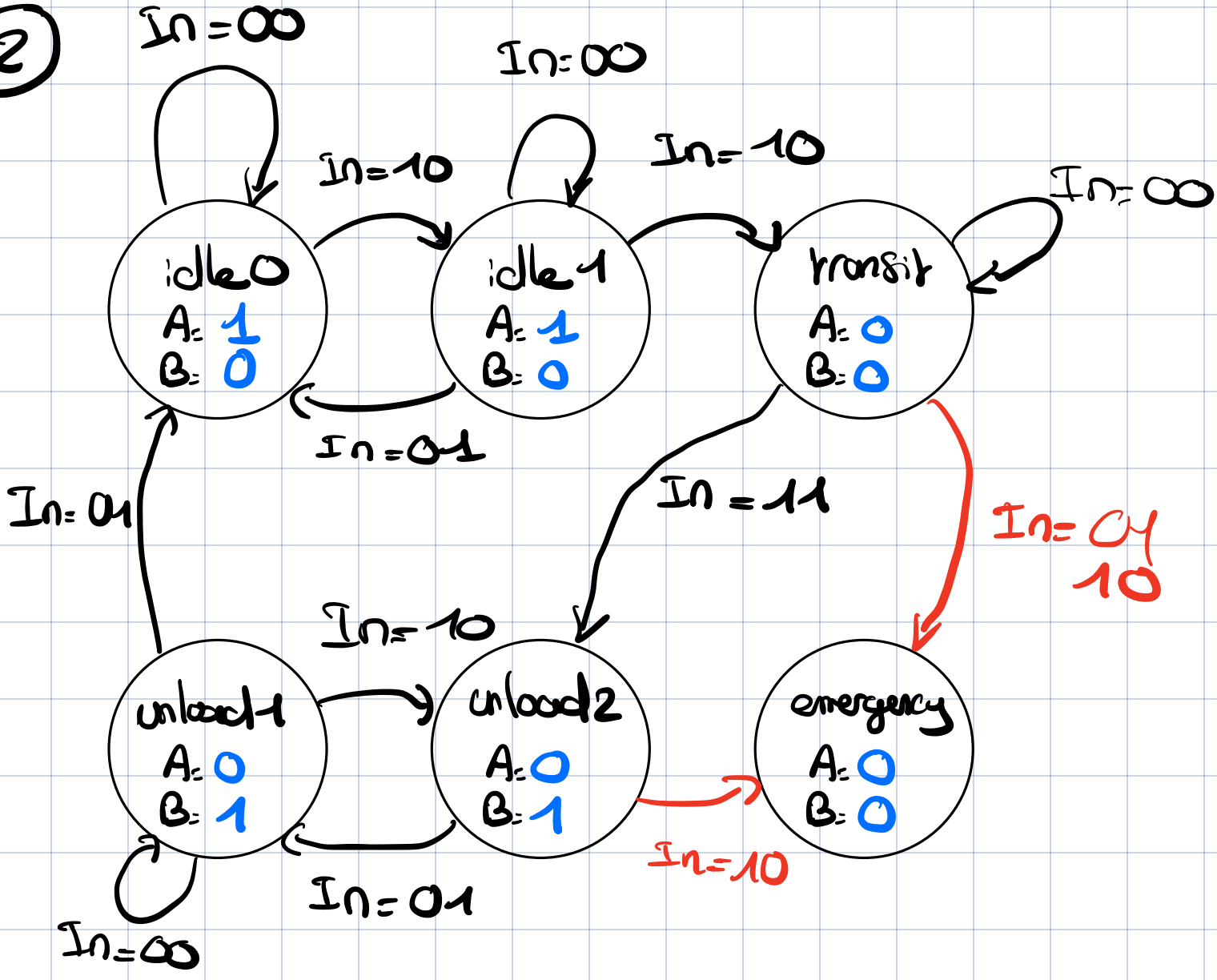
