Számítógépes alapismeretek

Heti óraszáma: 2 (Bagoly Zsolt, Papp Gábor) + 1 (Barnaföldi Gergely)

A tantárgy célja:

- **korszerű információtechnológiai alapismeretek elsajátítása**
- **megismerkedés az informatikai alapfogalmakkal**
- **megismerkedés a személyi számítógép vázlatos felépítésével**
- **áttekintés a számítógépes hálózatok alapjairól**
- **alapszintű biztonsági, adatvédelmi és jogi kérdések**

A tananyag sikeres elvégzése segíti az ECDL követelmények teljesítését.

írásbeli vizsga (legalább elégséges gyakorlati jegy követelmény!!!)

Irodalom:
ECDL 1-7 tankönyvek (Neumann János Számítógép-tudományi Társaság)
Ajánlott:
Papp Gábor és Bagoly Zsolt: Ablakok a hálózaton: az X11 rendszer
Bagoly Zsolt és Papp Gábor: UNIX alapismeretek

itl7.elte.hu

SZÁMLÁLÁS A MATEMATIKA ALAPJA

Nézzük meg mi történik törzsvendégek esetén egy kocsmában.

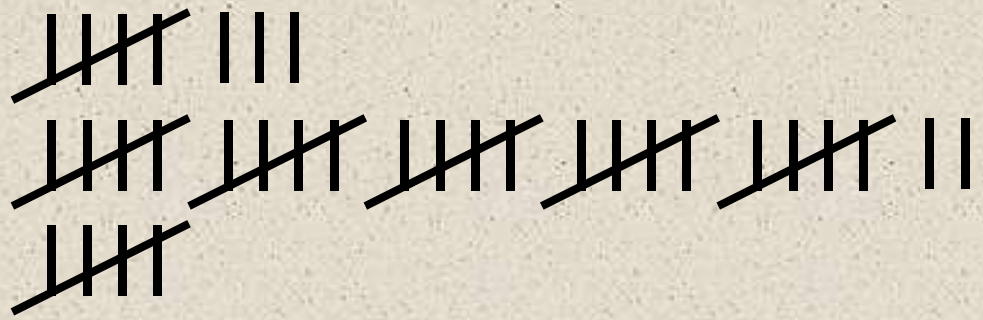
A pintek száma egy középkori kocsmában:

Arató András
Bornemissza Péter
Cuczor Áron

Az események (itt fogyasztott pintek) számának leképzése rovással

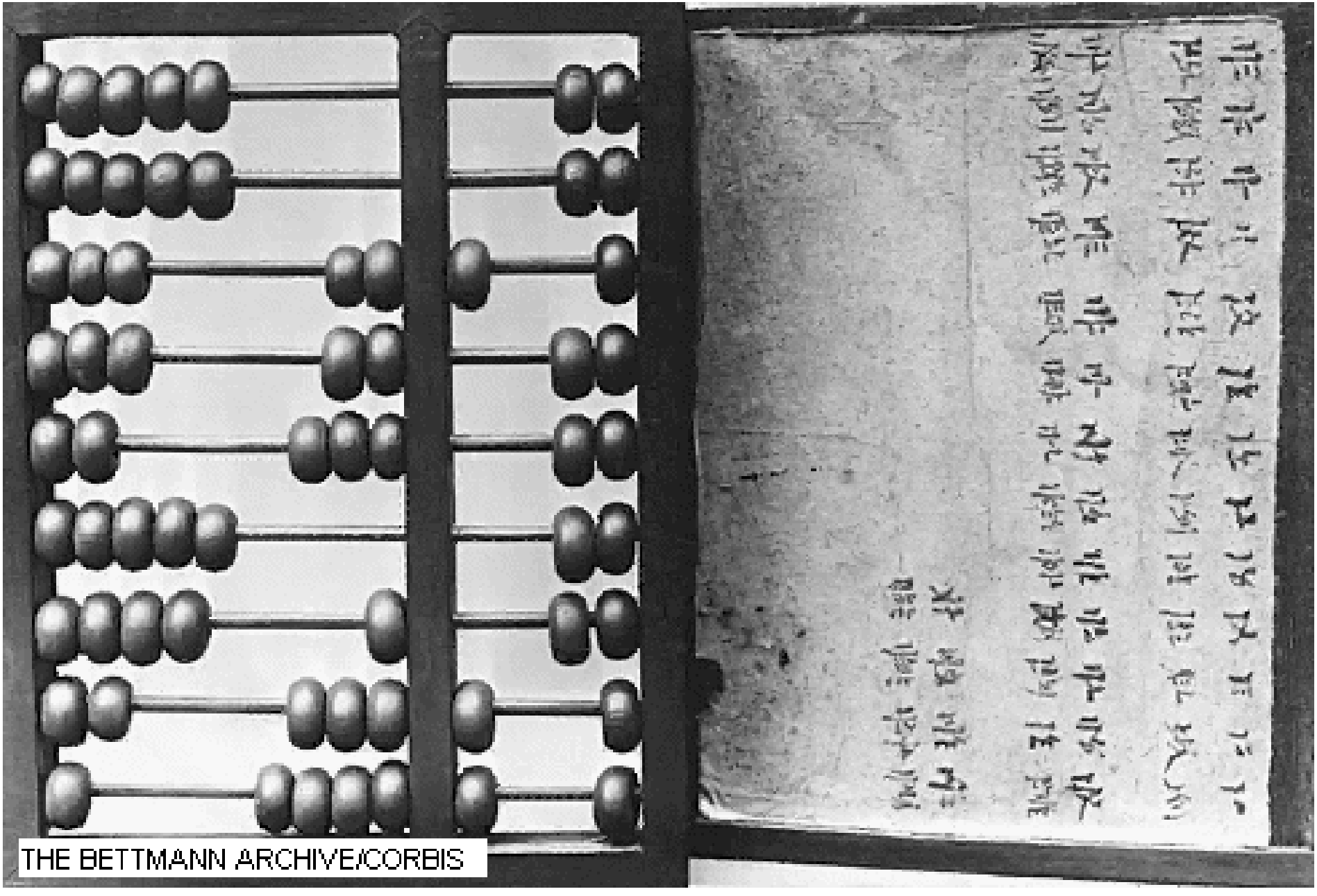
A pintek száma megint, de jobb lenne így:

Arató András
Bornemissza Péter
Cuczor Áron



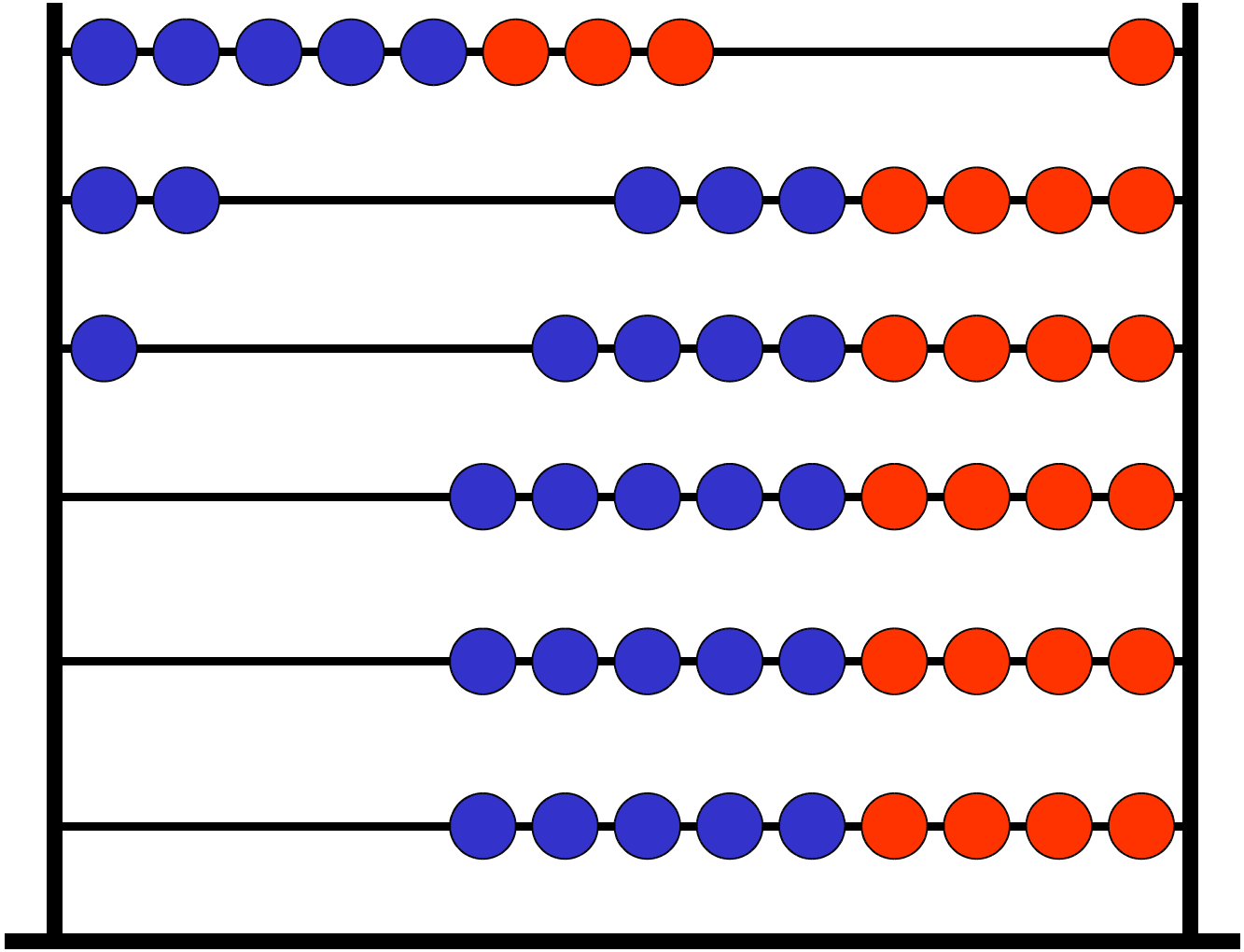
Feltéve persze, hogy 5-ös vagy 10-es számrendszert használunk

ABACUS



SZÁMOLÓGÉP

8
2
1
0
0
0



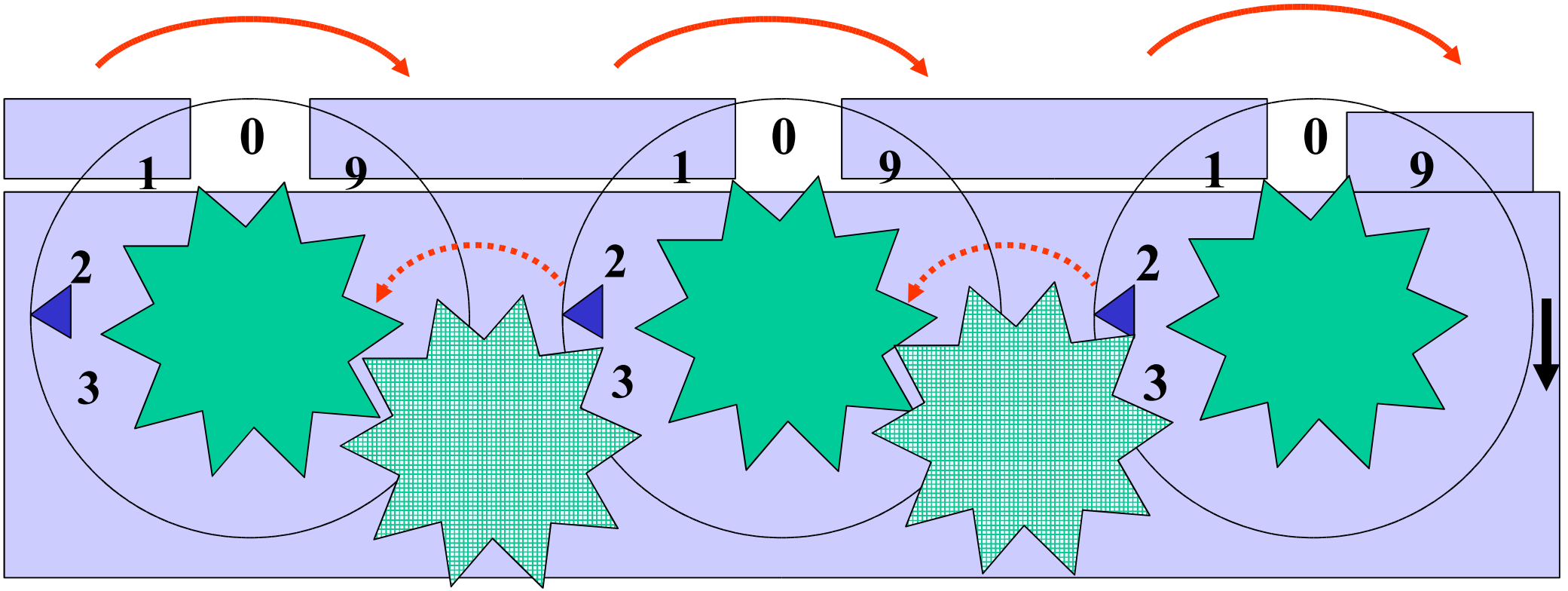
PASCAL in 1642

Pascal, **Blaise** 1623-1662

Francia filozófus és matematikus.
A tizes számrendszer jegyeit egy körtárcsa kerületére írva megoldja az átviteljegy (carry) automatizálását s ezzel a számlálás, valamint az erre alapuló összeadás illetve kivonás gépesítését.

Blaise PASCAL

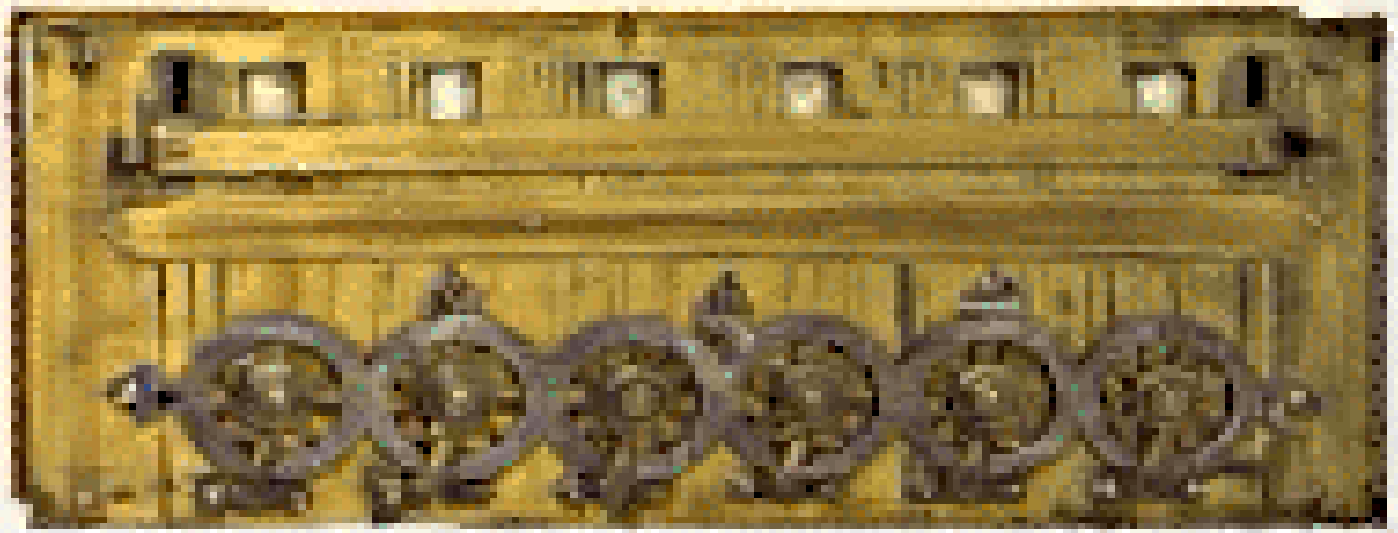
PASCAL KALKULÁTORA (1642)



Az „átvitel-jegy” automatikusan tovább vonul !

