ARHITECTURA SISTEMELOR DE CALCUL - CURS 0x03

ABSTRACTIZAREA DIGITALĂ ȘI CIRCUITE COMBINAȚIONALE

Cristian Rusu

DATA TRECUTĂ

- am discutat despre conceptul de informație
- am văzut cum măsurăm informația
- am calculat entropia lui Shannon
- am văzut algoritmul Huffman pentru codarea variabilă a datelor
- am folosit distanța Hamming între două șiruri de biți
- azi, vom vedea cum implementăm acești biți în circuite

CUPRINS

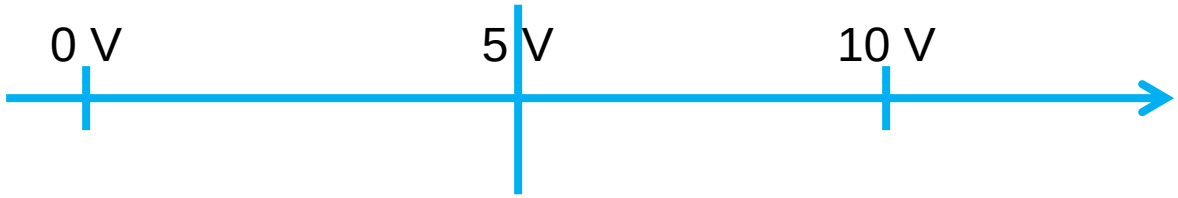
ABSTRACTIZAREA DIGITALĂ

- **avem nevoie de o reprezentare fizică a biților**
- **avem nevoie de o metodă de stocare**
- **ce caracteristici am dori?**
  - să se poată stoca mulți biți
  - să fie ieftin să stocăm per bit, pentru că avem mulți
  - să fie o reprezentare stabilă (să nu dispară sau să se degradeze în timp)
  - să îi putem manipula ușor

- **soluția: folosim proprietăți electrice**
  - de obicei voltaj
  - dar voltajul are valori continue (majoritatea efectelor în natură sunt continue) iar noi vrem binar
  - avem nevoie de o metodă de cuantizare

ABSTRACTIZAREA DIGITALĂ

- **de la continuu la digital**

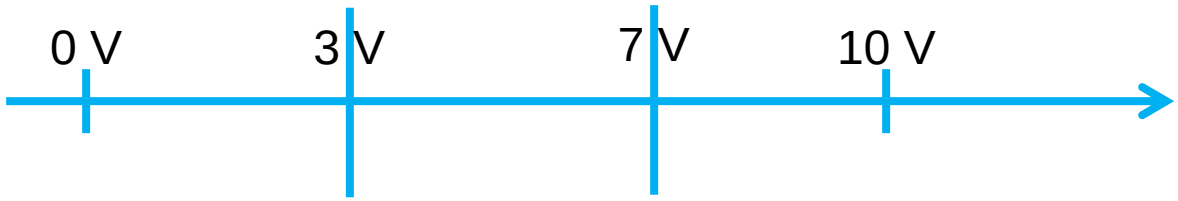


  - cea mai simplă ideea: avem un voltaj maxim care poate să fie atins: deci de la 0V la 5V codăm “0” iar de la 5V la 10V avem “1”

  - ce dificultăți avem în acestă situație?
    - e dificil să înțelegem ce se întamplă în jurul lui 5V

ABSTRACTIZAREA DIGITALĂ

- **de la continuu la digital**

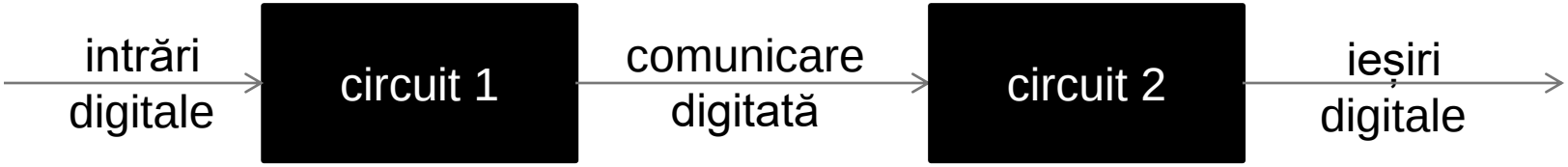


  - o soluție puțin mai sofisticată: avem două limite
  - de data asta: “0” este între 0V și 3V iar “1” este între 7V și 10V
  - intervalul între 3V și 7V este un “no man’s land”
    - nu putem decide voltajul
    - așteptăm stabilizarea la o valoare < 3V sau > 7V

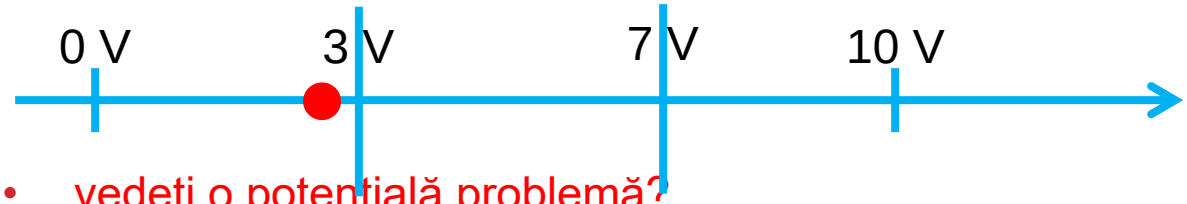
Atenție: sistemul nu este perfect, la 3V suntem “0” dar un zgomot de doar 4V din acest punct ne poate duce la 7V, deci “1”

ABSTRACTIZAREA DIGITALĂ

- **de la continuu la digital**
  - în cele mai multe cazuri, vom conecta dispozitive digitale între ele



  - de exemplu:
    - primul circuit scoate un “0”, dar la limita superioară

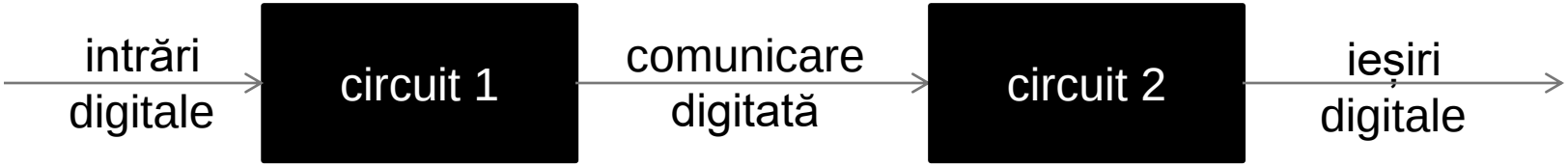


    - vedeți o potențială problemă?

