

ARHITECTURA SISTEMELOR DE CALCUL SEMINAR 0x02

NOTIȚE SUPORT SEMINAR

Cristian Rusu

TABELE DE ADEVĂR, EX 1

$$X = A + BC$$

[illegible]

TABELE DE ADEVĂR, EX 1

$$X = A + BC$$

A	B	C	BC	X
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

TABELE DE ADEVĂR, EX 1

$$X = A + BC$$

A	B	C	BC	X
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	
1	0	0	0	
1	0	1	0	
1	1	0	0	
1	1	1	1	

TABELE DE ADEVĂR, EX 1

$$X = A + BC$$

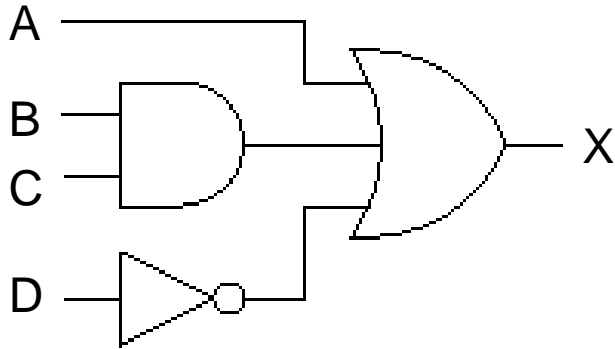
A	B	C	BC	X
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

DESENAȚI CIRCUITUL, EX 2

$$X = A + BC + \bar{D}$$

DESENAȚI CIRCUITUL, EX 2

$$X = A + BC + \bar{D}$$



DE MORGAN, EX 3

$$X = \overline{(\overline{A\overline{B}})(\overline{B} + C)}$$

=

=

=

=

=

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\overline{A\overline{B}})(\overline{B} + C)} \\ &= \overline{(\overline{A\overline{B}})} + \overline{(\overline{B} + C)} \\ &= \\ &= \\ &= \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\overline{A\overline{B}})(\overline{B} + C)} \\ &= \overline{(\overline{A\overline{B}})} + \overline{(\overline{B} + C)} \\ &= A\overline{B} + \overline{(\overline{B} + C)} \\ &= \\ &= \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\overline{A\overline{B}})(\overline{B} + C)} \\ &= \overline{(\overline{A\overline{B}})} + \overline{(\overline{B} + C)} \\ &= A\overline{B} + \overline{(\overline{B} + C)} \\ &= A\overline{B} + (\overline{\overline{B}}\overline{C}) \\ &= \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\overline{A\overline{B}})(\overline{B} + C)} \\ &= \overline{(\overline{A\overline{B}})} + \overline{(\overline{B} + C)} \\ &= A\overline{B} + \overline{(\overline{B} + C)} \\ &= A\overline{B} + (\overline{\overline{B}}\overline{C}) \\ &= A\overline{B} + (B\overline{C}) \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\overline{A\overline{B}})(\overline{B} + C)} \\ &= \overline{(\overline{A\overline{B}})} + \overline{(\overline{B} + C)} \\ &= A\overline{B} + \overline{(\overline{B} + C)} \\ &= A\overline{B} + (\overline{\overline{B}}\overline{C}) \\ &= A\overline{B} + (B\overline{C}) \\ &= A\overline{B} + B\overline{C} \end{aligned}$$

DE MORGAN, EX 3

$$X = \overline{(\bar{A} + C)(\overline{AB})}$$

=

=

=

=

=

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\bar{A} + C)(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + \overline{(\overline{AB})} \\ &= \\ &= \\ &= \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\bar{A} + C)(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + \overline{(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + (AB) \\ &= \\ &= \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\bar{A} + C)(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + \overline{(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + (AB) \\ &= (\overline{\bar{A}}\overline{C}) + (AB) \\ &= \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\bar{A} + C)(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + \overline{(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + (AB) \\ &= (\overline{\bar{A}}\overline{\bar{C}}) + (AB) \\ &= (A\overline{\bar{C}}) + (AB) \\ &= \end{aligned}$$

DE MORGAN, EX 3

$$\begin{aligned} X &= \overline{(\bar{A} + C)(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + \overline{(\overline{AB})} \\ &= \overline{(\bar{A} + C)} + (AB) \\ &= (\overline{\bar{A}}\overline{\bar{C}}) + (AB) \\ &= (A\overline{\bar{C}}) + (AB) \\ &= A(B + \overline{\bar{C}}) \end{aligned}$$

DE MORGAN, EX 3

- $\neg(\neg A + \neg B) = AB$
- $\neg(\neg A \neg B) = A + B$
- $\neg(A + B + C) = \neg A \neg B \neg C$
- $\neg(ABC) = \neg A + \neg B + \neg C$
- $\neg(A + B) \neg A \neg B = \neg A \neg B$
- $\neg(AB)(\neg A + \neg B) = \neg A + \neg B$
- $\neg(A + B)(\neg A + \neg B) = \neg A \neg B$
- $\neg A \neg B \neg(AB) = \neg A \neg B$
- $C + \neg(CB) = 1$
- $\neg(AB)(\neg A + B)(\neg B + \neg B) = \neg A \neg B$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

$$(A + C)A(D + \bar{D}) + AC + C \quad // \text{distribuim, invers}$$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

$$(A + C)A(D + \bar{D}) + AC + C \text{ //distribuim, invers}$$

$$(A + C)A + AC + C \text{ //suma variabila si complement}$$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

$$(A + C)A(D + \bar{D}) + AC + C \text{ //distribuim, invers}$$

$$(A + C)A + AC + C \text{ //suma variabila si complement}$$

$$A((A + C) + C) + C \text{ //distribuim, invers}$$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

$$(A + C)A(D + \bar{D}) + AC + C \text{ //distribuim, invers}$$

$$(A + C)A + AC + C \text{ //suma variabila si complement}$$

$$A((A + C) + C) + C \text{ //distribuim, invers}$$

$$A(A + C) + C \text{ //asociem, idempotent}$$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

$$(A + C)A(D + \bar{D}) + AC + C \text{ //distribuim, invers}$$

$$(A + C)A + AC + C \text{ //suma variabila si complement}$$

$$A((A + C) + C) + C \text{ //distribuim, invers}$$

$$A(A + C) + C \text{ //asociem, idempotent}$$

$$AA + AC + C \text{ //distribuim}$$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

