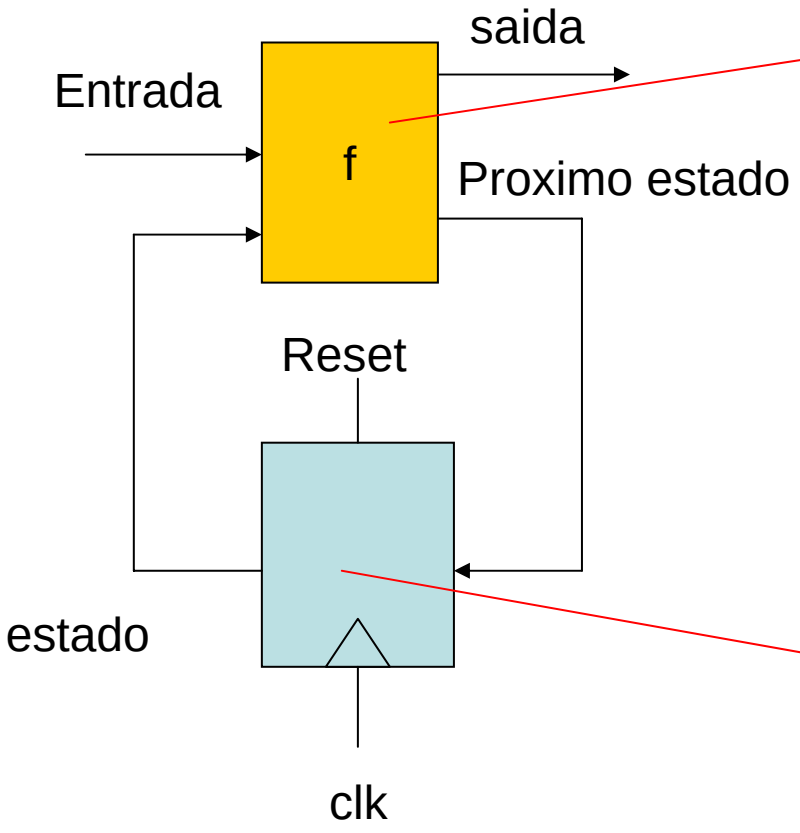
```
Process(clk, reset)
Begin

CASE entrada&estado IS
......
WHEN ... =>
estado <= Proximo estado ;
Saida <= .....;
.....
End process;
```

Registradores: saida e estado

Maquinas de Estado

Tipo Melay e Moore



LOGIC

```
Process(entrada, estado)
Begin
.....
Saida <= ...;
Proximo estado <= ....;
.....
End process;
```

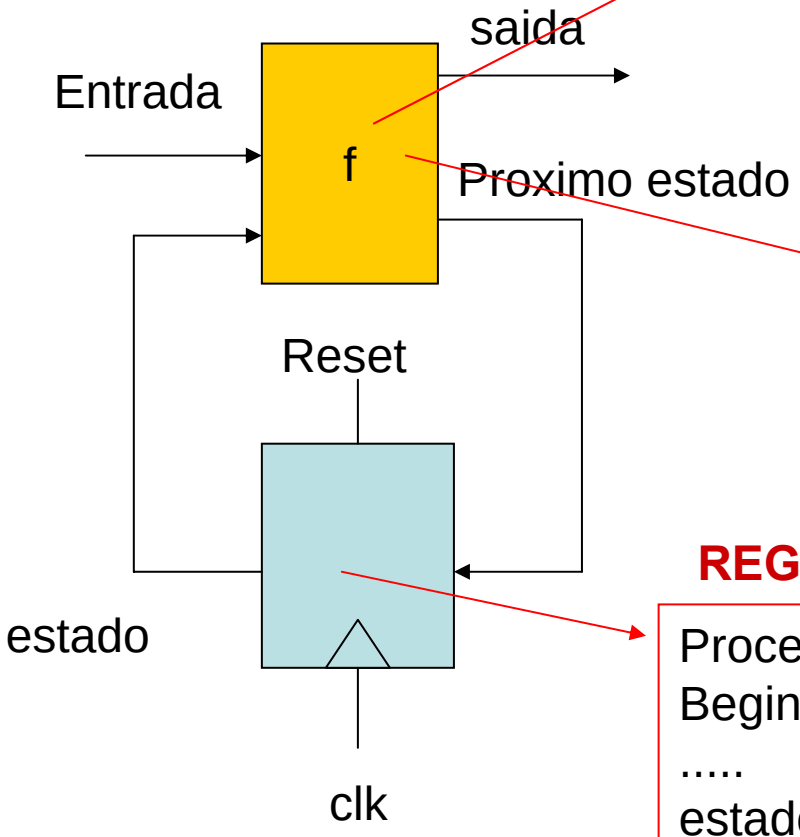
REGISTER

```
Process(clk, reset)
Begin
.....
estado <= Proximo estado ;
.....
End process;
```