Atenție: sistemul nu este perfect, la 3V suntem “0” dar un zgomot de doar 4V din acest punct ne poate duce la 7V, deci “1”

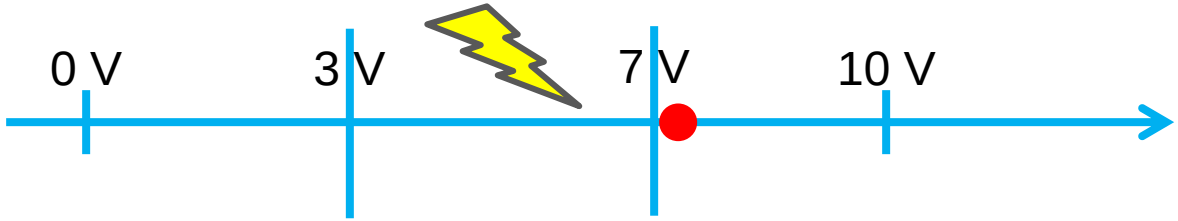
ABSTRACTIZAREA DIGITALĂ

- **de la continuu la digital**
  - în cele mai multe cazuri, vom conecta dispozitive digitale între ele



  - de exemplu:
    - primul circuit scoate un “0”, dar la limita superioară

    - semnalul este trimis către al doilea circuit, dar apare zgomot pe fir

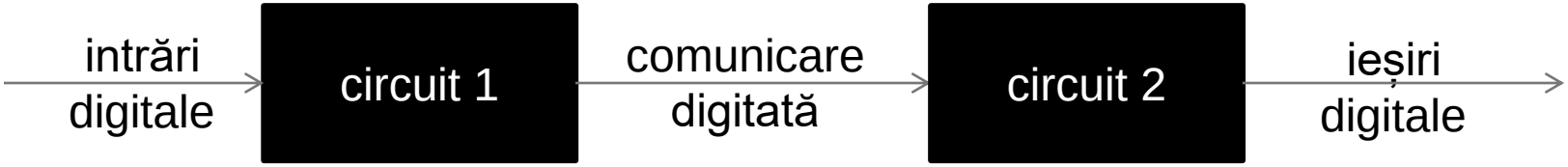


Atenție: sistemul nu este perfect, la 3V suntem “0” dar un zgomot de doar 4V din acest punct ne poate duce la 7V, deci “1”

Soluția?

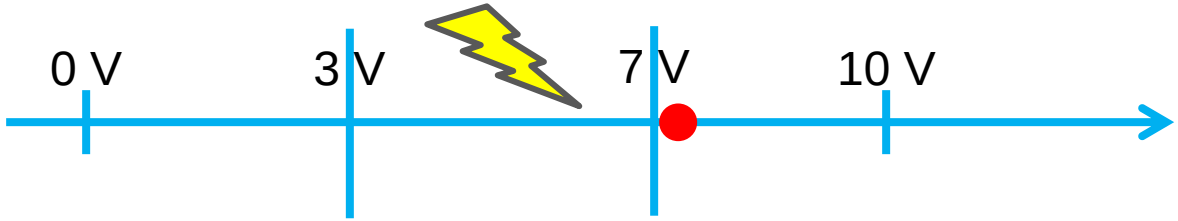
ABSTRACTIZAREA DIGITALĂ

- **de la continuu la digital**
  - în cele mai multe cazuri, vom conecta dispozitive digitale între ele



  - de exemplu:
    - primul circuit scoate un “0”, dar la limita superioară

    - semnalul este trimis către al doilea circuit, dar apare zgomot pe fir



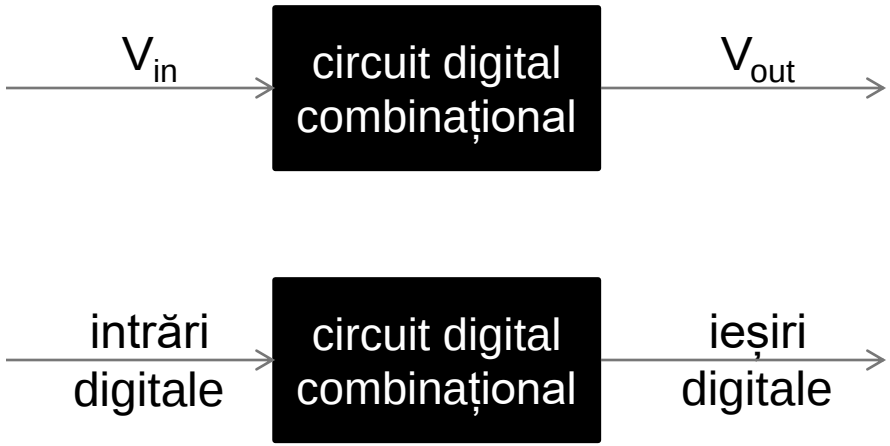
Atenție: sistemul nu este perfect, la 3V suntem “0” dar un zgomot de doar 4V din acest punct ne poate duce la 7V, deci “1”

Soluția? pentru ieșiri vom impune limite mai stricte (sub 2V, peste 8V)

ideal, am vrea ca limitele să fie cât mai înguste: 0V este “0” iar 10V este “1”

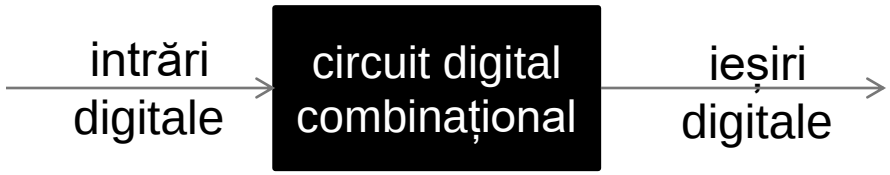
CIRCUITE DIGITALE

- **circuit digital combinațional**



CIRCUITE DIGITALE

- **circuit digital combinațional**



  - intrări digitale, pot fi multe
  - ieșiri digitale, pot fi multe
  - nu are stări interne
    - pui un semnal digital constant la intrare și ai un alt semnal digital constant la ieșire
    - dar nu poate “memora” nimic
    - nu are o “stare internă” (memorie)
  - avem un timp de propagare (t_p): timpul maxim necesar pentru a produce la ieșire semnale digitale corecte și valide din momentul în care la intrare s-au specificat semnale digitale corecte și valide

  - de ce se numesc circuite combinaționale?
    - pentru că ieșire este o combinație (o funcție logică care combină) toate (sau o parte) a intrărilor

