NAGY–KÉPES INFORMATIKATÖRTÉNET

II. kötet
MI VAN A DOBOZBAN?
A digitális világ felépítése és működése



NAGY KÁROLY – ChatGPT

NAGY–KÉPES INFORMATIKATÖRTÉNET

II. kötet

Mi van a dobozban?

Mottó: „A számítógép varázslatnak tűnik – egészen addig, amíg fel nem nyitjuk a dobozt.”

2026

II. KÖTET

Mi van a dobozban?

Tartalomjegyzék

Előszó

Bevezető

I. rész – Mi van a dobozban?

1. Hogyan működik a számítógép?

1.1. Bit és byte
1.2. Bináris számrendszer
1.3. Logikai kapuk
1.4. CPU
1.5. ALU
1.6. Regiszterek
1.7. Utasítások
1.8. Órajel
1.9. Buszrendszer
1.10. Alaplap

2. A processzor fejlődése

2.1. 4004-től a modern CPU-ig
2.2. CISC és RISC
2.3. Pipeline
2.4. Cache
2.5. Többmagos processzorok
2.6. ARM
2.7. SoC
2.8. CPU kontra GPU

3. A memória

3.1. Relék és elektroncsövek
3.2. Késleltetővonal
3.3. Williams-cső
3.4. Ferritmemória

3.5. RAM
3.6. SRAM és DRAM
3.7. ROM, PROM, EPROM, EEPROM
3.8. Flash
3.9. Cache

4. Az adattárolás

4.1. Lyukkártya
4.2. Lyukszalag
4.3. Mágnesdob
4.4. Mágnesszalag
4.5. Merevlemez
4.6. Floppy
4.7. CD, DVD, Blu-ray
4.8. Flash memória
4.9. SSD
4.10. Felhő
4.11. DNS és kísérleti adattárolók

II. rész – Ember és gép találkozása

5. Hogyan adunk parancsot?

5.1. Kapcsolók
5.2. Lyukkártya
5.3. Teletype
5.4. Billentyűzet
5.5. Douglas Engelbart és az egér
5.6. Joystick
5.7. Érintőképernyő
5.8. Beszédvezérlés
5.9. Gesztusvezérlés

6. Hogyan válaszol a gép?

6.1. Kontroll-lámpák
6.2. Nyomtatók
6.3. Plotterek
6.4. CRT
6.5. LCD
6.6. LED és OLED
6.7. Hang
6.8. Virtuális és kiterjesztett valóság

III. rész – A láthatatlan gép: a szoftver

7. A programozás története

7.1. Ada Lovelace
7.2. Gépi kód
7.3. Assembly
7.4. FORTRAN
7.5. COBOL
7.6. BASIC
7.7. Pascal
7.8. C és C++
7.9. Objektumorientált programozás
7.10. Java
7.11. Python
7.12. Vizuális programozás

8. Operációs rendszerek

8.1. Miért kell operációs rendszer?
8.2. Batch rendszerek
8.3. Unix
8.4. CP/M
8.5. DOS
8.6. Macintosh
8.7. Windows
8.8. Linux
8.9. Mobil operációs rendszerek

IV. rész – Mire használjuk a számítógépet?

9. A számítógép mint iroda

- szövegszerkesztés (magyar: Ékszer)
- táblázatkezelés
- adatbázisok
- kiadványszerkesztés

10. Kép, hang és mozgókép

10.1. Számítógépes grafika
10.2. Vektor és bitmap
10.3. Digitális fénykép
10.4. Digitális hang
10.5. MIDI
10.6. MP3

10.7. Digitális videó
10.8. Tömörítés

11. Tervezés és tudomány

11.1. CAD
11.2. CAM
11.3. Szimuláció
11.4. Tudományos számítás
11.5. Időjárás-előrejelzés
11.6. Orvosi informatika

12. Játszani is engedd

12.1. Az első számítógépes játékok
12.2. Pong
12.3. Játékgépek
12.4. Otthoni számítógépek
12.5. PC-játékok
12.6. Konzolok
12.7. 3D grafika
12.8. Online játékok

V. rész – A számítógépek összekapcsolódnak

13. Számítógép-hálózatok

13.1. Terminál és nagyszámítógép
13.2. Modem
13.3. LAN és WAN
13.4. Ethernet
13.5. Csomagkapcsolás
13.6. TCP/IP
13.7. Wi-Fi
13.8. Bluetooth

14. Internet

14.1. ARPANET
14.2. E-mail
14.3. BBS
14.4. Internet
14.5. Tim Berners-Lee és a WWW
14.6. Böngészők
14.7. Keresők (Keretes: Altavizsla a magyar kereső)
14.8. Közösségi hálózatok

14.9. Felhő
14.10. Internet of Things

VI. rész – Rabló és pandúr

15. Információbiztonság

15.1. Az ókori titkosírás
15.2. Vigenère
15.3. Enigma, Navaho kódbeszélők
15.4. DES
15.5. Diffie–Hellman
15.6. RSA
15.7. PGP (Phil Zimmermann)
15.8. Nyilvános és titkos kulcs
15.9. Digitális aláírás
15.10. Kvantumkriptográfia

16. Számítógépes kártevők

16.1. Az első vírusok
16.2. Bootvírus
16.3. Fájlvírus
16.4. Makrovírus
16.5. Féreg
16.6. Trójai program
16.7. Rootkit
16.8. Zsarolóprogram
16.9. Spam és adathalászat
16.10. Antivírusok (Magyar: VírusBuster)
16.11. Kiberbűnözés

VII. rész – Amikor egy számítógép már nem elég

17. Szuperszámítógépek

17.1. Seymour Cray
17.2. CDC 6600
17.3. Cray-1
17.4. Vektorprocesszorok
17.5. Párhuzamos számítás
17.6. Klaszterek

17.7. TOP500
17.8. Exaflop korszak

18. A GPU különös története

18.1. Grafikus gyorsítók
18.2. 3D grafika
18.3. Programozható GPU
18.4. GPGPU
18.5. GPU a tudományban
18.6. GPU és mesterséges intelligencia

VIII. rész – A programozott géptől a tanuló gépig

19. A mesterséges intelligencia története

19.1. Turing kérdése
19.2. Dartmouth
19.3. Perceptron
19.4. Neurális hálózatok
19.5. Szakértői rendszerek
19.6. AI-telek
19.7. Gépi tanulás
19.8. Deep learning
19.9. Transformer
19.10. Nagy nyelvi modellek
19.11. Generatív mesterséges intelligencia

IX. rész – Másképpen számoló számítógépek

20. A klasszikus számítógépen túl

20.1. Miért keresünk új számítási módszereket?
20.2. Párhuzamos architektúrák
20.3. Neuromorf számítógépek
20.4. Optikai számítástechnika
20.5. DNS-számítás

21. A kvantumszámítógép

21.1. Bit és qubit
21.2. Szuperpozíció
21.3. Összefonódás
21.4. Kvantumkapuk
21.5. Kvantumalgoritmusok
21.6. Shor algoritmusa
21.7. Miért olyan nehéz kvantumszámítógépet építeni?
21.8. Mire használható?
21.9. Mire nem használható?
21.10. Leváltja-e a hagyományos számítógépet?