Registradores: estado

Maquinas de Estado

Tipo Moore



Registradores: estado

LOGIC

```
Process(entrada, estado)
Begin
.....
Proximo estado <= ....;
.....
End process;
```

LOGIC

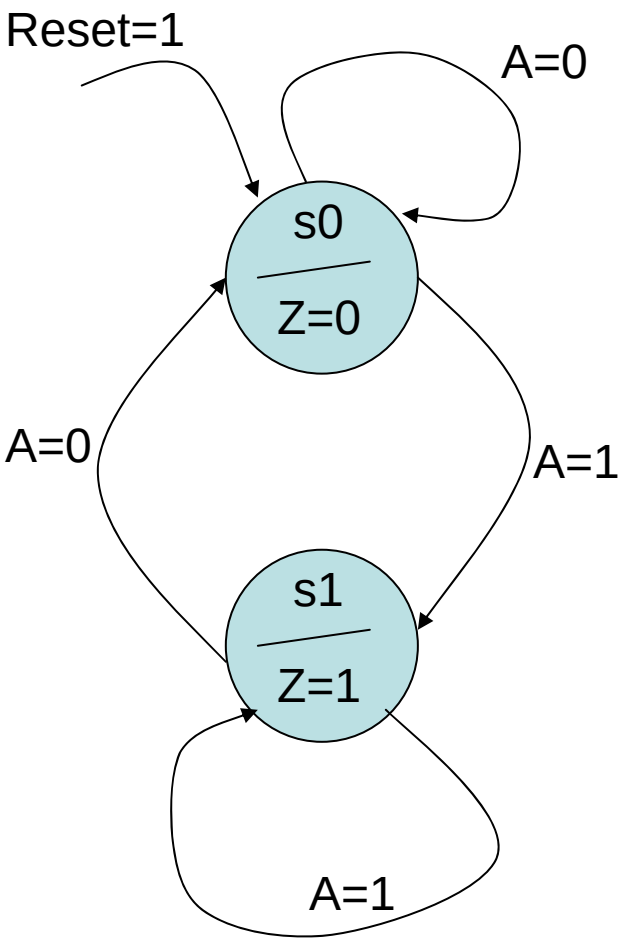
```
Process(estado)
Begin
.....
saida <= ....;
.....
End process;
```

REGISTER

```
Process(clk, reset)
Begin
.....
estado <= Proximo estado ;
.....
End process;
```