$$(A + C)A(D + \bar{D}) + AC + C \text{ //distribuim, invers}$$

$$(A + C)A + AC + C \text{ //suma variabila si complement}$$

$$A((A + C) + C) + C \text{ //distribuim, invers}$$

$$A(A + C) + C \text{ //asociem, idempotent}$$

$$AA + AC + C \text{ //distribuim}$$

$$A + (A + 1)C \text{ //idempotent, identitate, factor}$$

SIMPLIFICĂRI, EX 4

$$(A + C)(AD + A\bar{D}) + AC + C$$

$$(A + C)A(D + \bar{D}) + AC + C \text{ //distribuim, invers}$$

$$(A + C)A + AC + C \text{ //suma variabila si complement}$$

$$A((A + C) + C) + C \text{ //distribuim, invers}$$

$$A(A + C) + C \text{ //asociem, idempotent}$$

$$AA + AC + C \text{ //distribuim}$$

$$A + (A + 1)C \text{ //idempotent, identitate, factor}$$

$$A + C \text{ //identitate de doua ori}$$

SIMPLIFICĂRI, EX 4

$$\bar{A}(A + B) + (B + AA)(A + \bar{B})$$

SIMPLIFICĂRI, EX 4

$$\bar{A}(A + B) + (B + AA)(A + \bar{B})$$

$$\bar{A}(A + B) + (B + A)(A + \bar{B}) // AA \text{ este } A$$

SIMPLIFICĂRI, EX 4

$$\bar{A}(A + B) + (B + AA)(A + \bar{B})$$

$$\bar{A}(A + B) + (B + A)(A + \bar{B}) // AA \text{ este } A$$

$$(A + B)(\bar{A} + A + \bar{B}) // \text{factor } A + B$$

SIMPLIFICĂRI, EX 4

$$\bar{A}(A + B) + (B + AA)(A + \bar{B})$$

$$\bar{A}(A + B) + (B + A)(A + \bar{B}) // AA \text{ este } A$$

$$(A + B)(\bar{A} + A + \bar{B}) // \text{factor } A + B$$

$$(A + B)(1 + \bar{B}) // \text{variabila sau complement}$$

SIMPLIFICĂRI, EX 4

$$\bar{A}(A + B) + (B + AA)(A + \bar{B})$$

$$\bar{A}(A + B) + (B + A)(A + \bar{B}) // AA \text{ este } A$$

$$(A + B)(\bar{A} + A + \bar{B}) // \text{factor } A + B$$

$$(A + B)(1 + \bar{B}) // \text{variabila sau complement}$$

$$A + B // 1 \text{ sau orice este } 1$$

SIMPLIFICĂRI, EX 4

a) $A+0 = A$

b) $!Ax0 = 0$

c) $A+!A = 1$

d) $A+A = A$

e) $A+AB = A$

f) $A+!AB = A+B$

g) $A(!A+B) = AB$

h) $AB+!AB = B$

i) $(!A!B+!AB) = !A$

j) $A(A+B+C+...) = A$

k) subpuncte

a) $A+B$

b) 1

c) 1

l) $A+A!A = A$

m) $AB+A!B = A$

n) $!A+B!A = !A$

o) $(D+!A+B+!C)B = B$

p) $(A+!B)(A+B) = A$

q) $C(C+CD) = C$

r) $A(A+AB) = A$

s) $!(!A+!A) = A$

t) $!(A+!A) = 0$

u) $D+(D!CBA) = D$

v) $!D!(DBCA) = !D$

w) $AC+!AB+BC = AC+!AB$

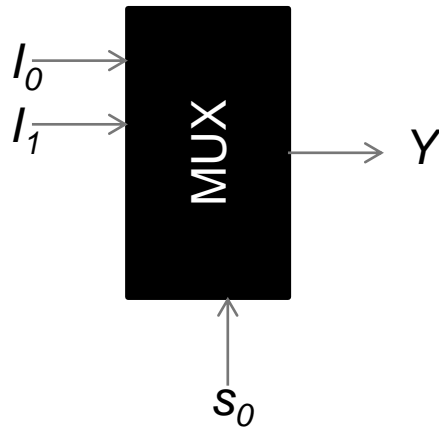
x) $(A+C)(!A+B)(B+C) = AB+!AC$

y) $!A+!B+AB!C = !A+!B+!C$

$(A+B)^2+(A+B)^3+A+3!A+A^3 = 1$

MUX, EX 5 A

- MUX, două intrări, un semnal s de selecție și o ieșire

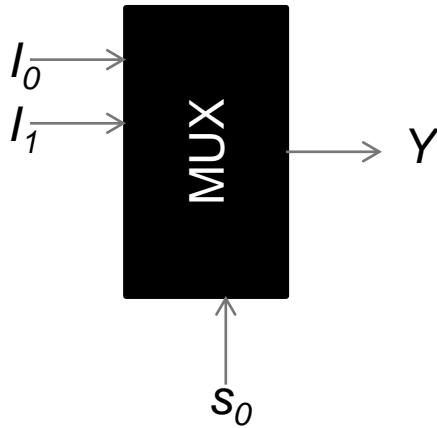


I_0	I_1	s_0	Y
*	*	0	I_0
*	*	1	I_1

- care este relația ieșire-intrare?
 - $Y = ?$

MUX, EX 5 A

- MUX, două intrări, un semnal s de selecție și o ieșire

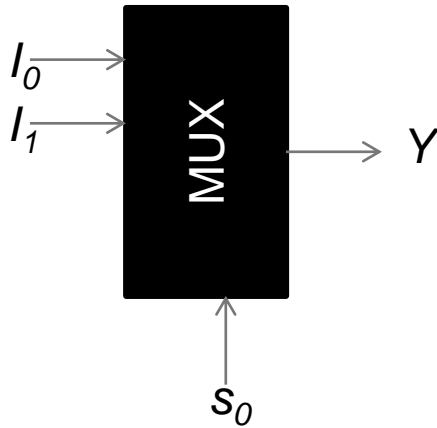


I_0	I_1	s_0	Y
*	*	0	I_0
*	*	1	I_1

- care este relația ieșire-intrare?
 - $Y = I_0\bar{s}_0 + I_1s_0$

MUX, EX 5 B

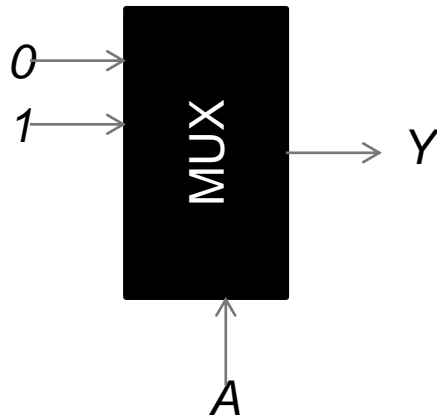
- MUX, două intrări, un semnal s de selecție și o ieșire



- un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND
 - implementați o poartă NOT cu un MUX

MUX, EX 5 B

- MUX, două intrări, un semnal s de selecție și o ieșire

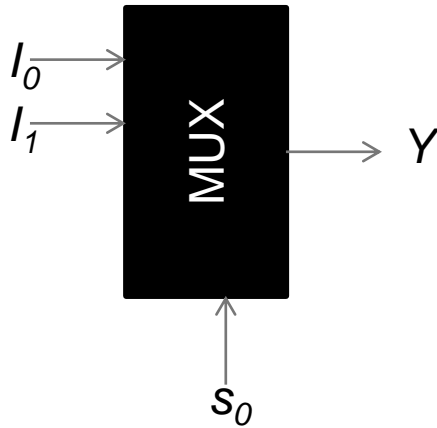


I_0	I_1	$s_0(A)$	Y
0	1	0	I_1
0	1	1	I_0

- un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND
 - implementați o poartă NOT cu un MUX: $Y = \text{NOT } A$

MUX, EX 5 B

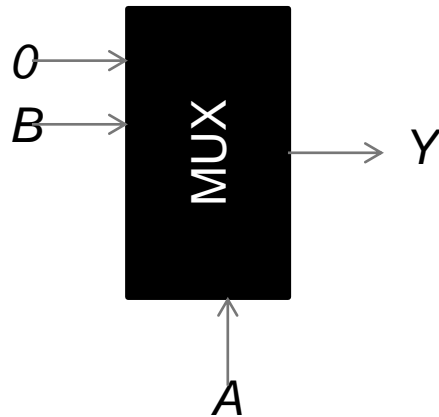
- MUX, două intrări, un semnal s de selecție și o ieșire



- un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND
 - implementați o poartă AND cu un MUX

MUX, EX 5 B

- MUX, două intrări, un semnal s de selecție și o ieșire

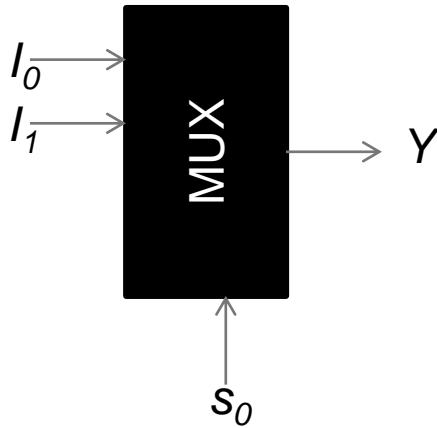


I_0	$I_1(B)$	$s_0(A)$	Y
0	B	0	$I_0(0)$
0	B	1	$I_1(B)$

- un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND
 - implementați o poartă AND cu un MUX: $Y = A \text{ AND } B$

MUX, EX 5 B

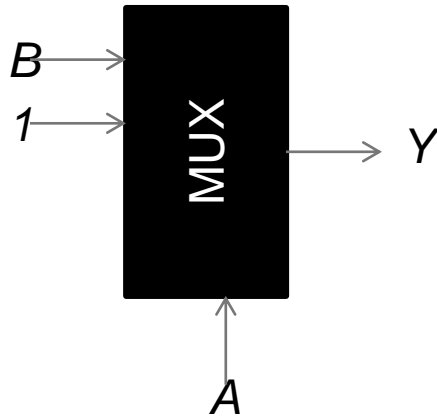
- MUX, două intrări, un semnal s de selecție și o ieșire



- un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND
 - implementați o poartă OR cu un MUX

MUX, EX 5 B

- MUX, două intrări, un semnal s de selecție și o ieșire

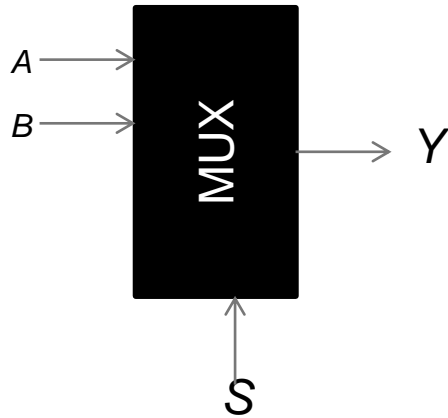


$I_0(B)$	I_1	$s_0(A)$	Y
B	1	0	$I_0(B)$
B	1	1	$I_1(1)$

- un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND
 - implementați o poartă OR cu un MUX: $Y = A \text{ OR } B$

MUX, EX 5 C

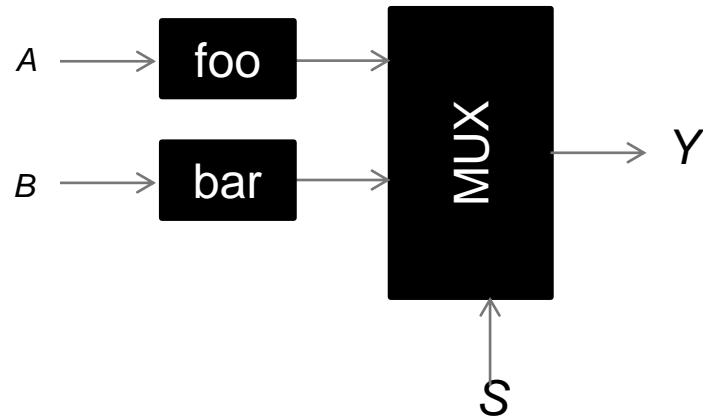
- $Y = S ? \text{foo}(A) : \text{bar}(B)$



- cum implementăm “if”-ul de mai sus?