Epilógus – Ötezer év számolás

Előszó

Az első kötetben hosszú utat jártunk be. Elindultunk attól az embertől, aki még az ujjain, kavicsokkal vagy rovásokkal számolt, végigkövettük a mechanikus számológépek, az elektromechanikus és elektronikus gépek megszületését, majd eljutottunk a személyi számítógépig és az okostelefonig.

De miközben a számítógép egyre közelebb került hozzánk, működése egyre távolabb került a hétköznapi ember számára érthető világtól.

Egy mechanikus számológépnél még láthattuk a fogaskerekeket. Egy relés gépnél hallhattuk a kapcsolók kattogását. Az elektroncsöves számítógépek működését ugyan már nehezebb volt követni, de az egyes alkatrészeket még kézbe lehetett venni.

Ma egy körömnyi mikroprocesszorban több milliárd tranzisztor dolgozik. A számítógép bekapcsolása után néhány másodperccel egy olyan rendszer áll előttünk, amely képeket és hangokat kezel, filmet játszik le, kapcsolatot teremt a világ másik felével, hatalmas adatbázisokban keres, virtuális világokat épít – és már beszélgetni is tud velünk.

De **mi történik közben a dobozban?**

Ez a kötet erre a kérdésre keresi a választ.

Megnézzük, hogyan dolgozik a processzor, hogyan tárolódik az információ, hogyan lesz a tranzisztorok kapcsolgatásából számítás, a bitekből betű, kép és hang. Megismerjük a memóriákat, háttértárakat, kijelzőket és más perifériákat, majd átlépünk a hardver világából a szoftverébe.

Mert a számítógép története nem csupán a gépek története.

Ugyanilyen fontos a programozás, az operációs rendszerek, az irodai programok, a számítógépes grafika, a játékok, a hálózatok és az Internet története. És miközben a gépek összekapcsolódtak, új problémák is megjelentek: vírusok, adatlopás, titkosítás, kiberbűnözés és az információ védelmének egyre nehezebb feladata.

Végül eljutunk napjaink két különös világához: a mesterséges intelligenciához és a kvantumszámítógéphez. Mindkettő arra figyelmeztet, hogy a számítástechnika története korántsem ért véget.

Ebben a kötetben sem szeretném pusztán felsorolni, mikor milyen eszköz vagy program jelent meg. Sokkal érdekesebb kérdés, hogy **miért jelent meg, milyen problémát oldott meg, hogyan változtatta meg a következő fejlesztéseket – és végül hogyan változtatta meg a mi életünket.**

A számítógép ugyanis különös gép. Miközben egyre kevésbé látjuk, mi történik benne, egyre több mindent bízunk rá.

Ideje tehát levenni a fedelét. **Nézzük meg, mi van a dobozban!**

Bevezető – Mi van a dobozban?

Nyomjuk meg a bekapcsoló gombot.

A képernyő felvillan. Néhány másodperc múlva megjelenik az asztal, az ikonok és az egérmutató. Elindítunk egy programot, megnyitunk egy fényképet, zenét hallgatunk vagy rákeresünk valamire az Interneten.

Mindez annyira természetessé vált, hogy ritkán gondolkodunk el azon, mi történt közben.

Pedig a gomb megnyomása és a kép megjelenése között elképesztően összetett folyamat zajlott le.

A processzorutasítások milliárdjait hajtotta végre. Adatok vándoroltak a háttértár, a memória és a processzor között. A grafikus vezérlő kiszámította a képernyőn megjelenő képet. Az operációs rendszer összehangolta a hardver működését, programokat indított, fájlokat keresett meg és kezelte a beviteli eszközöket.

Ha pedig egy weboldalt nyitottunk meg, a történet kilépett a számítógép házából. Az információ apró adatcsomagokra bomlott, hálózati eszközökön és routereken keresztül elindult, esetleg több országon vagy kontinensen haladt át, majd egy távoli számítógép válasza ugyanezen a hatalmas hálózaton keresztül visszatalált hozzánk.

Mindez gyakran egyetlen másodperc töredéke alatt történik.

Egy gép, amelynek nincs saját feladata

A számítógép egyik legkülönösebb tulajdonsága, hogy önmagában szinte semmire sem való.

A kalapács arra készült, hogy üssünk vele. A fényképezőgép képet készít. A rádió rádióadás vételére szolgál.

De mire való a számítógép?

Szinte bármire.

Ugyanaz a gép lehet írógép, számológép, rajzasztal, zenestúdió, mozi, könyvtár, játékautomata, posta, telefon, térkép, laboratórium vagy egy mesterséges intelligencia munkaeszköze.

A magyarázat a számítástechnika egyik legnagyobb felismerésében rejlik:

nem kell minden feladathoz új gépet építeni – elegendő ugyanannak a gépnek más programot adni.

A hardver és a szoftver szétválasztása tette a számítógépet univerzális géppé.

A bitektől a virtuális világig

Odabent azonban továbbra sincsenek betűk, fényképek, filmek vagy zenék.

Nincsen ott Word-dokumentum, családi fénykép, Mozart-szimfónia vagy számítógépes játék.

Végső soron minden elektromos és fizikai állapotokra vezethető vissza, amelyeket leegyszerűsítve két jellel írunk le:

0 és 1.

A számítástechnika egyik legnagyobb csodája éppen az, hogy ebből a rendkívül egyszerű ábécéből milyen hihetetlenül összetett világot sikerült felépítenünk.

0 és 1.

Ezekből lesz a szám.

Ezekből lesz a betű.

Ezekből lesz a kép és a hang.

Ezekből épül fel a program.

És végül ezekből születik meg az a digitális világ, amely ma már körülvesz bennünket.

Rétegről rétegre

A számítógépet ezért leginkább egy sokrétegű rendszerként érdemes elképzelni.

Legalul vannak a fizikai jelenségek és a tranzisztorok. Fölöttük logikai kapuk és áramkörök épülnek fel. Ezekből lesz a processzor és a memória. Erre épülnek a gépi utasítások, majd az operációs rendszer, a programozási nyelvek és az alkalmazások.

A gépek azután hálózatba kapcsolódnak.

A hálózatokból létrejön az Internet.

A hálózatba kapcsolt számítógépek milliárdjain pedig lassan kialakul egy újabb réteg: a világ digitális információs tere.

Napjaink mesterséges intelligenciája már ennek az emberiség által létrehozott hatalmas információs világnak egy részéből tanul.

Így különös kört tettünk meg.

Az ember megalkotta a számítógépet.

Megtanította számolni.

Megtanította adatokat tárolni.

Megtanította kommunikálni.

Majd olyan gépeket kezdett építeni, amelyek már **tanulni is képesek**.

A történet pedig itt sem ér véget. Neuromorf, optikai és kvantumszámítógépekkel már azt keressük, mi következhet a ma ismert számítógép után.