A kallantyúval számlálásra is alkalmazható

AZ ŐSLELET



PASCAL KALKULÁTORA

LEIBNITZ in 1692

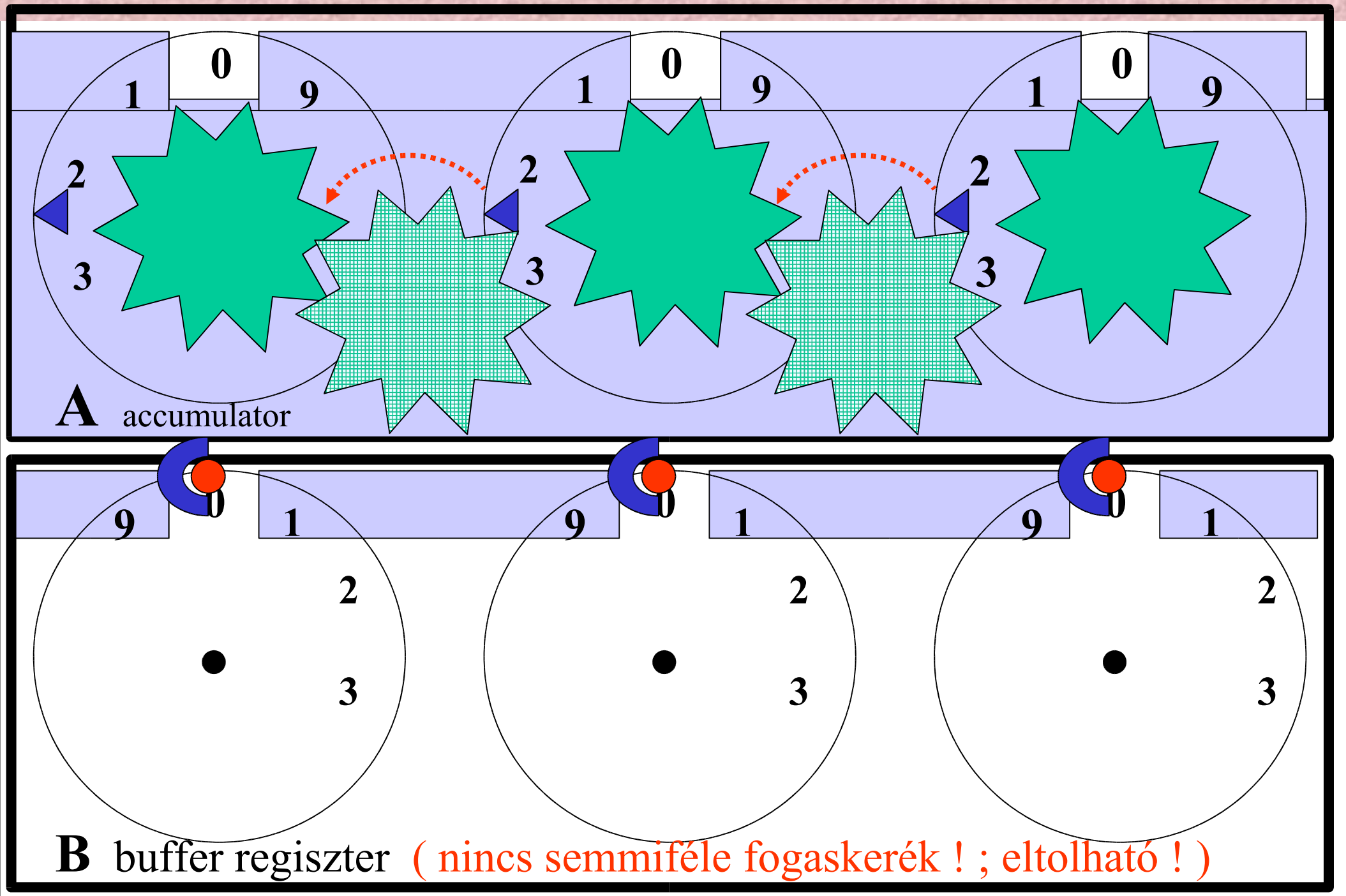
Gottried Wilhelm von LEIBNI(T)Z 1646-1716

Német filozófus és matematikus.

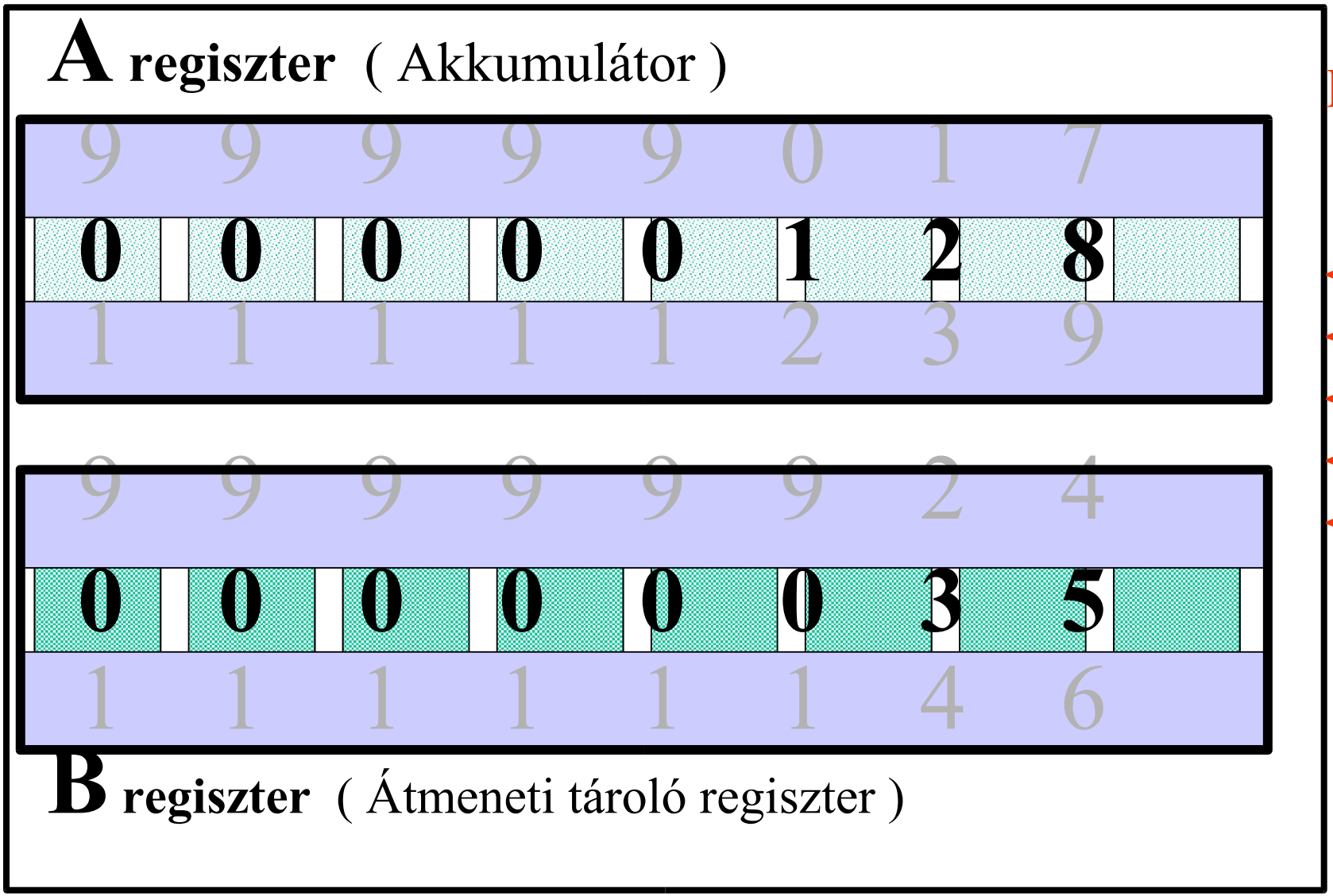
Pascal kalkulátorát továbbfejleszti, bevezetve az alapműveletek mindkét operandusa számára egy-egy regisztert, melyek közül az egyik eltól-ható. Így megvalósíthatja a tizzel, majd ismételt összeadás ill. kivonás segítségével a tetszőleges számmal való szorzást ill. osztást is.

Gottfried Wilhelm LEIBNIZ

LEIBNITZ KALKULÁTORA (1692)



ARITMETIKAI EGYSÉG



(A kerekek most élükkel állnak a két regiszterben)

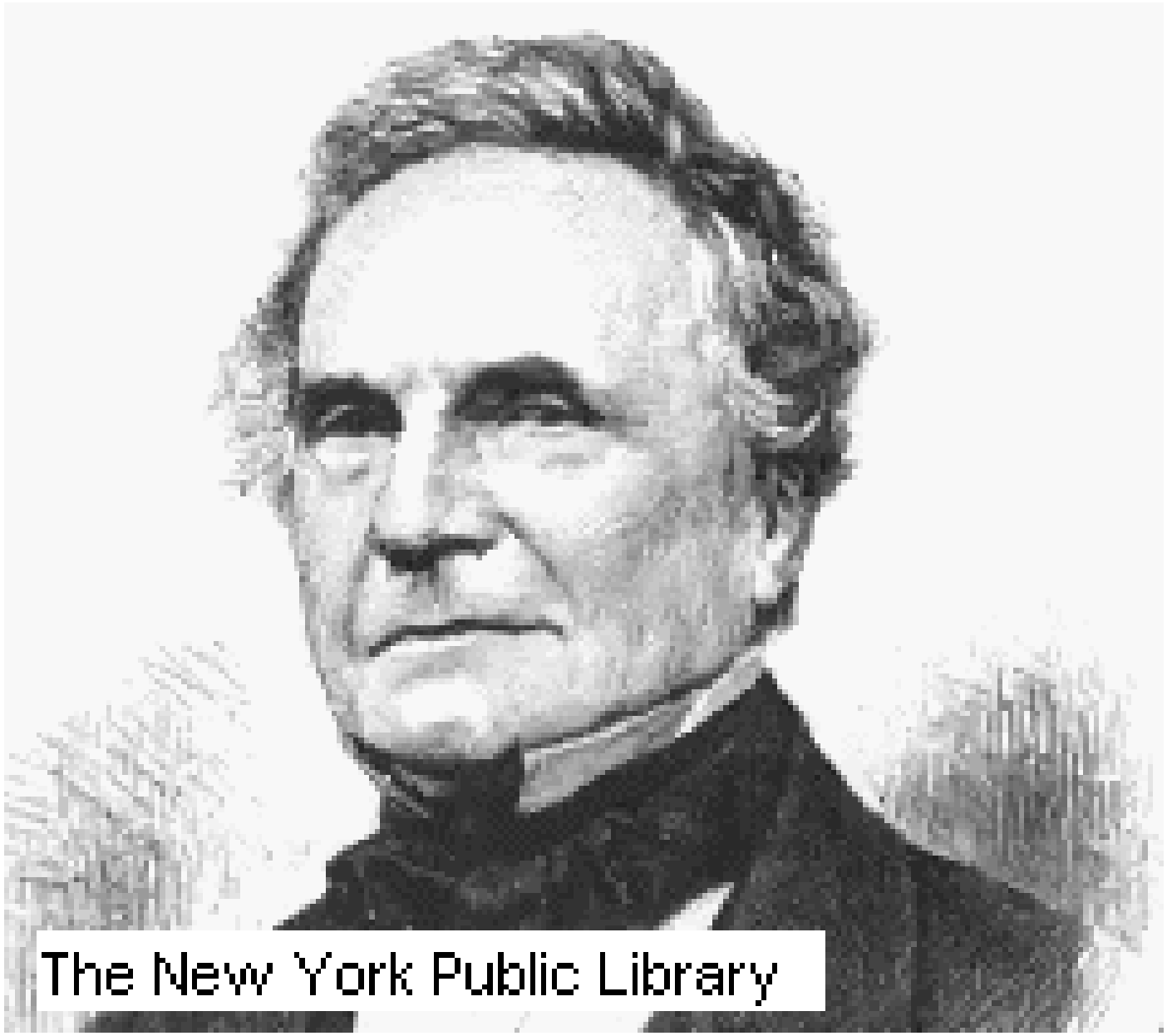
Charles Babbage

Charles Babbage 1792- 1871

A polinómok táblázatolására kifejleszti és megépíti az u.n. Differencia Gépet.

Ennek továbbfejlesztése képen az egymáshoz kapcsolt hat összeadó helyett egyetlen kalkulátort (aritmetikai egységet) és sok tároló rekeszt tartalmazó MEMÓRIÁT javasol, melyből az adatok lyukkártyákon tárolt utasítások nyomán jutnak az aritmetikai egységbe ill. abból vissza a memóriába. Ez a mai COMPUTER őse.

Charles BABBAGE



AZ ANALÍZIS MEGJELENÉSE

Leibnitz és Newton kialakítják az infinitézimális számítást.

Kialakul az analízis. Fel lehet írni differenciálegyenleteket.

Még a legegyszerübbeknek sincs zárt alakú megoldása:

$y' + y = 0$ megoldása $y = e^{-x}$; y nem számítható ki x-ből
$y'' + y = 0$ „ $y = \sin x$; $\cos x$ „

Ezeknek a függvényeknek az értékét táblázatok adják meg.

Ezeket a táblázatokat meg kellett csinálni. (Logar- tábla)

Babbage ezt a táblázatkészítést akarta automatizálni.

TAYLOR SOR

Babbage tudta, hogy
minden tisztességes függvény
hatványsorba fejthető:

$$f(x) = f(0) + f'(0).x/1! + f''(0).x^2/2! + \ldots..$$

Pl.:

$$\sin(x) = x - x^3/3! + x^5/5! - x^7/7! + \ldots..$$

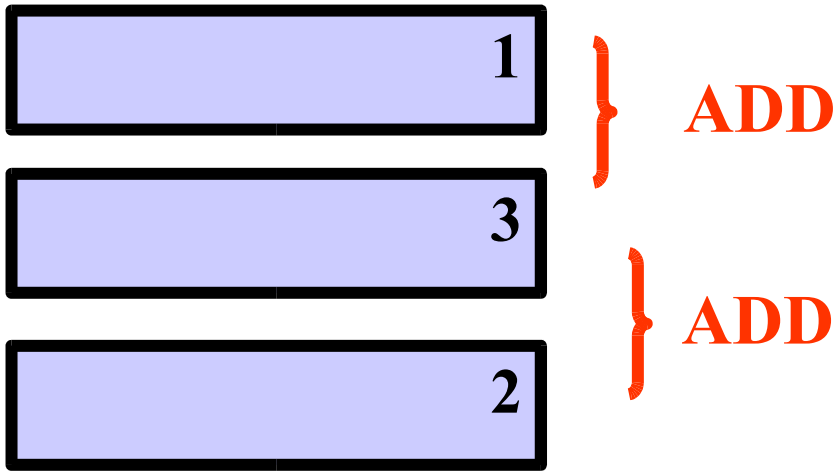
DIFFERENCIA GÉP

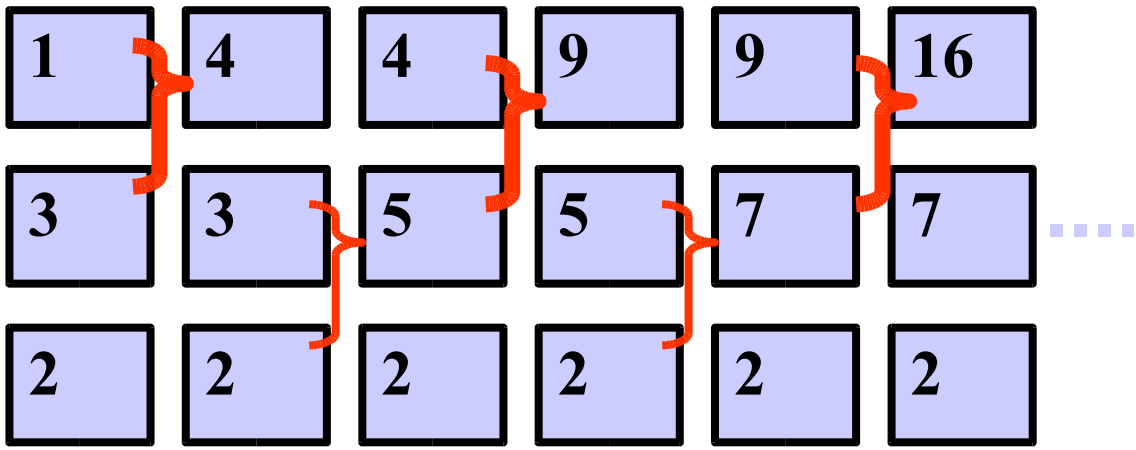
Babbage ugyancsak tudta, hogy egy

n-ed rendű polinóm n-edik differenciája konstans

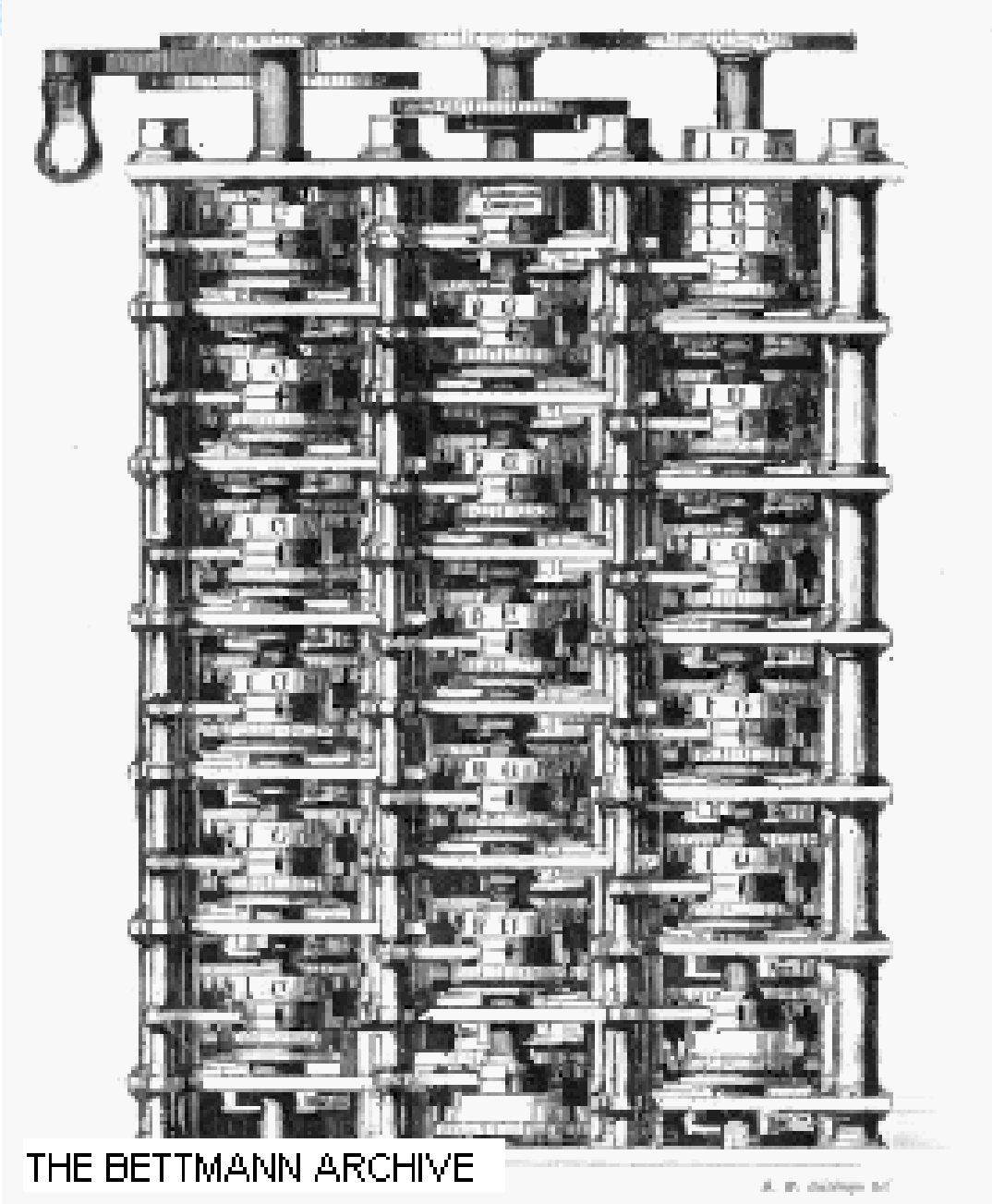
Pl.: $y = x^2 + 2x + 1$

x	y	dy	d^2y
0	1		
		3	
1	4		2
		5	
2	9		2
		7	
3	16		2
		9	
4	25		2
		11	
5	36		





BABBAGE's DIFFERENCE ENGINE



A MEMÓRIA MEGJELENÉSE

Babbage megépített hat regiszteres differencia gépe maximum ötödfokú polinómok táblázatolását engedte meg.

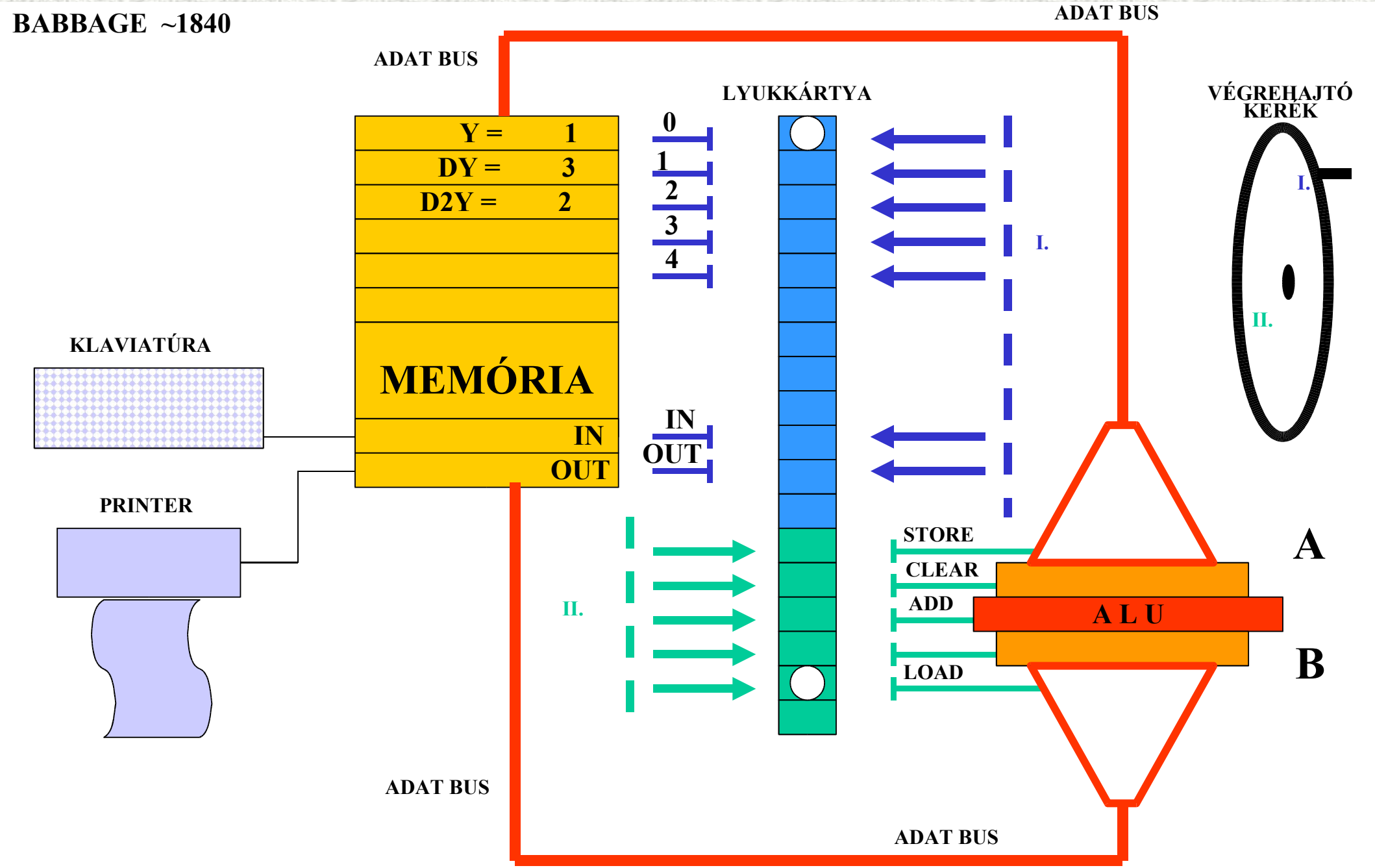
Bonyolultabb pl. trigonometrikus függvények megközelítéséhez nem elégséges egy ötödfokú hatványsor. 10, 20, 50 vagy tetszőleges n-ed fokú polinómokra lehet szükség. Ez ugyanennyi bonyolult fogaskerekes összeadómű megépítését tenné szükségessé.

És itt jött Babbage korszaknyitó ötlete: Használjunk csak egyetlenegy -- felépítésében a szűkséges fogaskerékrendszer miatt bonyolult -- összeadó-művet, vagy általában aritmetikai egységet, s n darab közönséges keréktárcsákból álló tároló regisztert. Ez lesz a MEMÓRIA.

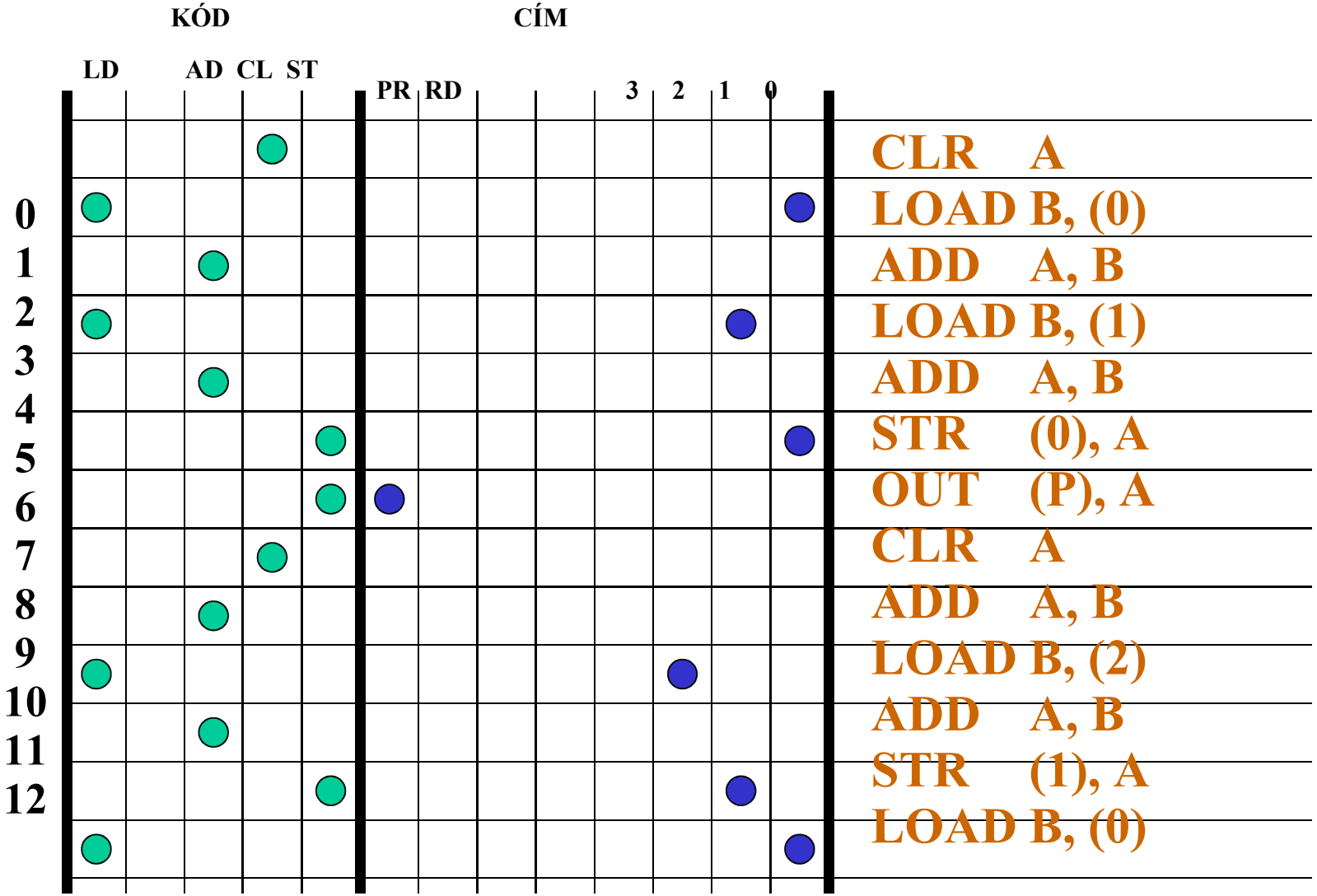
Meg kell oldani, hogy a kezdő és átmeneti értékeket tároló memória-regiszterek tartalma -- valamilyen módon megvalósítandó adatátvitel révén -- cserélhető legyen az aritmetikai egység A és B regisztereinek tartalmával.

Az adatátvitelek sorrendjét egy lyukkártyán tárolt program alapján működő vezérlőmű vezérelné. És ez lett a később megtáltosodó gondolatok magja.

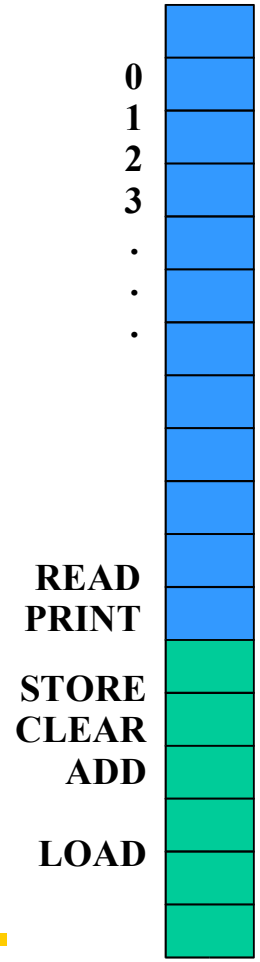
ANALYTICAL ENGINE



GÉPI és ASSEMBLY KÓD



LYUKKÁRTYA



KETTES SZÁMRENDSZER

A tizes számrendszerben működő aritmetikai egység és az ugyancsak tizes számrendszerű memóriarekeszek közötti

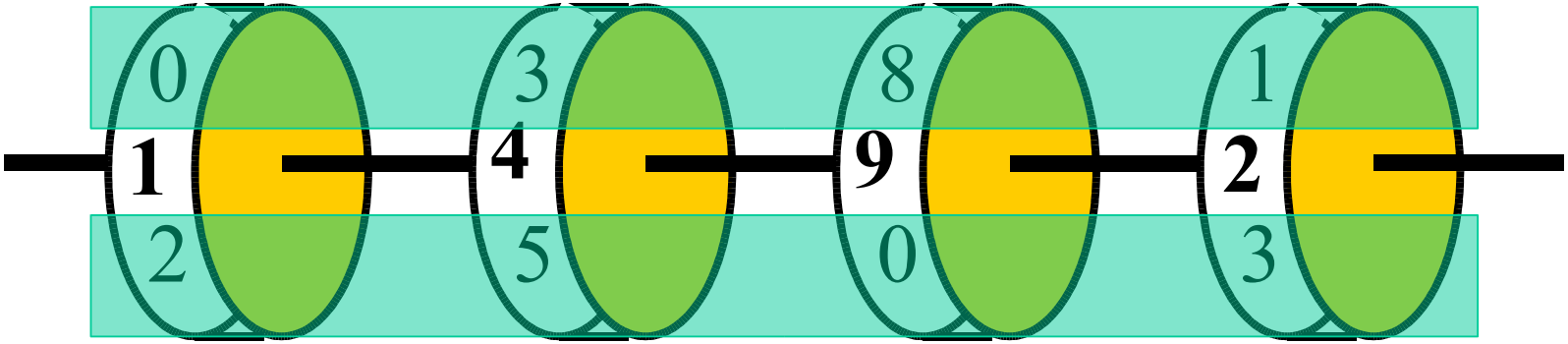
ADATÁTVITEL

nem volt megvalósítható a kor technológiai szinvonalán Babbage minden erőfeszítései ellenére sem.

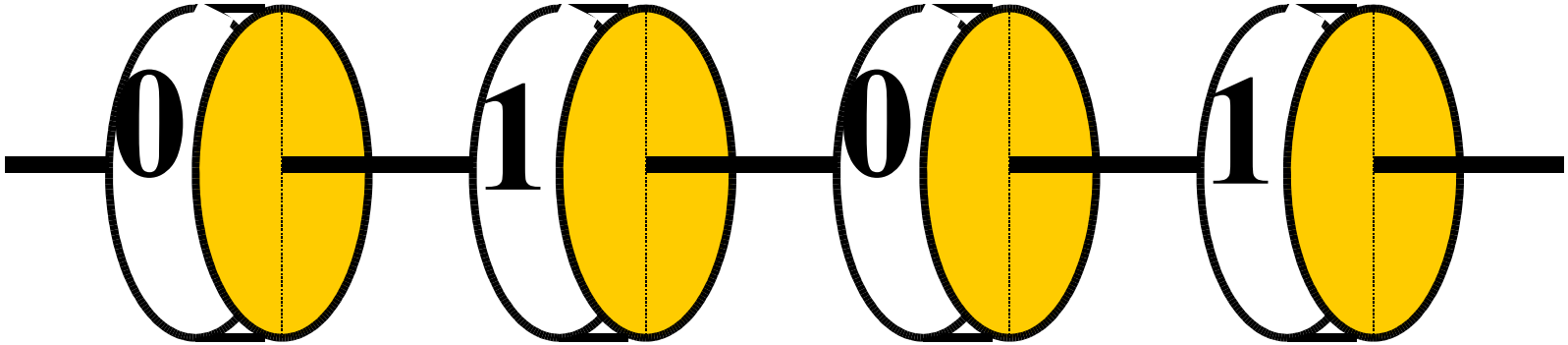
A minimális számjegyű, csak két számjegyet tartalmazó kerekek felvetése villantott fel a megoldható adatátvitelre némi reményt, de csak 100 év késéssel és kerekek nélkül.