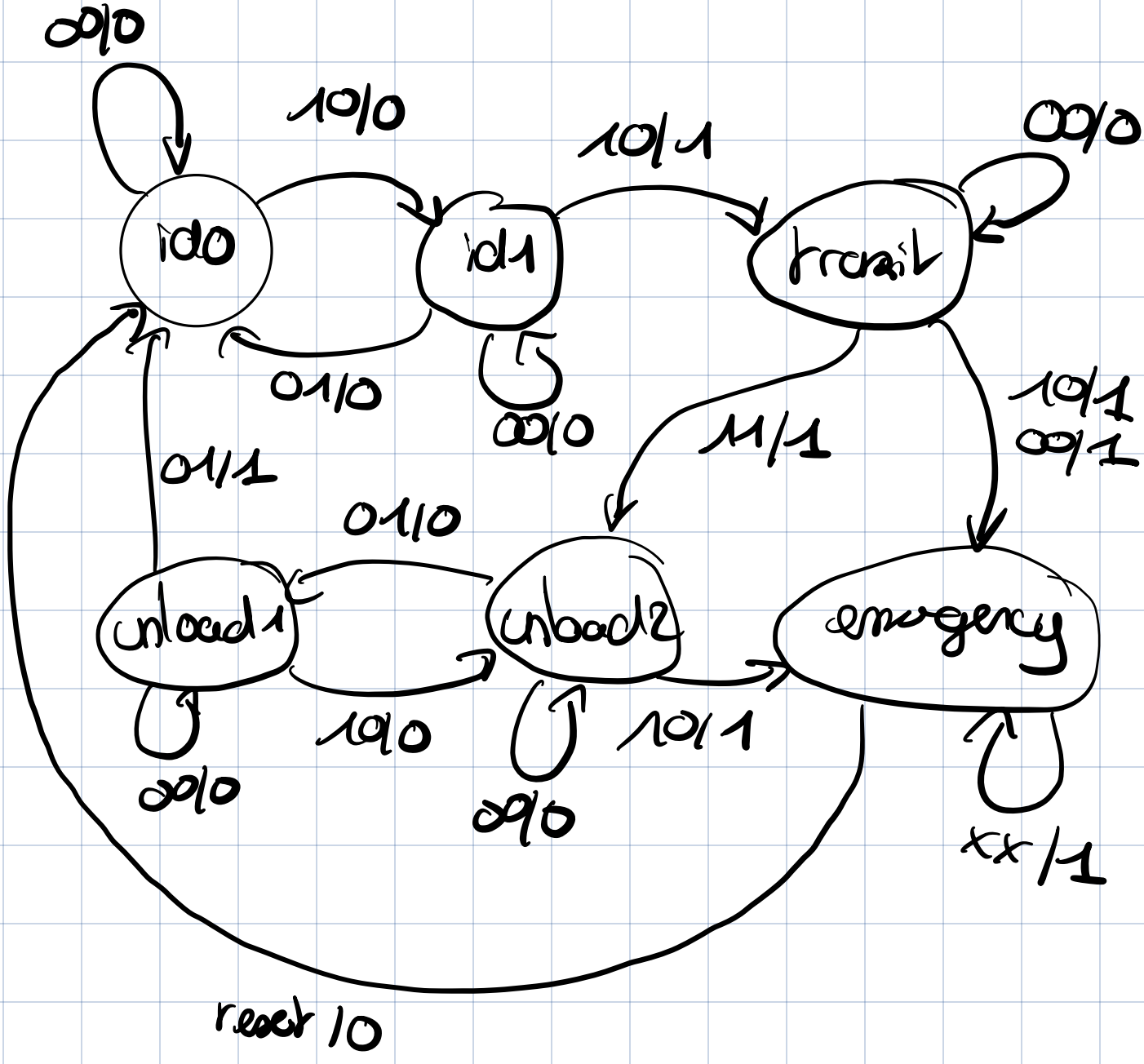
(2)