Maquinas (

Exemplo 1: Moore



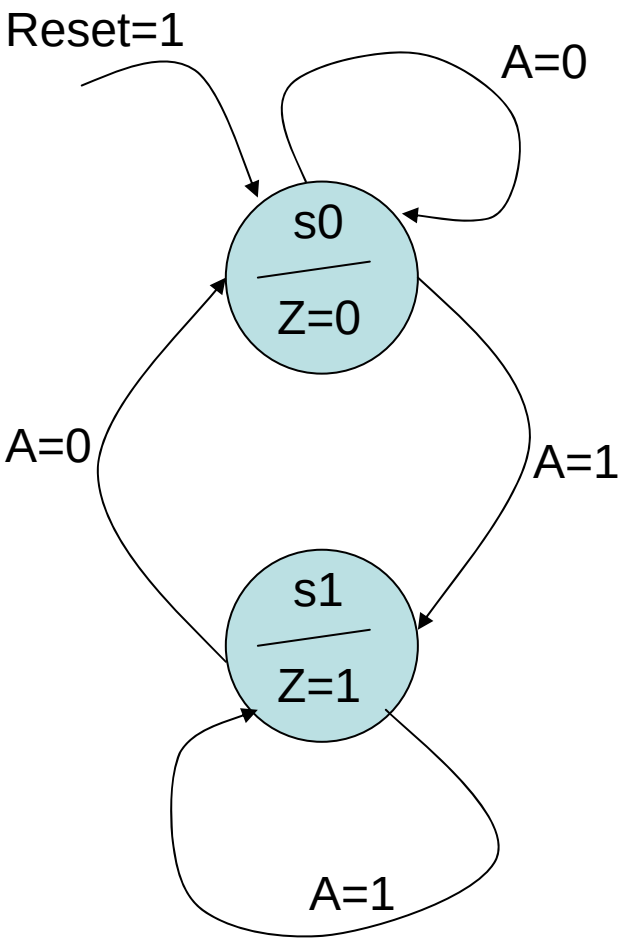

```
Process(clk, reset)
Begin
If reset='1' then
   estado <= s0;
Elsif (clk'event and clk='1') then
   estado <= prox_estado;
End if;
End process;

Process(A, estado)
Begin
CASE estado is
When s0 => if A='0' then prox_estado <= s0;
                    else prox_estado <= s1;
When s1 => if A='0' then prox_estado <= s0;
                    else prox_estado <= s1;
END CASE;
End process;

Process(estado)
Begin
If estado = s0 then
      Z<=0;
Else   Z<=1;
End if;
End process;
```

Maquinas de Estado

Exemplo 1: Moore

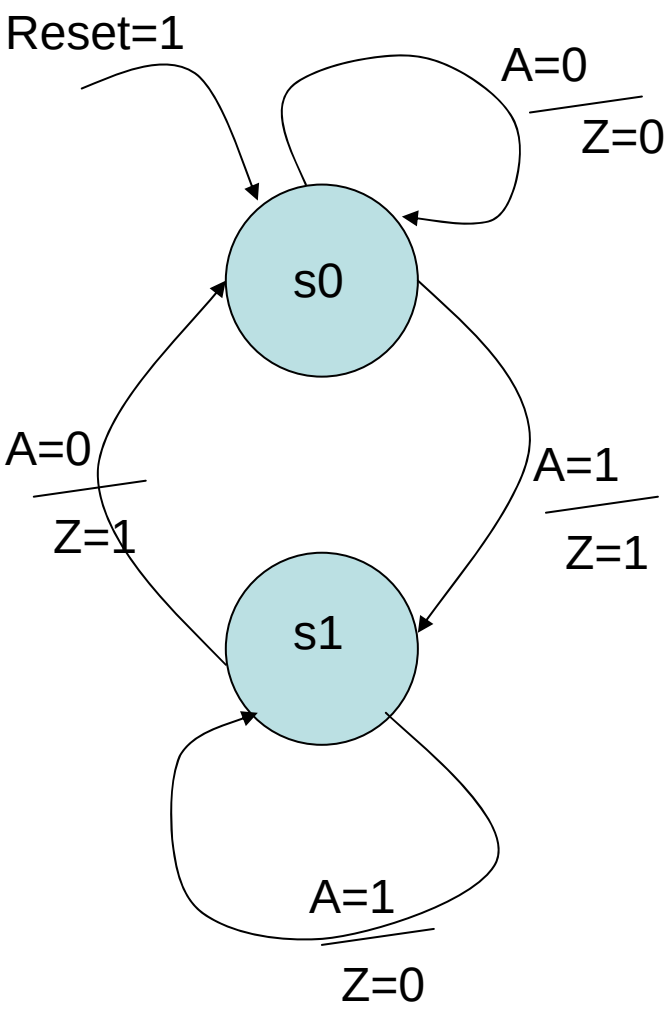



```
Process(clk, reset)
Begin
If reset='1' then
   estado <= s0;
Elsif (clk'event and clk='1') then
   estado <= prox_estado;
End if;
End process;

Process(A, estado)
Begin
CASE estado is
When s0 => Z<='0';
                if A='0' then prox_estado <= s0;
                else prox_estado <= s1;
When s1 => Z<='1';
                if A='0' then prox_estado <= s0;
                else prox_estado <= s1;
END CASE;
End process;
```

