

ATV 2

Nome: Amorim Brinio Rodrigues Dos Santos

- 01) Ao usar operadores lógicos, é necessário antes planejar quais operadores serão usados e ~~o que~~ em que ponto do Código esses operadores serão usados ~~e~~ de forma que o Código final realize o que desejamos. Típicamente a atribuição contínua de variáveis com operadores condicionais, a ~~sintaxe~~ sintaxe não é feita ~~automaticamente~~ pelo próprio HDL na modelagem HDL, evitando assim, projetos de mão, diferentes do uso de operadores lógicos, que exigem esse projeto.

02) Reinicialização síncrona

Aplicação de alguns bits estes em normal, ou seja, tudo em normal e aplicação, ou seja, tudo em inverso

module sincrona(input wire Clk, d, reset, output reg (2); } estabelecendo quem são os inputs e outputs do Código

```

always @ (posedge Clk) ← Clk se, só podemos mudar o
begin                                output apenas nos bordos positivos da
    if (reset) } clock, chamado de Clk, ou seja, quando
        begin                                     Clk está indo de 0 para 1
            a <= 1'b0; } → Esse if só irá rodar o Código que
        end                                         está dentro quando reset = 1, e
    else }                                       só com que o output volte para o
        begin                                     estado inicial, que é 0.
            a <= 1'b1; } → O Código dentro desse else só irá
        end                                         rodar quando reset = 0 e só com que
    end                                         o output seja sempre 1.
endmodule

```

Reiniciando o contador



module sincrono(input wire Clk, d, reset, a } estabelecendo os inputs
output reg a); } e outputs

always @ (posedge Clk, posedge reset) ← isso faz com que
begin a = 0 quando Clk mudar
if (reset) de 0 para 1 e reset = 0
begin e a = 0 quando reset = 1 e
a <= 1'b0; Clk mudar de 0 para 1 ou
end quando reset muda de 0
a <= 1'b0; para 1.
end } Sempre que reset for 1, a = 0
else begin } Sempre que reset for 0, a = 1
begin a <= 1'b1;
end
end
endmodule

03) Registro temporário

↓

~~temporário~~

always @ (posedge CLK) < Chi se, não trocamos os conteúdos dos registradores no bordo positivo do clock

begin

~~registers~~

regint * = regP;

regP = regRS;

regRS = regT;

end

O registro temporário ~~recebe~~ recebe o conteúdo do primeiro registro e o primeiro registro recebe o conteúdo do segundo registro, por fim, o segundo registro recebe o conteúdo do registro temporário, que é o conteúdo do 1º registro, realizando assim a troca

Sem registro temporário

↓

always @ (posedge CLK)

begin

regP <= regRS

regRS <= regP

end

temporário para auxiliar essa troca.

Neste ~~bloco~~ bloco, usamos atribuições sem bloqueio para que a troca do conteúdo seja feita ao final do código, para que o primeiro registro receba o conteúdo do segundo registro ao mesmo tempo em que o segundo registro recebe o conteúdo do primeiro, sem a necessidade de um registro temporário para auxiliar essa troca.

05) Combinacional, pois o bloco de lógica de estabelecido próximo depende apenas dos entradas atuais ou não, não depende das entradas anteriores.
E como a lógica sequencial depende das entradas anteriores, então é a opção falsa, restando apenas a opção da lógica Combinacional, que depende apenas das entradas atuais.

06) Como a lógica sequencial depende das entradas passadas e atuais para gerar o output atual, então a sequência de entradas irá influenciar o output. Já na lógica Combinacional, o output depende apenas da entrada atual, então a sequência de entradas não irá influenciar o output.