In=00
In=00
In=10
In=10
In=00
idle0
A=1
B=0
idle1
A=1
B=0
transit
A=0
B=0
In=01
In=01
In=11
In=04
10
In=10
unload1
A=0
B=1
unload2
A=0
B=1
emergency
A=0
B=0
In=01
In=10
In=00

b)


00/0
10/0
10/1
00/0
idle
idle1
travail
01/0
00/0
11/1
10/1
00/1
01/1
01/0
unload1
unload2
emergency
10/0
10/1
00/0
00/0
xx/1
reset 10

④

si en est actif
⇒ var1
si in[counter = 1
out = in[8*counter : 8]

0000 0011 | 0000 0010 | 0000 0001
31 30 29 28 27 26 25 24 | 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 8 | 7 6 5 4 | 3 2 1 0

ça fait la somme de

a bcd

[23 : 16]

8+: 8

[15:8]

clk	en	var	in2[var]	in1[var*8+: 8]	out
1	0	0	/	/	/
2	1	1	in2[1] = 1	in1[15:8] = 2	2

2	1	2	in2[2] = 0	m1[23:16] = 3	-1
3	0	3	m2[3] = 1	m1[~] = 4	3
4	0	4	m2[4] = 1	m[~] = 5	8

al	0	-1	3	8

↓

1111 1111 1111

ca sait $2^{11} - 1$

$= 2047$

0 | 2047 | 3 | 8

clk	en	var1	in1[var*8+1:8]	in2[var1]	av
0	0	0	/	/	0
1	1	0	in2[0] = 1	+1	1
2	0	1	in2[1] = 1	2	3
			0	-3	0
			1		
			1		
			1		0

a <= 1

a <= 0

(4.2)

- output reg
- reg [1:0]
- 3
- &
- |

abcd + efgh

wy

yz \ wx	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	0	0	1	1
10	0	0	1	1

$\bar{z}\,\bar{w}xy$

$$wy + w\bar{x} + \bar{x}\bar{y}\bar{z}$$

$$wxyz + wxy\bar{z} + w\bar{x}\bar{y}\bar{z} + w\bar{x}\bar{y}z + w\bar{x}yz + w\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}\bar{z}$$

$$= \bar{x}\bar{y}\bar{z}\underbrace{(w + \bar{w})}_{1}$$

(b)

$$F = \overline{\bar{A} + \overline{BC} + \overline{A\bar{C}}}$$

$$= \overline{\bar{A} + \left(\overline{B+\bar{C}}\right) + \left(\overline{\bar{A}+\bar{C}}\right)}$$

Verilog

(2.1) (1) output reg

(2) reg [31:0] nval

(3) default

(4) odata <= 0

(5) odata <= nval

State A

Si out contient que des 1 ⇒ on change le state

Si le state est 0 ⇒ on incrémente

clk	state	next_reg	next_state	out
0	0	01	0	00 0
1	0	10	0	01 1
2	0	11	0	10 2
3	0	00	1	(11) 3
4	1	11	0	00 0
5	0	00		3
6	1			
7	0			

