ARHITECTURA SISTEMELOR DE CALCUL - CURS 0x01

EVOLUȚIA SISTEMELOR DE CALCUL ȘI SISTEMUL BINAR DE CALCUL

Cristian Rusu

CUPRINS

- **scurt istoric al sistemelor de calcul**
- **sistemul binar**
  - reprezentarea binară
  - reprezentarea în complement față de doi
  - funcții logice
- **referințe bibliografice**

SCURT ISTORIC AL SISTEMELOR DE CALCUL

- **contribuții majore ale:**
  - Blaise Pascal
  - Gottfried Wilhelm von Leibniz
  - George Boole
  - Charles Babbage
  - Ada Lovelace
  - Konrad Zuse
  - Alan Turing
  - John von Neumann
  - Claude Shannon
  - după 1945 interestul în sisteme de calcul a crescut semnificativ iar progresul este dificil de atribuit doar unor indivizi

BLAISE PASCAL (1623 - 1662)

- **în 1642, când încă nu avea 19 ani, crează Pascaline**
  - un calculator mecanic
  - capabil de adunări/scăderi (utilizat pentru calcul de taxe)
  - nu a fost o mașină practică
  - mai puțin de 50 au fost create
  - era utilizată pe post de “jucarie” pentru aristocrație

- **limbajul Pascal e numit în onoarea lui**

https://en.wikipedia.org/wiki/Blaise_Pascal

GOTTFRIED WILHELM VON LEIBNIZ (1646 – 1716)

- **toate contribuțile lui sunt imposibil de enumerat**

- **două contribuți majore:**
  - studiază sistemul binar
  - extinde mașina lui Pascal, adăugând operațiile de înmulțire și împărțire – tot o mașină mecanică creată în 1673

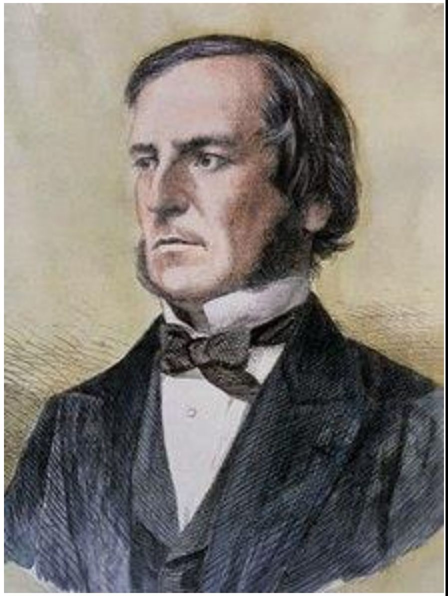
https://en.wikipedia.org/wiki/Gottfried_Wilhelm_Leibniz
http://www.leibniz-translations.com/binary.htm

GEORGE BOOLE (1815 – 1864)

- **scrie "The Laws of Thought" (1854)**

- **introduce logica booleană și analizează operațiile de bază**
  - negația
  - conjuncția
  - disjuncția
  - disjuncția exclusivă

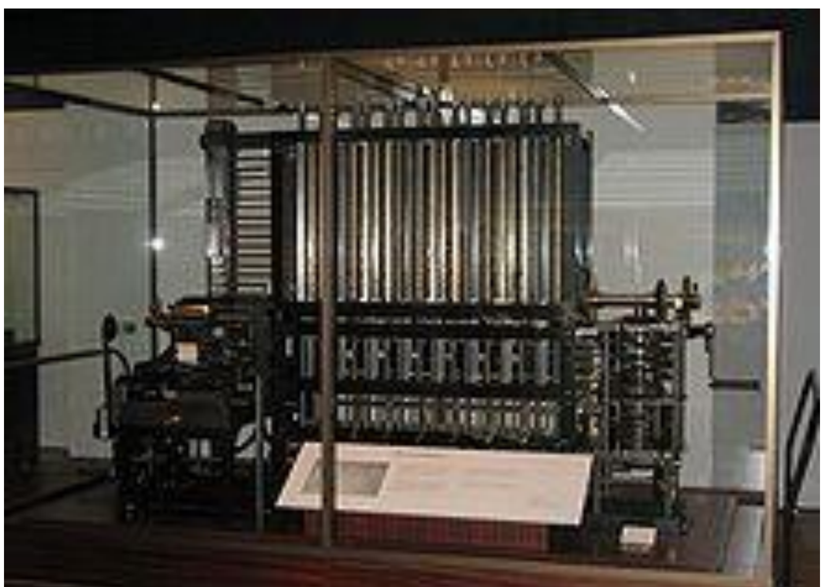
- **toate acestea stau la baza teoriei informației**

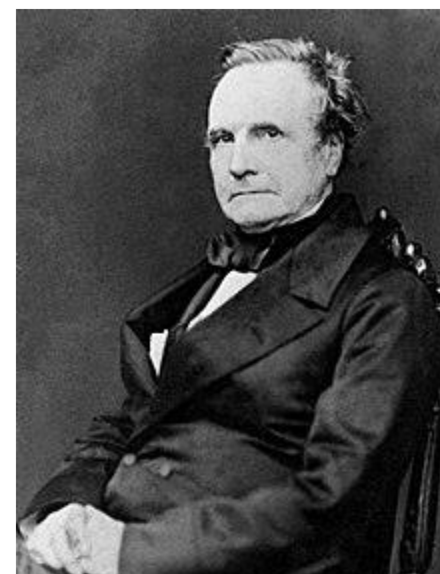


https://en.wikipedia.org/wiki/George_Boole

CHARLES BABBAGE (1791 – 1871)

- **proiectează Mașina Diferențială Nr. 2 (Difference Engine No. 2)**
- **doar teoretic, design-ul este realizat de abia în 1991**
- **totuși, este prima mașină de calcul (mecanică) programabilă**
- **prototipurile sale aveau deja peste 13 tone**
- **este considerat “tatăl calculatoarelor moderne”**