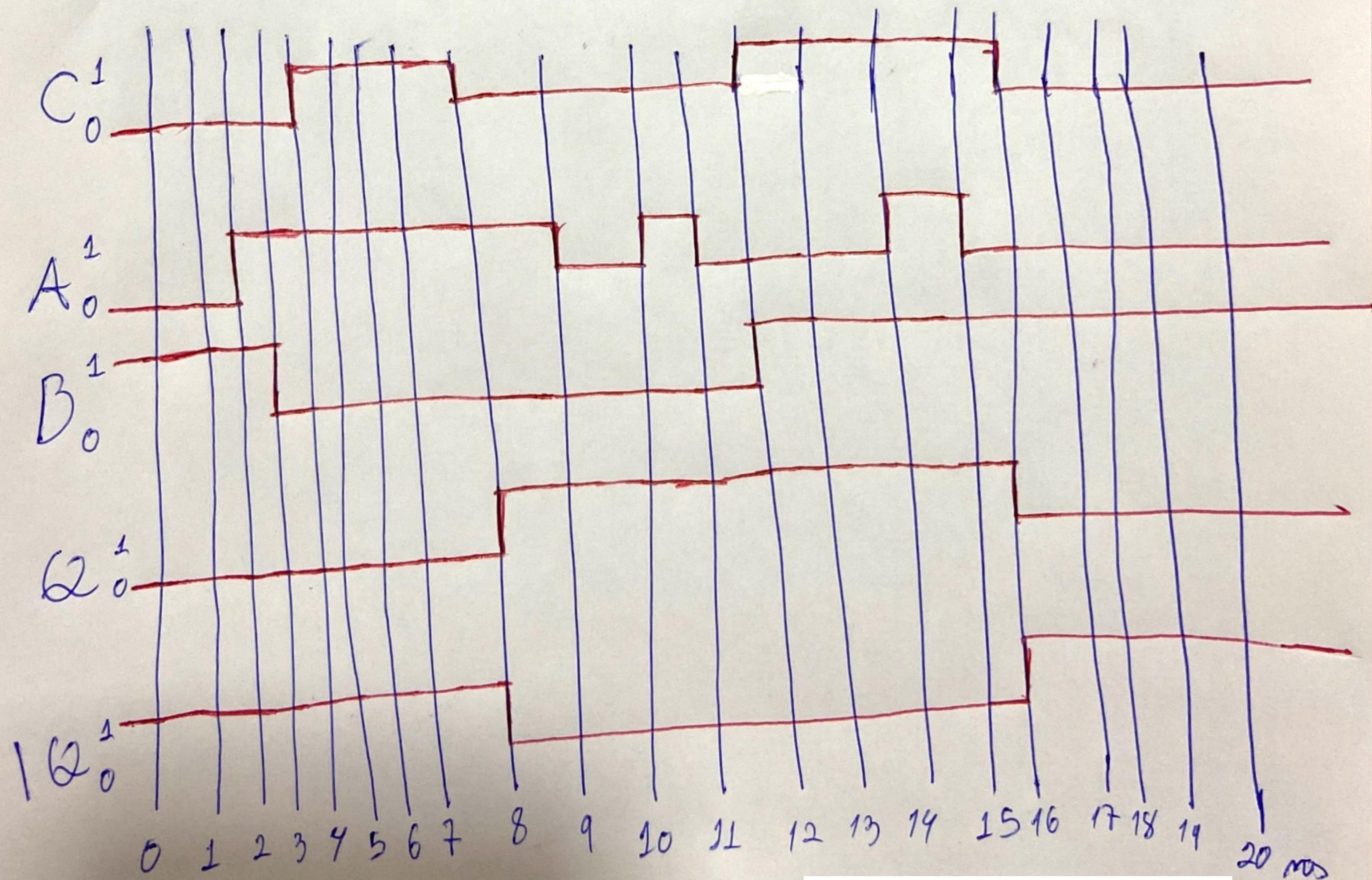
No Coxo I (Telefone), nota-se que ~~é~~ o output de um input também depende dos inputs anteriores, pois Coxo um dos inputs seja ~~seja~~ diferente ou Coxo a sequência dos inputs seja diferente automaticamente o output será diferente, portanto o Coxo I utiliza lógica sequencial (S)

To no Coxo II (Codebook), nota-se que temos 3 inputs, sendo Cada input uma das rotas com números do Codebook da imagem, e que ~~é~~ a rota atual depende apenas dos inputs ~~atual~~ atuais, visto que para o Codebook abrir é necessário por os 3 inputs certos sem precisar de precisar com os inputs anteriores, logo o Coxo II irá lógica Combinacional.

To no Coxo III (Segredo de Cofre), nota-se que temos apenas 1 input e que para gerar o ~~out~~ output precisamos da sequência de inputs para saber se é a sequência correta (o que gera true o output do cofre aberto) ou se é a sequência errada (o que gera false o output do cofre não aberto), desse modo, nota-se que o Coxo III usa lógica sequencial ^(S).

Portanto a resposta dera questão é S, C, S.