MUX, EX 5 C

- $Y = S ? \text{foo}(A) : \text{bar}(B)$

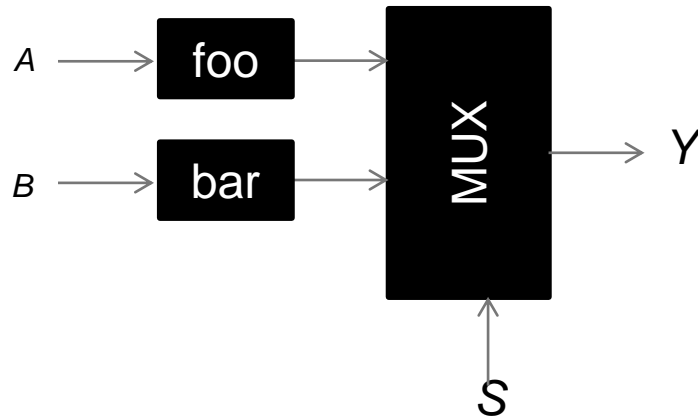


I_0 <i>foo(X)</i>	I_1 <i>bar(Y)</i>	S	Y
*	*	0	I_1
*	*	1	I_0

- care e diferența cu un limbaj de programare?

MUX, EX 5 C

- $Y = S ? \text{foo}(A) : \text{bar}(B)$

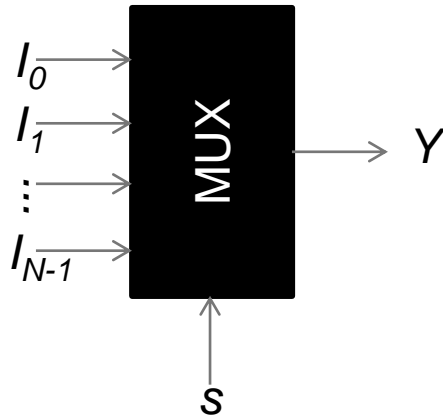


I_0 <i>foo(X)</i>	I_1 <i>bar(Y)</i>	S	Y
*	*	0	I_1
*	*	1	I_0

- **care e diferența cu un limbaj de programare?**
 - indiferent de valoarea lui S, se execută $\text{foo}(A)$ și $\text{bar}(B)$
 - doar că la ieșire vedem doar una dintre funcții (cea selectată de S)

MUX, EX 5 D

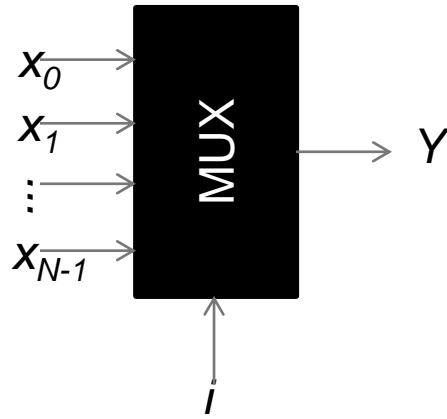
- vrem să accesăm un element al unui vector x_i



- ce putem la intrări?
- ce este semnalul s ?

MUX, EX 5 D

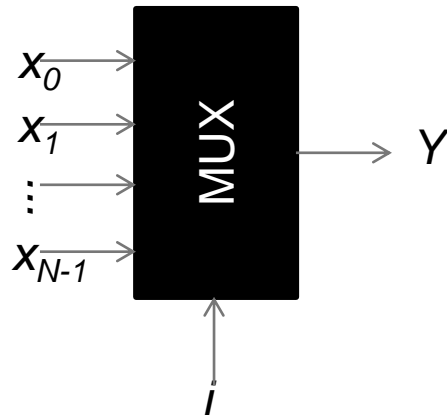
- vrem să accesăm x_i



- ce putem la intrări? **punem vectorul x**
- ce este semnalul s ? **punem index-ul i**
- care este dimensiunea intrării?
- care este dimensiunea lui s ?

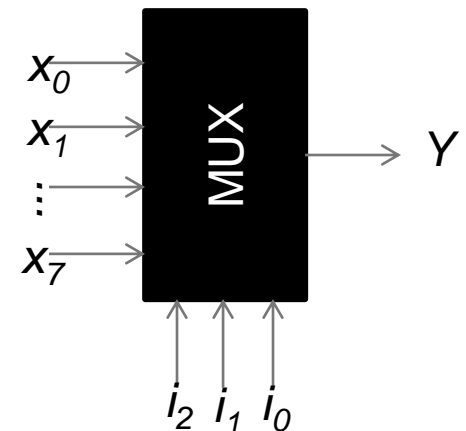
MUX, EX 5 D

- vrem să accesăm x_i



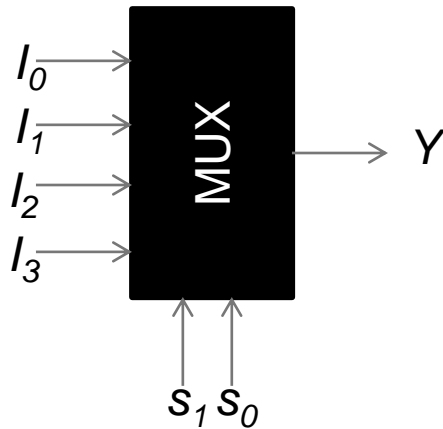
- ce putem la intrări? **punem vectorul x**
- ce este semnalul s ? **punem index-ul i**
- care este dimensiunea intrării? **N**
- care este dimensiunea lui s ? **$\text{ceil}(\log_2 N)$**

$s_0 (i)$	Y
000	$I_0 (x_0)$
001	$I_1 (x_1)$
010	$I_2 (x_2)$
011	$I_3 (x_3)$
100	$I_4 (x_4)$
101	$I_5 (x_5)$
110	$I_6 (x_6)$
111	$I_7 (x_7)$