CIRCUITE DIGITALE

- **circuit digital combinațional**

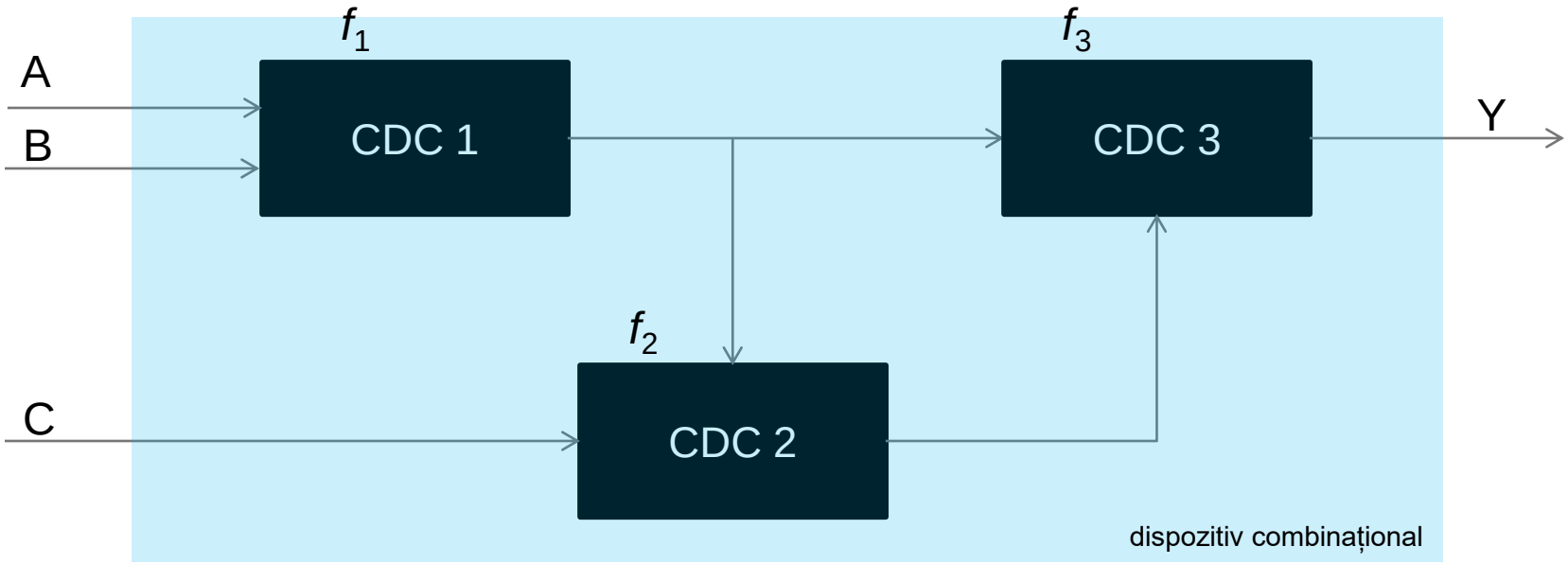
- de ce se numesc circuite combinaționale?
  - pentru că ieșire este o combinație (o funcție logică care combină) toate (sau o parte) a intrărilor
  - deci, pentru fiecare intrare, trebuie sa știm care e ieșirea

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

tabel de adevăr

CIRCUITE DIGITALE

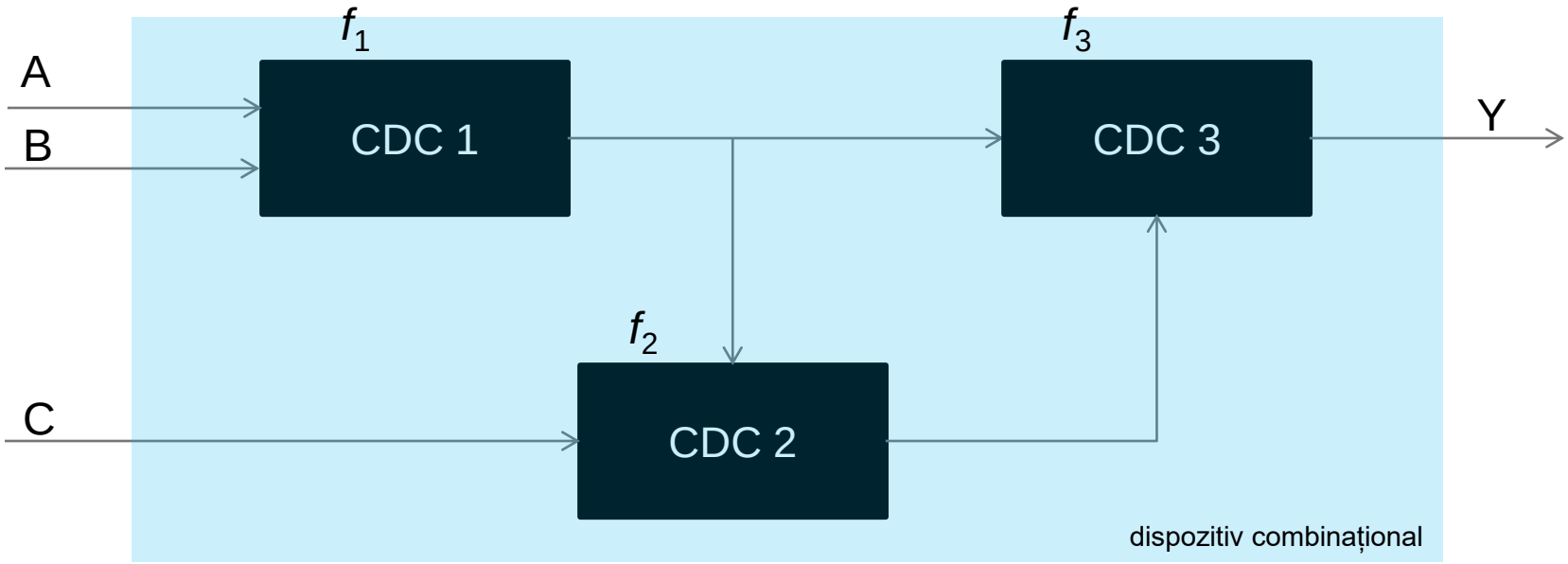
- **dispozitiv combinațional**
  - fiecare element este un circuit combinațional
  - fiecare intrare este conectată la exact o ieșire sau la o constantă
  - nu există niciun ciclu în graful direcțional al dispozitivului



- care este funcția dispozitivului?