07) Sej um flip-flop, pos note-se que temos um Clock ~~que~~
 fala. Com que o flip-flop só muda quando houver bordo no Clock e
 note-se também que há 2 flip-flops do tipo D que não formam um flip-flop do
 tipo JK (Mestre-exceção) onde $J = A$ e $K = \bar{A}$, onde ~~que~~
 a saída do 1º flip-flop D é $B_m = \bar{B}$ e a saída do 2º flip-flop D, é $B_F = B$



10) É uma máquina do tipo Moore, pois a entrada não influencia a saída de forma direta.

~~$$\text{Ecuación Característica: } Q^* = JQ' + K'Q$$~~

Portanto a ¹ Ecuación de Estado e transição:

$$\begin{array}{l} \left\{ \begin{array}{l} Q_1^* = JQ_1' + K'Q_1 \\ Q_0^* = JQ_0' + K'Q_0 \end{array} \right. \\ \text{Logo } Q_0^* = XQ_0' + X'Q_0 \quad \left| \begin{array}{l} Q_1^* = JQ_1' + K'Q_1 \\ \text{Logo } Q_1^* = Q_0Q_1' + (XQ_0)'Q_1 \end{array} \right. \end{array}$$

Assim a equação de saída é:

$$Z = (Q_0 \oplus Q_1)$$

Tabela de transição e saída:

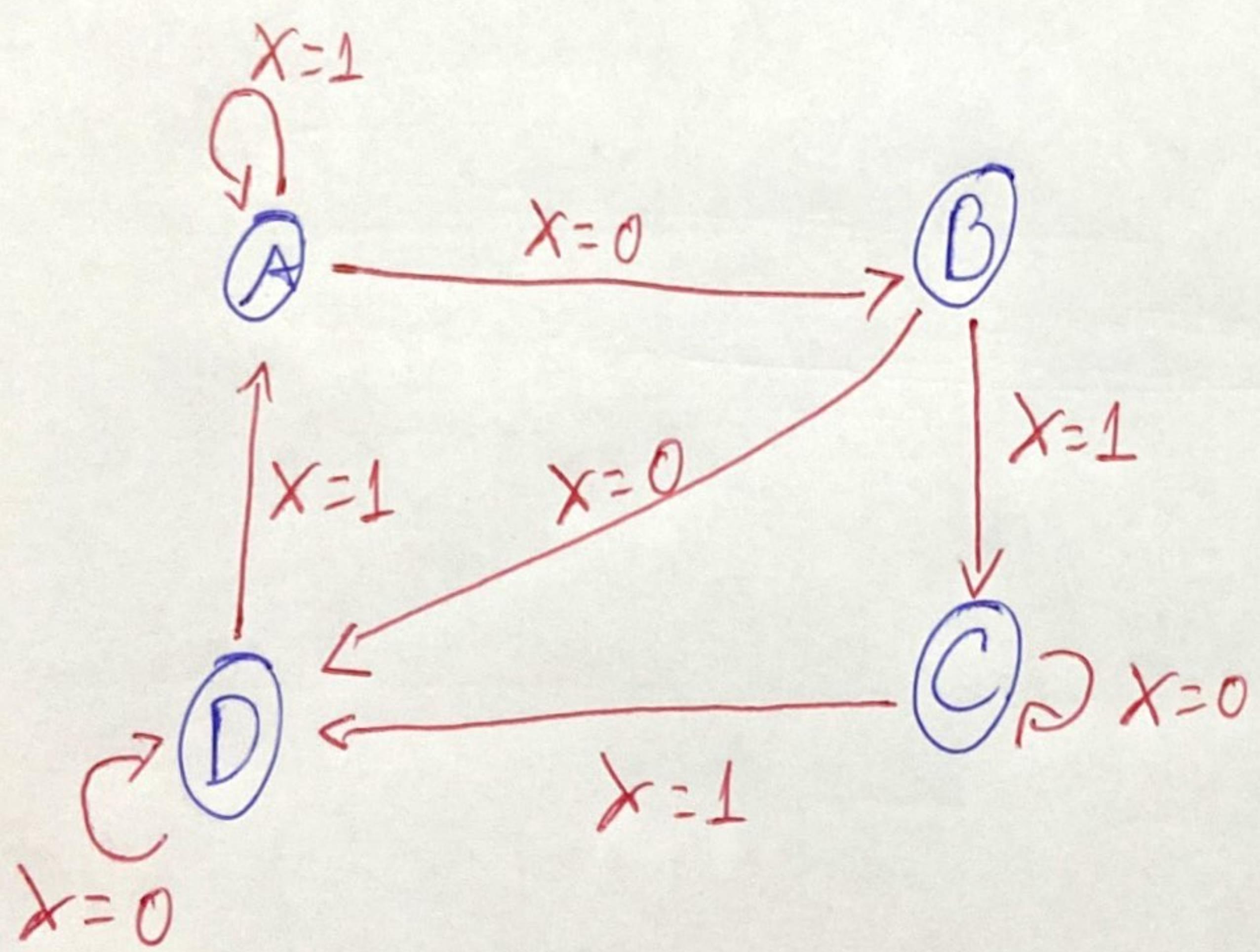
| | | X | | | | Z |
|-------|-------|---------|-------|---------|---|---|
| | | 0 | 1 | 0 | 1 | |
| Q_1 | Q_0 | 0 | 0 0 0 | 0 1 | | 0 |
| | | 1 | 1 1 1 | 1 0 | | 1 |
| 0 | 0 | 0 1 0 | 1 1 1 | 1 1 | | 1 |
| 1 | 1 | 1 1 0 | - - 0 | - - 0 | | 0 |
| | | Q_1^* | | Q_0^* | | |