Maquinas de Estado

Exemplo 2: Mealy

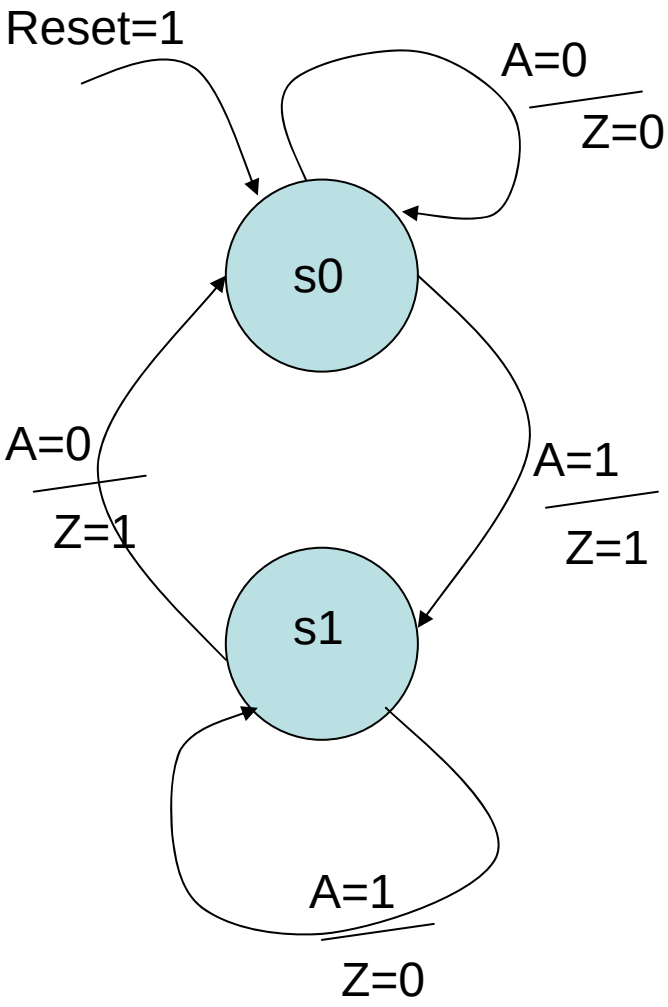



```
Process(clk, reset)
Begin
If reset='1' then
   estado <= s0;
Elsif (clk'event and clk='1') then
   estado <= prox_estado;
End if;
End process;

Process(A, estado)
Begin
CASE estado is
When s0 => if A='0' then prox_estado <= s0; Z<='0';
                    else prox_estado <= s1; Z<='1';
When s1 => if A='0' then prox_estado <= s0; Z<='1';
                    else prox_estado <= s1; Z<='0';
END CASE;
End process;
```

Maquinas de Estado

Exemplo 3: Mealy




```
Process(clk, reset)
Begin
If reset='1' then
   estado <= s0;
Elsif (clk'event and clk='1') then
CASE estado is
When s0 => if A='0' then estado <= s0; Z<='0';
                    else estado <= s1; Z<='1';
When s1 => if A='0' then estado <= s0; Z<='1';
                    else estado <= s1; Z<='0';
END CASE;
End process;
```

Codificação FSM

- **1) Não defina previamente os estados com uma codificação em binário, e sim deixe o nome do estado textual**

```
type T_STATE is (RESET, START,
   EXECUTE, FINISH);

signal estado, prox_estado : T_STATE ;
```