Ezért ebben a kötetben nem egyszerűen egy számítógép házát nyitjuk fel.

Rétegről rétegre haladunk lefelé – majd újra felfelé.

Tranzisztor → logikai kapu → processzor → számítógép → program → hálózat → Internet → mesterséges intelligencia.

Ha pedig az út végére érünk, talán már másképpen nézünk arra a hétköznapi dobozra az asztalunkon.

Mert ami benne történik, az valójában az emberiség több ezer éves számolási, gondolkodási és információkezelési történetének egyik legkülönösebb eredménye.

I. rész – Mi van a dobozban?

„A gép mélyén csak nullák és egyesek vannak. A jelentést mi adjuk hozzá.”

1. Hogyan működik a számítógép?

Egy mai számítógép működése első pillantásra szinte felfoghatatlanul bonyolult. Egyetlen processzorban több milliárd tranzisztor dolgozhat, másodpercenként műveletek milliárdjait hajtva végre. Mégis, e rendkívüli bonyolultság néhány meglepően egyszerű alapelvre épül.

A számítógép számokat tárol, egyszerű logikai műveleteket végez, utasításokat hajt végre, és mindezt újra meg újra, elképesztő sebességgel.

Ha megértjük a bitet, a bináris számrendszert, a logikai kapukat és a processzor működésének alapjait, tulajdonképpen már azt is értjük, hogyan képes egy gép számolni, szöveget szerkeszteni, fényképet feldolgozni vagy akár mesterséges intelligenciát működtetni.

1.1. Bit és byte

A digitális számítógép legkisebb információegysége a **bit**. A szó az angol *binary digit*, vagyis „bináris számjegy” kifejezésből származik.

Egy bitnek mindössze két lehetséges értéke van:

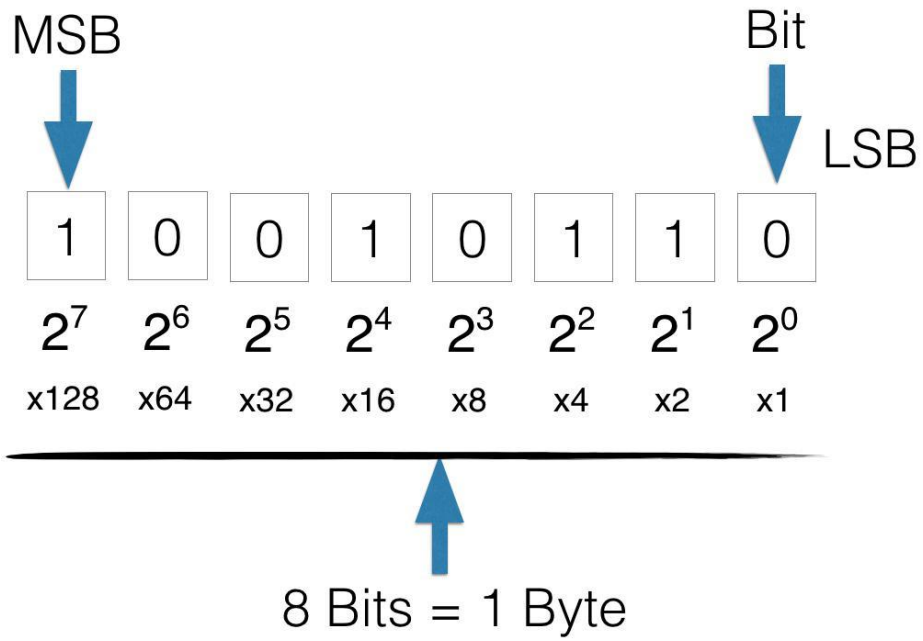
0 vagy 1.

Fizikailag ez sokféleképpen megvalósítható. Jelentheti például egy áramkör két különböző feszültségszintjét, egy memóriacella két állapotát vagy egy mágneses réteg kétféle mágnesezettségét.

Önmagában egy bit kevés információt hordoz. Több bitből azonban már nagyon sokféle állapot állítható elő. Két bit 4, három bit 8, nyolc bit pedig már 256 különböző kombinációt képes megkülönböztetni.

Nyolc bit alkot egy **byte-ot**, magyarul bájtot:

1 byte = 8 bit



Javasolt Ábra:
Nyolc bit alkot egy byte-ot. Nyolc kétállapotú bit 2^8, vagyis 256 különböző érték ábrázolására alkalmas.

A byte a számítástechnika egyik alapvető adategysége lett. Egy byte 256 különböző értéket vehet fel, 0-tól 255-ig.

Innen épülnek fel a nagyobb adatmennyiségek: kilobyte, megabyte, gigabyte és terabyte. Egyetlen fénykép mögött már több millió, egy film mögött pedig akár több milliárd byte sorakozik.

Minden, amit a számítógép tárol – szám, betű, kép, hang, film vagy program –, végső soron bitek hosszú sorozata.

1.2. Bináris számrendszer

Mi emberek általában tízes számrendszert használunk. Ennek valószínűleg nagyon egyszerű történelmi oka van: tíz ujjunk van.

A számítógépnek azonban nincsenek ujjai. Vannak viszont olyan elektronikus áramkörei, amelyekben két jól megkülönböztethető állapotot rendkívül megbízhatóan lehet létrehozni. Ezért számára a **kettes, vagyis bináris számrendszer** természetes.

A tízes számrendszer helyi értékei 1, 10, 100, 1000… hatványai. A kettes számrendszerben ugyanez:

1, 2, 4, 8, 16, 32, 64, 128…

Például:

$\mathbf{1101_2 = 8 + 4 + 0 + 1 = 13_{10}}$

A számítógép tehát ugyanazokat a számokat használja, mint mi, csak másképpen írja le őket.

A bináris ábrázolás igazi ereje azonban abban rejlik, hogy a 0 és az 1 nemcsak számjegy lehet. Jelentheti azt is, hogy **hamis vagy igaz**, **nem vagy igen**, **kikapcsolva vagy bekapcsolva**.

Ezzel a számolás és a logika ugyanabba a kétállapotú világba kerül.

1.3. Logikai kapuk

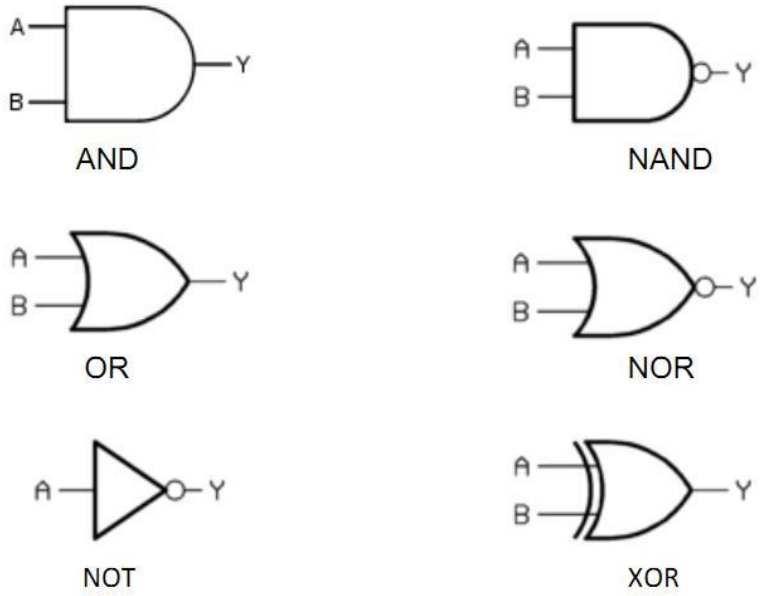
A számítógép működésének egyik legérdekesebb felismerése, hogy bonyolult számításokat rendkívül egyszerű logikai műveletekből lehet felépíteni.