Digamos que 0 0 = A, 0 1 = B, 1 0 = C e 1 1 = D,

Logo a tabela de estados e saída é:

| | | | | |
|-----|-----|-------|-------|-----------------|
| S | $ $ | $X=0$ | $X=1$ | $\} \mathbb{Z}$ |
| A | $ $ | A | B | $ $ |
| B | $ $ | D | C | $ $ |
| C | $ $ | C | D | $ $ |
| D | $ $ | D | A | $ $ |
| | $ $ | | | 0 |

«Lo Diagramma vero»:



8) ~~$t_{prop} < 10\text{ms}$~~

$5 < t_p < 10\text{ms} \rightarrow$ atraso de propagação ~~t_{prop}~~

$\hookrightarrow t_{pmin} \hookrightarrow t_{pmadC}$

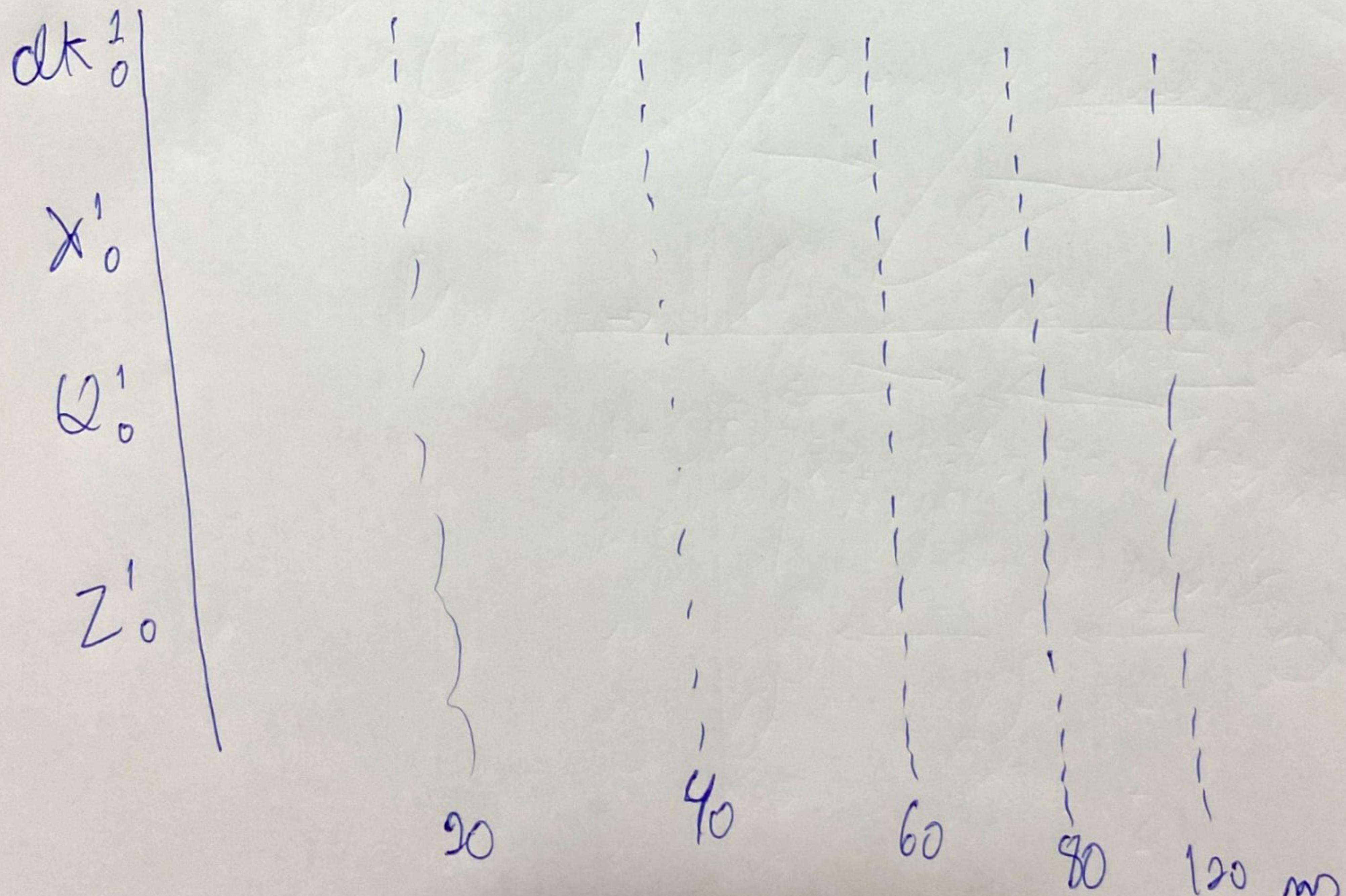
$5 \text{ a } 10\text{ms} \rightarrow$ atraso do relogio - delay
 $\hookrightarrow delay_{min} \hookrightarrow delay_{max}$