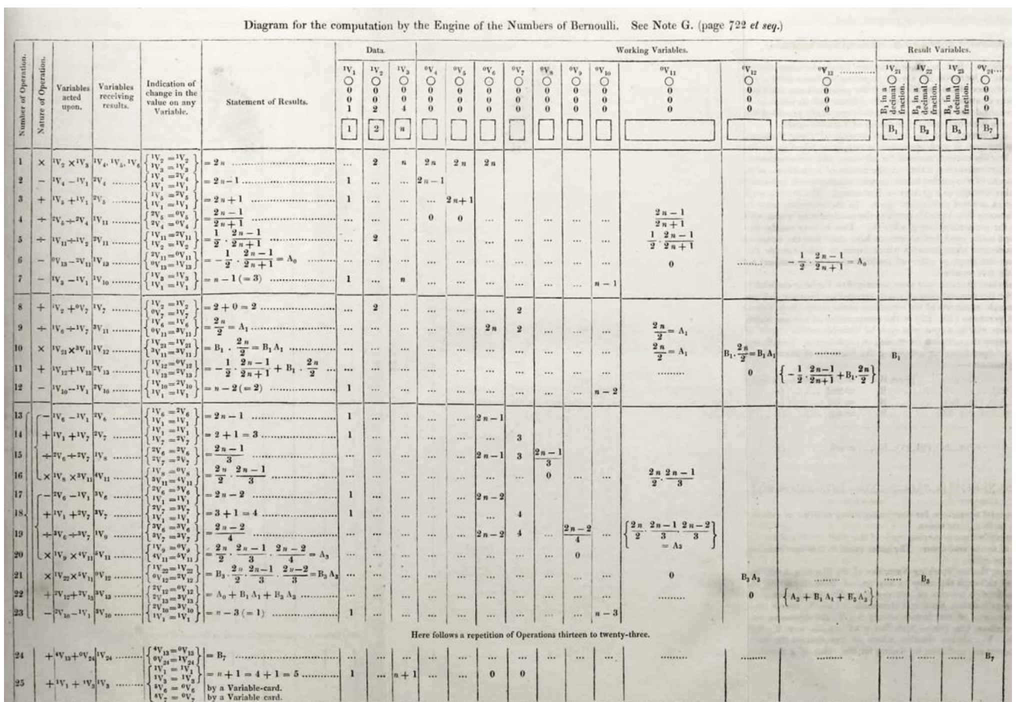




https://en.wikipedia.org/wiki/Charles_Babbage

ADA LOVELACE (1815 – 1852)

- **colaboratoare a lui Babbage**
- **scrie primul program, calculează numere Bernoulli**
- **nu existau limbaje de programare, dar ea a descris o serie de pași care sa fie executați de o mașină**
- **este considerată primul “programator”**



https://en.wikipedia.org/wiki/Ada_Lovelace

KONRAD ZUSE (1910 – 1995)

- introduce o serie de calculatoare: Z1, Z2, Z3 si Z4
- primele prototipuri in 1940-1941
- folosește sistemul binar
- instrucțiunile sunt stocate pe o bandă
- introduce reprezentarea și calculul în virgulă mobilă
- face aproape totul în izolare (1936-1945)

https://en.wikipedia.org/wiki/Konrad_Zuse

ALAN TURING (1912 – 1954)

- **celebru pentru publicul larg pentru contribuția lui în spargerea rapidă a mesajelor Enigma utilizând mașina "The Bombe"**
  - practic, mașina făcea un brute-force search pentru a reduce numărul de posibilități în decriptarea mesajelor

- **introduce Mașina Turing**
  - un model teoretic pentru a implementa orice algoritm
  - conceptul de Turing-complete
    - intuiția: un sistem care poate recunoaște și analiza seturi de reguli pentru manipularea datelor (o cantitate infinită, teoretic)

- **introduce Testul Turing**
  - The imitation game: "The original question, 'Can machines think!' I believe to be too meaningless to deserve discussion" A. Turing

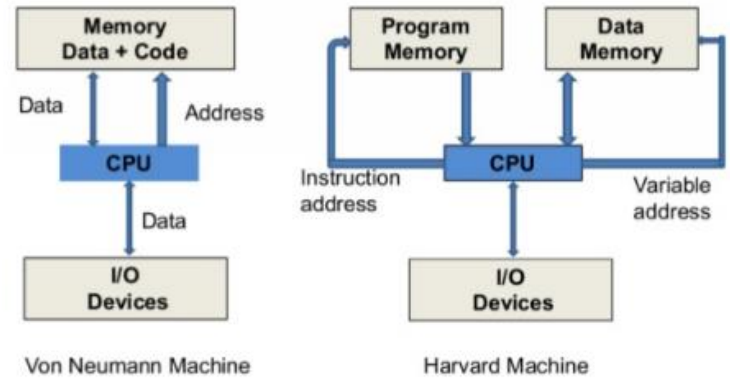
https://en.wikipedia.org/wiki/Alan_Turing
https://academic.oup.com/mind/article/LIX/236/433/986238

JOHN VON NEUMANN (1903 – 1957)

- **considerat unul dintre cei mai buni matematicieni ai ultimului secol, aduce contribuții în numeroase domenii**

- **ajută la crearea primul calculator electronic ENIAC (Electronic Numerical Integrator And Computer), 1939-1944**
- **îmbunătățește ENIAC ajutând la crearea EDVAC (Electronic Discrete Variable Automatic Computer), sistemul este binar și are programe stocate**

- **introduce arhitectura von Neumann**

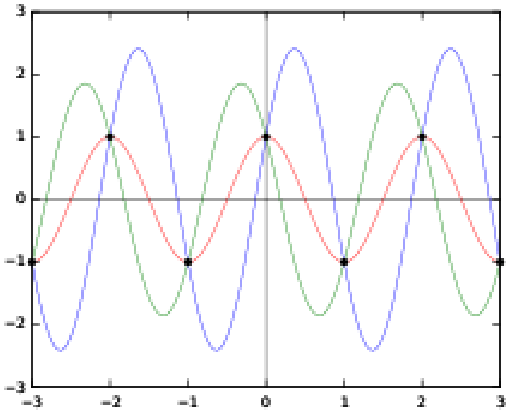


https://en.wikipedia.org/wiki/John_von_Neumann
https://vivadifferences.com/5-major-difference-between-von-neumann-and-harvard-architecture/