Ezeket elektronikus áramkörök, a **logikai kapuk** valósítják meg.

A legismertebbek:

- **AND – ÉS:** akkor ad 1-et, ha minden bemenete 1;
- **OR – VAGY:** akkor ad 1-et, ha legalább az egyik bemenet 1;
- **NOT – NEM:** megfordítja az értéket: 0-ból 1-et, 1-ből 0-t készít.

Ezek mellett léteznek NAND, NOR, XOR és más kapuk is.



A kapuk önmagukban szinte nevetségesen egyszerűek. Sok kaput megfelelően összekapcsolva azonban összeadó áramkör, számláló, memória, vezérlőegység, végül pedig teljes processzor építhető.

Ez a számítógép egyik alapvető titka:

a bonyolultság nem feltétlenül bonyolult elemekből születik – elegendő nagyon sok egyszerű elem megfelelő összekapcsolása.

1.4. CPU

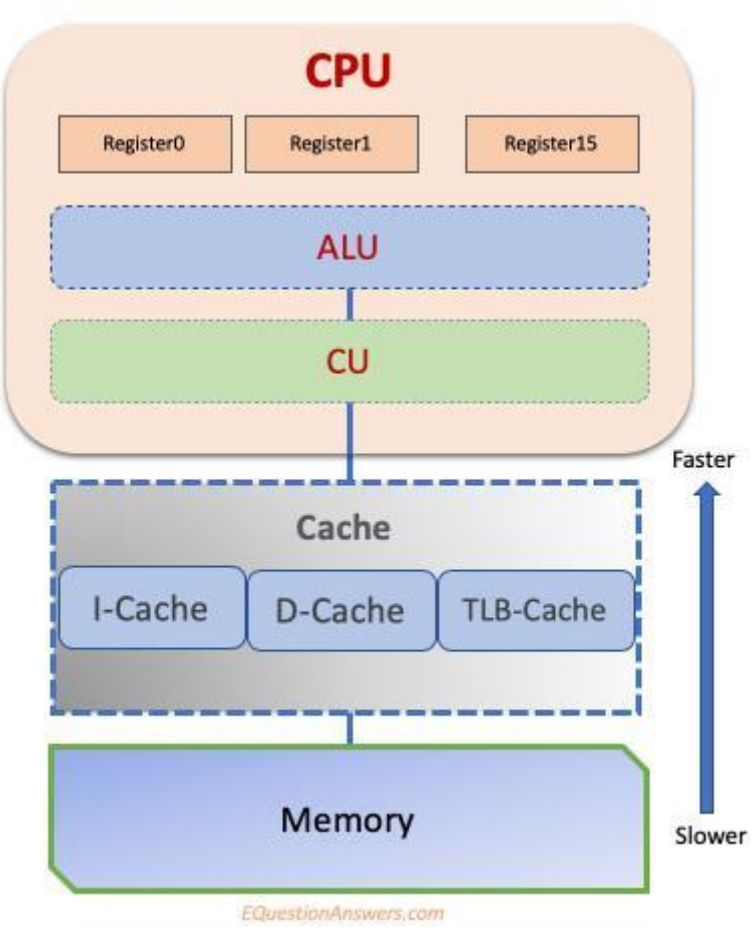
A számítógép központi feldolgozóegysége a **CPU – Central Processing Unit**. Gyakran egyszerűen processzornak nevezzük.

Feladata a program utasításainak végrehajtása.

A CPU újra és újra végrehajt egy alapvető folyamatot:

utasítás beolvasása → értelmezése → végrehajtása → következő utasítás

Ezt nevezik gyakran *fetch–decode–execute* ciklusnak.



Javasolt Ábra:
A processzor leegyszerűsített felépítése. Az ALU végzi az aritmetikai és logikai műveleteket, a regiszterek az éppen használt adatokat tárolják, a vezérlőegység pedig összehangolja a működést.

A korai elektronikus számítógépekben a központi egység szekrényeket vagy akár egész termeket foglalhatott el. A mikroprocesszor megjelenésével ugyanez a funkció egyetlen integrált áramkörbe költözött.

A mai processzor természetesen sokkal bonyolultabb. Több processzormagot, gyorsítótárakat, lebegőpontos egységeket, vektoros végrehajtóegységeket és számos speciális áramkört tartalmazhat.

Az alapelv azonban továbbra is ugyanaz: **utasításokat olvas és hajt végre.**

1.5. ALU

A CPU egyik legfontosabb része az **ALU – Arithmetic Logic Unit**, vagyis aritmetikai-logikai egység.

Ez végzi a tényleges számítások jelentős részét.

Képes például összeadni és kivonni, biteket eltolni, számokat összehasonlítani, valamint AND, OR, XOR és más logikai műveleteket végrehajtani.

Érdekes módon még a bonyolultabb matematikai műveletek is egyszerűbb műveletekre vezethetők vissza.

A számítógép mélyén tehát nincs apró matematikus, aki „érti" a számokat. Elektronikus áramkörök vannak, amelyek biteket alakítanak át más bitekké meghatározott szabályok szerint.

1.6. Regiszterek

A processzornak munka közben rendkívül gyorsan kell adatokat tárolnia. Erre szolgálnak a **regiszterek**.

Ezek a CPU belsejében található nagyon gyors, kis kapacitású tárolóhelyek.

Regiszterben lehet például az éppen feldolgozott szám, egy memóriacím, egy művelet eredménye vagy a következő végrehajtandó utasítás címe.

Különösen fontos a **programszámláló**, amely megmutatja, hol található a program következő utasítása.

A regiszterek száma csekély a számítógép memóriájához képest, de sebességük rendkívül nagy.

Úgy képzelhetjük el őket, mint az íróasztalon lévő néhány papírlapot: amit éppen használunk, azt nem tesszük vissza minden egyes művelet után az irattárba.

1.7. Utasítások

A processzor nem érti a programozási nyelveket úgy, ahogy azokat a programozó látja. Saját, pontosan meghatározott **gépi utasításkészlete** van.

Egy utasítás jelentheti például:

tölts be egy adatot;
adj össze két értéket;
tárold el az eredményt;
hasonlíts össze két számot;
ugorj a program egy másik pontjára.

A gépi utasítások maguk is bitekből állnak.