CIRCUITE DIGITALE

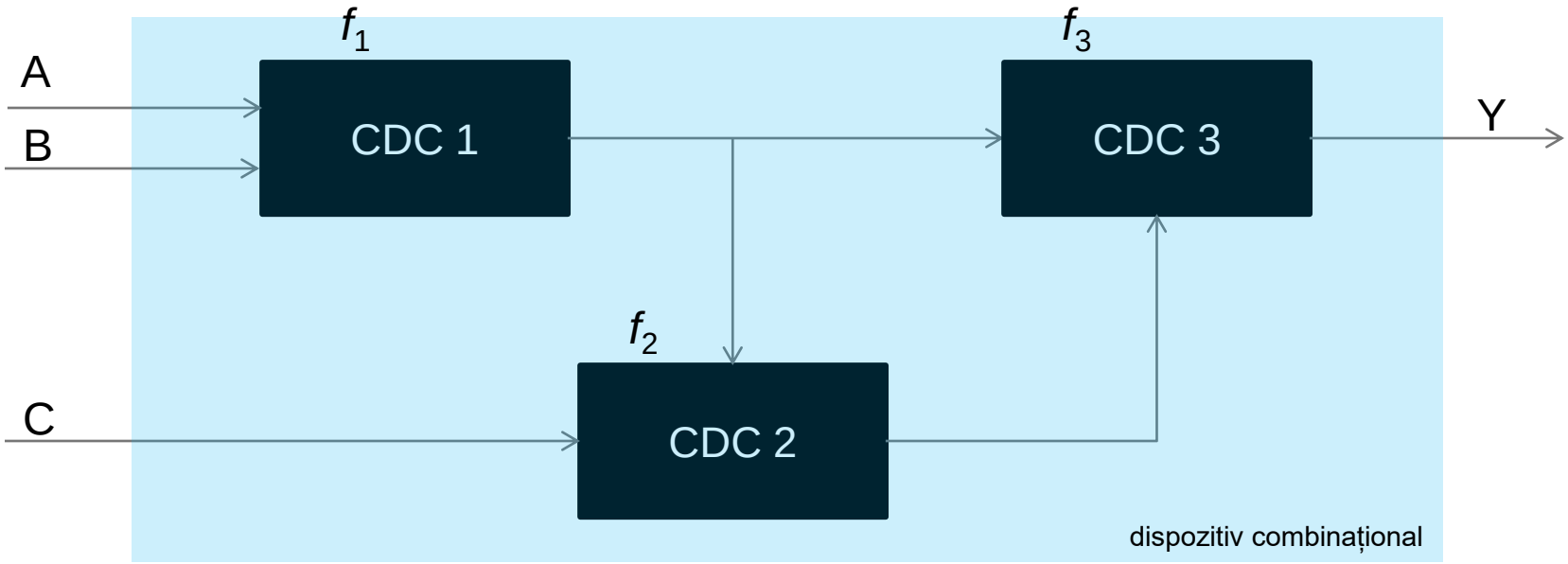
- **dispozitiv combinațional**
  - fiecare element este un circuit combinațional
  - fiecare intrare este conectată la exact o ieșire sau la o constantă
  - nu există niciun ciclu în graful direcțional al dispozitivului



- care este funcția dispozitivului? Y = $f_3(f_1(A, B), f_2(f_1(A, B), C))$

CIRCUITE DIGITALE

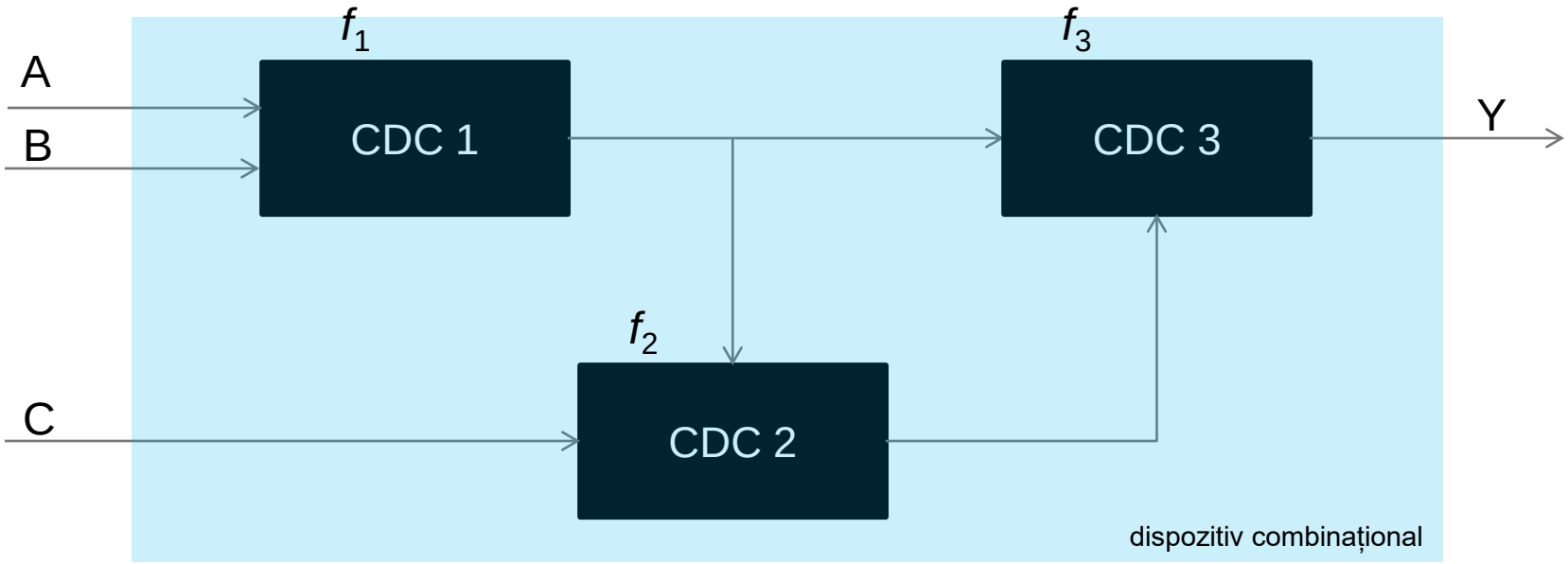
- **dispozitiv combinațional**
  - fiecare element este un circuit combinațional
  - fiecare intrare este conectată la exact o ieșire sau la o constantă
  - nu există niciun ciclu în graful direcțional al dispozitivului



  - timpul total de propagare?