REGISZTEREK

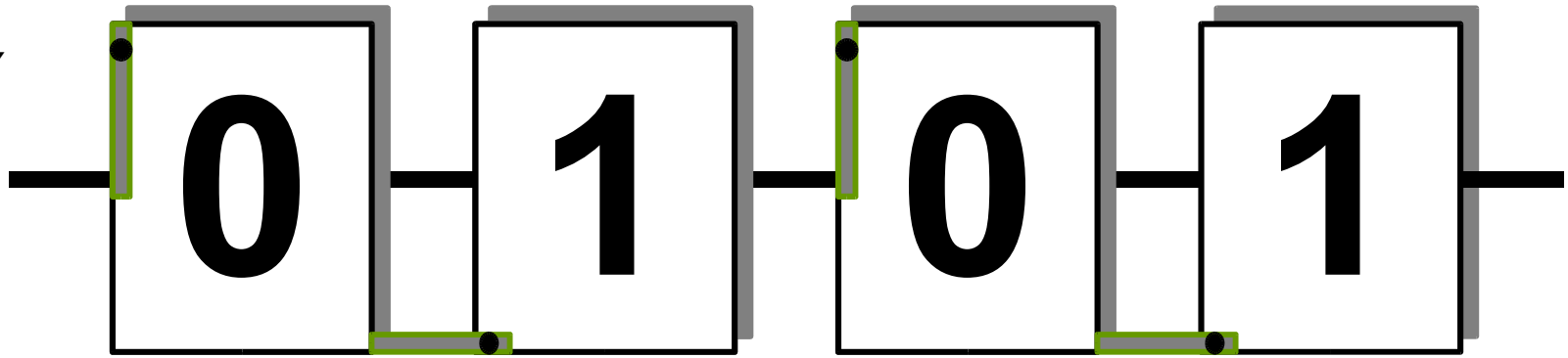
DECIMÁLIS
(TIZES)



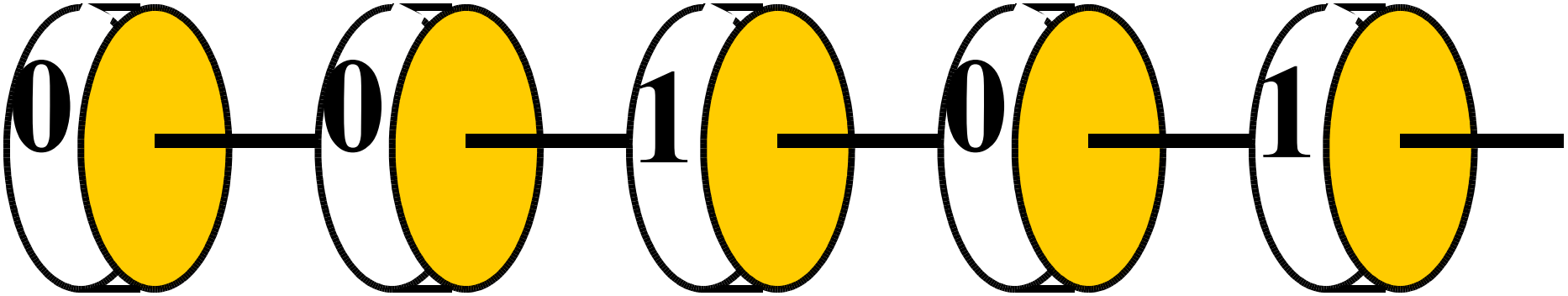
BINÁRIS
(KETTES)



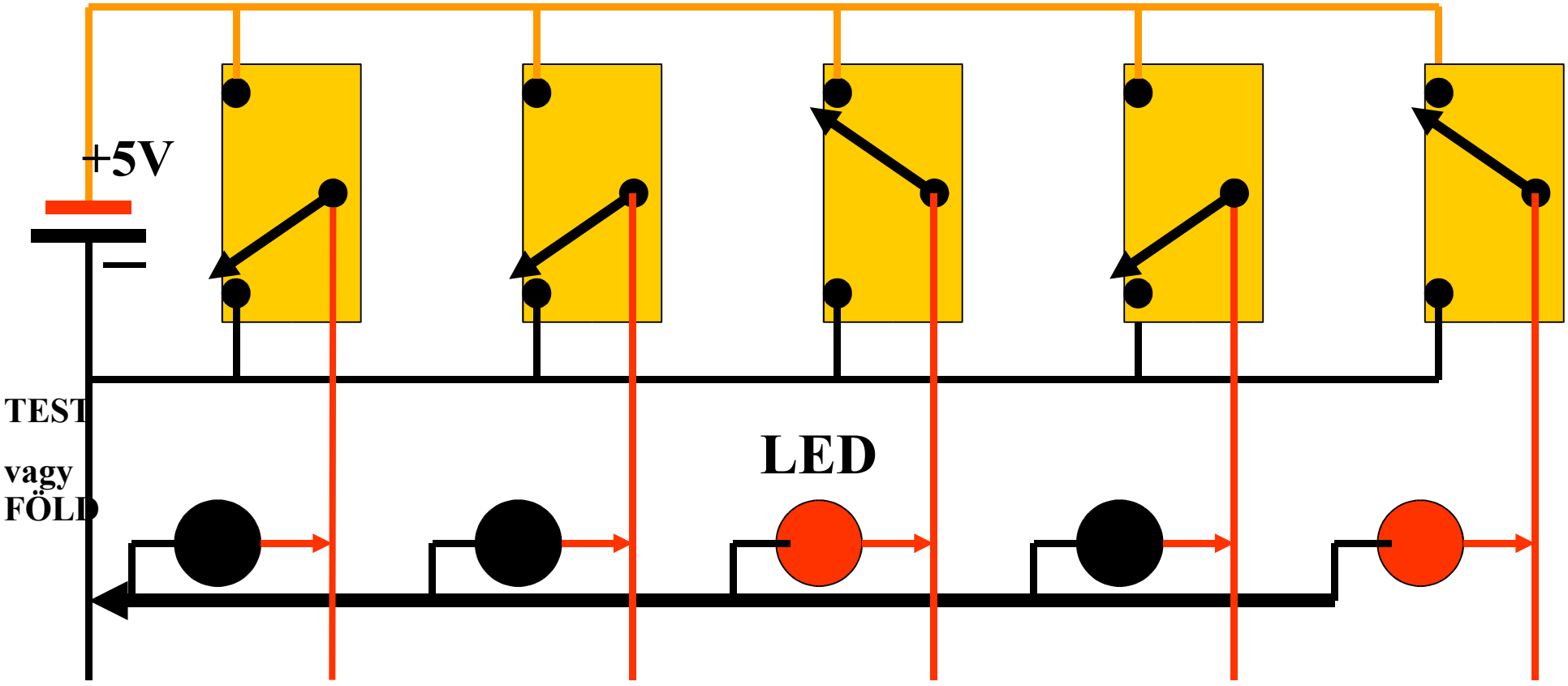
HA BINÁRIS, ÚGY
DOMINÓKBÓL
IS LEHETNE



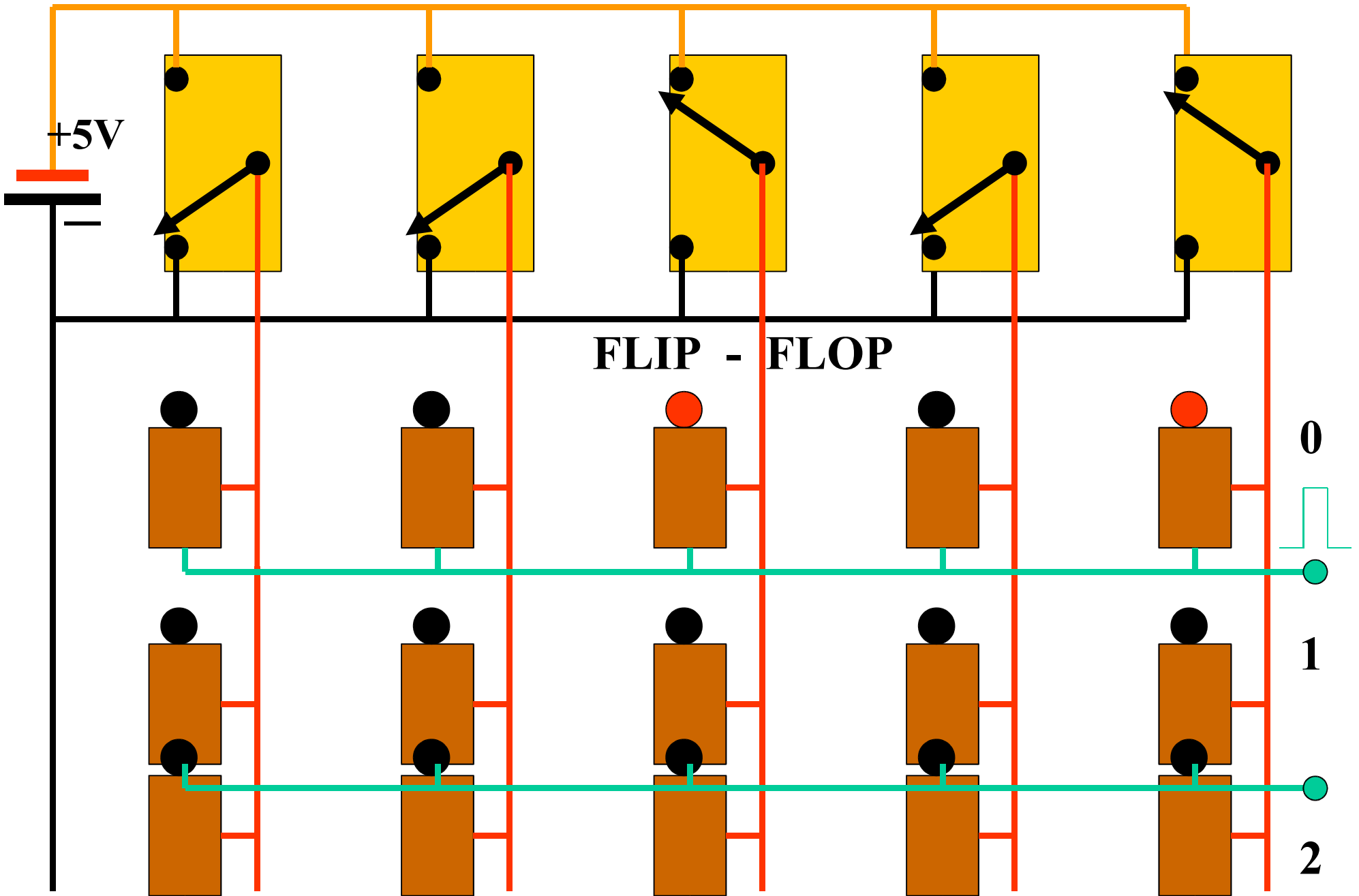
BINÁRIS REGISZTER



KAPCSOLÓ REGISZTER



MEMÓRIA ’FLIP - FLOP’-OKBÓL



ÍRHATÓ - OLVASHATÓ REGISZTEREK

Két értéket (bináris jegyet, bitet) tároló (elektromosan írható és olvasható) eszközt először relékkel, majd elektronikusan, úgy nevezett FLIP-FLOP-okkal tudtak megvalósítani. A kerekek alkalmazása helyett ezekkel a működési sebesség jelentősen megugorhatott.

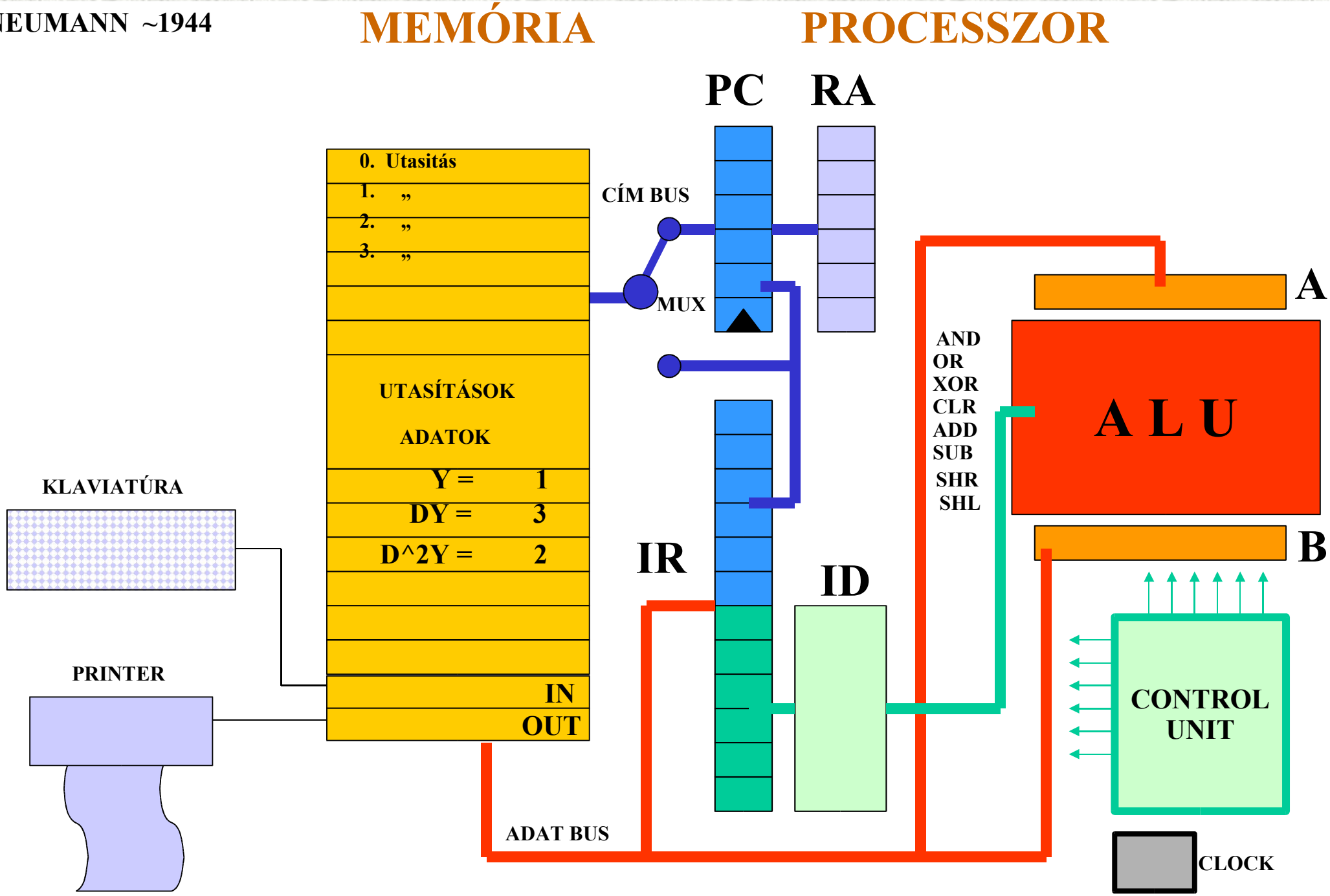
1944-46 között NEUMANN János elemezte, hogy ilyen ígéretes sebességek mellett milyen kell legyen egy számítógép méltó architektúrája. Megszületett a tárolt programmozású COMPUTER gondolata.
A több száz millió PC is 'von Neumann computer'.

NEUMANN János

Budapest 1903 - Princeton (USA) 1957

Von NEUMANN COMPUTER

NEUMANN ~1944

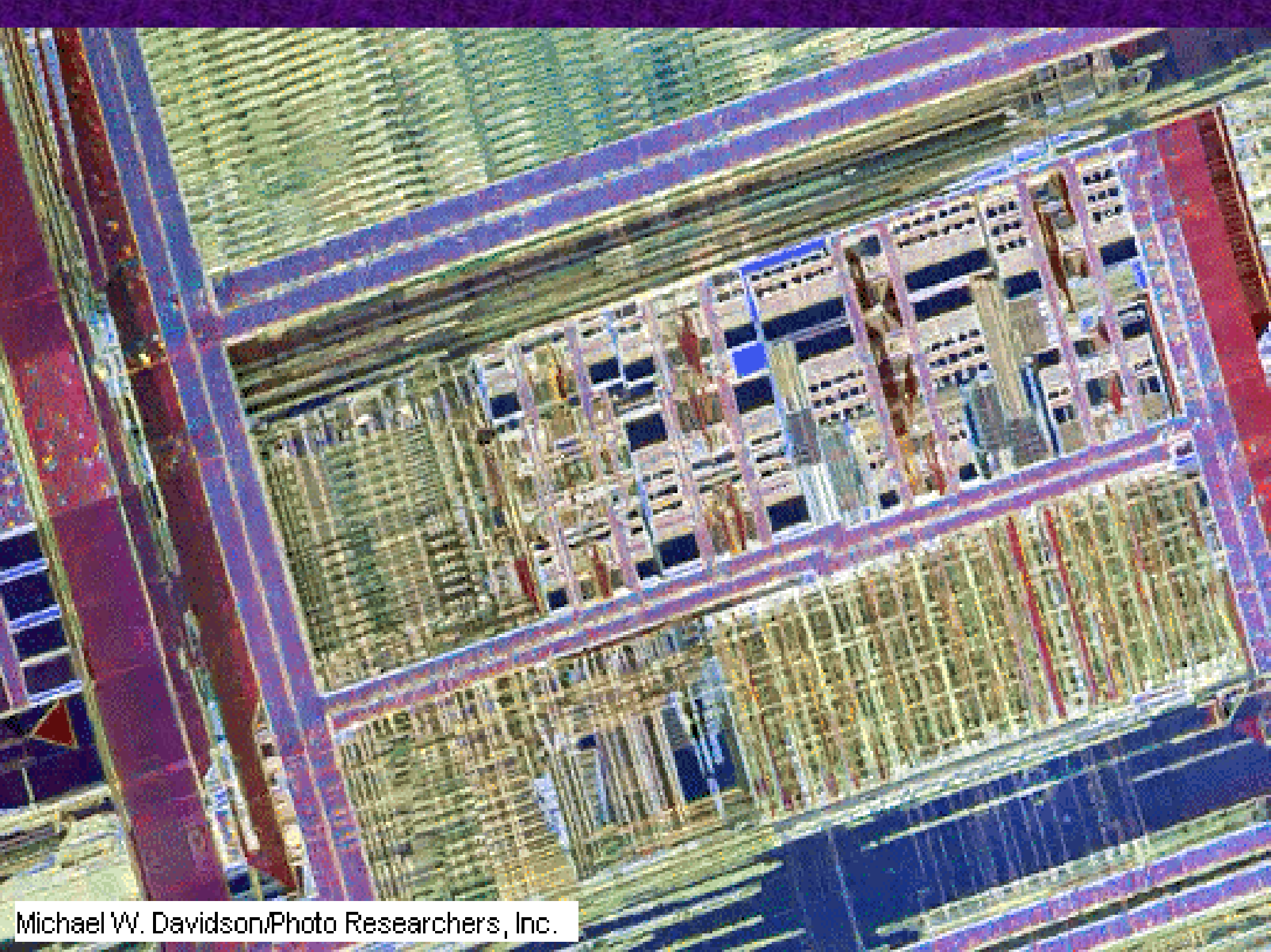


A MIKROPROCESSZOR

Az Aritmetikai és Logikai, valamint a Vezérlő Egység közös neve: PROCESSZOR.