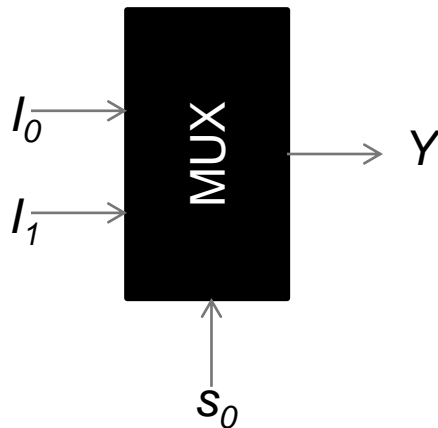


MUX, EX 5 E

- un MUX cu 4 intrări
 - automat știm că semnalul s are doi biți

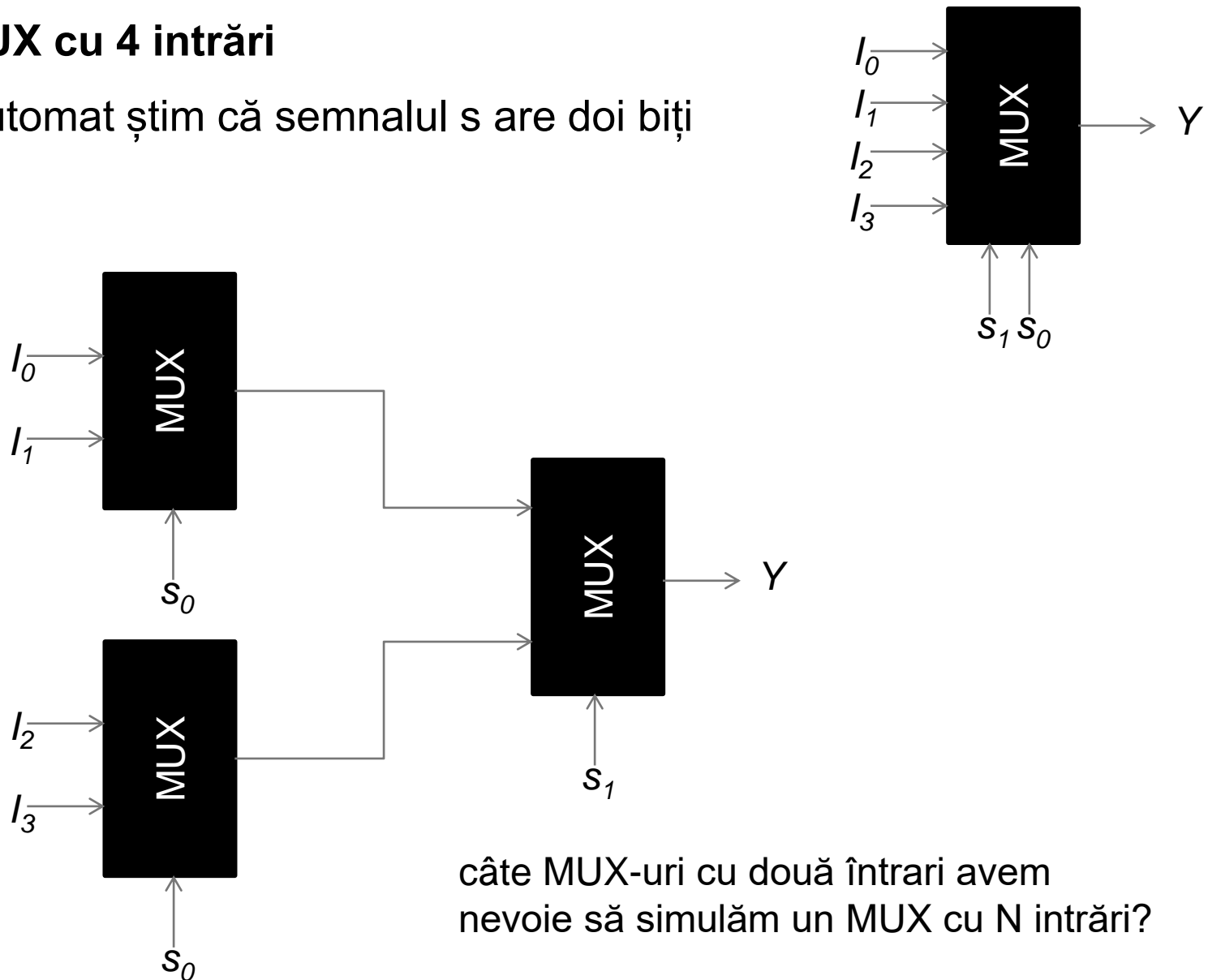


- avem la dispoziție un MUX cu 2 intrări



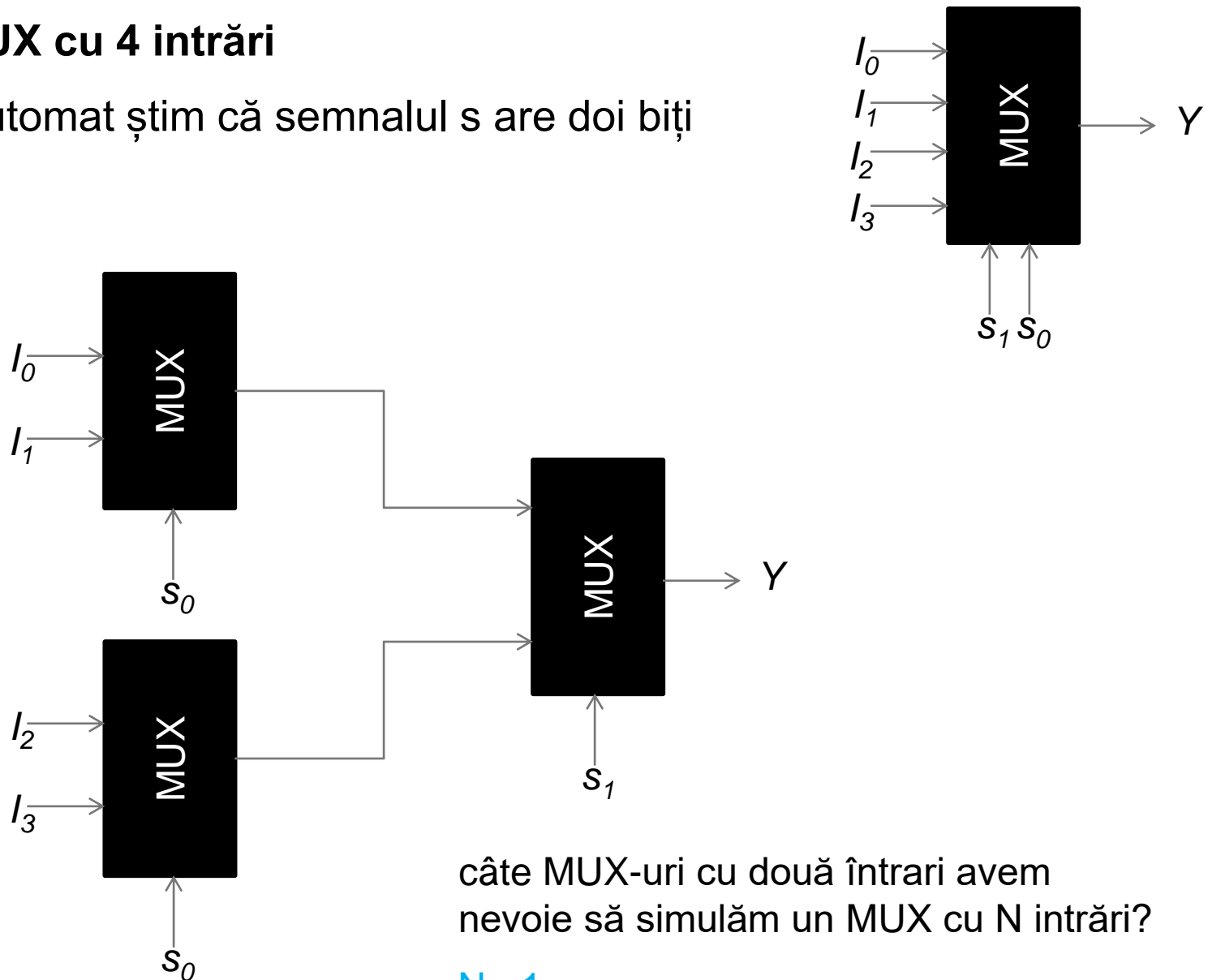
MUX, EX 5 E

- un MUX cu 4 intrări
 - automat știm că semnalul s are doi biți



MUX, EX 5 E

- un MUX cu 4 intrări
 - automat știm că semnalul s are doi biți

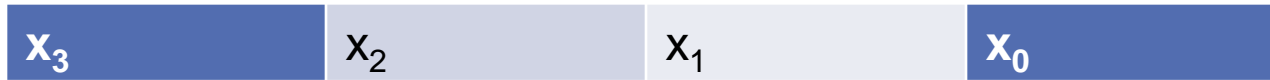


câte MUX-uri cu două intrări avem
nevoie să simulăm un MUX cu N intrări?