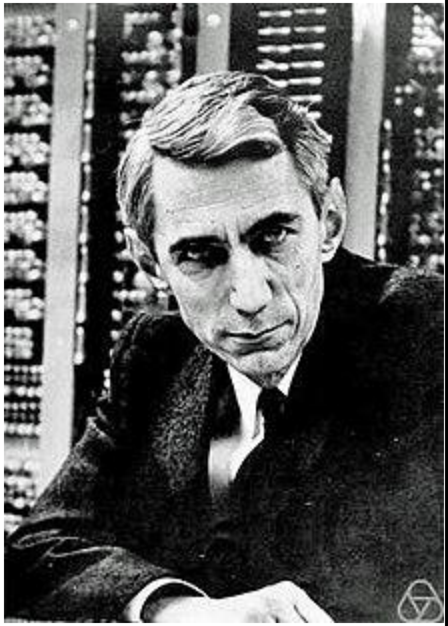
CLAUDE SHANNON (1916 – 2001)

- **considerat “părintele teoriei informației”**

- **trei contribuții exceptionale:**
  - demonstrează faptul că probleme de logică Booleană pot fi rezolvate cu circuite electronice
  - teorema de eșantionare Shannon-Nyquist



  - inventează teoria informației
    - cursul următor discutăm detaliat



https://en.wikipedia.org/wiki/Claude_Shannon
https://web.archive.org/web/19980715013250/http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf

IDEILE MARI

- **de la mecanic la electric**
- **de la o mașină care face un singur lucru automat, la o mașină care este programabilă**
- **design modular**
- **teorie despre ce este posibil pe aceste mașini**
- **dorința de a face lucrurile optim, la limită și fără risipă**

POST SHANNON ...

- după al doilea razboi mondial, cercetarea în domeniul calculatoare începe un ritm exponential

- sunt multe, mici invenții și discoperiri tehnologice pe parcurs

- nu avem cum să le acoperim pe toate

- actorii importanți în domeniu au devenit grupuri profesionale (ex: IEEE, ACM, etc.) și state (ex: State Unite, programe de cercetare DARPA, etc.)

- la baza acestui progres stau niște concepte de matematică fundamentale, începem cu sistemul binar

SISTEMUL BINAR

- **este baza sistemelor moderne de calcul**

- **orice număr (întreg sau, general, real) poate fi reprezentat printr-un număr (potential infinit) de biți**

- **bit = *bi*nary digi*t***

- **ne aflăm în sistemul de numărare cu baza *B* = *2***

- **avem disponibile doar două cifre: 0 și 1**

SISTEMUL BINAR

- **un număr natural reprezentat în baza *B = 2***

bit b_i:	…	0	1	1	1	1	0	0	0	1
2^i:	…	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- $$x = \sum_{i=0}^{N-1} b_i 2^i$$

- ***N*** **este numărul de biți pe care îl folosim în reprezentare**

- **în exemplul de mai sus:**
  - care este numărul reprezentat?
  - pe câți biți este reprezentat acest număr?

SISTEMUL BINAR

- **un număr natural reprezentat în baza *B = 2***

bit b_i:	…	0	1	1	1	1	0	0	0	1
2^i:	…	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- $$x = \sum_{i=0}^{N-1} b_i 2^i$$

- ***N* este numărul de biți pe care îl folosim în reprezentare**

- **în exemplul de mai sus:**
  - $0 \times 2^8 + 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 241$
  - mai sus avem *N* = *9*, dar defapt avem nevoie de *N* = *8*

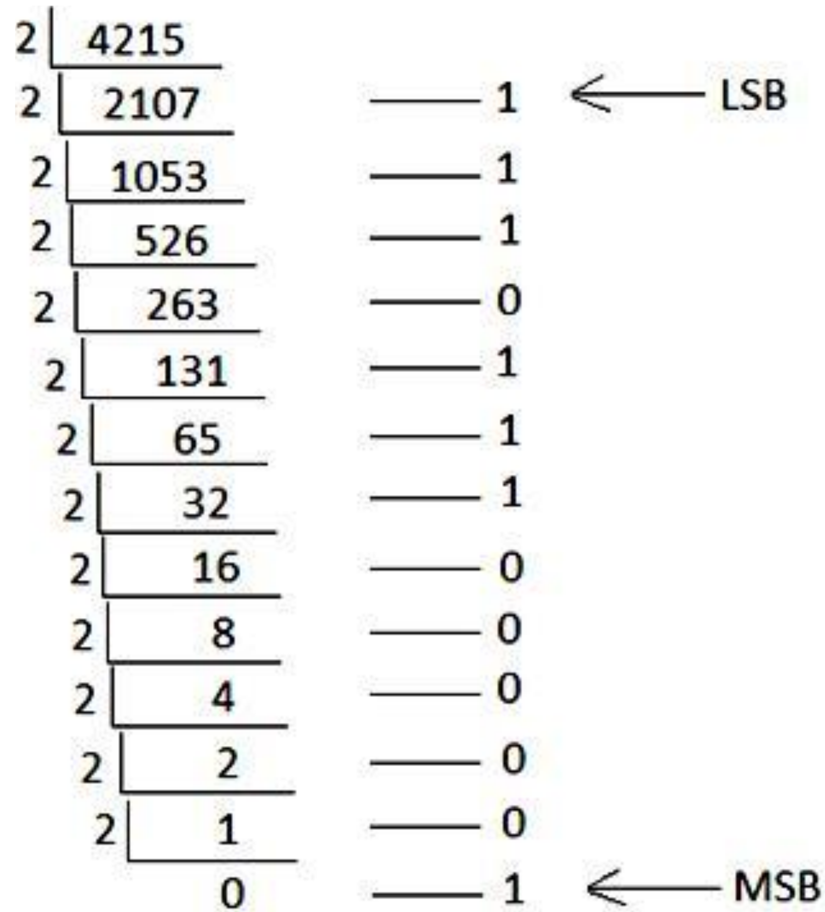
SISTEMUL BINAR

- **intuiția noastră este în baza *B = 10***
- **este folositor să abstractizăm și să considerăm baza generală *B***