CIRCUITE DIGITALE

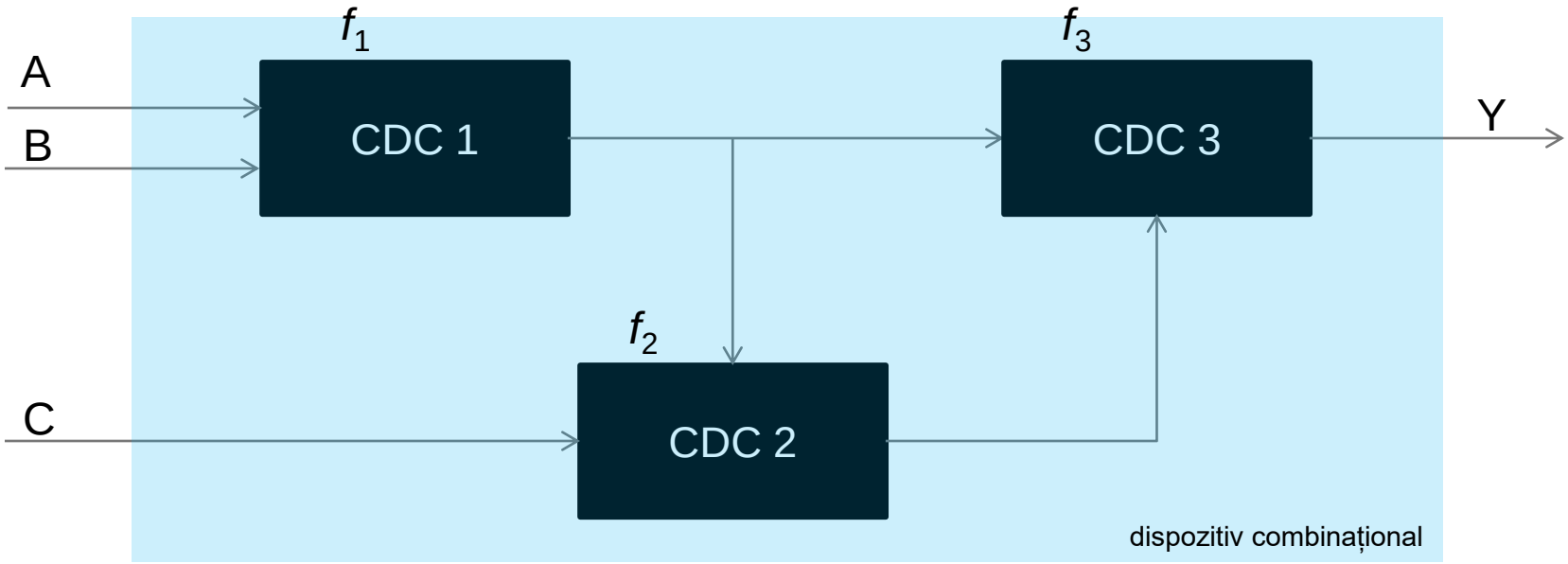
- **dispozitiv combinațional**
  - fiecare element este un circuit combinațional
  - fiecare intrare este conectată la exact o ieșire sau la o constantă
  - nu există niciun ciclu în graful direcțional al dispozitivului



- timpul total de propagare? $t_{p,total} = t_{p,1} + t_{p,2} + t_{p,3}$ (longest path)

CIRCUITE DIGITALE

- **dispozitiv combinațional**
  - fiecare element este un circuit combinațional
  - fiecare intrare este conectată la exact o ieșire sau la o constantă
  - nu există niciun ciclu în graful direcțional al dispozitivului

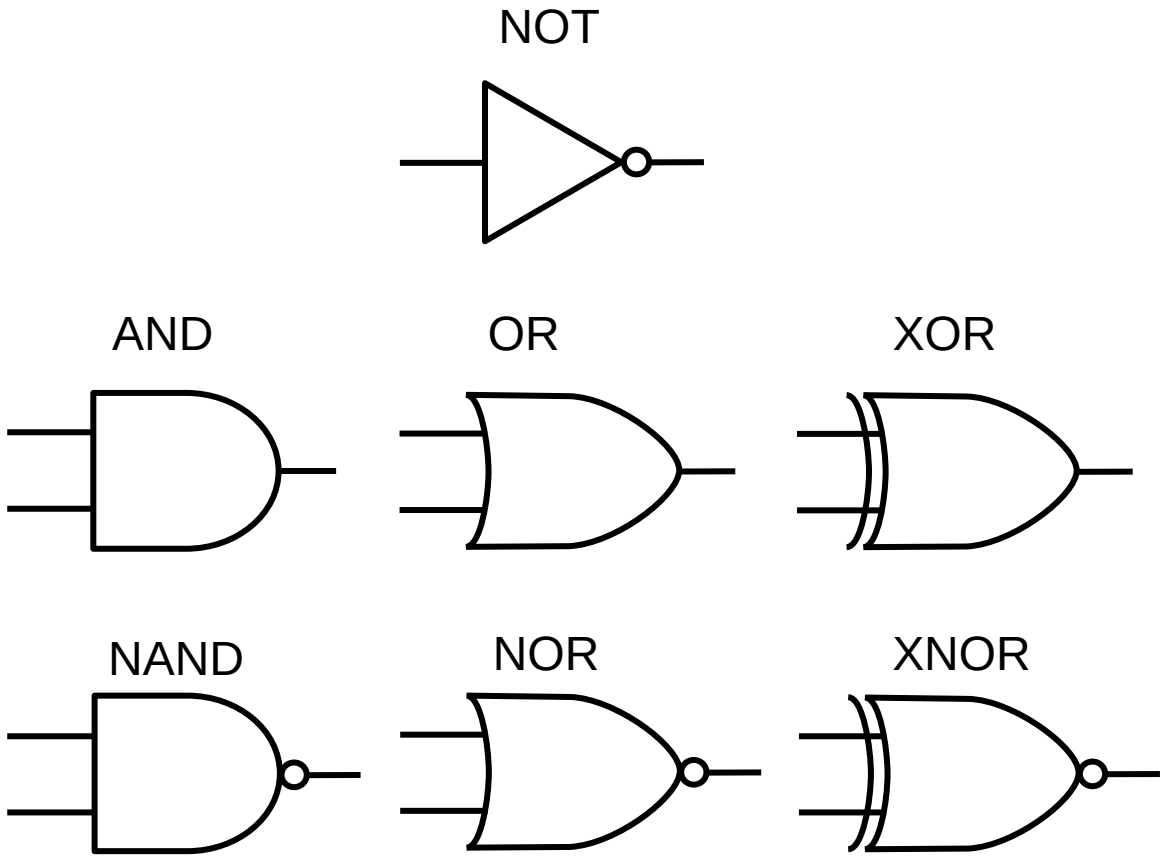


- timpul total de propagare? $t_{p,total} = t_{p,1} + t_{p,2} + t_{p,3}$ (longest path)