$N - 1$

MUX, EX 7 A

- numarul nostru x este



- deplasare normală cu 2 la dreapta



echivalent cu o împărțire la 2^2

- deplasare aritmetică cu 2 la dreapta



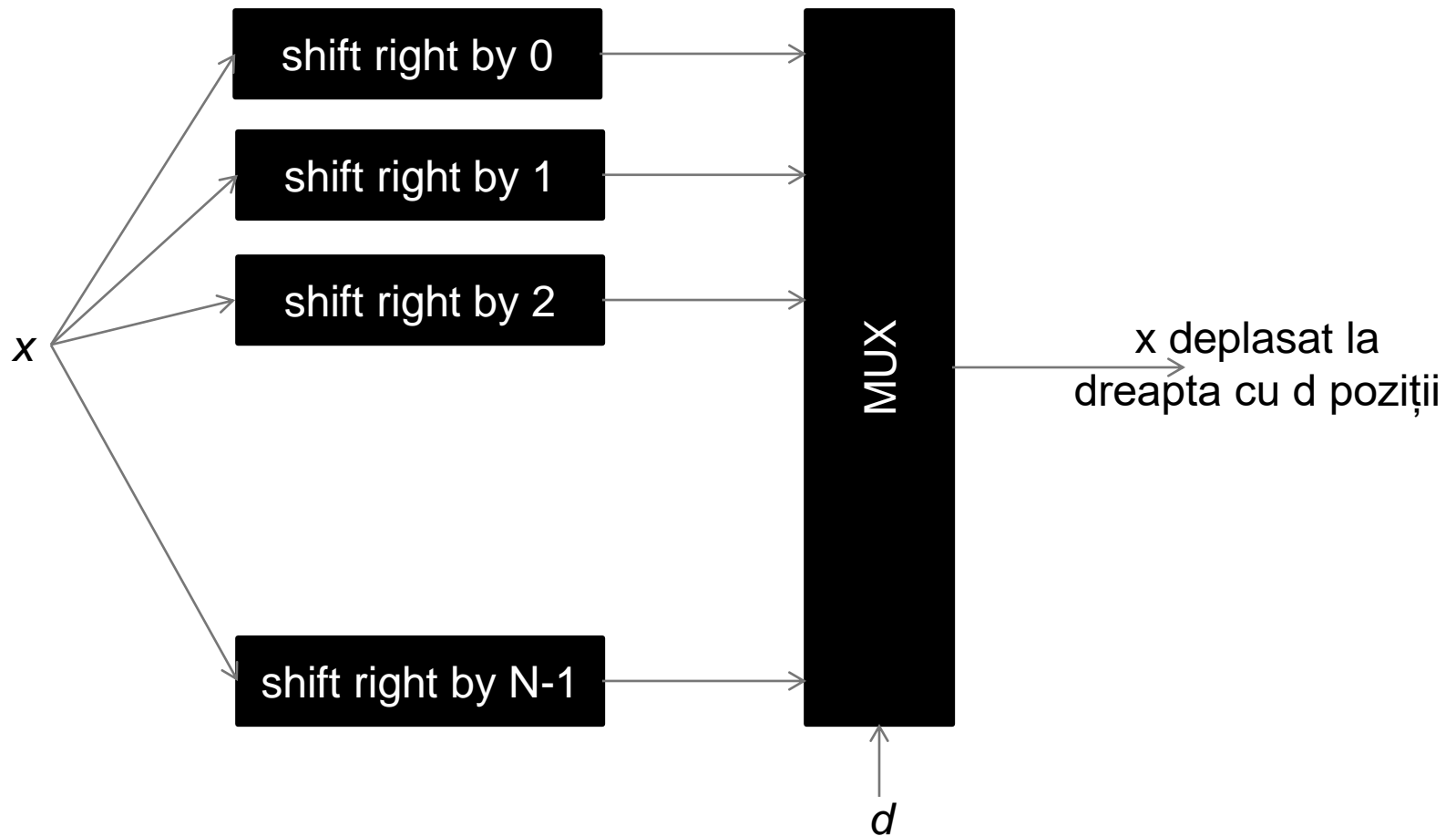
echivalent cu o împărțire la 2^2

- deplasare circulară cu 2 la dreapta



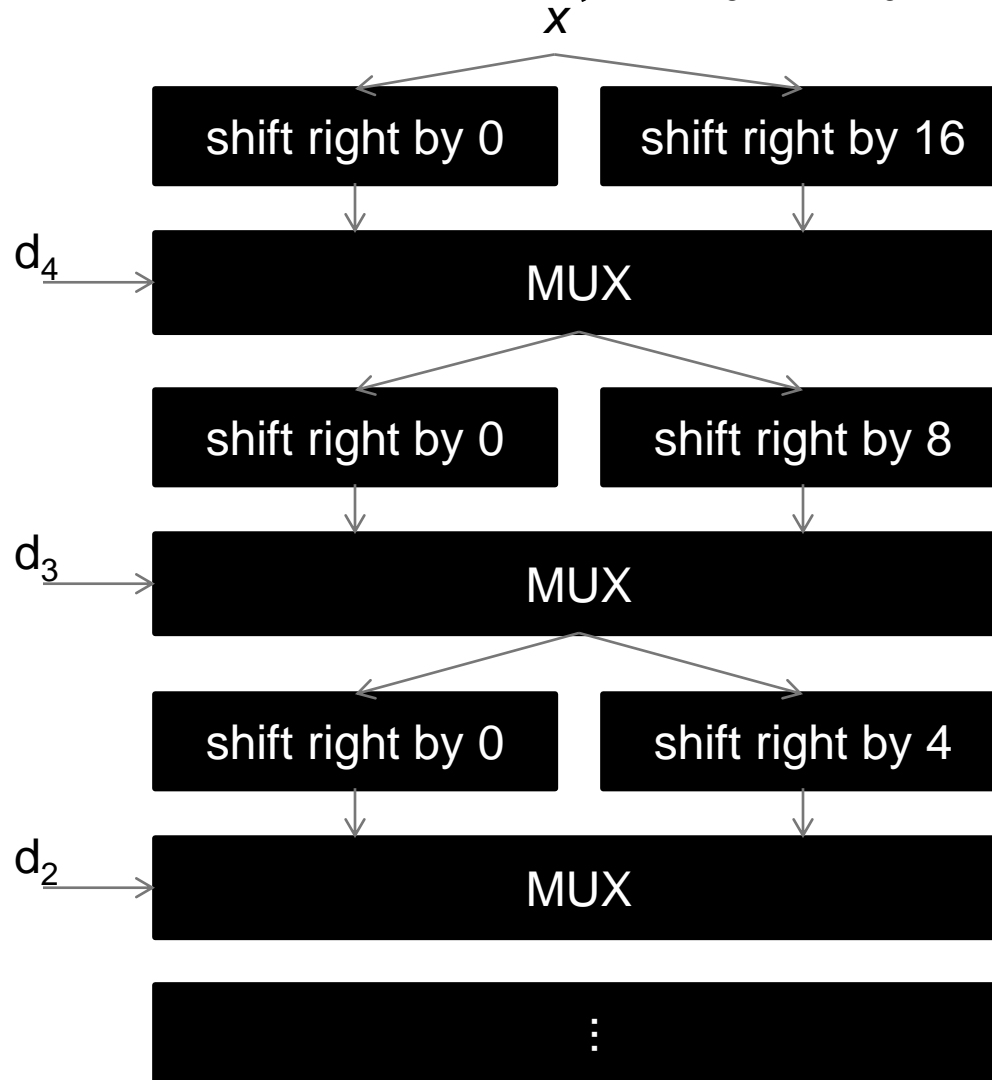
MUX, EX 7 B

- deplasare a unui numar x cu d poziții la dreapta



MUX, EX 7 C

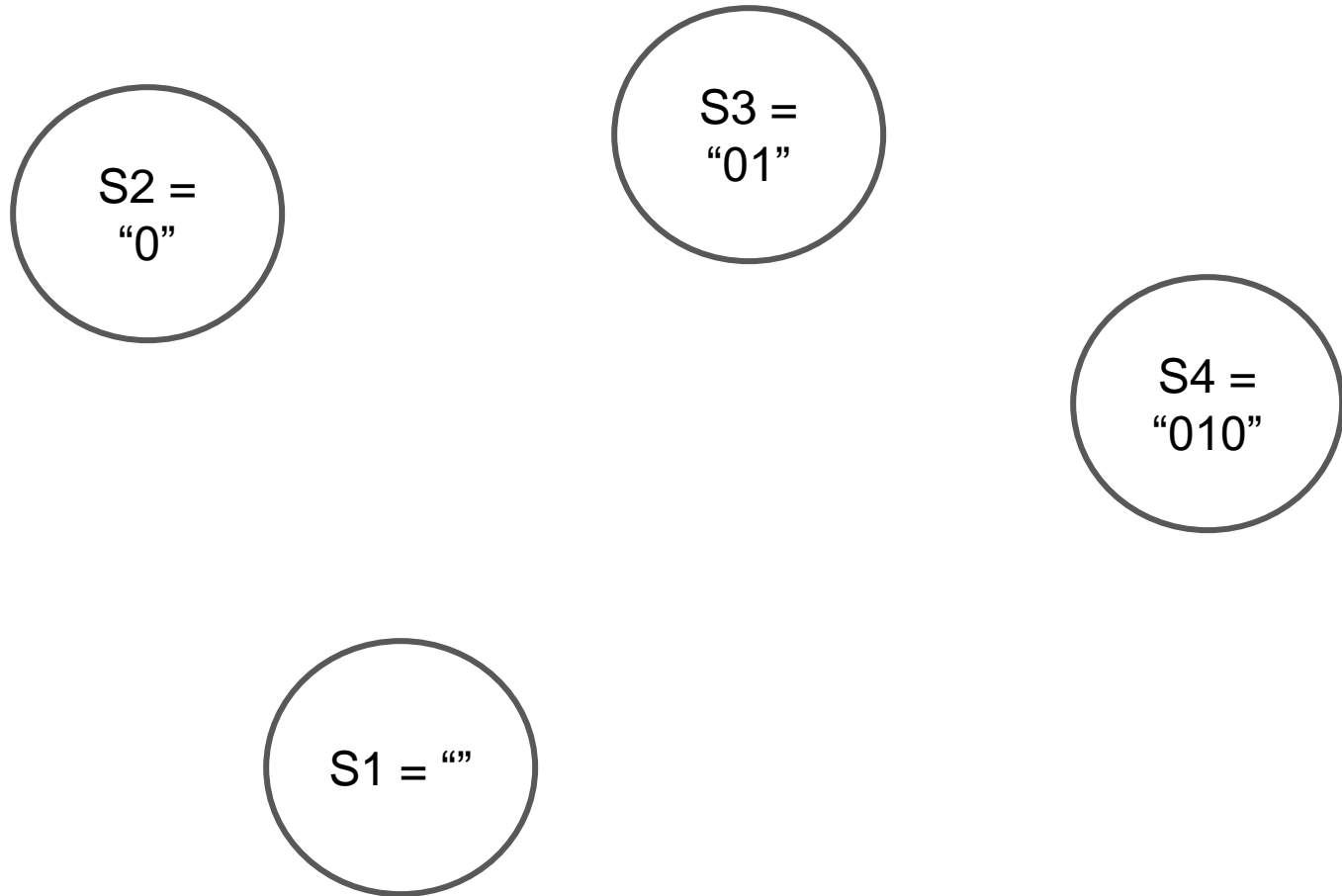
- presupunem că x este pe 32 de biți
- deplasarea d este pe 5 biți: $d_4d_3d_2d_1d_0$