- **în baza *B* avem:**
  - cifre de la *0* la *B-1* (restul se numesc numere)
  - reprezentarea este $x = \sum_{i=0}^{N-1} b_i B^i$
  - bitul b_0 se numește Least Significand Bit (LSB) iar bitul b_{N-1} se numește Most Significant Bit (MSB)
  - cum reprezentăm un număr din baza 10 în baza *B*?
    - împărțiri repetate cu *B* și păstrăm restul

SISTEMUL BINAR

- **un exemplu explicit: $(4215)_{10} = (1000001110111)_2$**



https://www.math-only-math.com/conversion-of-numbers.html

SISTEMUL BINAR

- **conversia între sisteme de numărare este foarte folositoare**

- **ce se întâmplă în baza $B = 10$?**
  - ce se întâmplă dacă vrem să trecem din baza $B_{old} = 10$ în noua bază $B_{new} = 100$?
    - câte cifre sunt în baza $B_{new} = 100$?

SISTEMUL BINAR

- **conversia între sisteme de numărare este foarte folositoare**

- **ce se întâmplă în baza $B = 10$?**
  - ce se întâmplă dacă vrem să trecem din baza $B_{old} = 10$ în noua bază $B_{new} = 100$?
    - câte cifre sunt în baza $B_{new} = 100$? 100
    - care sunt ciferele în baza $B_{new} = 100$?

.

SISTEMUL BINAR

- **conversia între sisteme de numărare este foarte folositoare**

- **ce se întâmplă în baza $B = 10$?**
  - ce se întâmplă dacă vrem să trecem din baza $B_{old} = 10$ în noua bază $B_{new} = 100$?
    - câte cifre sunt în baza $B_{new} = 100$? 100
    - care sunt ciferele în baza $B_{new} = 100$? de la cifra “0” la cifra “99”
    - primesc un număr în baza 10, cum îl transform în baza 100?
    - ex: $(4837103)_{10} = (?????)_{100}$

SISTEMUL BINAR

- **conversia între sisteme de numărare este foarte folositoare**

- **ce se întâmplă în baza $B = 10$?**
  - ce se întâmplă dacă vrem să trecem din baza $B_{old} = 10$ în noua bază $B_{new} = 100$?
    - câte cifre sunt în baza $B_{new} = 100$? 100
    - care sunt ciferele în baza $B_{new} = 100$? de la cifra “0” la cifra “99”
    - primesc un număr în baza 10, cum îl transform în baza 100?
    - ex: $(4837103)_{10}$ = $(“4”\ \ “83”\ \ “71”\ \ “3”)_{100}$
    - cum am obținut rezultatul? doar am grupal cifre consecutive, câte două – de ce două?

SISTEMUL BINAR

- **conversia între sisteme de numărare este foarte folositoare**

- **ce se întâmplă în baza *B* = *10*?**
  - ce se întâmplă dacă vrem să trecem din baza $B_{old} = 10$ în noua bază $B_{new} = 100$?
    - câte cifre sunt în baza $B_{new} = 100$? 100
    - care sunt ciferele în baza $B_{new} = 100$? de la cifra “0” la cifra “99”
    - primesc un număr în baza 10, cum îl transform în baza 100?
    - ex: $(4837103)_{10}$ = (“4” “83” “71” “3”)$_{100}$
    - cum am obținut rezultatul? doar am grupal cifre consecutive, câte două – de ce două?

Regula generală: când trecem din baza B în baza B^p trebuie doar sa grupăm noul număr în cate p cifre

SISTEMUL BINAR

- cineva spune: “Am cheltuit 1.000.000 de euro. O cifră enormă!!!”

  este corect?

SISTEMUL BINAR

- **cineva spune: “Am cheltuit 1.000.000 de euro. O cifră enormă!!!”**

- **1.000.000 e număr, nu cifră**
- **când poate să fie 1.000.000 cifră?**
  - doar dacă cel care vorbește se refera la numere într-o baza numerică $B \geq 1.000.001$
  - și atunci, ar trebui să spună “1.000.000”

- **pentru ultima dată**
  - în baza $B = 10$, cifrele sunt de la “0” la “9”
  - restul sunt numere
  - conceptul se generalizează pentru orice B

SISTEMUL BINAR

- **un exemplu explicit: $(111011100001)_2$**
  - în baza 4:
  - în baza 8:
  - în baza 16 (hexazecimal):

$\mathbf{0}_{hex}$	=	0_{dec}	=	0_{oct}	0	0	0	0
$\mathbf{1}_{hex}$	=	1_{dec}	=	1_{oct}	0	0	0	1
$\mathbf{2}_{hex}$	=	2_{dec}	=	2_{oct}	0	0	1	0
$\mathbf{3}_{hex}$	=	3_{dec}	=	3_{oct}	0	0	1	1
$\mathbf{4}_{hex}$	=	4_{dec}	=	4_{oct}	0	1	0	0
$\mathbf{5}_{hex}$	=	5_{dec}	=	5_{oct}	0	1	0	1
$\mathbf{6}_{hex}$	=	6_{dec}	=	6_{oct}	0	1	1	0
$\mathbf{7}_{hex}$	=	7_{dec}	=	7_{oct}	0	1	1	1
$\mathbf{8}_{hex}$	=	8_{dec}	=	10_{oct}	1	0	0	0
$\mathbf{9}_{hex}$	=	9_{dec}	=	11_{oct}	1	0	0	1
$\mathbf{A}_{hex}$	=	10_{dec}	=	12_{oct}	1	0	1	0
$\mathbf{B}_{hex}$	=	11_{dec}	=	13_{oct}	1	0	1	1
$\mathbf{C}_{hex}$	=	12_{dec}	=	14_{oct}	1	1	0	0
$\mathbf{D}_{hex}$	=	13_{dec}	=	15_{oct}	1	1	0	1
$\mathbf{E}_{hex}$	=	14_{dec}	=	16_{oct}	1	1	1	0
$\mathbf{F}_{hex}$	=	15_{dec}	=	17_{oct}	1	1	1	1

https://en.wikipedia.org/wiki/Binary_number

https://www.ascii-code.com/

SISTEMUL BINAR

- **un exemplu explicit: $(1110111000001)_2$**
  - în baza 4: $(“1” “11” “01” “11” “00” “00” “01”)_4 = (1313001)_4$
  - în baza 8:
  - în baza 16 (hexazecimal):