  timpul maxim după care avem o ieșire validă dacă avem intrări valide

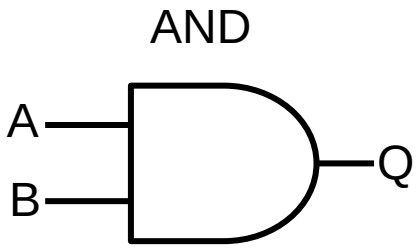
CIRCUITE DIGITALE

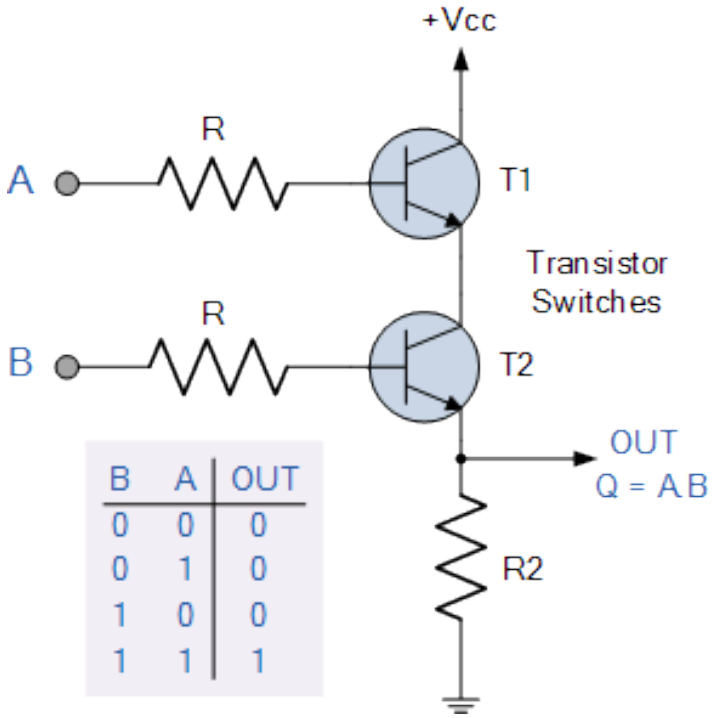
- **circuit digital combinațional**
  - exemple fundamentale



CIRCUITE DIGITALE

- **circuit digital combinațional**
  - exemple fundamentale
  - la baza tuturor se află circuite eletronice bazate de tranzistor

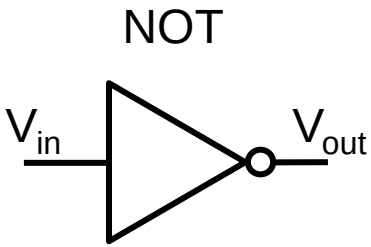


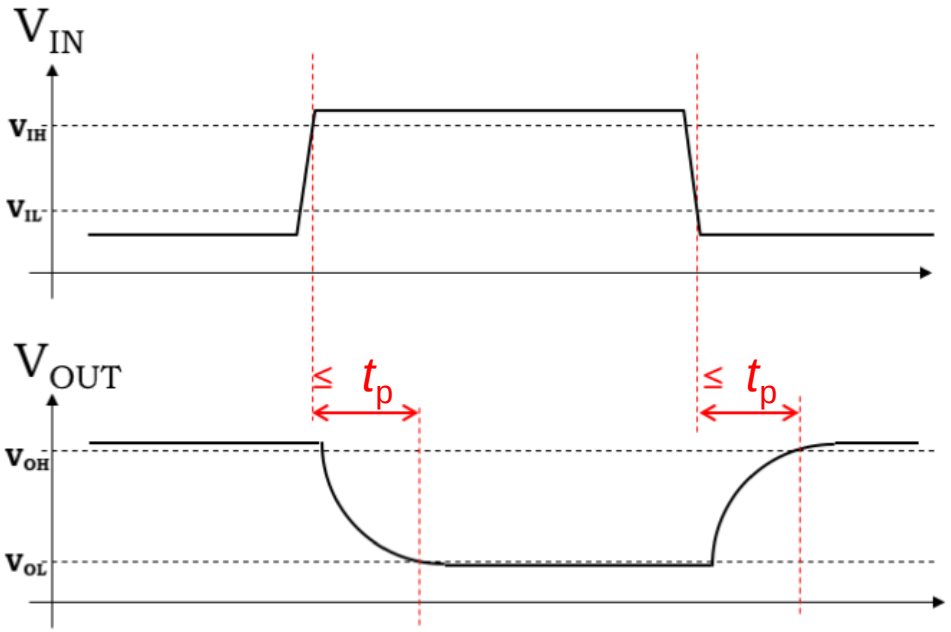


https://www.electronics-tutorials.ws/logic/logic_2.html

CIRCUITE DIGITALE

- **circuit digital combinațional**
  - exemple fundamentale: să nu uităm că totul este analogic

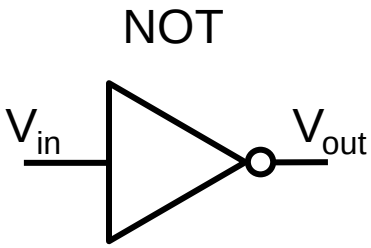


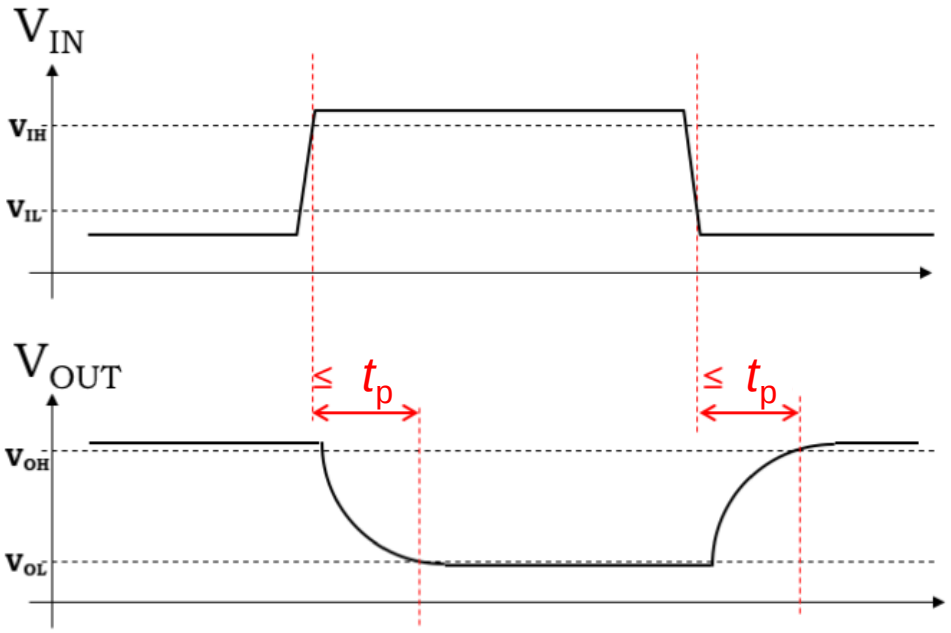


https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-004-computation-structures-spring-2017/c3/c3s1/

CIRCUITE DIGITALE

- **circuit digital combinațional**
  - exemple fundamentale: să nu uităm că totul este analogic