avem nevoie de $\log_2 d$ MUX-uri

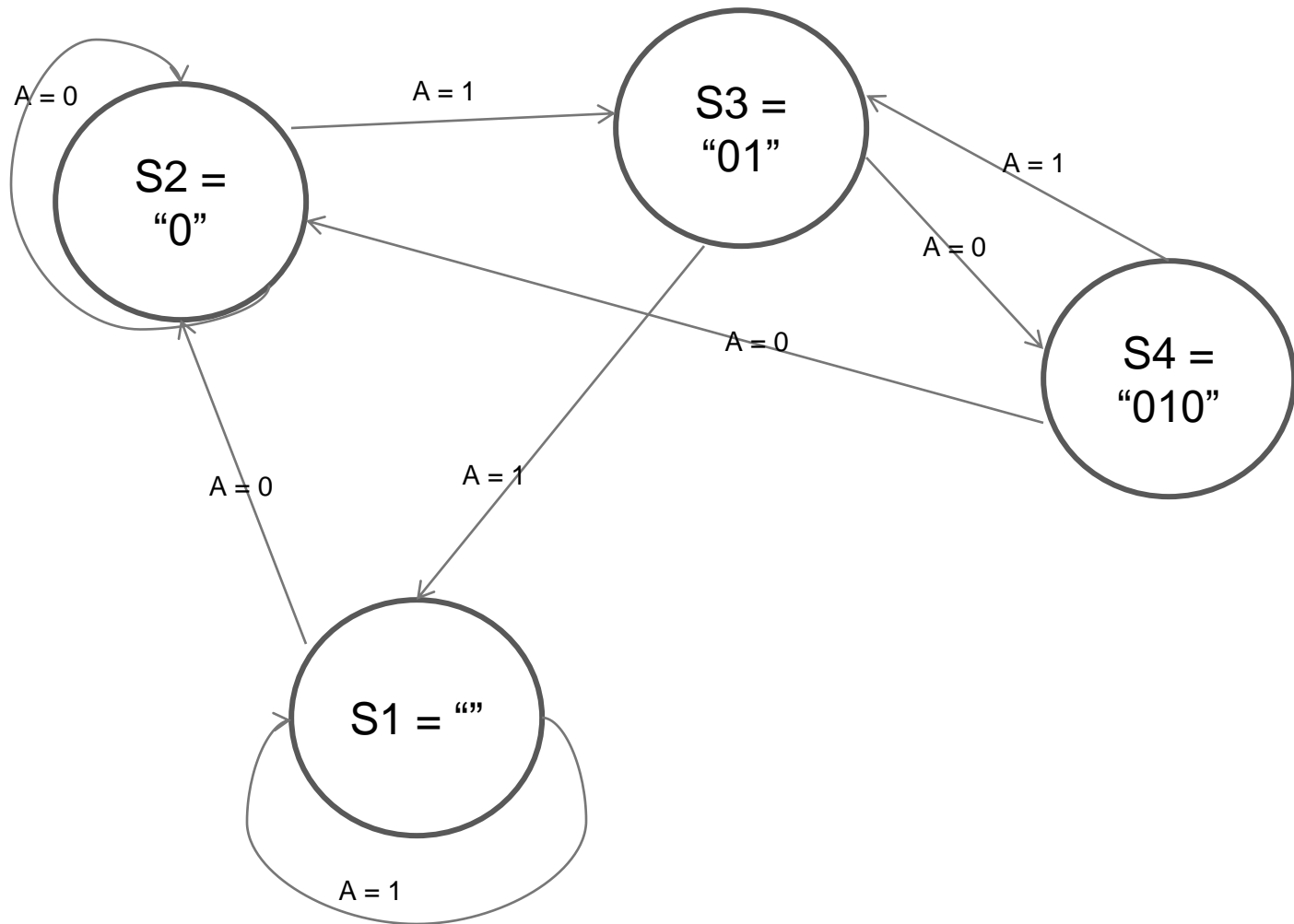
SECVENȚIAL, EX 10

- definim 4 stări

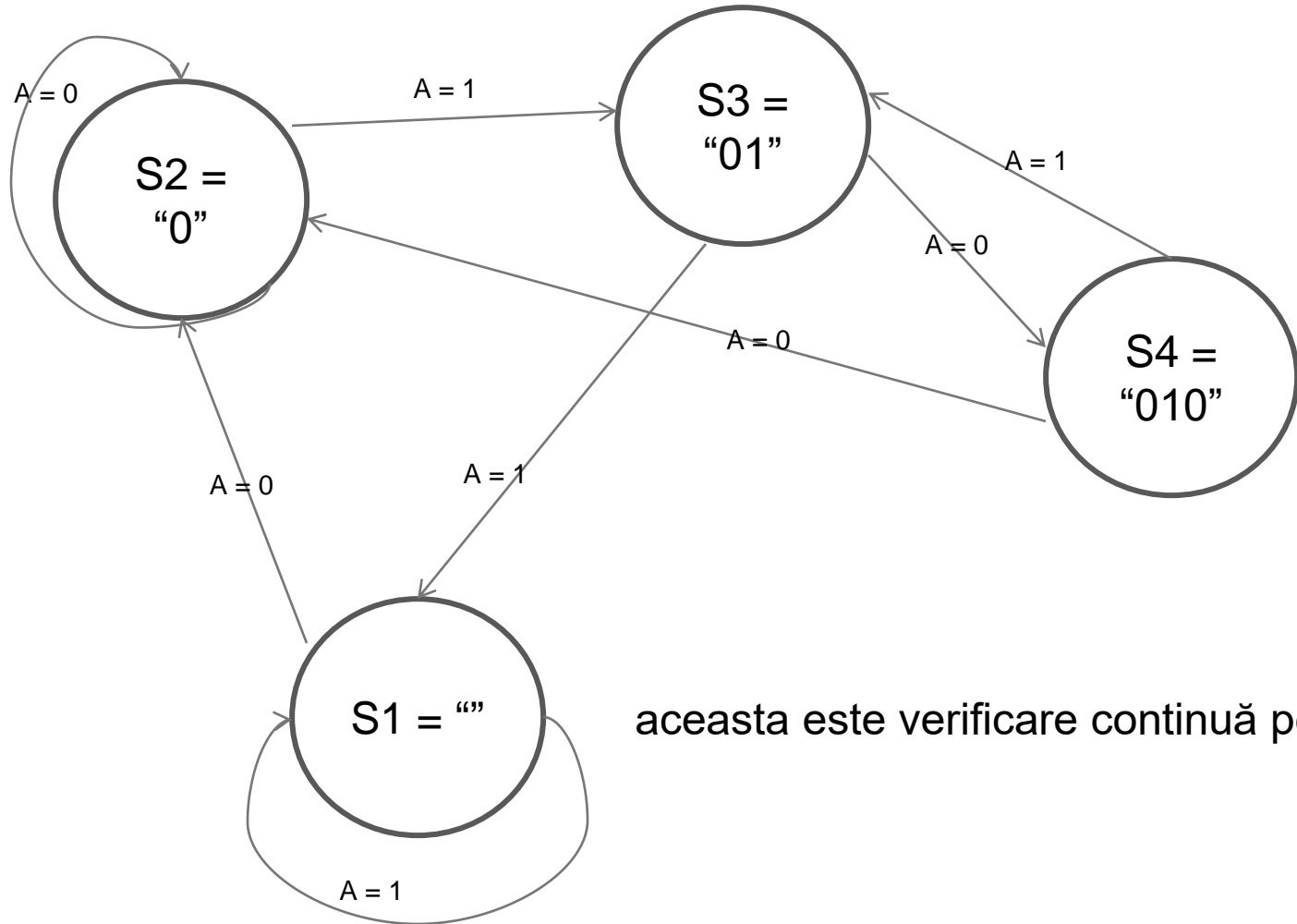


- care sunt tranzițiile între aceste stări?