$\mathbf{0}_{hex}$	=	0_{dec}	=	0_{oct}	0	0	0	0
$\mathbf{1}_{hex}$	=	1_{dec}	=	1_{oct}	0	0	0	1
$\mathbf{2}_{hex}$	=	2_{dec}	=	2_{oct}	0	0	1	0
$\mathbf{3}_{hex}$	=	3_{dec}	=	3_{oct}	0	0	1	1
$\mathbf{4}_{hex}$	=	4_{dec}	=	4_{oct}	0	1	0	0
$\mathbf{5}_{hex}$	=	5_{dec}	=	5_{oct}	0	1	0	1
$\mathbf{6}_{hex}$	=	6_{dec}	=	6_{oct}	0	1	1	0
$\mathbf{7}_{hex}$	=	7_{dec}	=	7_{oct}	0	1	1	1
$\mathbf{8}_{hex}$	=	8_{dec}	=	10_{oct}	1	0	0	0
$\mathbf{9}_{hex}$	=	9_{dec}	=	11_{oct}	1	0	0	1
$\mathbf{A}_{hex}$	=	10_{dec}	=	12_{oct}	1	0	1	0
$\mathbf{B}_{hex}$	=	11_{dec}	=	13_{oct}	1	0	1	1
$\mathbf{C}_{hex}$	=	12_{dec}	=	14_{oct}	1	1	0	0
$\mathbf{D}_{hex}$	=	13_{dec}	=	15_{oct}	1	1	0	1
$\mathbf{E}_{hex}$	=	14_{dec}	=	16_{oct}	1	1	1	0
$\mathbf{F}_{hex}$	=	15_{dec}	=	17_{oct}	1	1	1	1

https://en.wikipedia.org/wiki/Binary_number

https://www.ascii-code.com/

SISTEMUL BINAR

- **un exemplu explicit: $(1110111000001)_2$**
  - în baza 4: $(“1” “11” “01” “11” “00” “00” “01”)_4 = (1313001)_4$
  - în baza 8: $(“1” “110” “111” “000” “001”)_8 = (16701)_8$
  - în baza 16 (hexazecimal):

$\mathbf{0}_{hex}$	=	0_{dec}	=	0_{oct}	0	0	0	0
$\mathbf{1}_{hex}$	=	1_{dec}	=	1_{oct}	0	0	0	1
$\mathbf{2}_{hex}$	=	2_{dec}	=	2_{oct}	0	0	1	0
$\mathbf{3}_{hex}$	=	3_{dec}	=	3_{oct}	0	0	1	1
$\mathbf{4}_{hex}$	=	4_{dec}	=	4_{oct}	0	1	0	0
$\mathbf{5}_{hex}$	=	5_{dec}	=	5_{oct}	0	1	0	1
$\mathbf{6}_{hex}$	=	6_{dec}	=	6_{oct}	0	1	1	0
$\mathbf{7}_{hex}$	=	7_{dec}	=	7_{oct}	0	1	1	1
$\mathbf{8}_{hex}$	=	8_{dec}	=	10_{oct}	1	0	0	0
$\mathbf{9}_{hex}$	=	9_{dec}	=	11_{oct}	1	0	0	1
$\mathbf{A}_{hex}$	=	10_{dec}	=	12_{oct}	1	0	1	0
$\mathbf{B}_{hex}$	=	11_{dec}	=	13_{oct}	1	0	1	1
$\mathbf{C}_{hex}$	=	12_{dec}	=	14_{oct}	1	1	0	0
$\mathbf{D}_{hex}$	=	13_{dec}	=	15_{oct}	1	1	0	1
$\mathbf{E}_{hex}$	=	14_{dec}	=	16_{oct}	1	1	1	0
$\mathbf{F}_{hex}$	=	15_{dec}	=	17_{oct}	1	1	1	1

https://en.wikipedia.org/wiki/Binary_number

https://www.ascii-code.com/

SISTEMUL BINAR

- **un exemplu explicit: $(1110111000001)_2$**
  - în baza 4: $(\text{“1” “11” “01” “11” “00” “00” “01”})_4 = (1313001)_4$
  - în baza 8: $(\text{“1” “110” “111” “000” “001”})_8 = (16701)_8$
  - în baza 16 (hexazecimal): $(\text{“1” “1101” “1100” “0001”})_{16} = (1DC1)_{16}$

0_{hex}	=	0_{dec}	=	0_{oct}	0	0	0	0
1_{hex}	=	1_{dec}	=	1_{oct}	0	0	0	1
2_{hex}	=	2_{dec}	=	2_{oct}	0	0	1	0
3_{hex}	=	3_{dec}	=	3_{oct}	0	0	1	1
4_{hex}	=	4_{dec}	=	4_{oct}	0	1	0	0
5_{hex}	=	5_{dec}	=	5_{oct}	0	1	0	1
6_{hex}	=	6_{dec}	=	6_{oct}	0	1	1	0
7_{hex}	=	7_{dec}	=	7_{oct}	0	1	1	1
8_{hex}	=	8_{dec}	=	10_{oct}	1	0	0	0
9_{hex}	=	9_{dec}	=	11_{oct}	1	0	0	1
A_{hex}	=	10_{dec}	=	12_{oct}	1	0	1	0
B_{hex}	=	11_{dec}	=	13_{oct}	1	0	1	1
C_{hex}	=	12_{dec}	=	14_{oct}	1	1	0	0
D_{hex}	=	13_{dec}	=	15_{oct}	1	1	0	1
E_{hex}	=	14_{dec}	=	16_{oct}	1	1	1	0
F_{hex}	=	15_{dec}	=	17_{oct}	1	1	1	1

intr-un slide anterior am spus despre “99” ca e cifra in baza 100, pentru ca nu am litere pana la 99

pentru “99” putem continua pe lista ASCII extinsa (pornind de la F care este “15”) pana ajungem la “99”: š

https://en.wikipedia.org/wiki/Binary_number

https://www.ascii-code.com/

SISTEMUL BINAR

- **numere întregi negative**
  - până acum am văzut doar numere naturale
  - ce ați face voi că să reprezentați numere negative?
  - care este prima (cea mai simplă) idee?