Logo o tempo de Configuração, $10 \xrightarrow{t_{pmadC}} 5\text{ms}$

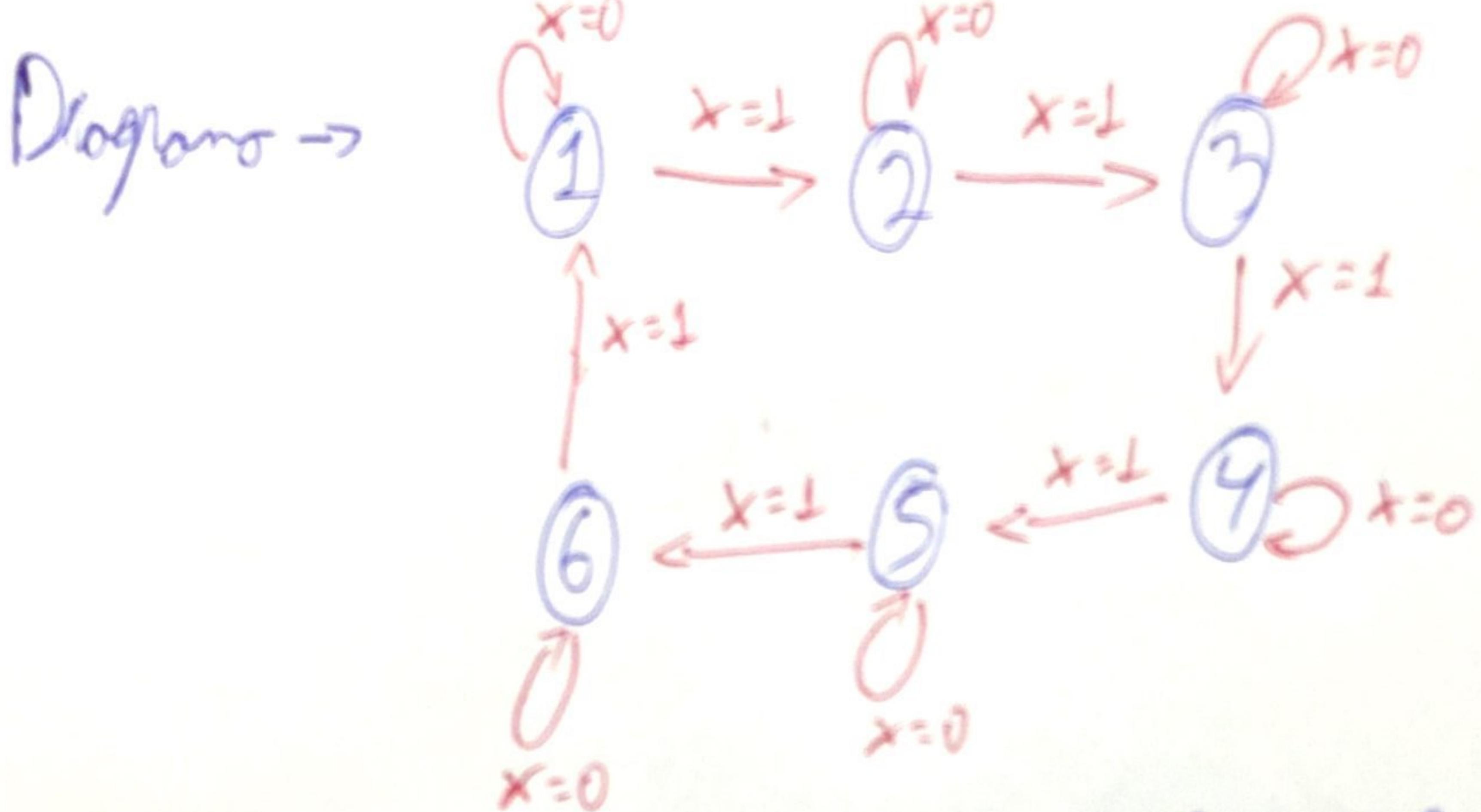
d o tempo de espera é $\cancel{t_{pmin} + t_{hold}}$ $t_{pmin} \geq t_{hold} + delay_{max}$