SECVENȚIAL, EX 10

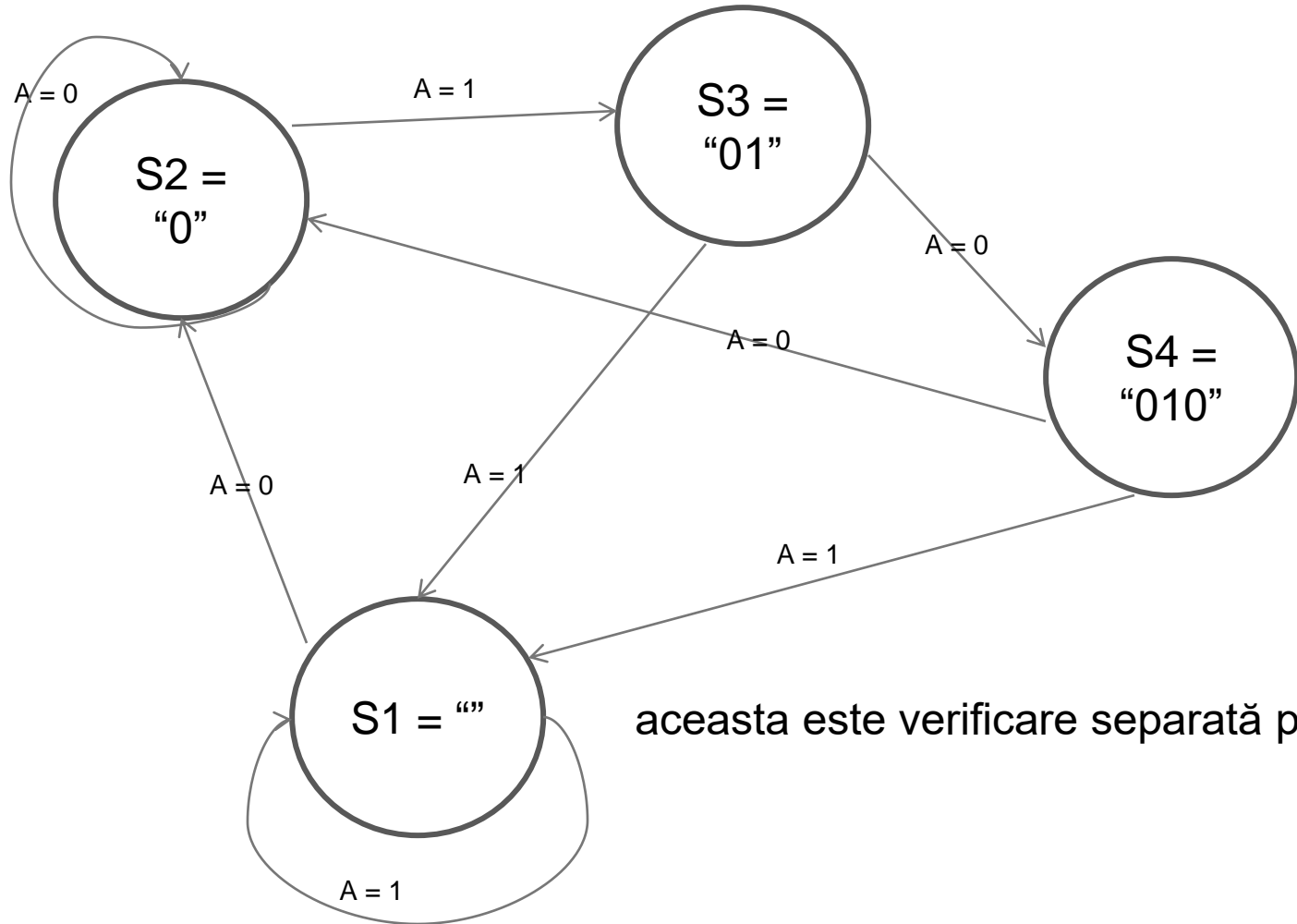


SECVENȚIAL, EX 10



aceasta este verificare continuă pentru 010

SECVENȚIAL, EX 10

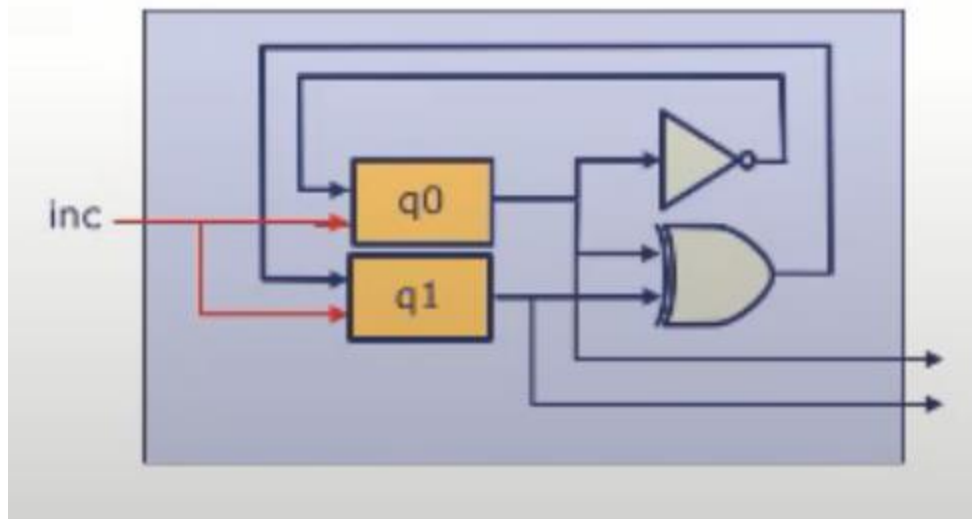


aceasta este verificare separată pentru 010

COUNTER 2 BIȚI, EX 12

- $q_0^{(t+1)} = !INC \times q_0^{(t)} + INC \times !q_0^{(t)}$, $q_1^{(t+1)} = !INC \times q_1^{(t)} + INC \times (q_1^{(t)} \otimes q_0^{(t)})$

$q_1^{(t)}$	$q_0^{(t)}$	$q_1^{(t+1)}$	$q_0^{(t+1)}$
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0



- aici INC e pe post de semnal Enable

COUNTER 2 BİTİ, EX 12

- counter cod Gray

$q_2^{(t)}$	$q_1^{(t)}$	$q_0^{(t)}$	$q_2^{(t+1)}$	$q_1^{(t+1)}$	$q_0^{(t+1)}$
0	0	0	0	0	1
0	0	1	0	1	1
0	1	0	1	1	0
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	0	1

- $$q_0^{(t+1)} = !q_2^{(t)}!q_1^{(t)}!q_0^{(t)} + !q_2^{(t)}!q_1^{(t)}q_0^{(t)} + q_2^{(t)}q_1^{(t)}!q_0^{(t)} + q_2^{(t)}q_1^{(t)}q_0^{(t)}$$
$$= q_2^{(t)}q_1^{(t)} + !q_2^{(t)}!q_1^{(t)}$$
- $q_1^{(t+1)} = \dots$
- $q_2^{(t+1)} = \dots$

CMMDC, EX 13

- implementați algoritmul lui Euclid pentru CMMDC folosind un circuit secvențial.
- exemplu: se dau $a = 15$, $b = 6$
 - pasul 1: $a = 9$, $b = 6$ ($a := a - b$)
 - pasul 2: $a = 3$, $b = 6$ ($a := a - b$)
 - pasul 3: $a = 6$, $b = 3$ ($a, b := b, a$)
 - pasul 4: $a = 3$, $b = 3$ ($a := a - b$)
 - pasul 5: $a = 0$, $b = 3$
 - răspunsul este 3
- codul python:

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

CMMDC, EX 13

- **codul python:**

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

- avem variabilele $a^{(t)}$ și $b^{(t)}$
- conform algoritmului de mai sus avem ecuațiile de evoluție
 - facem ceva doar dacă $a^{(t)} \neq 0$

CMMDC, EX 13

- **codul python:**

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

- avem variabilele $a^{(t)}$ și $b^{(t)}$
- conform algoritmului de mai sus avem ecuațiile de evoluție
 - facem ceva doar dacă $a^{(t)} \neq 0$
 - $p = (a^{(t)} > b^{(t)})$
 - $a^{(t+1)} = ?$
 - $b^{(t+1)} = ?$

CMMDC, EX 13

- **codul python:**

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

- avem variabilele $a^{(t)}$ și $b^{(t)}$
- conform algoritmului de mai sus avem ecuațiile de evoluție
 - facem ceva doar dacă $a^{(t)} \neq 0$
 - $p = (a^{(t)} > b^{(t)})$
 - $a^{(t+1)} = p \times (a^{(t)} - b^{(t)}) + !p \times b^{(t)}$
 - $b^{(t+1)} = p \times b^{(t)} + !p \times a^{(t)}$

CMMDC, EX 13

- **codul python:**

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

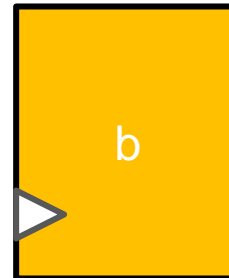
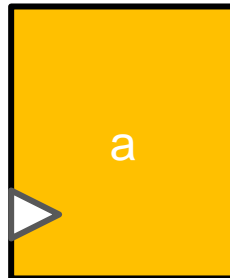
ce fel de circuite sunt a și b?

CMMDC, EX 13

- **codul python:**

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

când sunt acești regiștri activi?