SISTEMUL BINAR

- **numere întregi negative**
  - până acum am văzut doar numere naturale
  - ce ați face voi că să reprezentați numere negative?
  - care este prima (cea mai simplă) idee?
    - pai trebuie să salvăm semnul numărului
    - cât spațiu ocupă asta? 1 bit
    - deci 1 bit pentru semn, restul pentru valoarea absolută
      - 1 101 ar fi -5
      - 0 101 ar fi 5
    - cum arată “zero” reprezentat asa?

SISTEMUL BINAR

- **numere întregi negative**
  - până acum am văzut doar numere naturale
  - ce ați face voi că să reprezentați numere negative?
  - care este prima (cea mai simplă) idee?
    - pai trebuie să salvăm semnul numărului
    - cât spațiu ocupă asta? 1 bit
    - deci 1 bit pentru semn, restul pentru valoarea absolută
      - 1 101 ar fi -5
      - 0 101 ar fi 5
    - cum arată “zero” reprezentat așa?

SISTEMUL BINAR

- **numere întregi negative**
  - până acum am văzut doar numere naturale
  - ce ați face voi că să reprezentați numere negative?
  - care este prima (cea mai simplă) idee?
    - pai trebuie să salvăm semnul numărului
    - cât spațiu ocupă asta? 1 bit
    - deci 1 bit pentru semn, restul pentru valoarea absolută
      - 1 101 ar fi -5
      - 0 101 ar fi 5
    - cum arată “zero” reprezentat așa?
      - 1 000 ar fi -0
      - 0 000 ar fi 0
      - este redundant
    - și mai este o problemă: avem nevoie de circuite speciale pentru a face operații cu aceste numere (trebuie verificat primul bit și in funcție de asta trebuie decise operațiile)

SISTEMUL BINAR

- **numere întregi negative**

bit b_i:	1	1	1	1	0	0	0	1
2^i:	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- **ce se întâmplă? MSB este negativ**

- $x = -b_{N-1}2^{N-1} + \sum_{i=0}^{N-2} b_i 2^i$

- **în exemplul de mai sus:**
  - care este numărul reprezentat?
  - pe cati biți este reprezentat acest număr?

SISTEMUL BINAR

- **numere întregi negative**

bit b_i:	1	1	1	1	0	0	0	1
2^i:	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- **ce se întâmplă? MSB este negativ**

- $$x = -b_{N-1}2^{N-1} + \sum_{i=0}^{N-2} b_i 2^i$$

- **în exemplul de mai sus:**
  - $-1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = -15$
  - 8 biți

- **acesta este sistemul de reprezentare în complement față de doi**

SISTEMUL BINAR

- **numere întregi negative**

bit b_i:	1	1	1	1	0	0	0	1
2^i:	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- $$x = -b_{N-1}2^{N-1} + \sum_{i=0}^{N-2} b_i 2^i$$

- **reprezentarea se numește *în complement față de doi***
  - numerele în intervalul -2^{N-1} pana la $2^{N-1} - 1$
  - se "pierde" un bit pentru semn, e optim
  - MSB este semnul, restul biților sunt valoarea
  - ca să putem folosi numere înscrise în operații aritmetice avem nevoie să facem niște transformări

vedeți tabelul din dreapta, numerele pozitive sunt la fel ca înainte (dar pe 3 biți doar), cele negative arată la fel doar că primul bit este "1" (jumătatea de sus a tabelului are MSB "0", jumătatea de jos este identică, dar cu MSB "1")

ce se întâmplă dacă adunăm 1 la 7?

complement față de doi	zecimal
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	−1
1110	−2
1101	−3
1100	−4
1011	−5
1010	−6
1001	−7
1000	−8

https://en.wikipedia.org/wiki/Two%27s_complement

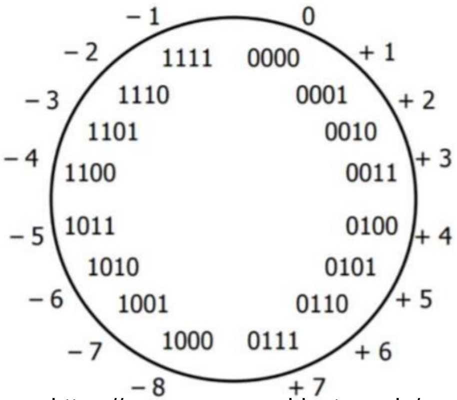
SISTEMUL BINAR

- **numere întregi negative**

bit b_i:	1	1	1	1	0	0	0	1
2^i:	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- $x = -b_{N-1}2^{N-1} + \sum_{i=0}^{N-2} b_i 2^i$

- **reprezentarea se numește *în complement față de doi***
  - numerele în intervalul -2^{N-1} pana la $2^{N-1} - 1$
  - se "pierde" un bit pentru semn, e optim
  - MSB este semnul, restul biților sunt valoarea
  - ca să putem folosi numere înscrise în operații aritmetice avem nevoie să facem niște transformări



complement față de doi	zecimal
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	−1
1110	−2
1101	−3
1100	−4
1011	−5
1010	−6
1001	−7
1000	−8

https://en.wikipedia.org/wiki/Two%27s_complement

https://courses.cs.washington.edu/courses/cse351/19au/sections/03/cse351_sec03_au19.pdf

SISTEMUL BINAR

- **numere întregi negative**

bit b_i:	1	1	1	1	0	0	0	1
2^i:	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- $x = -b_{N-1}2^{N-1} + \sum_{i=0}^{N-2} b_i 2^i$