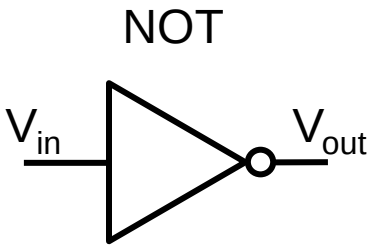


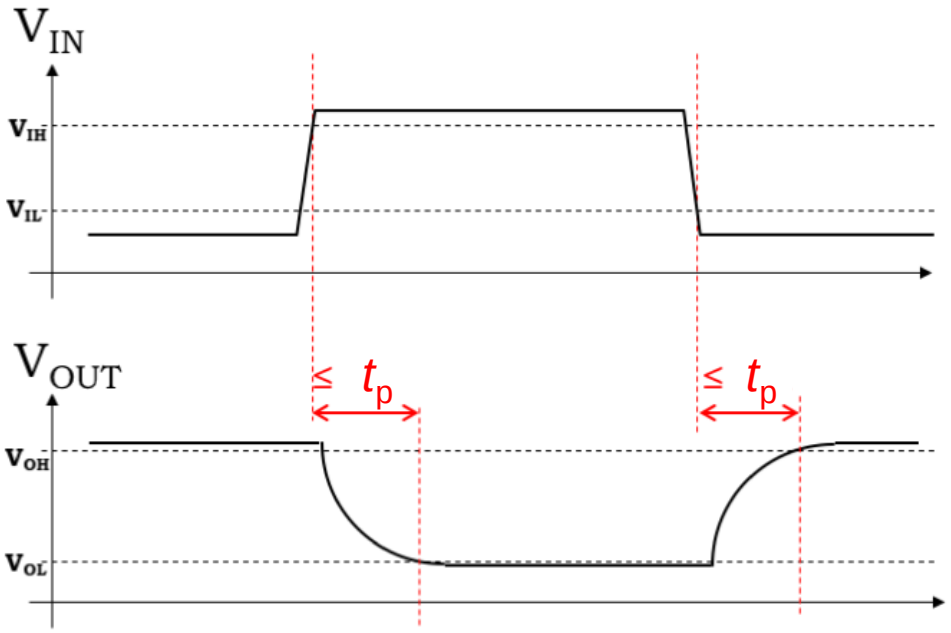
de ce este important acest t_p pentru arhitectura calculatoarelor?

https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-004-computation-structures-spring-2017/c3/c3s1/

CIRCUITE DIGITALE

- **circuit digital combinațional**
  - exemple fundamentale: să nu uităm că totul este analogic



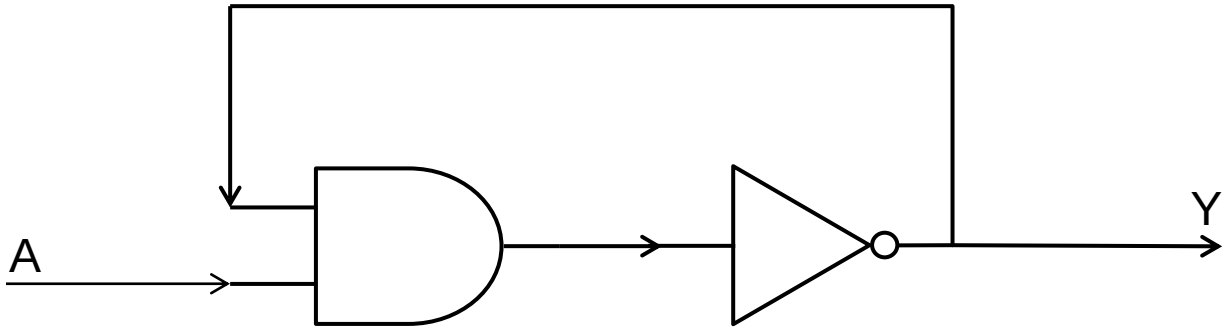


un computer care funcționează la 1GHz trimite comenzi o dată la 1ns

https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-004-computation-structures-spring-2017/c3/c3s1/

CIRCUITE DIGITALE

- **circuit digital combinațional**
  - ce se întâmplă dacă avem un ciclu în circuit?

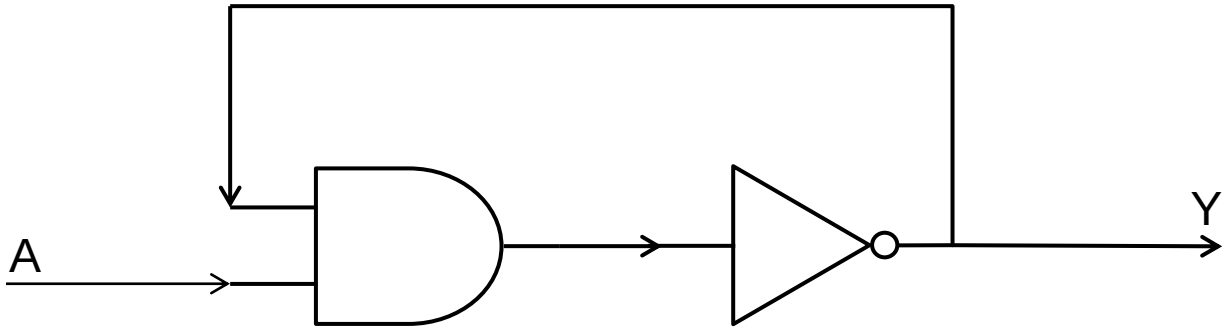


  - ce se întâmplă dacă A = 0?
  - ce se întâmplă dacă A = 1?

(în ambele cazuri considerăm cealalaltă intrare în AND ca 0)

CIRCUITE DIGITALE

- **circuit digital combinațional**
  - ce se întâmplă dacă avem un ciclu în circuit?



  - ce se întâmplă dacă A = 0?
  - ce se întâmplă dacă A = 1?

(în ambele cazuri considerăm cealalaltă intrare în AND ca 0)

  - totuși, vom folosi cicluri pentru a crea stări în circuite digitale

ABSTRACTIZAREA DIGITALĂ

- **două întrebări importante:**
  - de ce folosim semnale digitale în loc de analogice?
  - de ce folosim sistemul binar? ar fi mai avantajos să folosim hex?