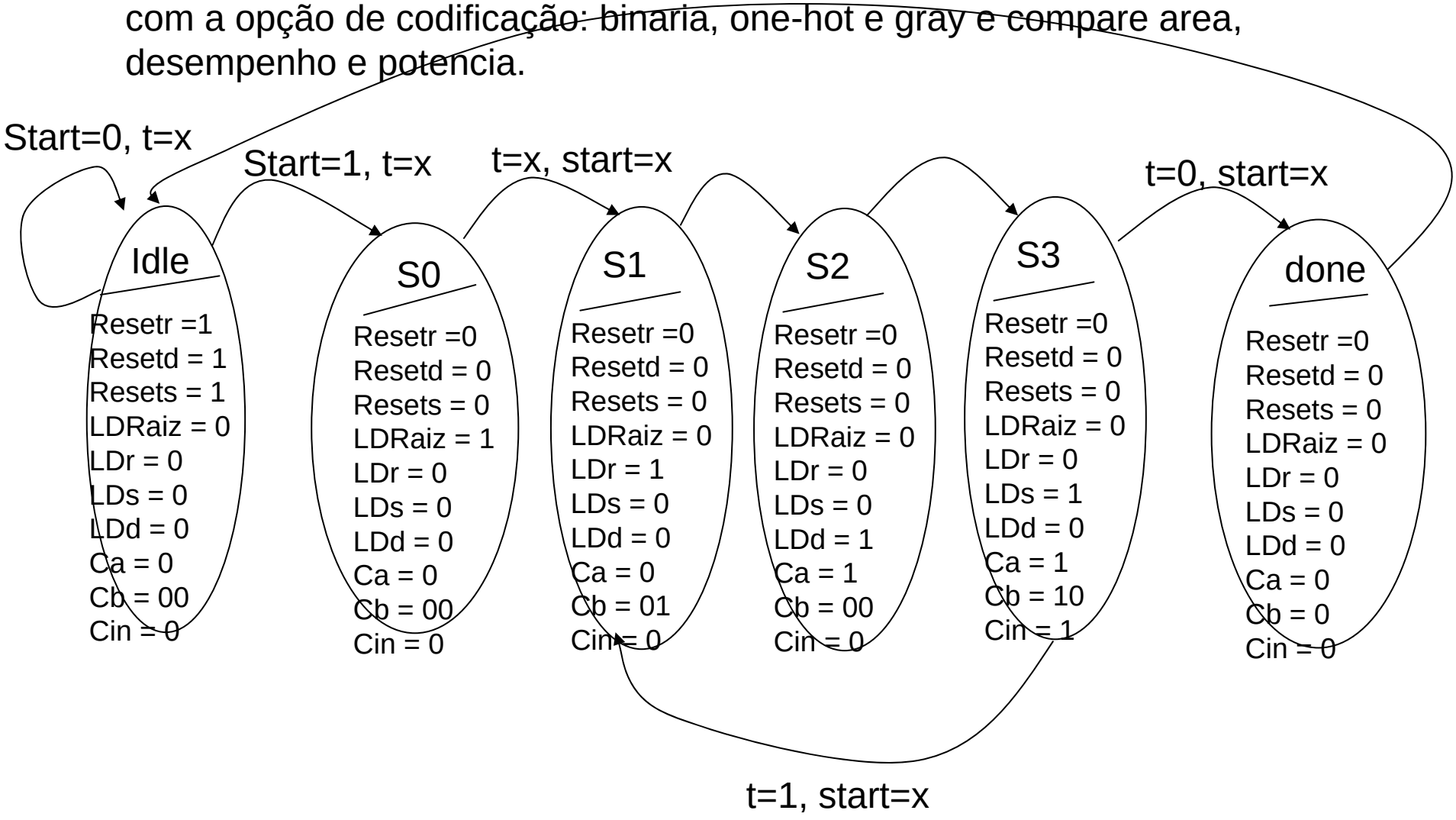
- **2) Deixe a ferramenta codificar o estado**

Codificação de Estados

- Binária:
  - Usa um baixo numero de bits para armazenar o estado. Ou seja, n bits de armazenamento representa 2^n estados.
  - Pode ser na ordem crescente ou não
  - Existem algoritmos para determinar a melhor ordem de codificação de estados para minimizar área e maximizar o desempenho.
- One-hot
  - Cada estado tem o seu bit na decodificação, logo uma máquina de n estados vai precisar de n bits para armazenar o estado.
  - Apenas um bit é setado por vez.
  - Apresenta um baixo consumo de potencia na transíção de estados pois apenas dois bits são