Exercise 3

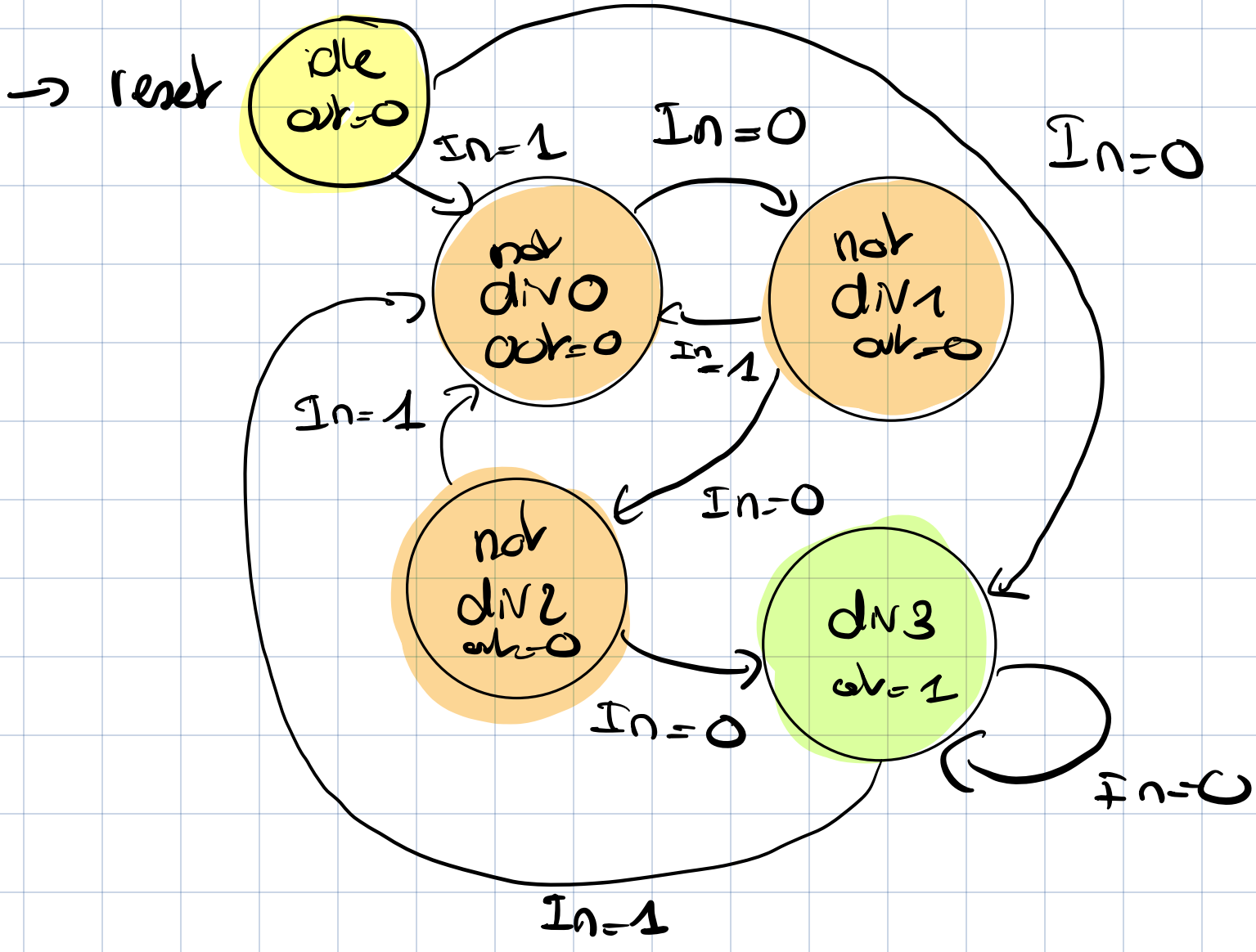
3_1

CURR	NEXT		OUT	
	In=0	In=1	In=0	In=1
A	A	B	0	1
B	B	B	1	0
C	D	A	1	0
D	A	B	0	1

10110100

01001100

1000



NAND $\overline{abc}$

(a)

$$\overline{\overline{AB} + C + AC}$$

$$= \left(\overline{\overline{\overline{AB} \cdot \overline{C}}}\right)\left(\overline{\overline{AC}}\right)$$

(b)

CD \ AB	00	01	11	10
00	1	1	1	1
01	0	1	1	0
11	0	0	0	0
10	0	0	0	0

$\overline{C}\overline{D} + \overline{C}B$

② Verilog

it has to be combinational because it only depends on the input, no state

ⓑ
```
always @(posedge clk) begin
    if (reset) curVal <= S0;
    else curVal <= nextVal;
end
```