- **cum reprezentăm un număr negativ zecimal x? (ex: x = -30)**
  - scriem |x| în binar
  - setăm MSB și inversăm restul biților
  - adunăm unu

	0	0	1	1	1	1	0
1	1	1	0	0	0	0	1
1	1	1	0	0	0	1	0

  - deci $(-30)_{10} = (11100010)_2$

SISTEMUL BINAR

- **numere întregi negative**

bit b_i:	1	1	1	1	0	0	0	1
2^i:	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- $x = -b_{N-1}2^{N-1} + \sum_{i=0}^{N-2} b_i 2^i$

- **cum reprezentăm un număr binar x? (ex: x = 1011 1010)**
  - scriem x în binar
  - MSB = 1, deci x este negativ (dacă MSB = 0 atunci x e pozitiv)
  - inversăm biții
  - adăugăm unu

scriem x în binar	1	0	1	1	1	0	1	0
inversăm biții	0	1	0	0	0	1	0	1
adăugăm unu	0	1	0	0	0	1	1	0

  - deci $(10111010)_2 = (-70)_{10}$ (negativ, 64 + 4 + 2 = 70)

SISTEMUL BINAR

- **operații aritmetice, numere naturale exemplu**

	1	0	1	1	0	0	1	0	178
	0	0	1	1	0	1	1	0	54
+	1	1	1	0	1	0	0	0	232

SISTEMUL BINAR

- de ce folosim acest sistem în complement față de 2?
- pentru că algoritmul de adunare este la fel ca pentru numere naturale, nu trebuie să schimbăm nimic
- circuitele anterioare care realizează adunarea naturală pot fi folosite și acum

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - numărul negativ este mai mic (magnitudine) decât cel pozitiv

	1	1	0	1	1	0	0	1	-39
	0	1	0	1	1	1	0	0	92
+	0	0	1	1	0	1	0	1	53

- **cum am obținut reprezentarea pentru -39?**

	0	1	0	0	1	1	1	39
1	1	0	1	1	0	0	0	inversarea de biți
1	1	0	1	1	0	0	1	plus unu

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - numărul negativ este mai mare (magnitudine) decât cel pozitiv

	1	1	0	1	1	0	0	1	-39
	0	0	0	1	0	1	0	1	21
+	1	1	1	0	1	1	1	0	-18

- **de unde am obținut -18?**

1	1	1	0	1	1	1	0	rezultatul, primul bit e 1
0	0	0	1	0	0	0	1	inversarea de biți
0	0	0	1	0	0	1	0	plus unu

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - ambele numere sunt negative

	1	1	0	1	1	0	0	1	-39
	1	1	1	0	1	1	1	0	-18
+	1	1	0	0	0	1	1	1	-57

- **de unde am obținut -57?**

1	1	0	0	0	1	1	1	rezultatul, primul bit e 1
0	0	1	1	1	0	0	0	inversarea de biti
0	0	1	1	1	0	0	1	plus unu

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - overflow la limită

	0	1	1	1	1	1	1	1	127
	0	0	0	0	0	0	0	1	1
+	1	0	0	0	0	0	0	0	-128

- **de unde am obținut -128?**

1	0	0	0	0	0	0	0	rezultatul, primul bit e 1
0	1	1	1	1	1	1	1	inversarea de biți
1	0	0	0	0	0	0	0	plus unu: 128

-128 are aceeși reprezentare

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - ambele numere pozitive, mari

	0	1	1	1	0	1	1	1	119
	0	1	0	0	1	0	1	1	75
+	1	1	0	0	0	0	1	0	-62

- **de unde am obținut -62?**

1	1	0	0	0	0	1	0	rezultatul, primul bit e 1
0	0	1	1	1	1	0	1	inversarea de biți
0	1	1	1	1	1	1	0	plus unu

- **119 + 75 = 194 (nu încap pe 7 biți)**

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - ambele numere pozitive, mari

	0	1	1	1	0	1	1	1	119
	0	1	0	0	1	0	1	1	75
+	1	1	0	0	0	0	1	0	-62

- **intuiția, de unde am obținut -62?**
  - 119 + 75 = 194 (nu încap pe 7 biți)
  - maximum e 127, deci avem 194 - 127 = 67 “extra”
  - overflow începe după 127, după 127 este -128 (folosim un extra)
  - deci 66 extra rămași pornesc de la -128
  - deci avem -128 + 66 = -62

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - overflow la limită

	0	1	1	1	1	1	1	1	127
	0	0	0	0	0	0	0	1	1
+	1	0	0	0	0	0	0	0	-128

- **intuiția, de unde am obținut -128?**
  - 127 + 1 = 128 (nu încap pe 7 biți)
  - maximum e 127, deci avem 128 - 127 = 1 “extra”
  - overflow începe dupa 127, după 127 este -128 (folosim un extra)
  - deci 0 extra ramași pornesc de la -128
  - deci avem -128 + 0 = -128

SISTEMUL BINAR

- **operații aritmetice, numere întregi exemple**
  - ambele numere negative, mari

	1	0	0	0	1	0	0	1	-119
	1	0	1	1	0	1	0	1	-75
+	0	0	1	1	1	1	1	0	62

- **intuiția, de unde am obținut 62?**
  - -119 - 75 = -194 (nu încap pe 7 biți)
  - minimul e -128, deci avem +194 - 128 = 66 “extra”
  - underflow începe înainte de -128, înainte de -128 este 127 (folosim un extra)
  - deci 65 extra rămași pornesc de la 127
  - deci avem 127 - 65 = 62

SISTEMUL BINAR

- **extinderea numărului de biți pentru reprezentare**
  - să presupunem că avem numărul 01 11 11 reprezentat pe 6 biți
  - ni se spune că numărul este natural
  - ni se spune că putem folosi încă 2 biți pentru reprezentare
    - noua reprezentare este:

SISTEMUL BINAR

- **extinderea numărului de biți pentru reprezentare**
  - să presupunem că avem numărul 01 11 11 reprezentat pe 6 biți
  - ni se spune că numărul este natural
  - ni se spune că putem folosi încă 2 biți pentru reprezentare
    - noua reprezentare este: 0001 1111
  - ce se întâmplă acum dacă avem numărul 10 00 11 reprezentat pe 6 biți (dar știm că suntem în complement față de doi)
  - ni se spune din nou că putem folosi încă 2 biți pentru reprezentare
    - noua reprezentare este: 0010 0011 ?