Logo $5 \geq t_{espera} + 10$

$t_{espera} \leq 5\text{ms}$ logo o tempo de espera é no mínimo 5ms.



9) Como o minero mais próximo ~~é~~^{tem} o motor que 6 gal é uma
potência de 2^1 é 8, então precisamos de 3 flip-flops, pois $2^3 = 8$.



Com $1 = S_0$, $2 = S_1$, $3 = S_2$, $4 = S_3$, $5 = S_7$, $6 = S_5$
 logo temos a seguinte tabela de ~~transição~~ estado

| S | $x=0$ | $x=1$ | Z |
|-------|-------|-------|---|
| S_0 | S_0 | S_1 | 0 |
| S_1 | S_1 | S_2 | 1 |
| S_2 | S_2 | S_3 | 1 |
| S_3 | S_3 | S_4 | 1 |
| S_4 | S_4 | S_5 | 1 |
| S_5 | S_5 | S_6 | 1 |

$$S = F_1 F_2 F_3$$

\downarrow \downarrow \downarrow
 F_2 F_3

Flip flop 1

| S | X | | Z |
|-----|-----|-----|---|
| | 0 | 1 | |
| 000 | 000 | 001 | 0 |
| 001 | 001 | 010 | 1 |
| 010 | 010 | 011 | 1 |
| 011 | 011 | 100 | 1 |
| 100 | 100 | 101 | 1 |
| 101 | 101 | 110 | 1 |
| 110 | 110 | 111 | 1 |
| 111 | 111 | 001 | 1 |

Moto:

Mapa: $f_1 \rightarrow$

| $\bar{F}_1 \bar{F}_2$ | $\bar{F}_3 X$ | 00 | 01 | 11 | 10 |
|-----------------------|---------------|----|----|----|----|
| 00 | | 0 | 0 | 0 | 0 |
| 01 | | 0 | 0 | 1 | 0 |
| 11 | | 1 | 1 | 0 | 1 |
| 10 | | 1 | 1 | 1 | 1 |

$$f_1 = \bar{F}_1 \cdot \bar{F}_3 + \bar{F}_1 \cdot \bar{F}_2 + \bar{F}_1 \cdot F_3 \cdot \bar{X} + \bar{F}_1 \cdot F_2 \cdot F_3 \cdot X$$

| $f_2 \rightarrow$ | $\bar{F}_3 \bar{X}$ | 00 | 01 | 11 | 10 |
|-------------------|---------------------|----|----|----|----|
| $F_1 F_2$ | 00 | 0 | 10 | 1 | 0 |
| 01 | 1 | 1 | 0 | 1 | |
| 11 | 1 | 1 | 0 | 1 | |
| 10 | 0 | 0 | 1 | 0 | |

$$f_2 = F_2 \cdot \bar{F}_3 + F_2 \cdot \bar{X} + \bar{F}_2 \cdot F_3 \cdot X$$