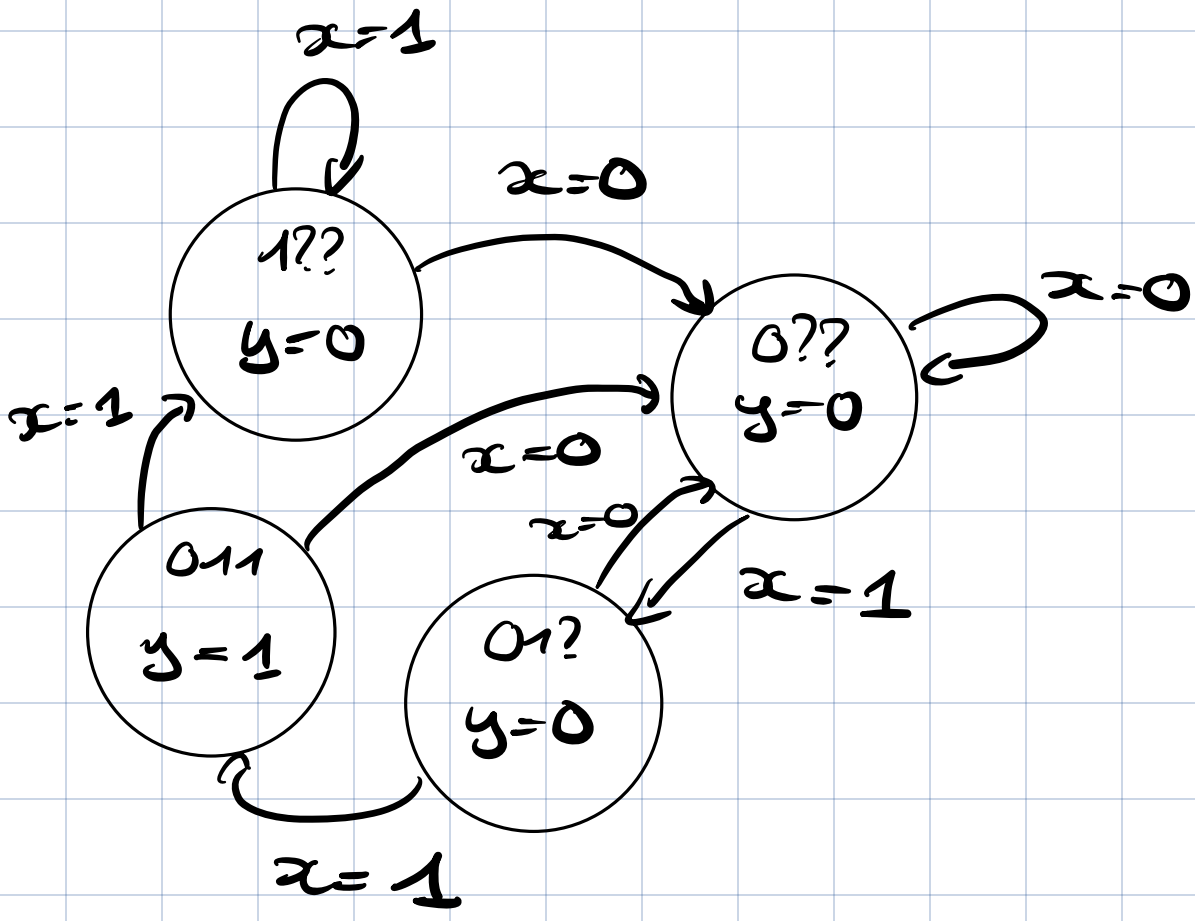
ⓒ

out = 0	tmp = 1	because b not sensitivity
out = 0	tmp = 1	

(d)

- wires can not be assigned in always blocks (rmp)
- there should be only one always block for z[0] and z[1]
- s has to be a wire

(3) Finite State Machines



(3.2) This is a Mealy because outputs depend on the inputs

(3.2.b) S_0 can be removed.