ABSTRACTIZAREA DIGITALĂ

- **două întrebări importante:**
  - de ce folosim semnale digitale în loc de analogice?
    - din cauza zgomotului
    - într-un sistem analogic zgomotul se acumulează
    - într-un sistem digital, avem corecțiile de zgomot (avem margini)
  - de ce folosim sistemul binar? ar fi mai avantajos să folosim hex?
    - da, ar fi mai avantajos să folosim hex (e de 4 ori mai avantajos)
    - problema este că în loc de două stări ar trebui acum să avem 16
    - asta înseamnă că trebuie să distingem 16 nivele de voltaj în prezența zgomotului (adică cu tot cu margini de zgomot)
    - probabil 16 nivele e prea mult ... dar probabil 4 nivele ar fi fezabil
    - dacă am avea 4 nivele (adică baza $B = 4$) am fi de două ori mai eficienți

CIRCUITE DIGITALE

- **hex în media**
  - The Martian Hexadecimal Scene, https://www.youtube.com/watch?v=k-GH3mbvUro
  - care e problema?
  - de ce folosește hex?
  - cum traduce din hex în litere?
  - ce face la sfârșit? (aparent scrie hex direct ca să programeze)

.

CIRCUITE DIGITALE

- **hex în media**
  - The Martian Hexadecimal Scene, https://www.youtube.com/watch?v=k-GH3mbvUro

- care e problema? 27 de caractere sunt prea multe
- de ce folosește hex? pentru că în loc de 27 are doar 16 caractere
- cum traduce din hex în litere? tabela ASCII
- la sfârșit, scrie cod mașină (și noi vom face asta, puțin)

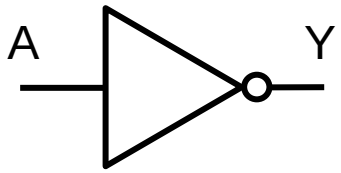
ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

https://simple.wikipedia.org/wiki/ASCII

PORȚILE LOGICE DE BAZĂ

- **NOT**

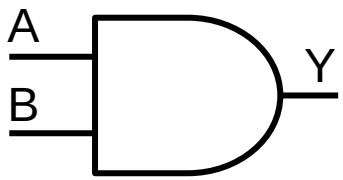


A	Y
0	1
1	0

- **notația:**
  - NOT A (explicit operația)
  - $\overline{A}$ **(A bar, A complement)**
  - ¬A (notația din logică)
  - ~A (not A)
  - - A (minus A)
  - A' (A prime, A complement)
  - !A (bang A)

PORȚILE LOGICE DE BAZĂ

- **AND**

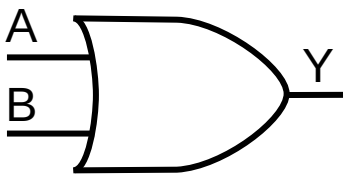


A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

- **notația:**
  - A AND B (explicit operația)
  - A x B (înmulțire)
  - A • B (înmulțire)
  - A * B (înmulțire)
  - A . B (înmulțire)
  - **AB (înmulțire)**
  - A ∧ B (notația din logică)
  - A & B (operația pe biți, C)
  - A && B (operația logică, C)

PORȚILE LOGICE DE BAZĂ

- **OR**

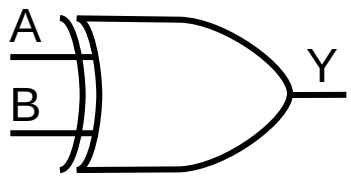


A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

- **notația:**
  - A OR B (explicit operația)
  - **A + B (adunare)**
  - A ∨ B (notația din logică)
  - A | B (operația pe biți, C)
  - A || B (operația logică, C)

PORȚILE LOGICE DE BAZĂ

- **XOR**

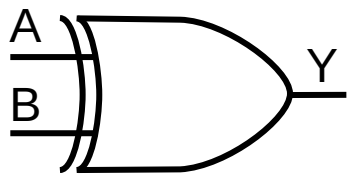


A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

- **notația:**
  - A XOR B (explicit operația)
  - **A ⊕ B (adunare XOR)**
  - A ^ B (operația pe biți, C)
  - A ^^ B (există așa ceva ???)

PORȚILE LOGICE DE BAZĂ

- **XOR**



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

- **notația:**
  - A XOR B (explicit operația)
  - **A ⊕ B (adunare XOR)**
  - A ^ B (operația pe biți, C)
  - A != B (operația logică, C: sau A <> B, sau A ~= B, sau …)

ALGEBRĂ BOOLEANĂ

- **aveți un curs de Logică în acest semestru**
- **deci, din acest moment presupun că logica/algebra booleană este cunoscută de voi**
- **nu vom repeta materia de la logică, dar dacă este ceva foarte important vom trece rapid în revistă conceptele**

CIRCUIT DIGITAL COMBINAȚIONAL

- **circuit digital combinațional**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **o expresie booleană care conține regulile din tabel?**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$
  - Y = ... (execițiu pentru voi)

forma normală, suma de produse

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

- **concluzia: NOT, AND și OR sunt universale (pot implementa orice circuit combinațional)**
- **de ce lipsește XOR?**

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

- **concluzia: NOT, AND și OR sunt universale (pot implementa orice circuit combinațional)**
- **de ce lipsește XOR? $A \oplus B = \overline{A}B + A\overline{B}$**

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC$

- **o potențială problemă: dacă avem N intrări, expresia pentru X depinde de un număr (potențial) mare de sume de produse**
- **care e numărul maxim de termeni în sumele din X?**

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC$

- **o potențială problemă: dacă avem N intrări, expresia pentru X depinde de un număr (potențial) mare de sume de produse**
- **care e numărul maxim de termeni în sumele din X? $½2^N$**

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

- **o potențială problemă: dacă avem N intrări, expresia pentru X depinde de un număr (potențial) mare de sume de produse**
- **soluția generală: vrem reprezentări minime**

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

- **reprezentarea minimă a lui X?**

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

    $=$

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

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

    $=$

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

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

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

CIRCUIT DIGITAL COMBINAȚIONAL

- **tabel de adevăr**

A	B	C	X	Y
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

- **expresia booleană echivalentă**
  - $X = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$

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

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

    $= A \oplus B \oplus C$

CE AM FĂCUT ASTĂZI

- abstractizarea digitală
- discuție binary vs hex
- circuite digitale
- circuite combinaționale
- am văzut un singur exemplu de circuit pe bază de tranzistor

DATA VIITOARE ...

- mai mult despre circuite de bază
- logică booleană
- sinteză logică
- circuite mai complexe
- circuite secvențiale

LECTURĂ SUPLIMENTARĂ

- **PH book**
  - Appendix B

- **Computerphile, Why Use Binary?, https://www.youtube.com/watch?v=thrx3SBEpL8**

- **Real Engineering, Transistors - The Invention That Changed The World, https://www.youtube.com/watch?v=OwS9aTE2Go4**

- **Ben Eater, Making logic gates from transistors, https://www.youtube.com/watch?v=sTu3LwpF6XI**

- **DrPhysicsA, An Introduction to Logic Gates, https://www.youtube.com/watch?v=95kv5BF2Z9E**

.

.