Cheat Sheet Seminar

- Seminar 0x00

7. Am discutat la curs de modul în care aflăm valoarea unui număr scris în format complement față de 2: inversăm biții și adunăm 1. Demonstrați că această procedură este corectă.

Let's examine two useful shortcuts when working with two's complement numbers. The first shortcut is a quick way to negate a two's complement binary number. Simply invert every 0 to 1 and every 1 to 0, then add one to the result. This shortcut is based on the observation that the sum of a number and its inverted representation must be $111 \ldots 111_{\text{two}}$, which represents -1. Since $x + \bar{x} = -1$, therefore $x + \bar{x} + 1 = 0$ or $\bar{x} + 1 = -x$. (We use the notation $\bar{x}$ to mean invert every bit in x from 0 to 1 and vice versa.)

care este cantitatea de informație din pachet când scoatem cărțile pentru prima dată?

este 0, cărțile sunt ordonate crescător

amestecăm aleator cărțile, câtă informație avem acum?

- în câte feluri putem combina cele 52 de cărți?
- 52!
- deci informația este $\log_2(52!)$
    - cum calculăm valoarea asta?
    - $\log_2(a \times b) = \log_2(a) + \log_2(b)$
    - $\log_2(52!) = 225.6$ biți
    - cu aproximarea lui Stirling: $\log_2(52!) \approx 52\log_2(52) - 52\log_2 e = 221.4$ biți
- algoritmic, cum amestecăm cărțile (eficient)?
    - aveți la dispoziție o funcție care returnează o valoare aleatoare în intervalul [0,1]

- Seminar 0x01

5. Se dă un număr natural x pe N biți. Câtă informație am câștigat dacă:

(a) ni se spune despre x că are exact două valori "1" în reprezentarea sa binară;

(b) ni se spune despre x că are exact $N/2$ valori "1" în reprezentarea sa binară;

(c) ni se spune despre x că are o secvență continuă de $N/4$ biți de "1" în reprezentarea sa binară (restul biților sunt "0");

(d) ni se spune despre x că are MSB setat la "1";

(e) ni se spune despre x că este impar;

(f) ni se spune despre x că este o putere a lui 2;

(g) ni se spune despre x că are primii $N/2$ biți din reprezentarea sa binară setați la "0";

(h) ni se spune despre x că este un număr prim (aici doar o estimare aproximativă este posibilă);

(i) ni se spune despre x că are în reprezentarea sa binară un număr par de biți setați la "1";

(j) ni se spune că $x = 42$.

a) $p = C_2^N / 2^N$

b) $p = C_{N/2}^N / 2^N$, **presupunem N par**

c) $p = (\frac{3}{4}N + 1) / 2^N$, **presupunem N divizibil la 4**

d) $p = 2^{N-1} / 2^N = 1 / 2$

e) $p = 2^{N-1} / 2^N = 1 / 2$

f) $p = N / 2^N$

g) $p = 2^{N/2} / 2^N = 1 / 2^{N/2}$, **presupunem N par**

h) $p \approx [(2^N - 1) / \ln(2^N - 1)] / 2^N \approx [2^N / \ln(2^N)] / 2^N = 1 / \ln(2^N)$

i) $p = (1 + \sum_{i=1}^{N/2} C_{2i}^N) / 2^N = 1 / 2$, **presupunem N par**

j) $p = 1 / 2^N$

- Seminar 0x02

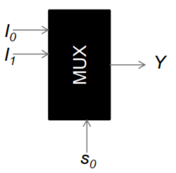
DE MORGAN, EX 3

- **!(!A+!B) = AB**
- **!(!A!B) = A+B**
- **!(A+B+C) = !A!B!C**
- **!(ABC) = !A+!B+!C**
- **!(A+B)!A!B = !A!B**
- **!(AB)(!A+!B) = !A+!B**
- **!(A+B)(!A+!B) = !A!B**
- **!A!B!(AB) = !A!B**
- **C+!(CB) = 1**
- **!(AB)(!A+B)(!B+!B) = !A!B**

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

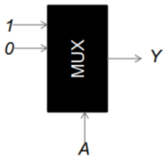


I_0	I_1	s_0	Y
*	*	0	I_0
*	*	1	I_1

care este relația ieșire-intrare?

- $Y = I_0\bar{s}_0 + I_1 s_0$

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

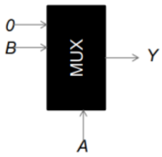


I_0	I_1	$s_0\,(A)$	Y
1	0	0	I_0
1	0	1	I_1

un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND

- implementați o poartă NOT cu un MUX: Y = NOT A
- Y = $I_0\bar{s}_0 + I_1 s_0 = 1\bar{A} + 0A = \bar{A}$

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

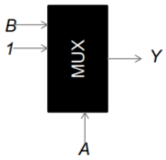


I_0	$I_1\,(B)$	$s_0\,(A)$	Y
0	B	0	I_0 (0)
0	B	1	I_1 (B)

un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND

- implementați o poartă AND cu un MUX: Y = A AND B
- Y = $I_0\bar{s}_0 + I_1 s_0 = 0\bar{A} + BA = AB$

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

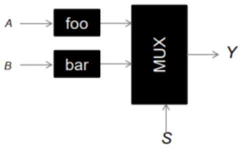


$I_0\,(B)$	I_1	$s_0\,(A)$	Y
B	1	0	I_0 (B)
B	1	1	I_1 (1)

un MUX este un circuit universal, adică poate implementa porți NOT, OR și AND

- implementați o poartă OR cu un MUX: Y = A OR B
- Y = $I_0\bar{s}_0 + I_1 s_0 = B\bar{A} + 1A = A + B\bar{A}$ = A + B [vezi ex. 4 f)]

- **Y = S ? foo(A) : bar(B)**



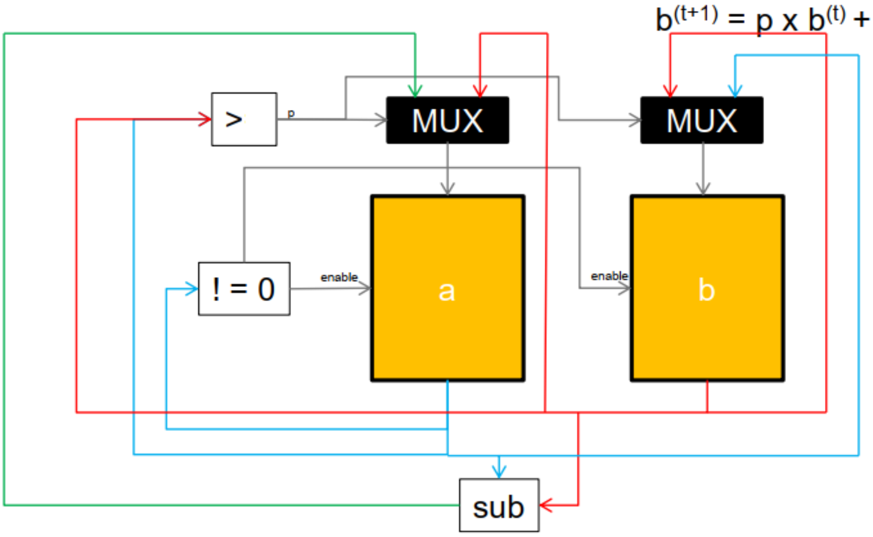
I_0 foo(X)	I_1 bar(Y)	S	Y
*	*	0	I_1
*	*	1	I_0

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

- **codul python:**

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

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



- Seminar 0x03