(1) Boolean Algebra

$$(\bar{A} + B + C)(\cancel{A} + B + \cancel{C})C + A$$

$$BC + A$$

(2)

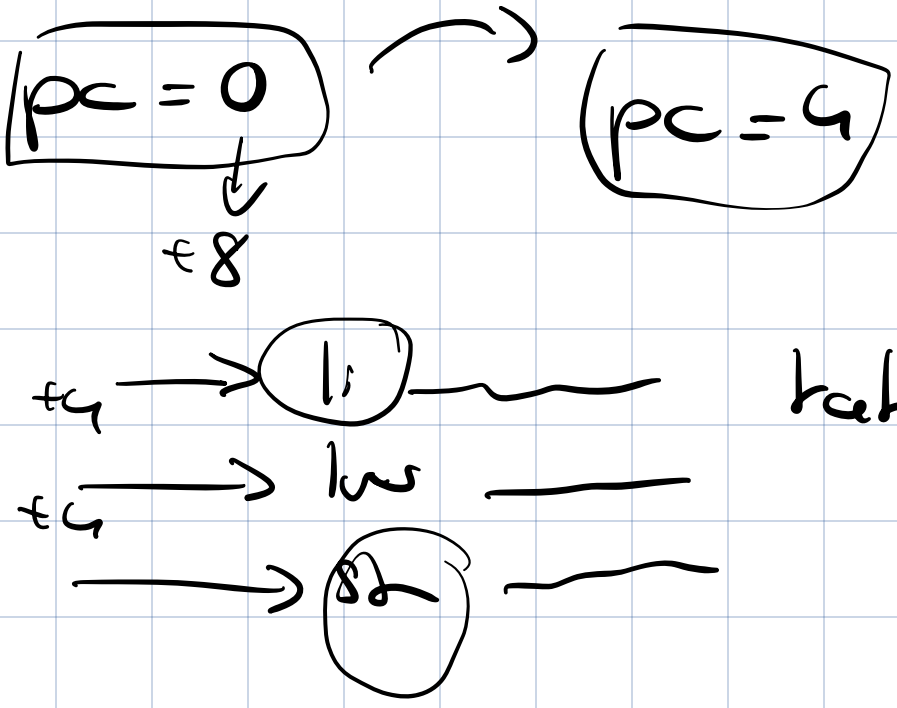
$$F = \overline{\bar{A} + (BC + \overline{AC})}$$

$$= \frac{\overline{\bar{A} + (\overline{BC + \overline{AC}})}}{}$$

$$= \frac{A\overline{(BC + \overline{AC})}}{}$$

$$= A\left(\overline{BC}\,\overline{\overline{AC}}\right)$$

→ 1, ——
→ 1, ——
) 32 bits



takes 4 bytes in the memory

② Verilog

ⓐ no because of negedge reset
2 bits wide

ⓑ Sequential because data_out[0]
can be 1 or 0 depending on
the fact that mask was enabled
before or not