A hetvenes években sikerült a számítógép processzorát egy körömnyi nagyságú szilicium-lapka felületén létrehozni.
Ekkortól beszélünk MIKROPROCESSZOR-ról.

Ma ezek már több millió tranzisztort tartalmaznak és évi több tiz-milliós szériákban készülnek.



Michael W. Davidson/Photo Researchers, Inc.

MIKROSZÁMÍTÓGÉPEK

Amint idővel tekintélyesebb (>kByte) mennyiségű memóriát is sikerült a processzor mellett egy chipen megvalósítani, úgy megszülettek az egy chipes MICROCOMPUTER-ek.

Ezek ma már évi milliárdos szériákban készülnek.

Egy népszerű MIKROSZÁMÍTÓGÉP

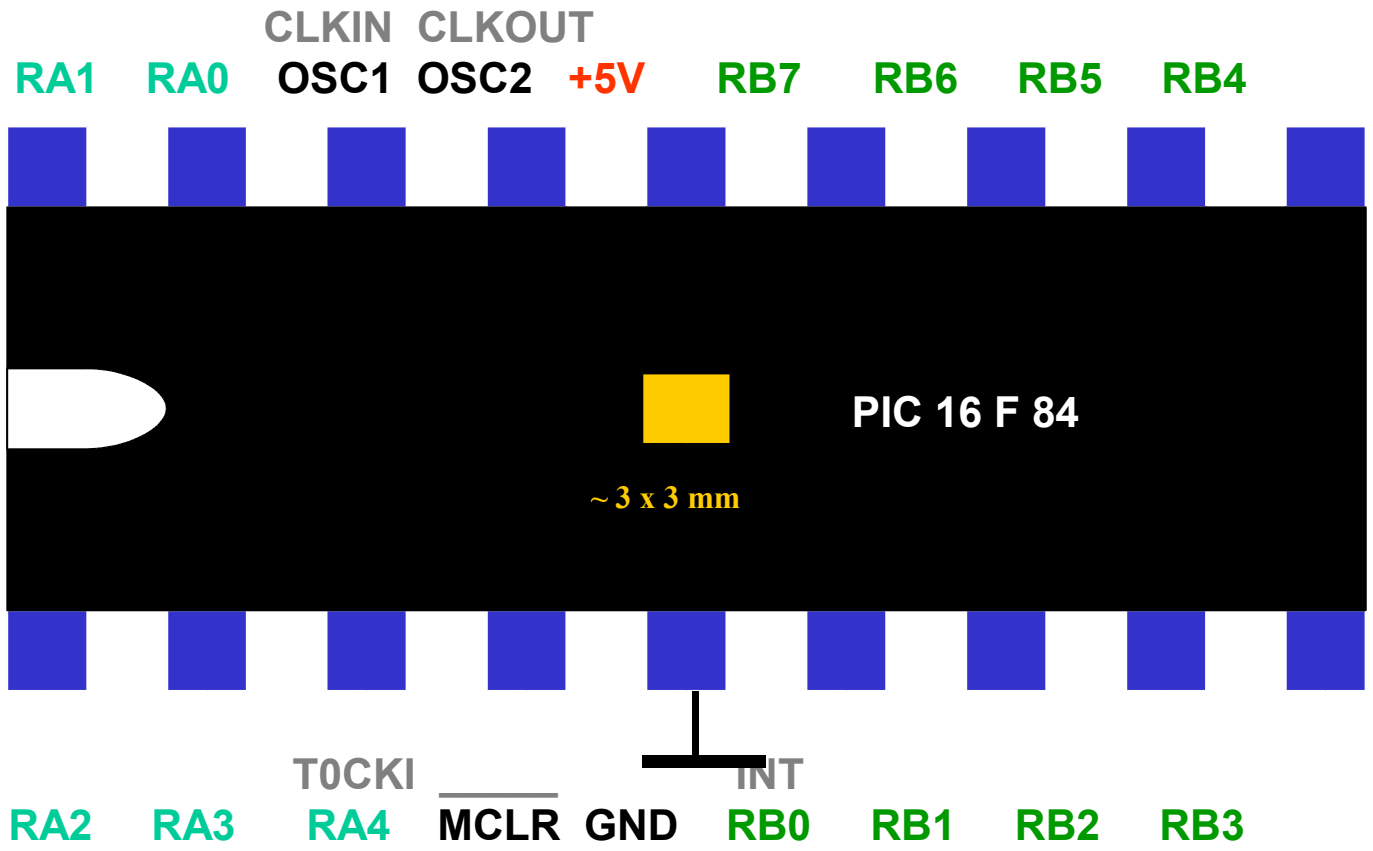
PIC 16 F 84

8 bites RISC processzor
1 K * 14 bit FLASH Program Memória
68 byte Adat RAM
64 byte Adat EEPROM
13 I/O PIN

Ára: 880 Ft

Hitelkártyába épített változata: 2500 Ft

ÚJRA PROGRAMOZHATÓ!



A LOGIKA KIMŰVELŐI

Az **ENCARTA** encyclopedia -ban megnézzük az:

- **ARISTOTLE**
- **BOOLE**
- **SHANNON**

cimszavakat

BOOLE ALGEBRA 1

- A három fundamentális logikai kapura a következő posztulátumok érvényesek:

- NOT 0’=1; 1’=0; Ha nem 0, úgy 1 kell legyen -- és fordítva.
- AND 0.0=0; 0.1=0; 1.0=0; 1.1=1
- OR 0+0=0; 0+1=1; 1+0=1; 1+1=1

BOOLE ALGEBRA 2

- Boole kifejezések egyszerűsítéséhez az (előbbi posztulátumok alapján bizonyítható) alábbi összefüggések használhatók:

• EGY és ZÉRUS szabály	0+A=A; 0.A=0; 1+A=1; 1.A=A
• Kommutativitás	A+B=B+A; AB=BA
• Asszociativitás	A+(B+C)=(A+B)+C; A(BC)=(AB)C
• Disztributivitás	A+BC=(A+B)(A+C); A(B+C)=AB+AC
• Idempotencia	A+A=A; A.A=A
• Komplementaritás	A+A'=1; A.A'=0
• Abszorpció	A+AB=A; A(A+B)=A; A+A'B=A+B
• Involució	(A')'=A
• Inverzió (de Morgan)	(A+B)'=A'.B'; (A.B)'=A'+B'

BOOLE ALGEBRA 3

- Egy példa Boole kifejezések egyszerűsítésére az elöbbi relációkkal:

• F = A + ABC + BCA’ + A’B + DA + D’A	
• = A + BC(A + A’) +A’B + A(D + D’)	disztribúció
• = A + BC + A’B +A	komplementaritás
• = A + BC + A’B	idempotencia
• = (A + A’B) + BC	asszociativitás
• = A + B + BC	abszorpció
• = A + (B + BC)	asszociativitás
• = A + B	abszorpció

AUTOMATA FORMALIZMUS

Egy **véges automata** egy **(A, S, V, k, f, g)** hatos rendszer ahol:

A egy véges elemszámú ábécé, amelynek szimbólumai képezik az automata bemeneteit;

S az automata lehetséges állapotainak halmaza;

V az automata egyes állapotaiban válaszként adható szimbólumok halmaza;

k az automata kezdô állapota, $k \in S$;

g az a leképzés, amellyel az **S** halmazt a **V** halmazra képezi le az automata, $v = g(s)$, $v \in V$ és $s \in S$;

f az a leképzés, amellyel az **A** és **S** halmazok szorzatát az **S** halmazra képezi le az automata, $q = f(s, a)$, $a \in A$, $s \in S$, $q \in S$.

AZ AUTOMATA MŰKÖDÉSE

Adott $x \in A$ esetén az automata mûködése a következôképen írható le:

Legyen $x = x_1, x_2 \ldots x_n$, és a válaszszöveg

$y = y_1, y_2 \ldots y_n$.

Jelölje az automata mindenkori állapotát s_i, $i = 1, 2, \ldots n$.

1. $i \leftarrow 1$,

2. $s_i \leftarrow k$,

3. $y_i \leftarrow g(s_i)$ **Moore, if** $g(s_i, x_i)$ **then Mealy,**

4. $s_{i+1} \leftarrow f(s_i, x_i)$,

5. $i \leftarrow i + 1$,

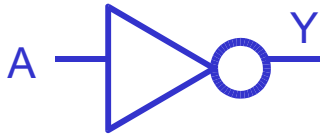
6. ha $i \leq n$, **ismételjük a feladatot a 3. ponttól.**

A LOGIKA ELEMEI

A logikai változók lehetséges értékei: **0 , 1**

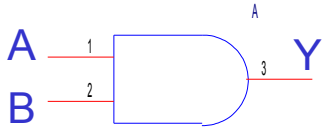
Ezeket 0 V illetve +5 V feszültségszinttel fogjuk reprezentálni

Egyváltozós logikai művelet a **NEGÁLÁS**: **Y = A'**

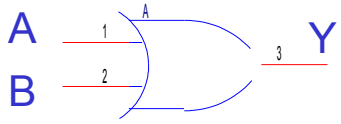


Kétváltozós logikai művelet az **ÉS** és **VAGY**:

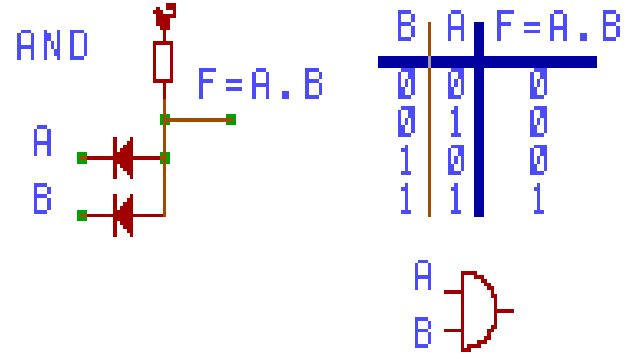
Y = A.B



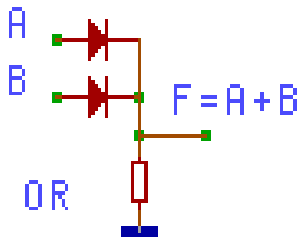
Y = A+B



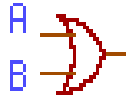
LOGIKAI KAPUK

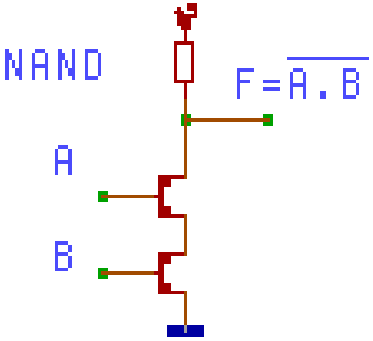


B	A	F=A.B
0	0	0
0	1	0
1	0	0
1	1	1

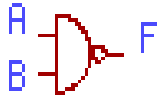


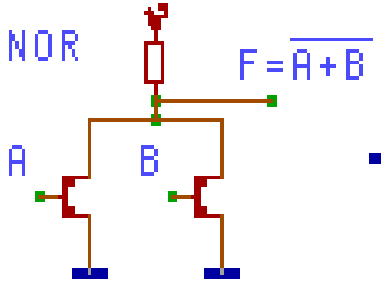
B	A	F=A+B
0	0	0
0	1	1
1	0	1
1	1	1



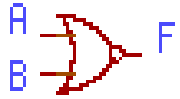


B	A	F=$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0





B	A	F=$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0



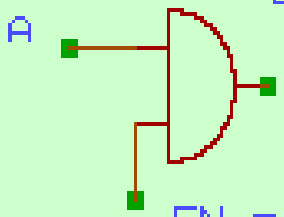
All possible two-variable logic (or Boolean) functions

which could be implemented by two-input logic gates

B	A	f_0	NOR	f_2	f_3	f_4	f_5	XOR	NAND	AND	EQU	f_{10}	f_{11}	f_{12}	f_{13}	OR	f_{15}
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

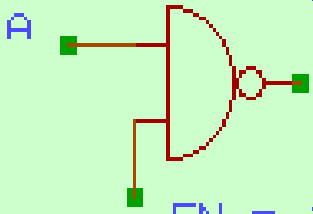
ENGEDÉLYEZÉS és NEGÁLÁS

ENABLE



EN = 1 / 0
EN (gedelyez)
EN (able)

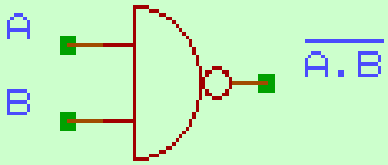
INVERT



EN = 1 / 0
EN (gedelyez)
EN (able)

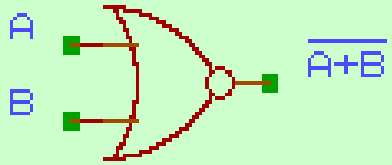
NAND

B	A	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0



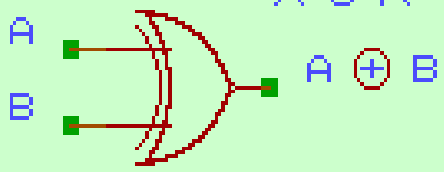
NOR

B	A	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0



XOR

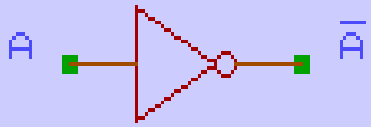
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0