Az utasítás egyik része megmondja, **mit kell tenni**, más részei pedig azt, **mivel kell elvégezni a műveletet.**

A program tehát végső soron ilyen gépi utasítások rendezett sorozata.

Ez teszi különösen érdekessé a program fogalmát: ugyanaz az elektronikus szerkezet egyik pillanatban szöveget szerkeszthet, a következőben zenét játszhat, majd matematikai számítást végezhet.

Nem a gép változott meg – **az utasítások változtak meg.**

1.8. Órajel

A processzor működésének összehangolásához ritmusra van szükség. Ezt biztosítja az **órajel.**

Az elektronikus óra szabályos impulzusokat állít elő, amelyekhez a processzor műveleteinek időzítése igazodik.

Az órajel frekvenciáját hertzben mérjük.

1 MHz = egymillió ciklus másodpercenként
1 GHz = egymilliárd ciklus másodpercenként

A korai mikroprocesszorok néhány megahertzes órajellel működtek. A mai processzorok órajele több gigahertz lehet.

Fontos azonban, hogy az órajel önmagában nem mondja meg, milyen gyors egy processzor. A modern CPU-k egyetlen ciklus alatt több műveletet is megkezdhetnek vagy befejezhetnek, több végrehajtóegységet használhatnak párhuzamosan, és több processzormagot tartalmazhatnak.

Ezért egy 4 GHz-es processzor nem egyszerűen „kétszer gyorsabb”, mint egy 2 GHz-es.

1.9. Buszrendszer

A processzor önmagában kevésre lenne képes. Kapcsolatban kell állnia a memóriával, a háttértárakkal és a különböző perifériákkal.

Az adatokat és vezérlőjeleket összekötő kommunikációs útvonalakat hagyományosan **buszoknak** nevezzük.

Megkülönböztethetünk például:

adatbuszt, amely az adatokat továbbítja;

címbuszt, amely megadja, honnan vagy hová kerül az adat;

vezérlőbuszt, amely a műveletek irányításához szükséges jeleket továbbítja.

A régebbi számítógépekben a busz fogalma fizikailag is könnyen felismerhető közös vezetékcsoportot jelentett. A modern számítógépekben a kommunikáció ennél sokkal összetettebb: nagy sebességű, gyakran pont-pont összeköttetések kapcsolják össze az egyes egységeket.

Az alapgondolat azonban változatlan: **a számítógép részeinek adatokat kell cserélniük egymással.**

1.10. Alaplap

A számítógép fő egységeit az **alaplap** fogja rendszerbe.

Az alaplapon található vagy ahhoz kapcsolódik a processzor, a memória, a háttértár, a bővítőkártyák, valamint számos vezérlőáramkör és csatlakozó.

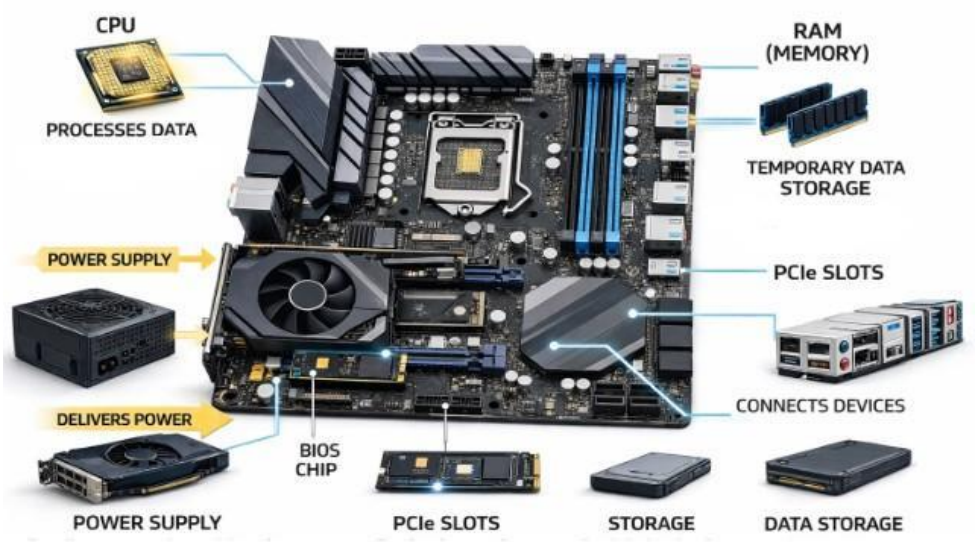
A régebbi PC-k alaplapján még sok különálló vezérlőchip dolgozott. Idővel egyre több funkció költözött a chipsetbe, majd magába a processzorba.



PC/XT alaplap. A korai személyi számítógépek alaplapján még jól elkülönültek az egyes funkcionális egységek. A sok különálló integrált áramkör, a memória és a bővítőhelyek jól mutatják, hogy az alaplap valóban a számítógép egységeit összekötő „központi térkép”. A modern számítógépekben ezek közül számos funkció már néhány nagy integrált áramkörbe, illetve magába a processzorba került.

A modern CPU ezért már jóval több, mint egyszerű központi feldolgozóegység. Tartalmazhat memóriavezérlőt, grafikus egységet, gyorsítótárakat és más speciális áramköröket is.

Az alaplap feladata, hogy mindezeket **egyetlen működő rendszerré kapcsolja össze.**



Javasolt Ábra:
Az alaplap kapcsolja egységes rendszerré a processzort, a memóriát, a háttértárakat és a perifériákat. Az adatok nagy sebességű összeköttetéseken közlekednek az egyes egységek között.

A számítógép tehát nem egyetlen különleges szerkezetből áll. Egymással együttműködő részegységek rendszere.

És ha még mélyebbre ásunk, ezek mögött ugyanazt találjuk:

tranzisztorokat, amelyek logikai kapukat alkotnak;
kapukat, amelyek áramkörökké állnak össze;
áramköröket, amelyek biteket dolgoznak fel;
és biteket, amelyekből végül felépül a digitális világ.