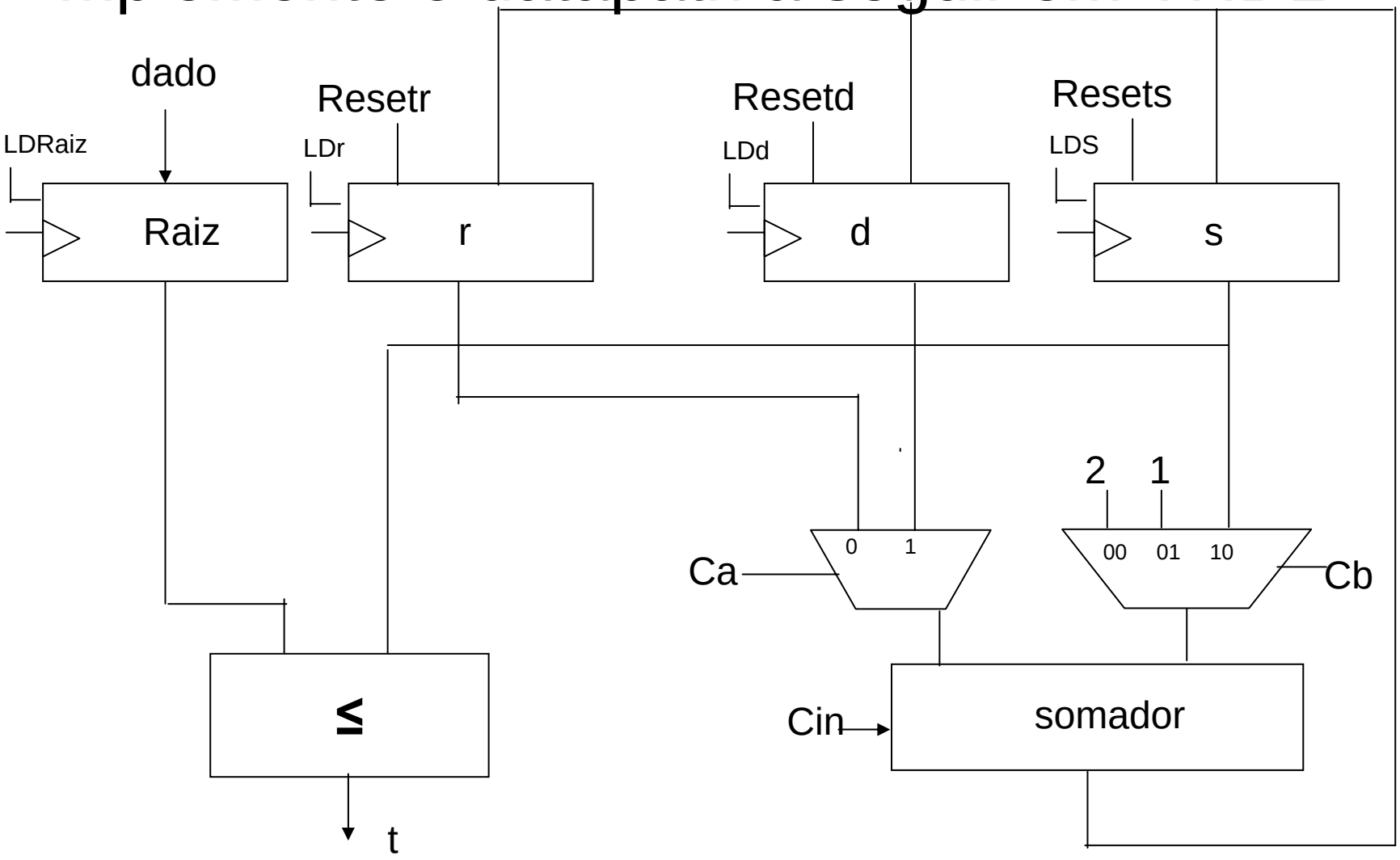
Exercicio 1:

- Represente a maquina de estados a seguir em VHDL e sintetize no ISE com a opção de codificação: binaria, one-hot e gray e compare area, desempenho e potencia.



Exercicio 2:

- Implemente o datapath a seguir em VHDL



Exercicio 3:

- Você acaba de implementar o controle (a maquina de estados) e a parte operativa (datapath) do extrator de raiz quadrada.

- Então una os dois modulos em um modulo de VHDL topo, e simule o extrator de raiz quadrata.

Exercício 4:

- Lembrando que o algoritmo da extração de raiz quadrada é:

```
Ler X
-- inicialização
r=1; d=2; s=4;
Enquanto (t==1) {
r=r+1;
d=d+2;
s=s+d+1;
t=ac(s,x);
}
devolve(r);
}
ac(a,b)
semsinal a, b;
{
Se (a<=b) retorna (1); Senão retorna (0);
}
```

Tente implementar em VHDL o algoritmo usando apenas um process

Conclusão

- Você acaba de implemetar de duas maneiras diferentes o mesmo circuito em VHDL.

- A primeira aboradagem é chamada de descrição RTL (Register Transfer Level), ou seja, a parte de controle e a parte operativa com os registradores esta toda muito bem definida.