SISTEMUL BINAR

- **extinderea numărului de biți pentru reprezentare**
  - să presupunem că avem numărul 01 11 11 reprezentat pe 6 biți
  - ni se spune că numărul este natural
  - ni se spune că putem folosi încă 2 biți pentru reprezentare
    - noua reprezentare este: 0001 1111
  - ce se întâmplă acum dacă avem numărul 10 00 11 reprezentat pe 6 biți (dar știm că suntem în complement față de doi)
  - ni se spune din nou că putem folosi încă 2 biți pentru reprezentare
    - noua reprezentare este: 0010 0011 ?
    - numărul acesta nici măcar nu este negativ (MSB este 0)
    - deci nu putem extinde cu “zero”
    - cu ce extindem?

.

SISTEMUL BINAR

- **extinderea numărului de biți pentru reprezentare**
  - să presupunem că avem numărul 01 11 11 reprezentat pe 6 biți
  - ni se spune că numărul este natural
  - ni se spune că putem folosi încă 2 biți pentru reprezentare
    - noua reprezentare este: 0001 1111

  - ce se întâmplă acum dacă avem numărul 10 00 11 reprezentat pe 6 biți (dar știm că suntem în complement față de doi)

  - ni se spune din nou că putem folosi încă 2 biți pentru reprezentare
    - noua reprezentare este: 0010 0011 ?
    - numărul acesta nici măcar nu este negativ (MSB este 0)
    - deci nu putem extinde cu “zero”
    - cu ce extindem? cu “unu”
    - noua reprezentare este: 1110 0011

SISTEMUL BINAR

- **înca un exemplu:**
- **extinderea numărului de biți pentru reprezentare**
  - să presupunem că avem numărul 10111 reprezentat pe 5 biți
  - trecem în baza *B* = *8*
    - numărul în baza *B* = *8* este 10 111 = 27
    - dar, dacă vedem $(27)_8$ atunci am putea crede ca în binar avem 010111
    - dar 010111 este pe 6 biți și este pozitiv

    - **ideea:** când trecem din binar în baza *B* = *8* (și știm că aici implicit avem 6 biți de reprezentare) atunci extindem reprezentarea
    - deci, pornim cu 110 111
    - iar în baza *B* = *8* atunci avem 110 111 = 67

    - problema este generată de faptul ca 3 nu îl împarte exact pe 5
    - ambele variante sunt corecte: $(27)_8$ dar știi că sunt 5 biți sau $(67)_8$ și crezi că sunt 6 biți (implicit când vezi două cifre în baza *B* = *8* crezi că sunt 6 biți)

SISTEMUL BINAR

- **logica binară (0 = False, 1 = True)**
- **operații logice:**
  - NOT (negația)
  - AND (conjuncția)
  - OR (disjuncția)
  - XOR (disjuncția exclusivă)

X	NOT X
0	1
1	0

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

- **pentru numere reprezentate binar operația logică se face pentru fiecare bit în parte (pentru numere pe *N* biți, sunt *N* operații)**

SISTEMUL BINAR

- logica binară, exemplu

	1	0	1	1	0	0	1	0	178
	0	0	1	1	0	1	1	0	54
AND:	0	0	1	1	0	0	1	0	50
OR:	1	0	1	1	0	1	1	0	182
XOR:	1	0	0	0	0	1	0	0	132

- aparent, valorile zecimale nu au interpretare clară
- totuși, putem spune ceva: OR încurajează apariția de biți “1”, AND o descurajează, iar la XOR … depinde
- totuși, vom vedea că logica binară (în combinații interesante) ne poate spune multe și despre numere zecimale

CE AM FĂCUT ASTĂZI

- **am discutat despre istoria sistemelor de calcul**

- **am acoperit elementele de bază ale sistemului binar**
  - reprezentarea binară (numere naturale)
  - calcule cu baze *B* diferite
  - reprezentarea complement față de doi (numere întregi)
  - operații binare elementare

- **la seminar, vom vedea cum facem operații în sistemul binar și cum folosim complementul față de doi**

DATA VIITOARE ...

- introducere în teoria informației
- cum măsurăm cantitatea de informație
- entropia lui Shannon
- continuăm să studiem sistemul binar
- … mai târziu: înmulțirea și împărțirea
- … și mai târziu: floating point

LECTURĂ SUPLIMENTARĂ

- **PH book**
  - 2.4 Signed and Unsigned Numbers
  - 2.17 Real Stuff: x86 Instructions
  - 3.2 Addition and Subtraction

- **Thomas Finley course notes, https://www.cs.cornell.edu/~tomf/notes/cps104/twoscomp.html**

- **mașina Turing și conceptul de Turing-complete:**
  - Turing Machines Explained – Computerphile, https://youtube.com/watch?v=dNRDvLACg5Q
  - Turing Complete – Computerphile, https://www.youtube.com/watch?v=RPQD7-AOjMI

LECTURĂ SUPLIMENTARĂ (NU INTRA IN EXAMEN)

- **dacă sunteți interesați de istoria sistemelor de calcul vă sugerez următoarele prezentări:**
  - Computer Pioneers: Pioneer Computers Part 1 (1935 – 1945), https://www.youtube.com/watch?v=qundvme1Tik
  - Computer Pioneers: Pioneer Computers Part 2 (1946 – 1950), https://www.youtube.com/watch?v=wsirYCAocZk
  - BBC History of Computers, https://www.youtube.com/playlist?list=PL1331A4548513EA81
  - https://www.gresham.ac.uk/lectures-and-events/a-very-brief-history-of-computing-1948-2015
  - The Grand Narrative of the History of Computing, https://www.youtube.com/watch?v=njwQgz63rIs

.