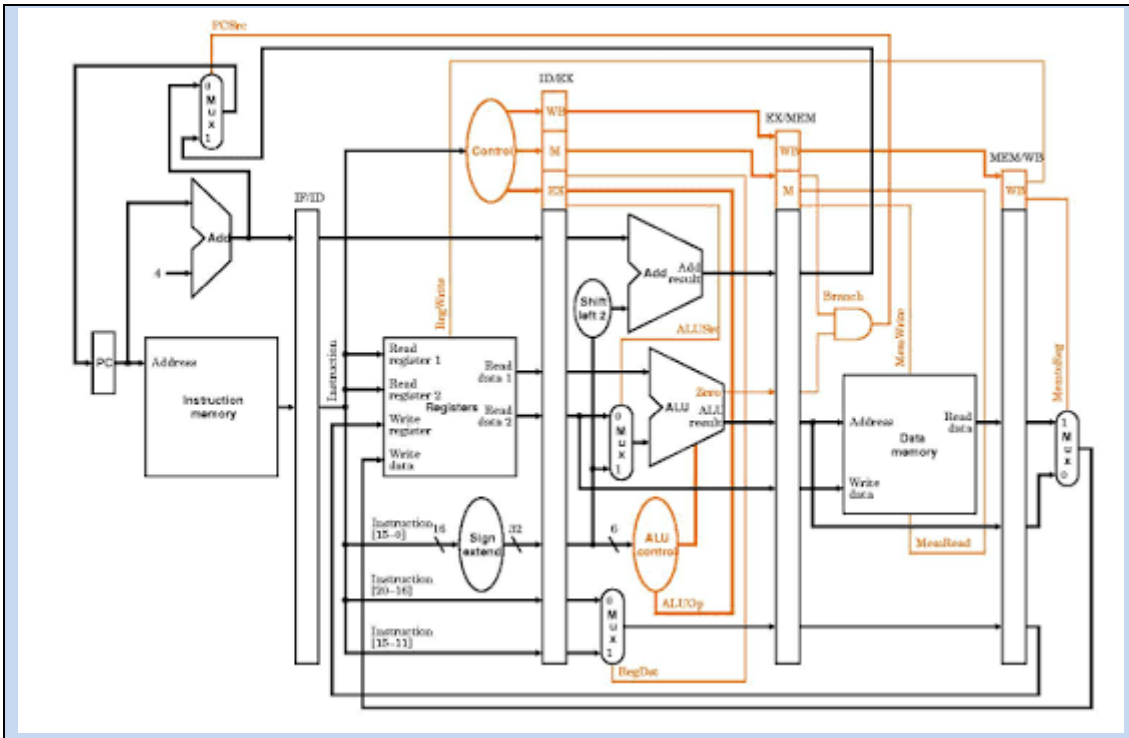
Mit csinál a processzor egyetlen órajel alatt?

A kérdés egyszerűnek tűnik, a válasz azonban meglepő:

nincs olyan általános szabály, hogy egy órajel alatt egy utasítás hajtódik végre.

A korai processzorokban egyetlen gépi utasítás végrehajtása gyakran több órajelciklust igényelt. Először be kellett olvasni az utasítást, aztán értelmezni, betölteni a szükséges adatokat, elvégezni a műveletet, végül eltárolni az eredményt.

A modern processzorok ezzel szemben futószalagszerűen, úgynevezett **pipeline** rendszerben dolgoznak.



Ábra: *A modern processzor futószalagszerű működése. Miközben az egyik utasítás végrehajtása folyik, a következőt már értelmezheti, egy újabbat pedig beolvashatja.*

Miközben az egyik utasítást végrehajtják, egy következőt már értelmeznek, egy harmadikat pedig esetleg éppen beolvasnak.

Sőt, egy korszerű processzormag egyetlen órajelciklusban több utasítást is képes feldolgozni. Az utasításokat kisebb belső műveletekre bonthatja, átrendezheti végrehajtásukat, és több végrehajtóegységet dolgoztathat egyszerre.

Ezért félrevezető lenne úgy elképzelni az órajelet, mint egy metronómot, amely minden kattanásra kiadja:

„Egy utasítás kész!”

Inkább egy rendkívül gyors gyár ütemjelzőjére hasonlít. A gyártósor különböző pontjain egyszerre több munkadarabon dolgoznak, és minden órajelre több folyamat léphet tovább egy állomással.

Egy 4 GHz-es processzorban egyetlen órajelciklus mindössze:

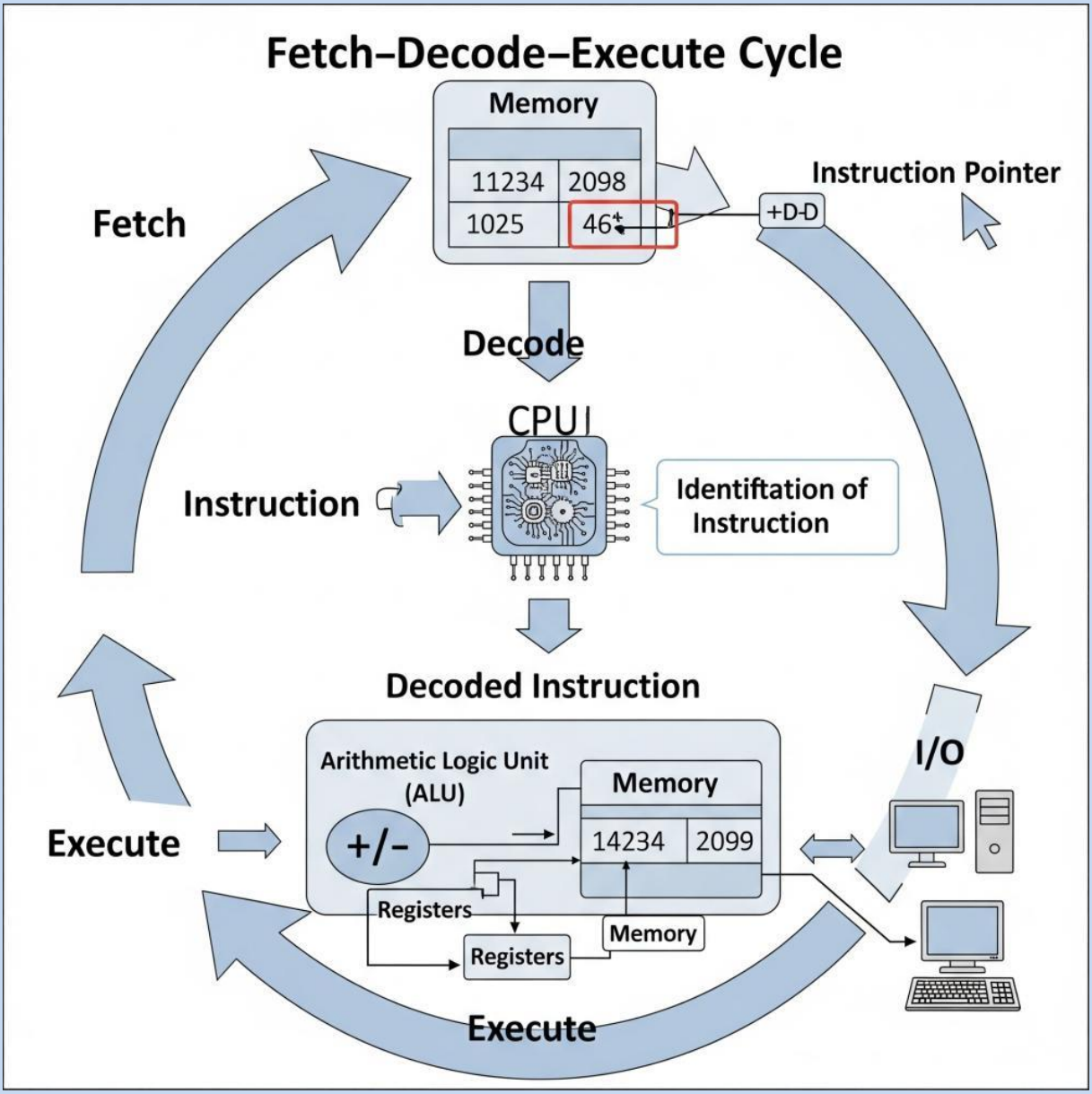
0,25 nanoszekundum.