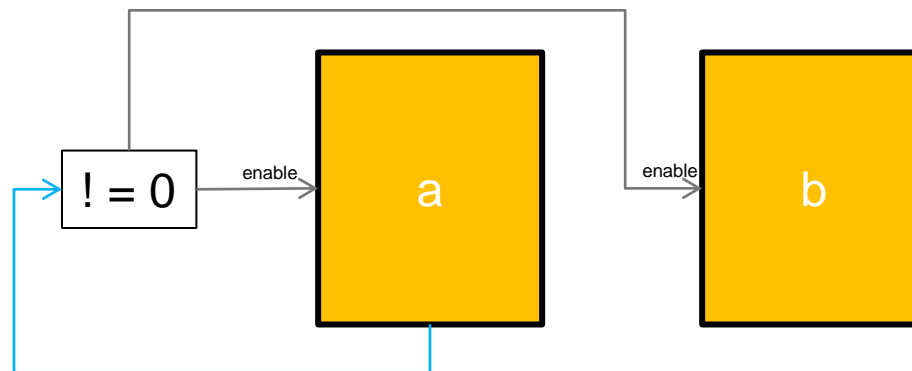
CMMDC, EX 13

- codul python:

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

$$p = (a^{(t)} > b^{(t)})$$
$$a^{(t+1)} = p \times (a^{(t)} - b^{(t)}) + !p \times b^{(t)}$$
$$b^{(t+1)} = p \times b^{(t)} + !p \times a^{(t)}$$

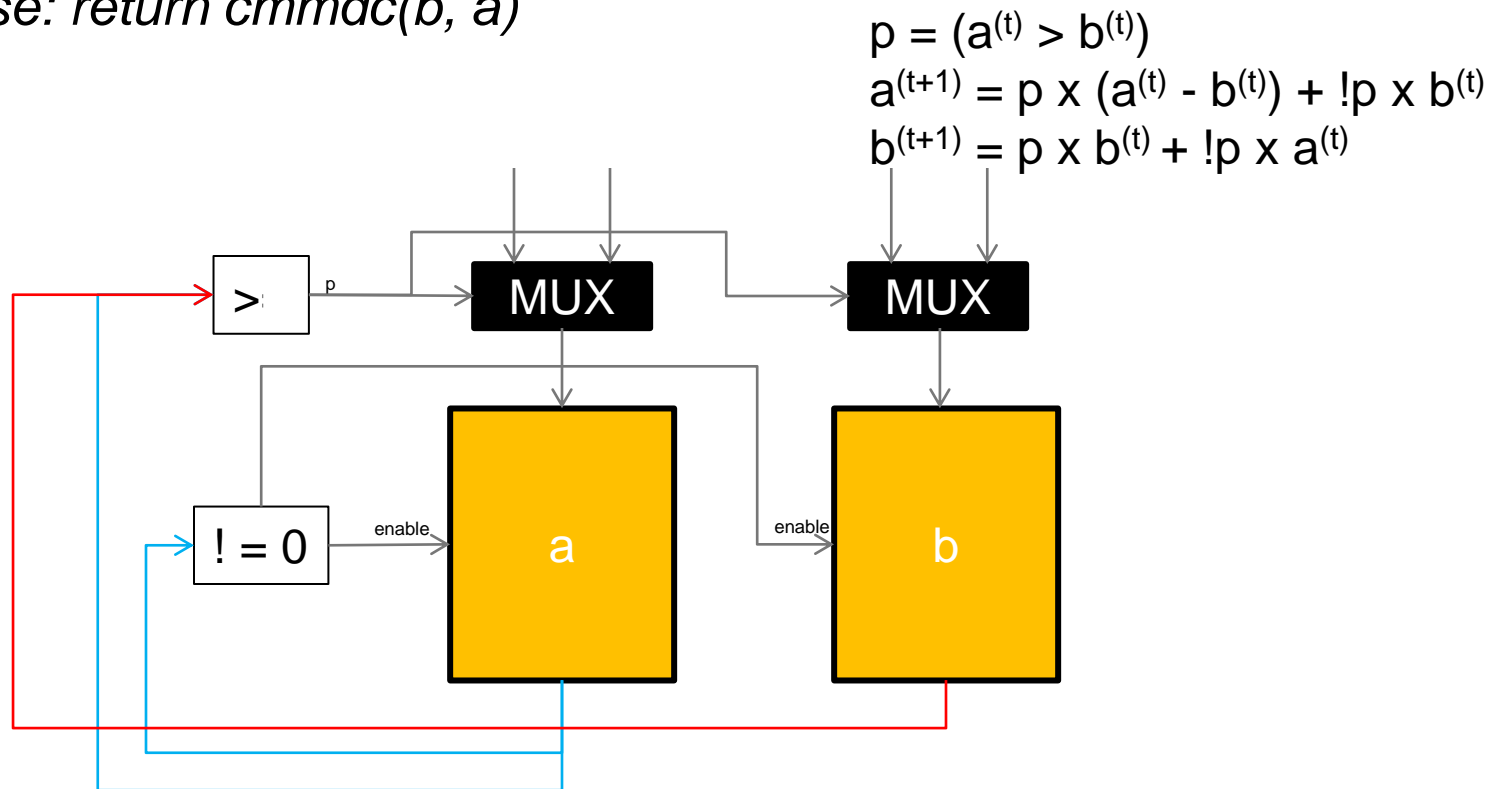
cum se schimbă a și b?



CMMDC, EX 13

- codul python:

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```

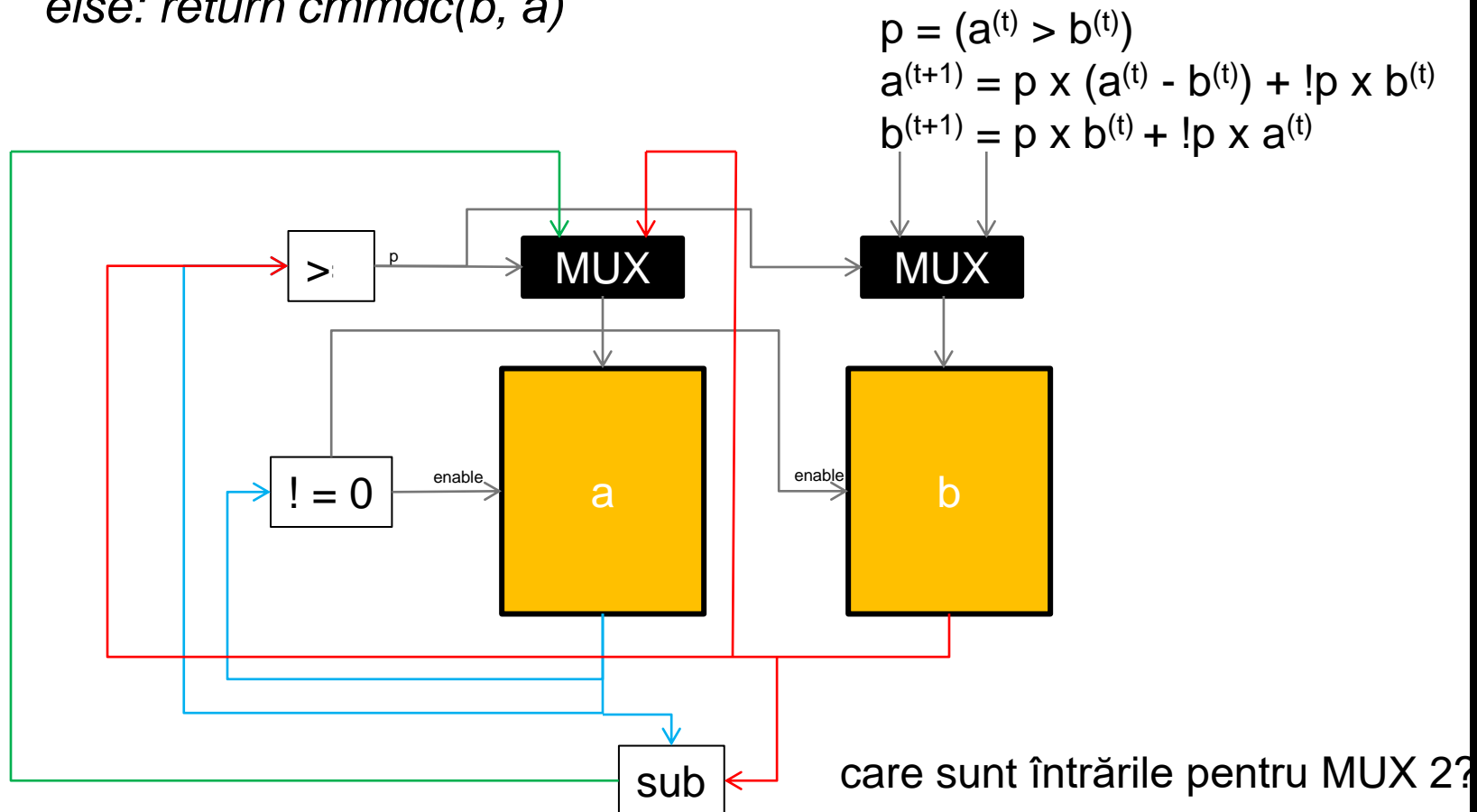


care sunt întrările pentru MUX 1?

CMMDC, EX 13

- codul python:

```
def cmmdc(a, b):  
    if a == b: return b  
    elif a > b: return cmmdc(a-b, b)  
    else: return cmmdc(b, a)
```



CMMDC, EX 13

- codul python:

```
def cmmdc(a, b):
```

```
    if a == b: return b
```

```
    elif a > b: return cmmdc(a-b, b)
```

```
    else: return cmmdc(b, a)
```