De MORGAN SZABÁLY

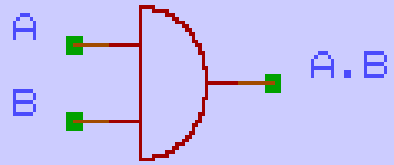
A	$\overline{A}$
0	1
1	0

INVERTER



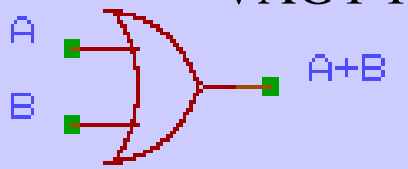
B	A	A.B
0	0	0
0	1	0
1	0	0
1	1	1

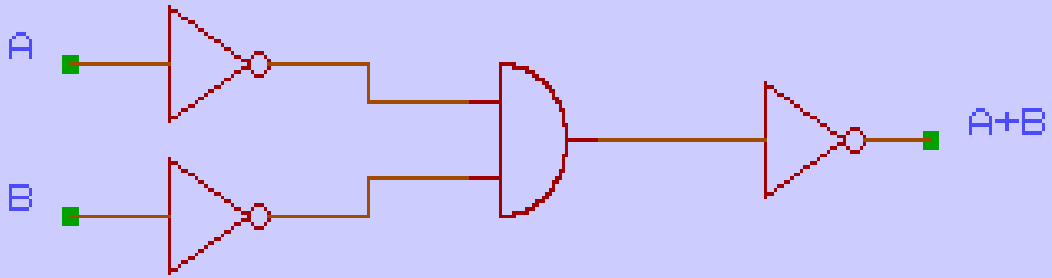
ÉS KAPU



B	A	A+B
0	0	0
0	1	1
1	0	1
1	1	1

VAGY KAPU





De MORGAN SZABÁLY $\overline{A . B} = A + B$

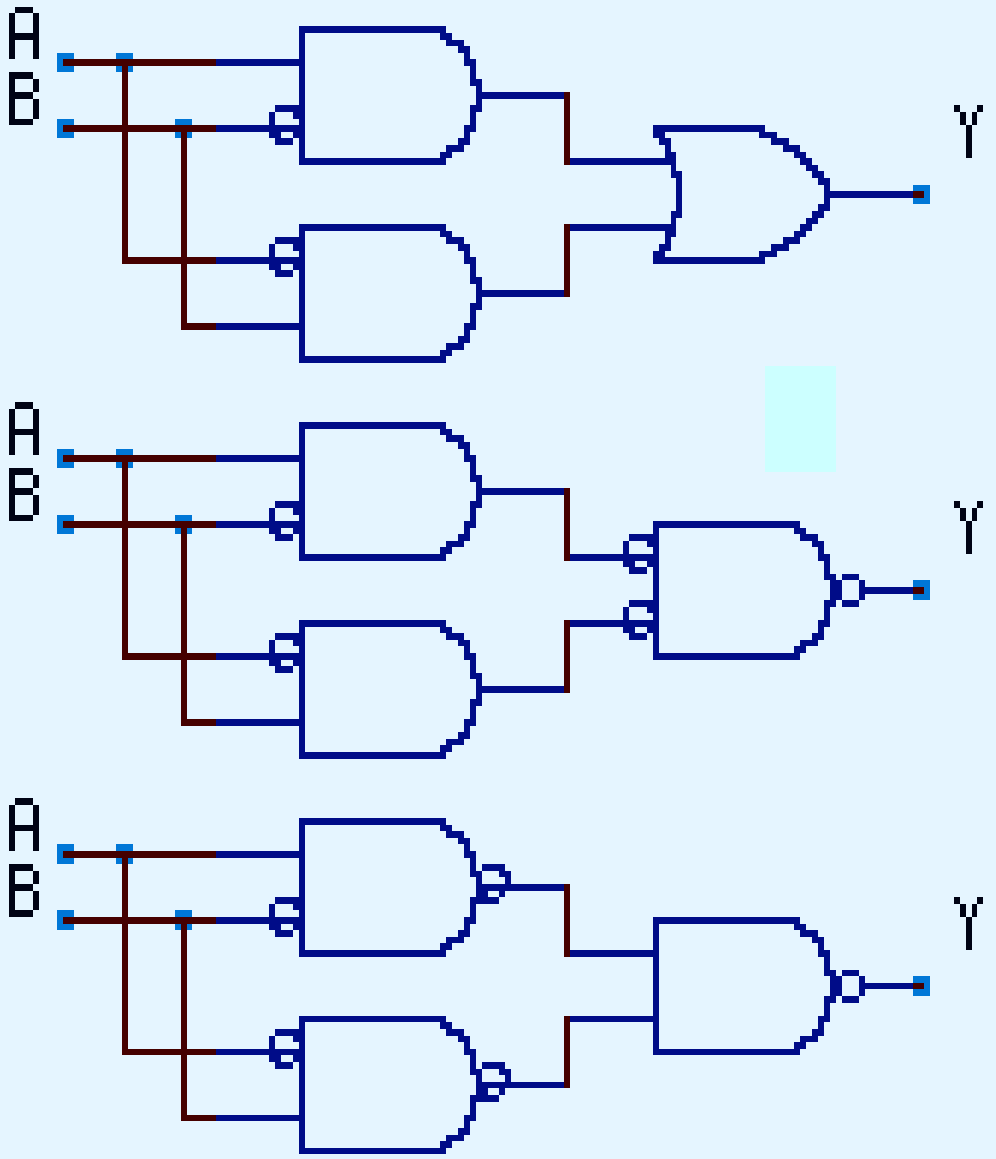
KOMBINÁCIÓS HÁLÓZATOK

KOMBINACIOS LOGIKAI
HALOZATOK TERVEZESE

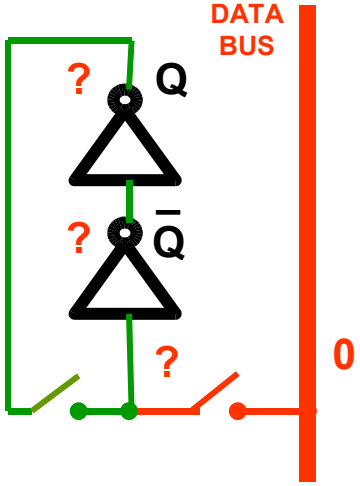
F1. a kovetkezo BOOLE fuggveny
leplesentalasa az igazsag-
tablazata alapjan az alabbiak
szerint tortenik:

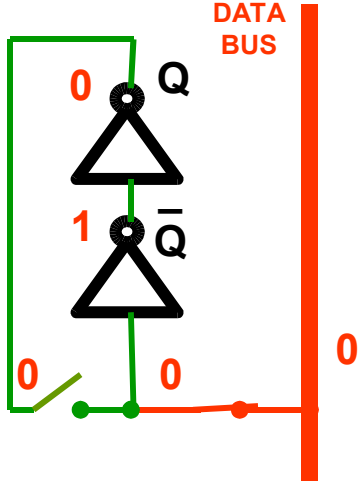
$Y = A\bar{B} + \bar{A}B$ (=A XOR B)

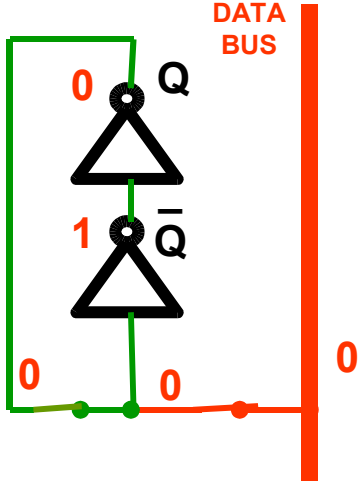
B	A	Y=XOR
0	0	0
0	1	1
1	0	1
1	1	0

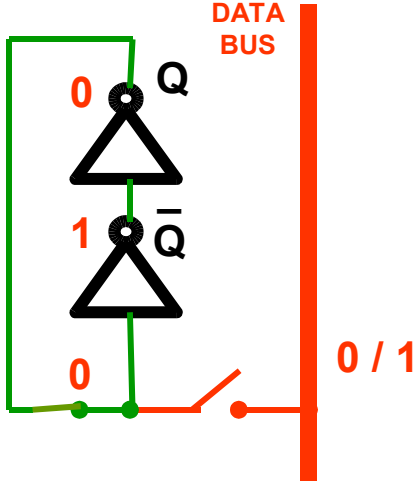


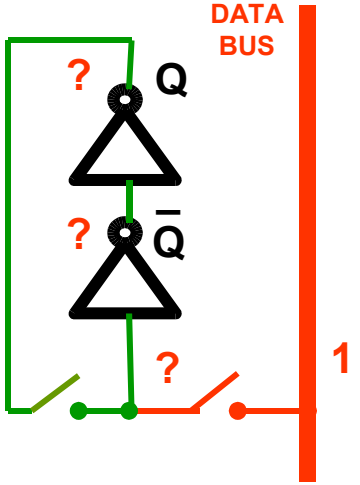
2 INVERTER = MEMORIA

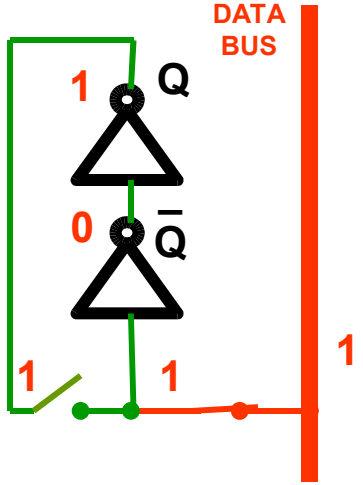


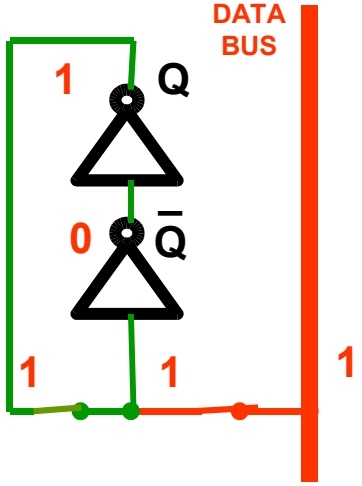


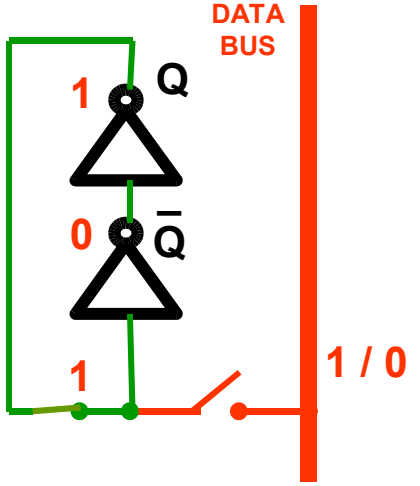




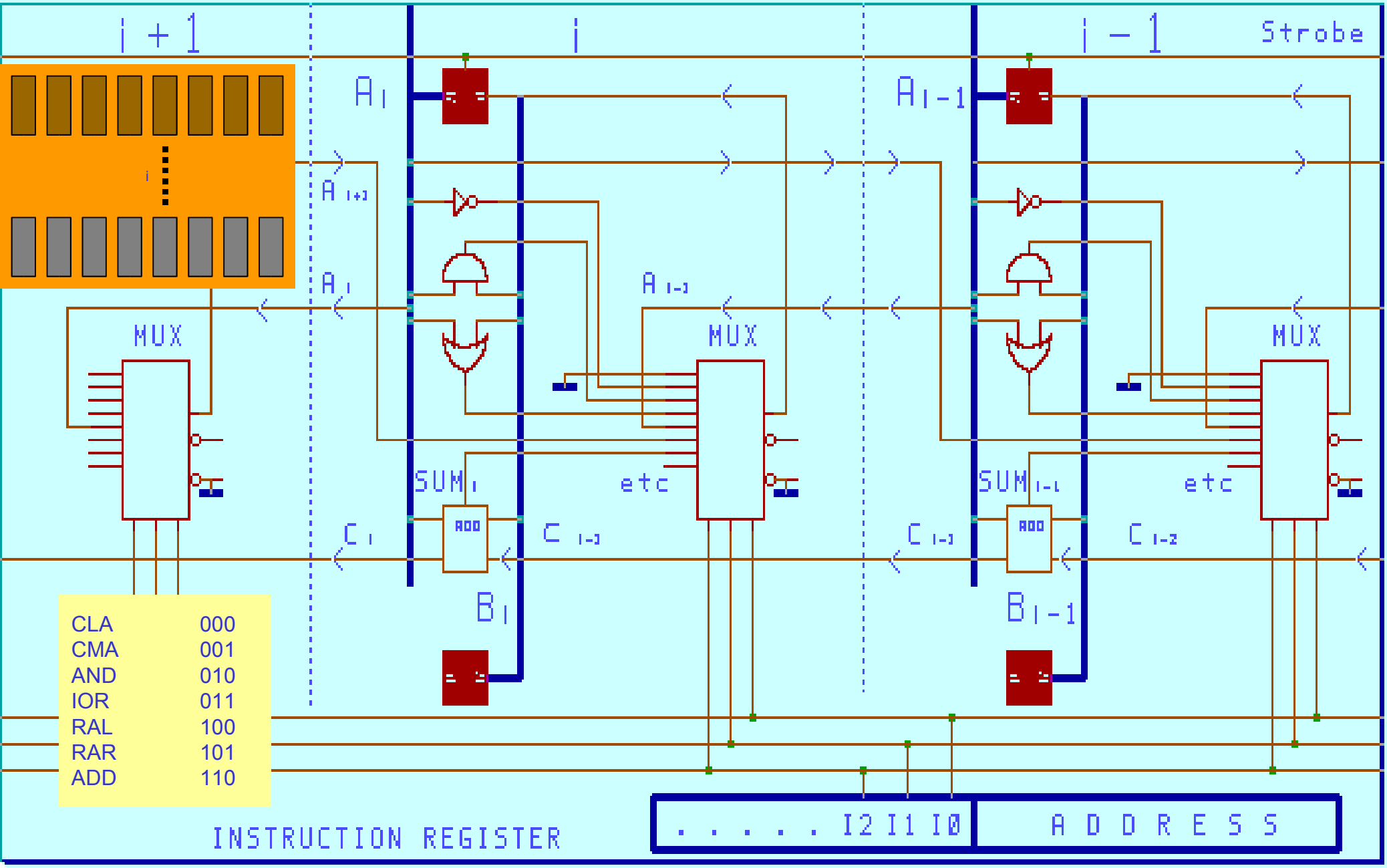




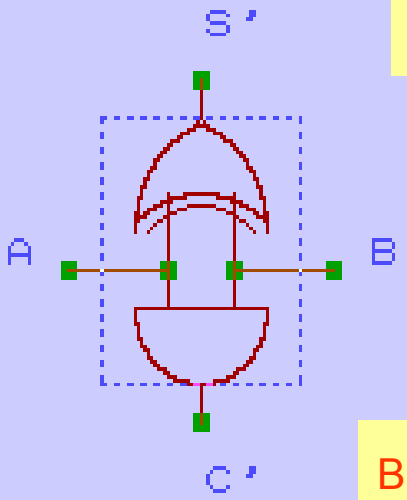




ARITMETIKAI-LOGIKAI EGYSÉG

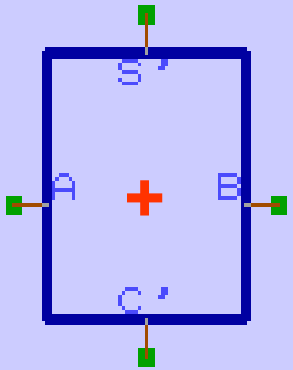


ÖSSZEADÓ



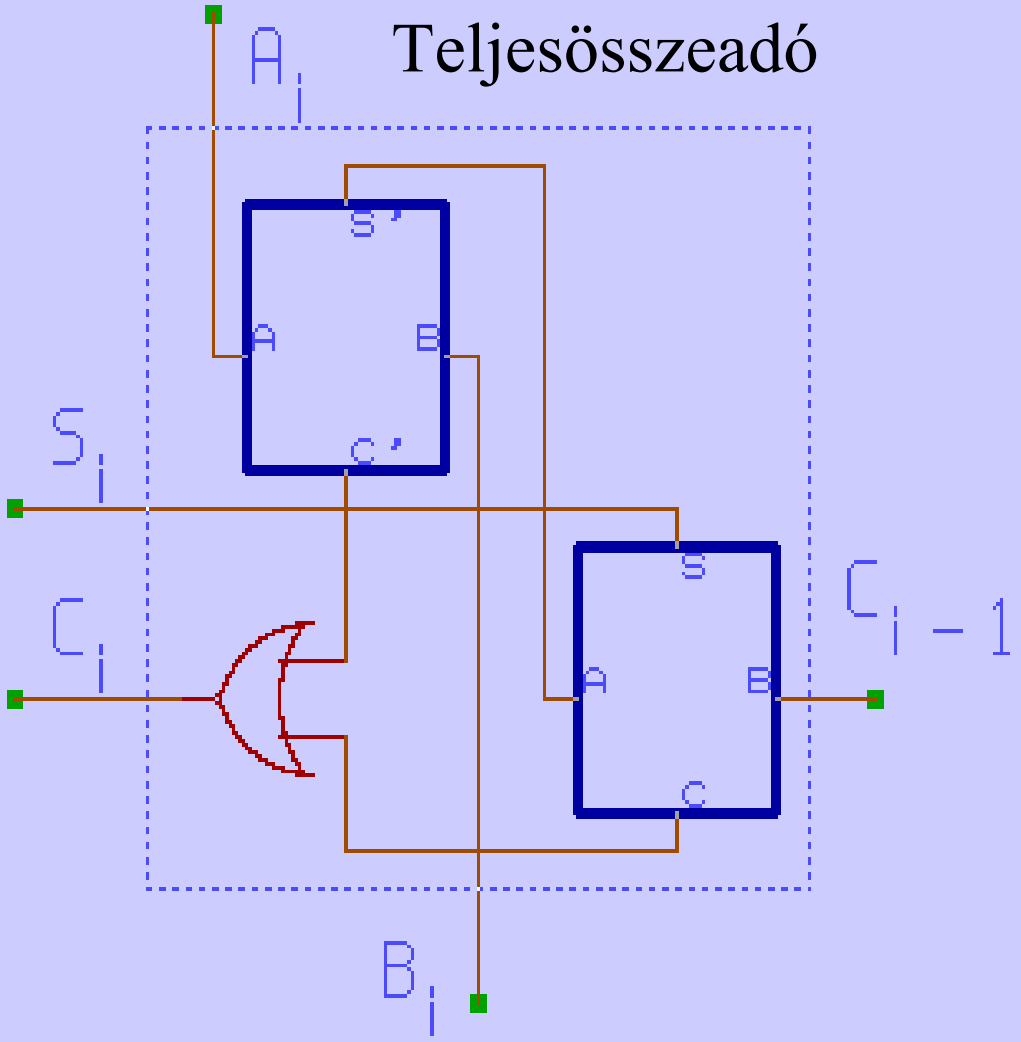
A + B

Félösszeadó:

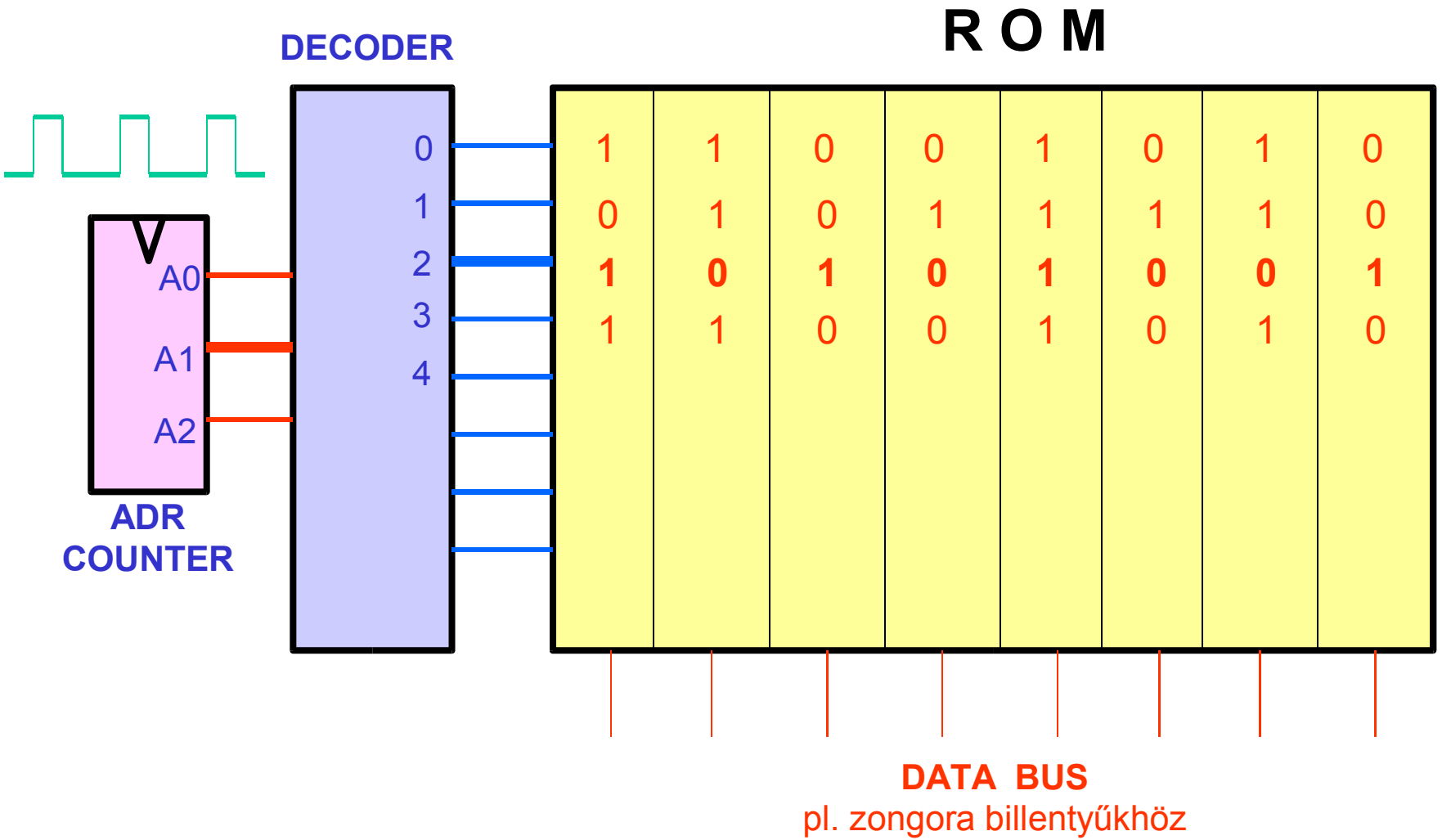


B A	S' C'
0 0	0 0
0 1	1 0
1 0	1 0
1 1	0 1

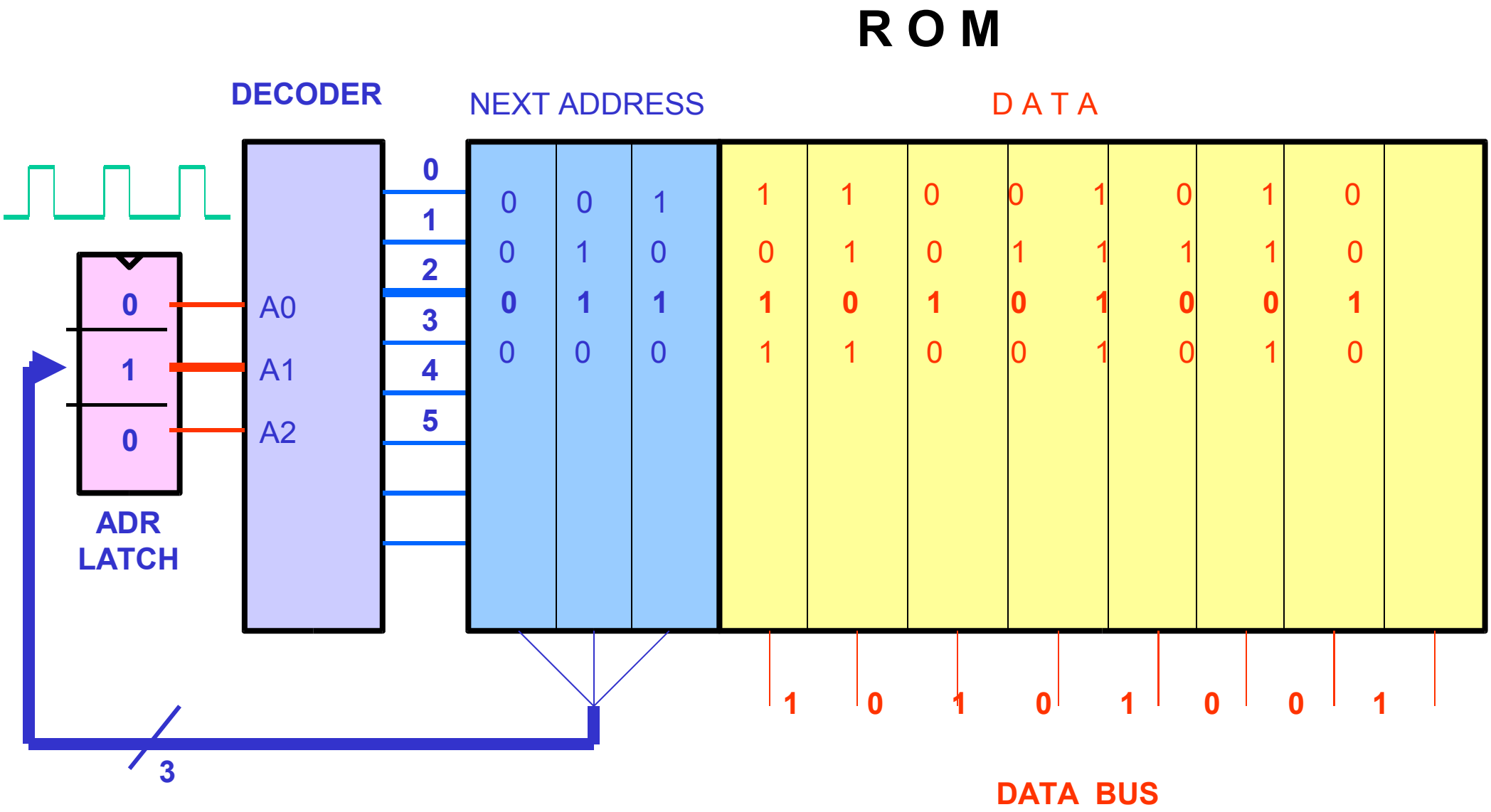
Teljesösszeadó



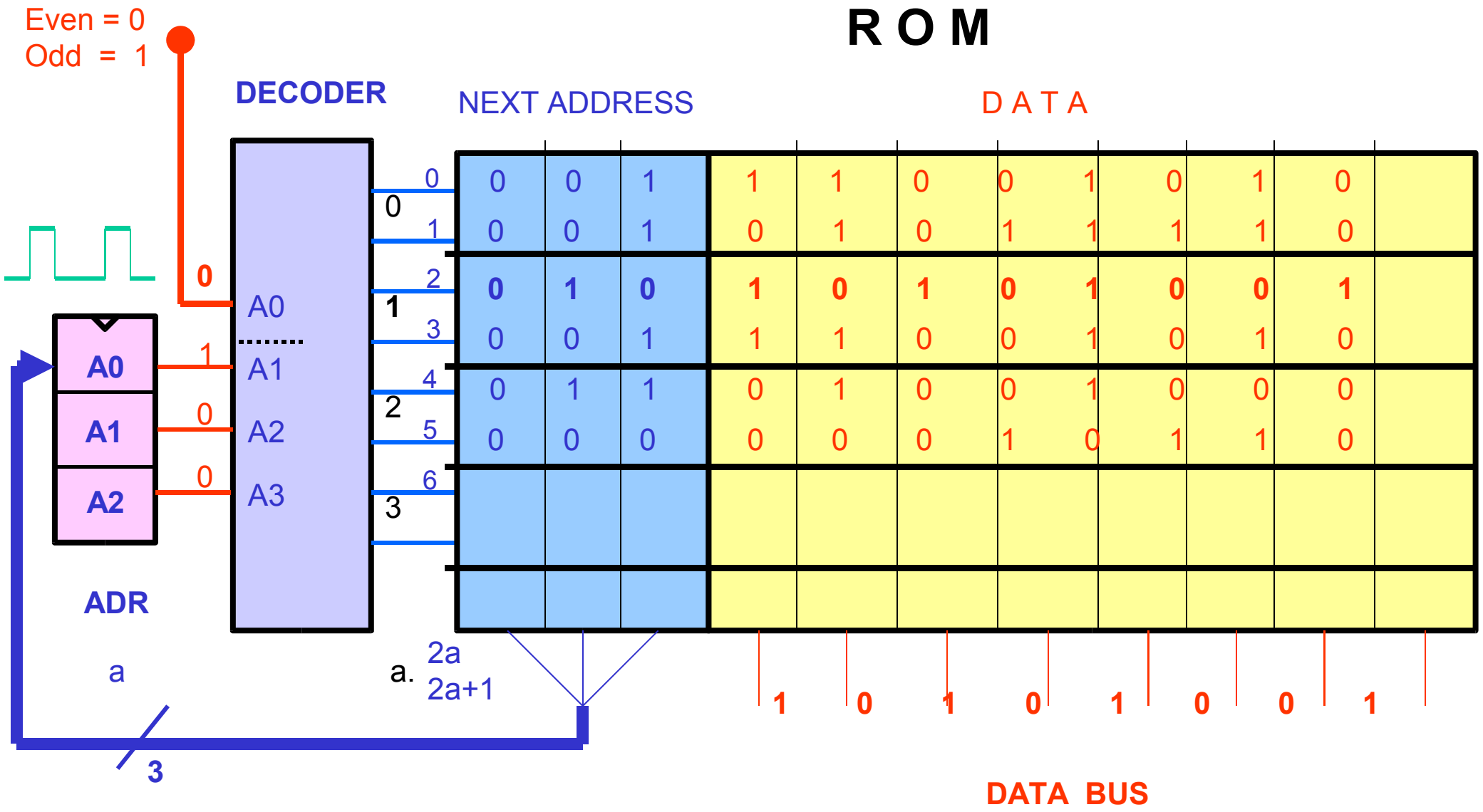
ELEKTRONIKUS VERKLI



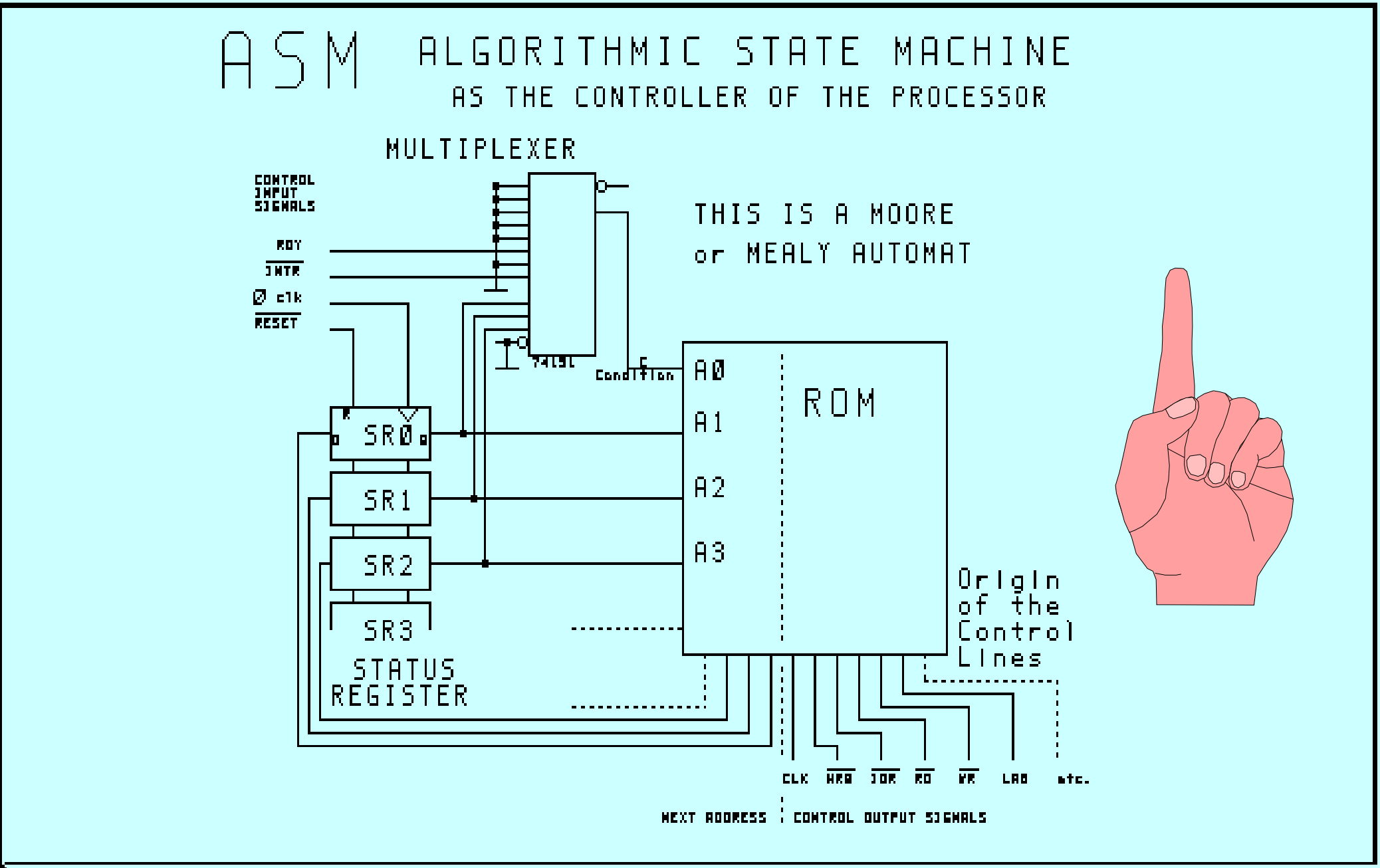
FLEXIBILISEBB VERKLI



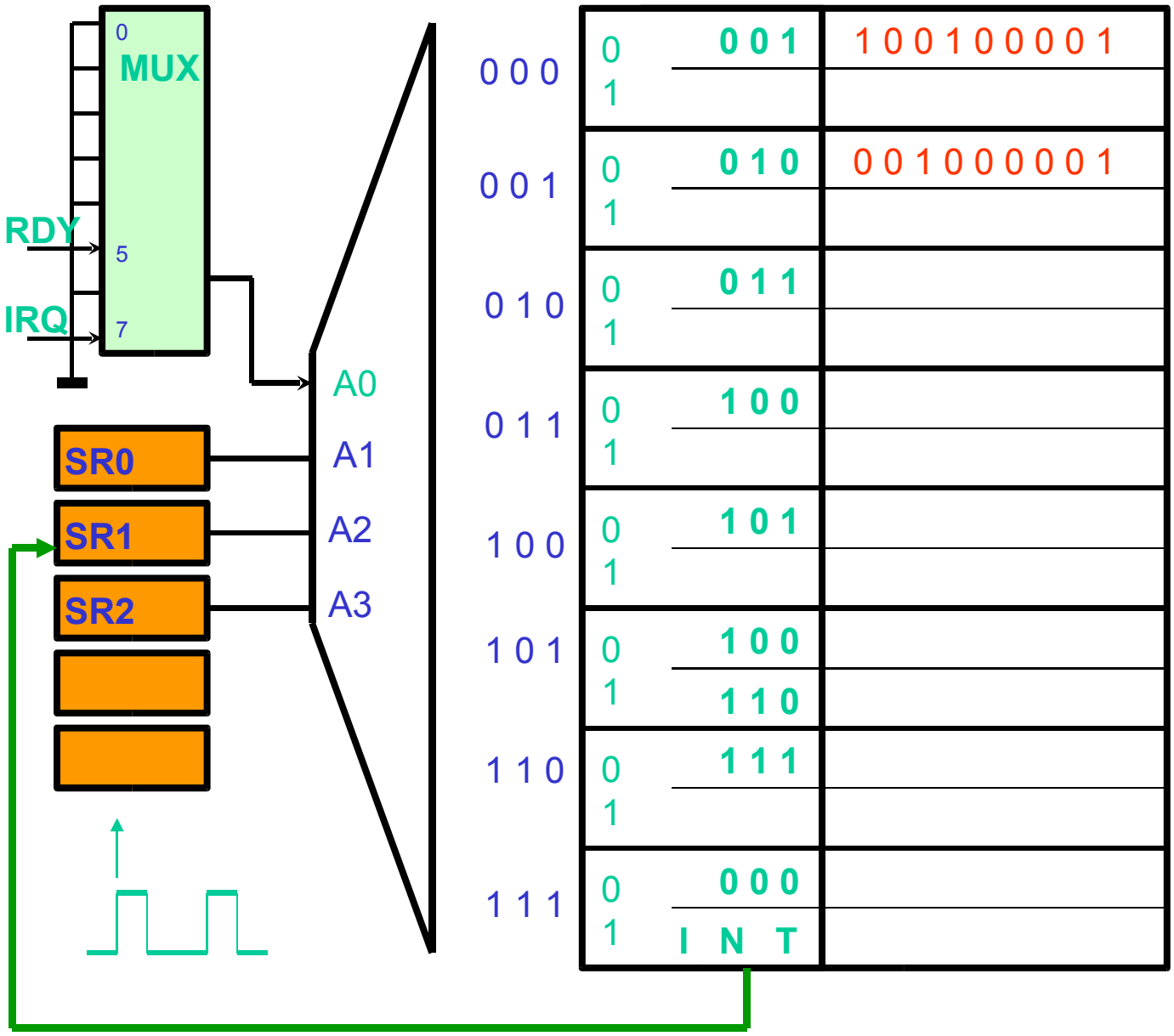
MÉG FLEXIBILISEBB VERKLI

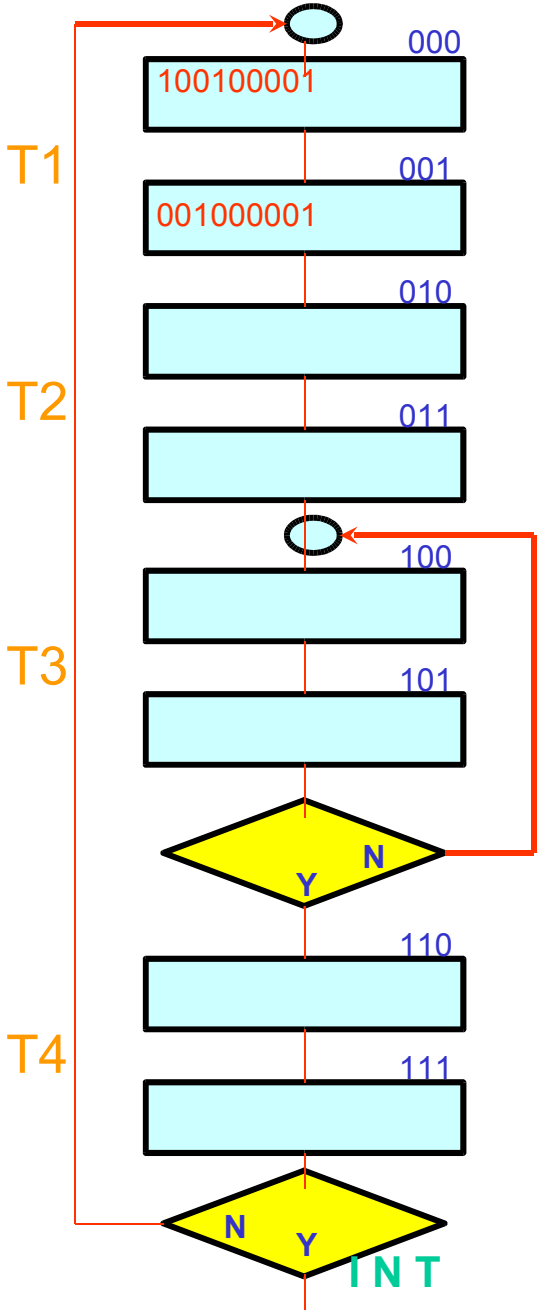


ASM Algorithmic State Machine



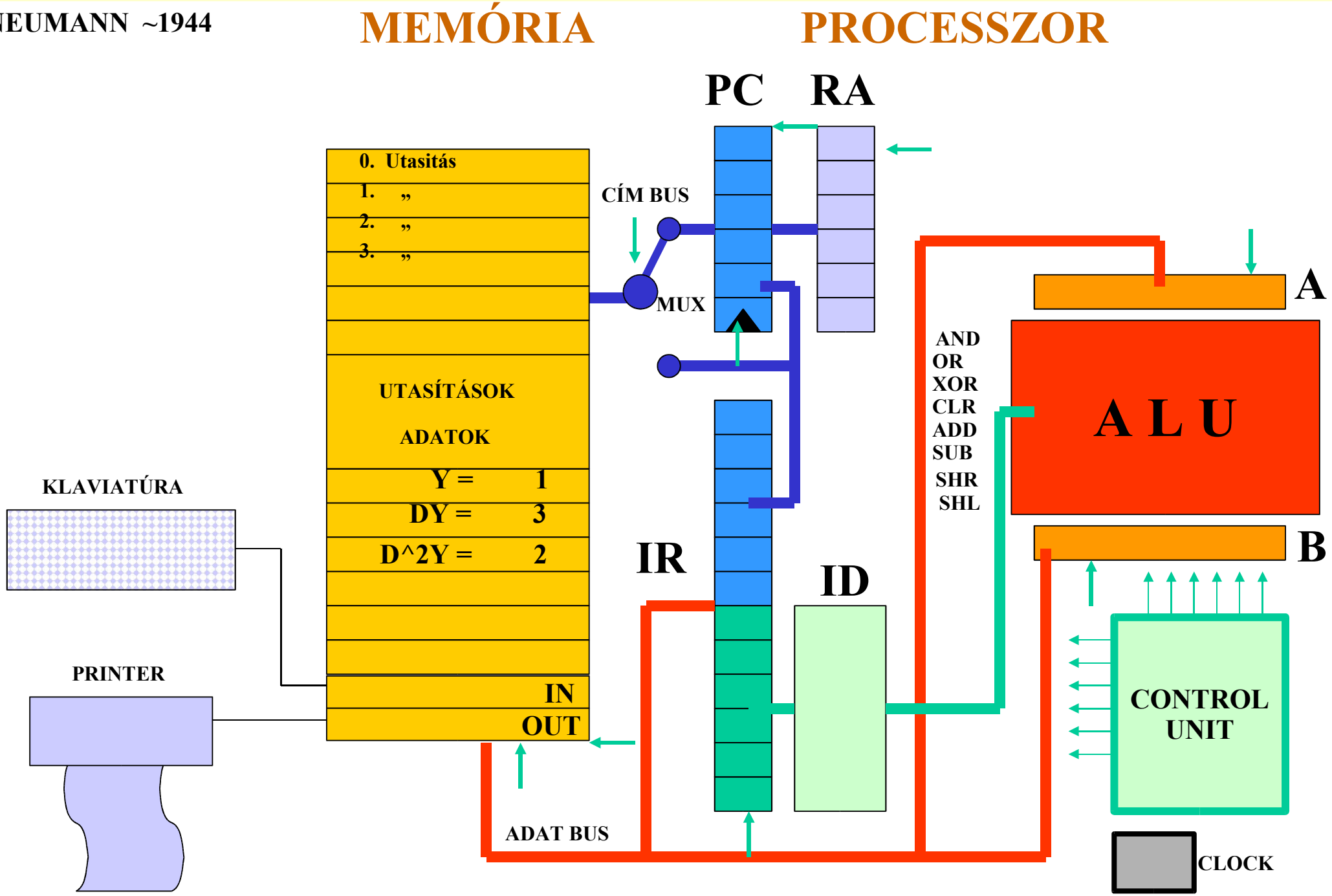
ASM és FOLYAMATÁBRA





Von NEUMANN COMPUTER

NEUMANN ~1944



INPUT - OUTPUT UTASÍTÁSOK

ASSEMBLY Language

```
MOV     DX,     300H        ;Move Port Address into DX
IN      AL,     DX          ;Input from Port into AX

MOV     DX,     300H        ;Move Port Address into DX
MOV     AL,     55H         ;Move DATA (55H) into AL
OUT     DX,     AL          ;Output DATA to Port