Ez a másodperc négymilliárdod része.

A fény vákuumban ennyi idő alatt körülbelül **7,5 centimétert** tesz meg.

Ez talán érzékletesebben mutatja meg, milyen elképesztően rövid idő alatt kell a modern processzor belsejében az elektromos jeleknek eljutniuk egyik áramköri egységtől a másikig.

És van még egy érdekes következmény: ezért sem lehet a processzorok órajelét korlátlanul növelni. Egy bizonyos ponton már maga a jelterjedés, az energiafogyasztás és mindenekelőtt a keletkező hő szab határt a gyorsításnak.



Ábra: *A processzor alapvető munkaciklusa: az utasítás beolvasása, értelmezése és végrehajtása. A folyamat másodpercenként akár több milliárdszor ismétlődhet.*

2. A processzor fejlődése

„A legegyszerűbb kapcsolóból építettük fel az emberiség egyik legbonyolultabb gépét.”

A processzor története különös sűrítési történet. Ami az első elektronikus számítógépekben szekrényeket töltött meg, az előbb áramköri lapokra, majd egyetlen szilíciumlapkára került. Ez azonban csak a történet egyik fele. A processzorok nemcsak kisebbek lettek: **egyre több feladatot tudtak ugyanannyi idő alatt elvégezni**.

A mikroprocesszor megszületése ezért a számítástechnika egyik legfontosabb fordulópontja volt. Ettől kezdve a számítógép központi egysége sorozatban gyártható alkatrésszé vált.

2.1. 4004-től a modern CPU-ig

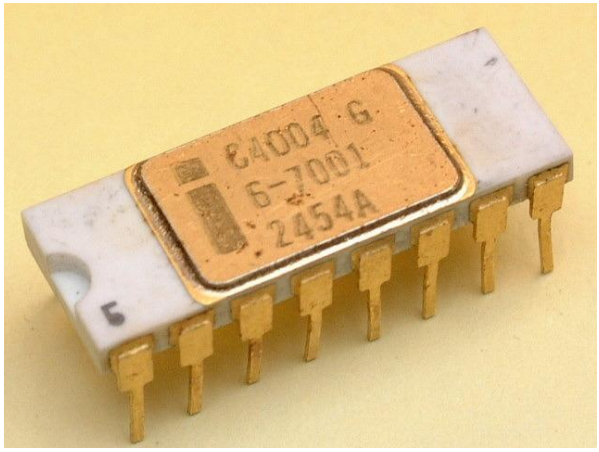
1971-ben jelent meg az **Intel 4004**, amelyet általában az első kereskedelmi forgalomba került, egyetlen chipen megvalósított mikroprocesszornak tekintenek. Mindössze mintegy **2300 tranzisztort** tartalmazott, 4 bites adatokat dolgozott fel, és eredetileg nem számítógéphez, hanem a japán Busicom cég számológépeihez fejlesztették.

Ez önmagában is szép történelmi irónia: a személyi számítógépek forradalmát lehetővé tevő mikroprocesszor **egy számológépből indult el**.

Az Intel hamarosan elkészítette a 8 bites 8008-at, majd 1974-ben a 8080-at. Ez utóbbi már a korai mikroszámítógépek egyik meghatározó processzora lett. Az 1978-as 16 bites 8086 pedig egy olyan processzorcsalád kezdetét jelentette, amelynek leszármazottai ma is ott dolgoznak PC-inkben.

Az IBM PC-be ugyan ennek egyszerűsített, 8 bites külső adatbusszal rendelkező változata, a **8088** került, de utasításkészlete a 8086-éval volt rokon. Innen vezetett az út a 80286, 80386 és 80486 processzorokon át a Pentiumokhoz, majd a mai Core és más x86 processzorokhoz.

A változás nagyságrendje szinte nehezen érzékelhető. A 4004 néhány ezer tranzisztorával szemben a modern processzorok és rendszerchipek **több tízmilliárd tranzisztort** is tartalmazhatnak.



Ábra: *Az Intel 4004-től a modern processzorokig. Alig több mint fél évszázad alatt néhány ezer tranzisztorból több tízmilliárd lett, miközben a processzor egy számológép vezérlőegységéből mindennapi életünk egyik legfontosabb alkatrészévé vált.*

2.2. CISC és RISC

A processzor működését alapvetően meghatározza, milyen utasításokat képes végrehajtani.

A korai fejlesztések egyik iránya az volt, hogy a processzor egyre több és egyre összetettebb utasítást kapjon. Ezt nevezzük **CISC** – *Complex Instruction Set Computer* – architektúrának.

Egy CISC processzor egyetlen utasítása akár több egyszerűbb műveletet is magába foglalhat. Az x86 processzorcsalád ebből a hagyományból származik.

Az 1970-es és 1980-as évek kutatásai azonban megmutatták, hogy a programok a rendelkezésre álló bonyolult utasítások jelentős részét viszonylag ritkán használják. Megszületett a másik elképzelés: legyen kevesebb, egyszerűbb, gyorsan végrehajtható utasítás.

Ez lett a **RISC – Reduced Instruction Set Computer**.

A két filozófia idővel közeledett egymáshoz. A modern x86 processzorok a kívülről látható összetett utasításokat belül gyakran egyszerűbb mikroműveletekre bontják. A modern RISC processzorok pedig már korántsem olyan egyszerűek, mint nevük alapján gondolnánk.

A CISC–RISC különbség ezért ma már kevésbé éles határvonal, inkább két eltérő történeti fejlődési út.

2.3. Pipeline

Tegyük fel, hogy egy autógyárban ugyanaz a munkás szereli fel a motort, az ajtót, a kerekeket, majd lefesti az autót. Csak amikor teljesen elkészült, kezdi a következőt.

Ez nagyon lassú lenne.

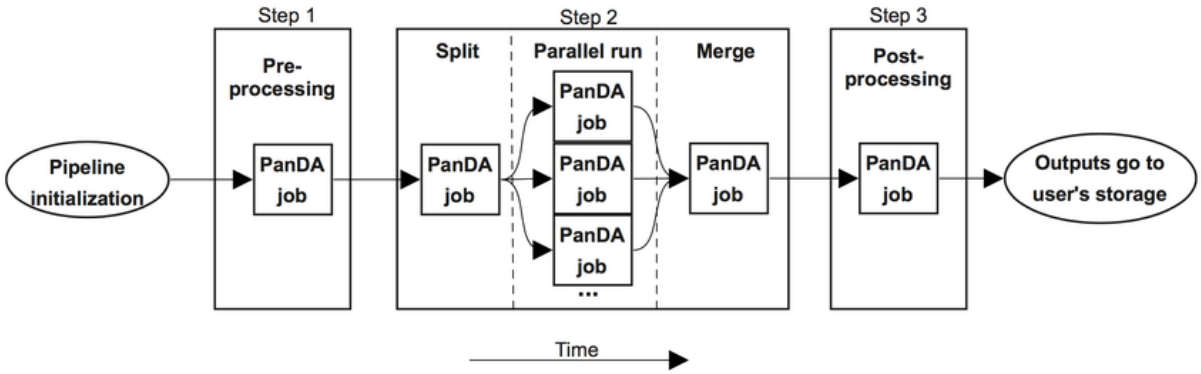
A gyártósor azért gyorsabb, mert egyszerre több autón dolgoznak, csak mindegyiken más műveletet végeznek.