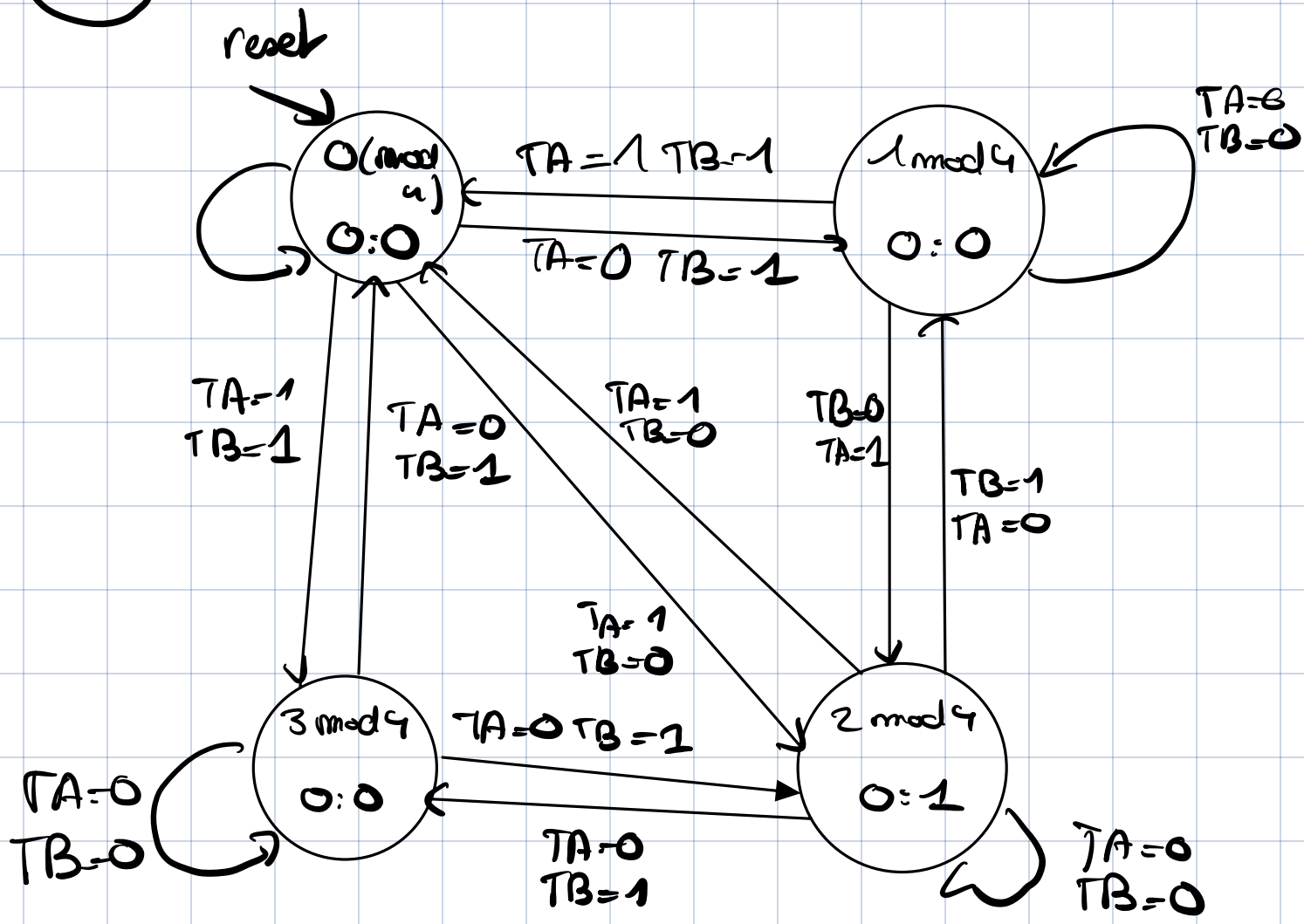
ⓒ S, cout have to be of type wire

input wire ... ??

-z should be .s

r2, r1 should be wire

(3)



(b)

In

$Y_2 Y_1$	$T_A T_B$ = 00	01	10	11	output
00 Z	00 Z	01 U	10 D	11 T	0
01 U	01 U	10 D	11 T	00 Z	0
10 D	10 D	11 T	00 Z	01 U	1
11 T	11 T	00 Z	01 U	10 D	0

$$Y_1^* = T_B \overline{T_1} + Y_1 \overline{T_B}$$
$$= T_B \oplus Y_1$$

$$Y_2^* = \overline{T_B}(T_A \oplus Y_2) + T_B(T_A \oplus (Y_1 \oplus Y_2))$$

Verilog

It is sequential because

(b)

E14 c 1110 0011
7 6 5 4 3 2 1 0
a 1000
b 0010

c = E3
c = {1110 0011, 0, 1}
= 1000 1101

c[0] = 0 ⇒ 1000 1100

(c)

- input wire [6:0] instr

- width of z is incorrect
- output reg while it should be output
- .z should be .s
- .instr should be .d → not the right size d

count_next = 1;

(3) (a)

$$B + A + \bar{C}$$

(b)