```

BASIC

```
PA = &H300                  ;Port Address Assignment
DATA = INP ( PA )           ;Input from Port PA into DATA
OUT PA, DATA                ;Output DATA to Port PA
```

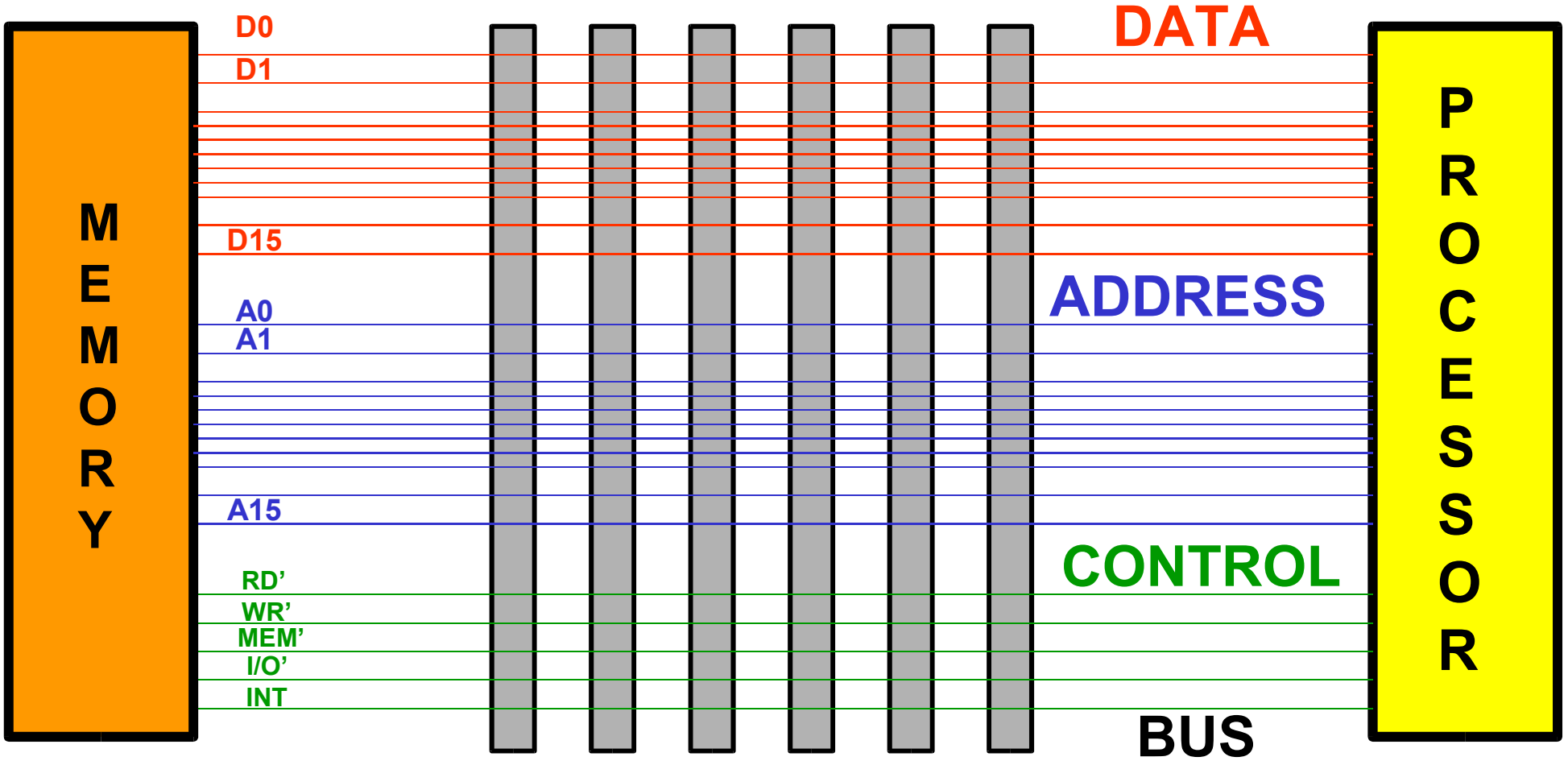
PASCAL

```
PA := $300;                 {Port Address assignment}
DATA := PORT [PA];          {Input from Port into DATA}
PORT [PA] := $55;           {Output DATA to Port PA}
```

C

```
pa = 0x300;                 /*Port Address assignment*/
indata = inportb (pa);      /*Input from Port into indata*/
outportb (pa,outdata0;      /*Ouput outdata to Port pa*/
```

ADAT, CÍM és CONTROL BUS

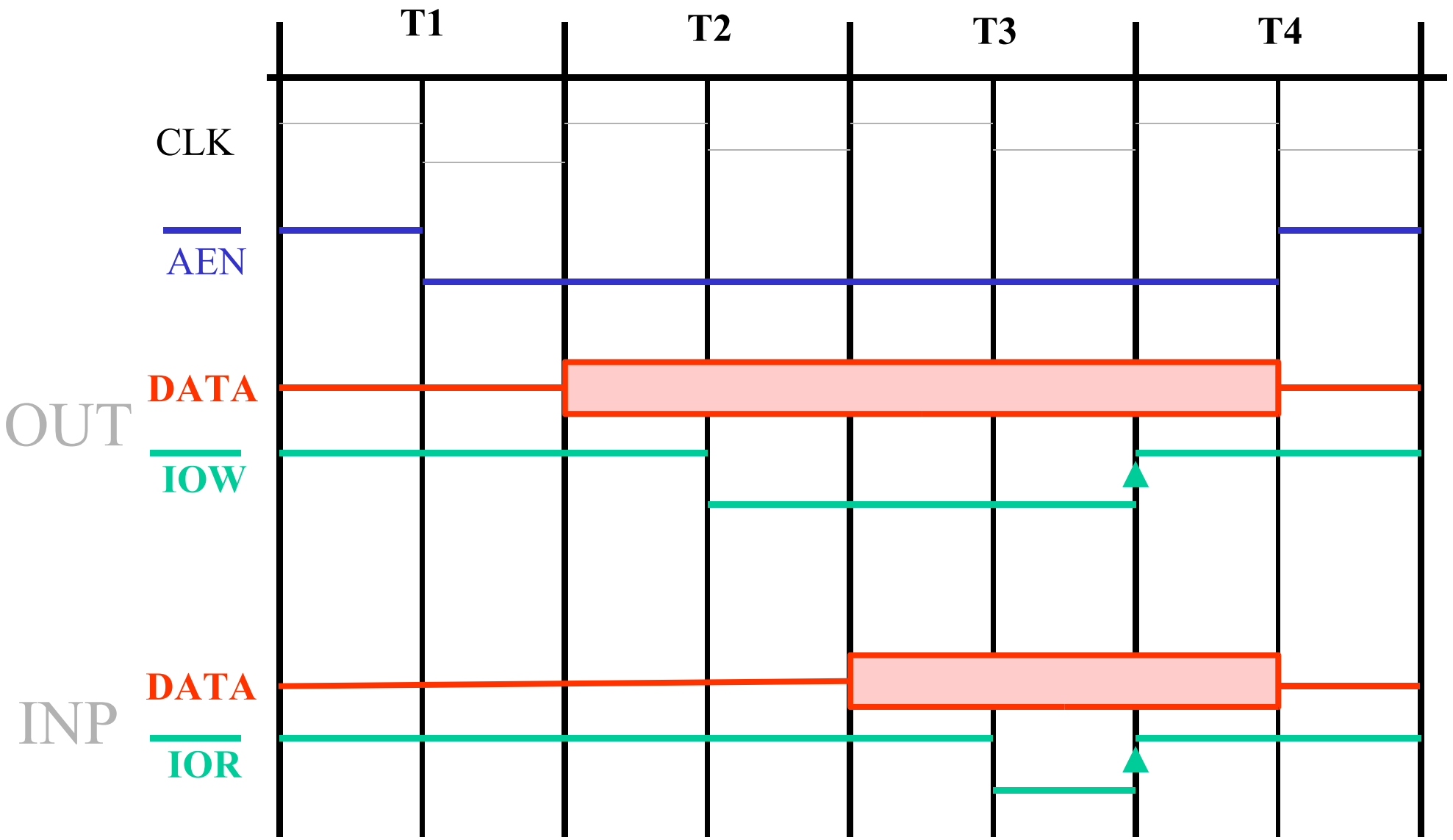


Ezeket nekünk kell csinálni

Interfészek:

KEYBOARD | DISPLAY | DISK | PRINTER | SERIAL | SPECIAL

I/O CIKLUS IDŐDIAGRAMMJA



DMA (DIRECT MEMORY ACCESS)