$$\overline{(A + BC) + \bar{C}}$$

$$= \overline{\bar{A}\,\overline{BC} + \bar{C}}$$

$$= \overline{\bar{A}\,\overline{BC}}\; C$$

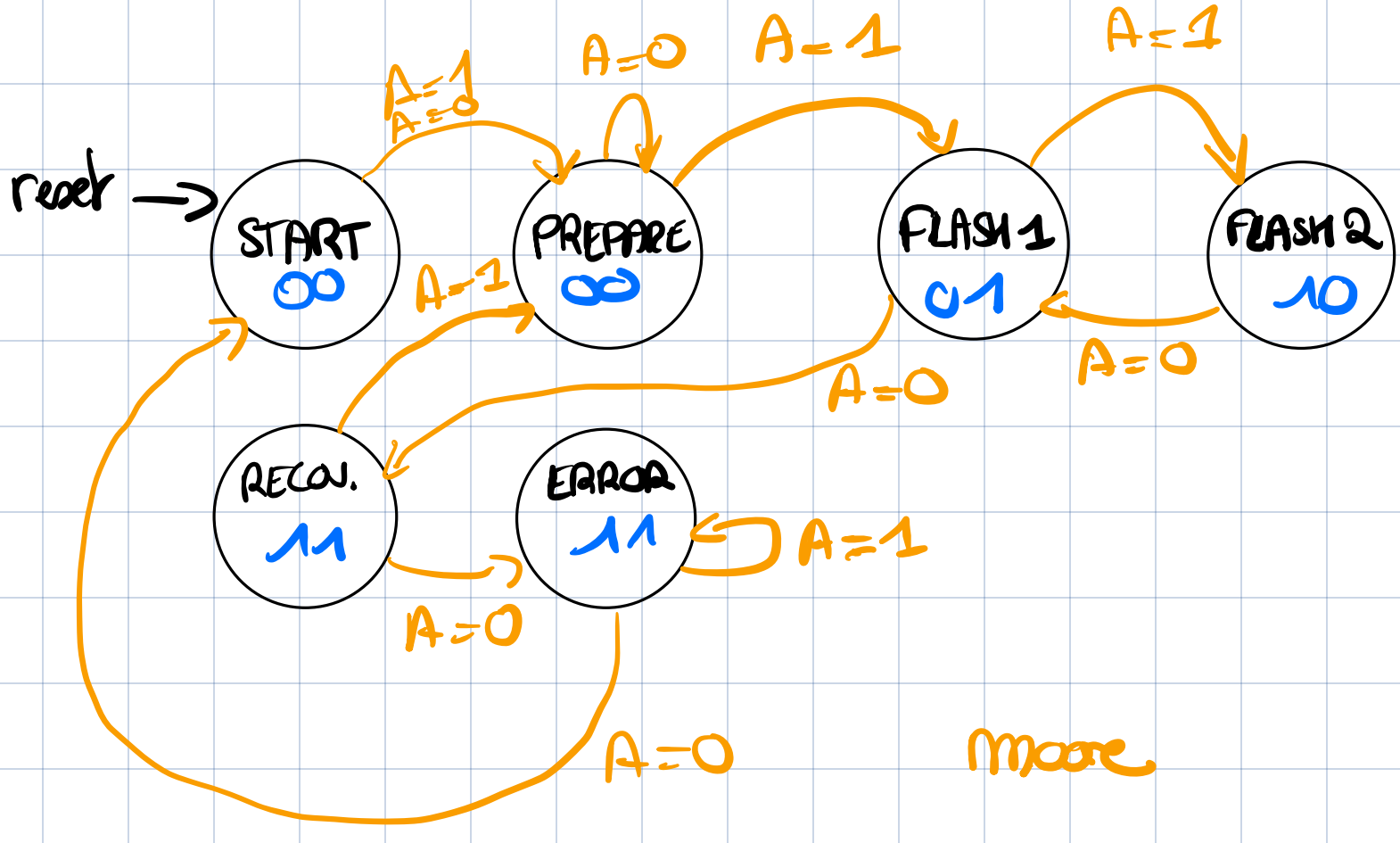
CD \ AB	00	01	11	10	00
00	0	0	0	0	0
01	0	1	1	0	0
11	1	1	1	1	1
10	0	0	0	0	0

(a) it is a Moore FSM

(c) One-hot : get an easy formula
Binary encoding: less bits
Output encoding: less bits

2017 (2)

- it can not be a Moore AND Mealy
- Some arrows are empty
- Some arrows are exiting the states with the same value
- no reset state



Sequential (changing from 11 to 00 then 01 to 00)

t