Ugyanezt az elvet alkalmazza a **pipeline**, vagyis futószalagos utasítás-végrehajtás.

Amíg az egyik utasítást végrehajtják, a következőt már dekódolhatják, egy harmadikat pedig beolvashatnak.

Ettől egyetlen utasítás végrehajtása nem feltétlenül lesz gyorsabb, de **adott idő alatt több utasítás készülhet el**.

A modern processzorok ezt még tovább fejlesztették: több végrehajtóegység dolgozhat párhuzamosan, és a CPU akár az utasítások eredeti sorrendjétől eltérően is elvégezhet műveleteket, ha ezzel időt nyer.



Ábra:*A pipeline működése egy gyártósorhoz hasonlítható. Egyszerre több utasítás van feldolgozás alatt, de mindegyik a végrehajtás más-más szakaszában jár.*

2.4. Cache

A processzor sebességének növekedése új problémát teremtett: a memória nem tudta ugyanolyan ütemben követni.

Hiába képes a CPU rendkívül gyorsan dolgozni, ha folyton várnia kell az adatokra.

Ezért a processzor és a főmemória közé rendkívül gyors, kisebb méretű memóriát helyeztek: ez a **cache**, vagyis gyorsítótár.

A modern processzorokban általában több szint található:

L1 → nagyon kicsi, de rendkívül gyors
L2 → nagyobb, valamivel lassabb
L3 → még nagyobb, gyakran több processzormag közösen használja

A cache működése azon a megfigyelésen alapul, hogy a processzor nagy valószínűséggel hamarosan újra használni fogja a nemrég használt adatokat, illetve az azok közelében lévőket.

Olyan, mint az íróasztal: az éppen használt könyvet nem visszük vissza minden mondat elolvasása után a könyvtár raktárába.

2.5. Többmagos processzorok

Sokáig a processzorok gyorsításának egyik legkézenfekvőbb módja az órajel növelése volt. Ennek azonban fizikai korlátai egyre erősebben jelentkeztek.

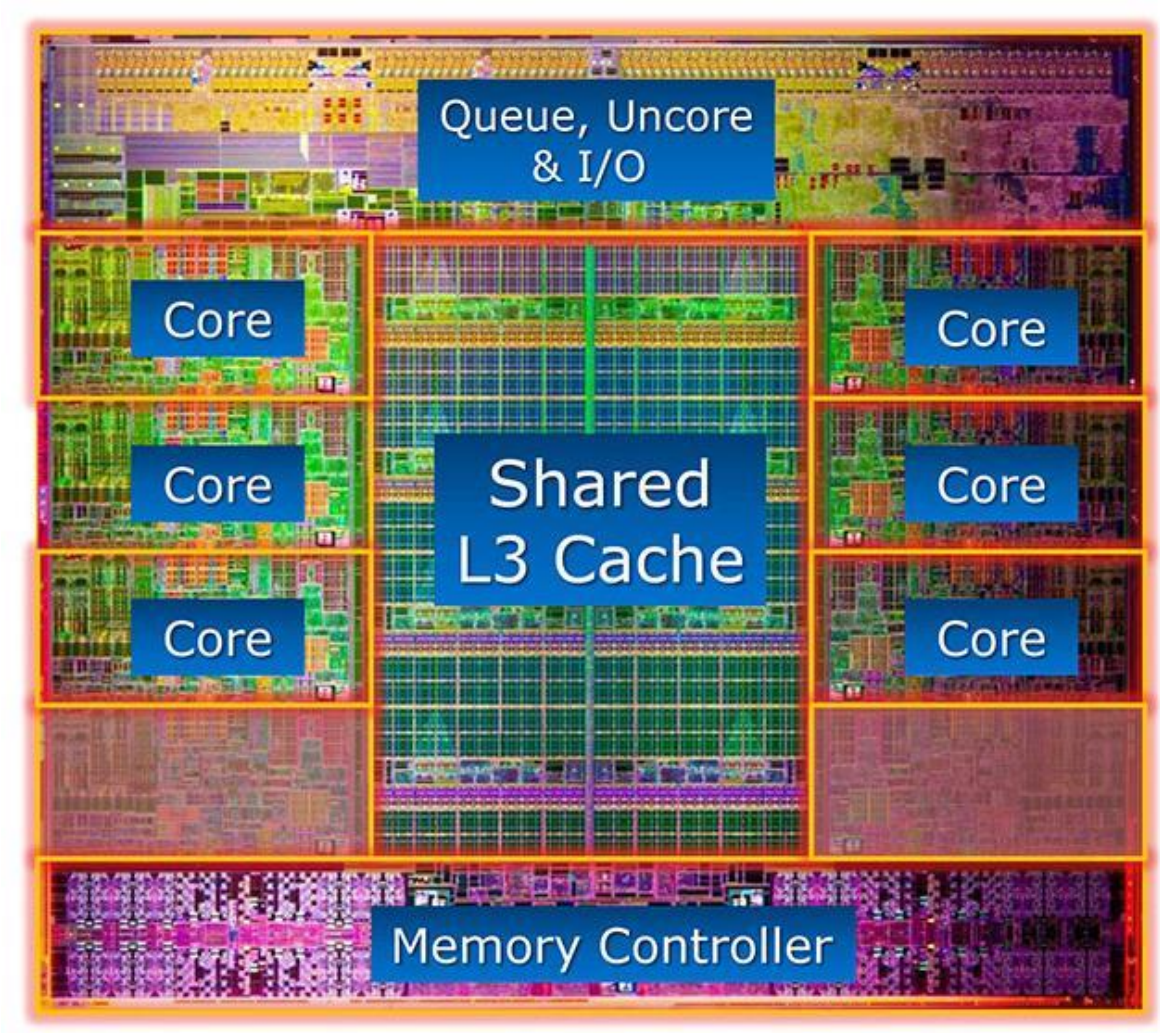
A megoldás az lett, hogy egyetlen lapkára **több teljes processzormagot** helyeztek.

Ha egy processzornak például nyolc magja van, bizonyos feladatok nyolc részre oszthatók, és párhuzamosan végezhetők.

Ez azonban nem jelenti azt, hogy egy nyolcmagos processzor minden esetben nyolcszor gyorsabb. Ehhez a programnak is párhuzamosíthatónak kell lennie.

A processzorfejlesztés történetében ezért fontos szemléletváltás történt:

nemcsak gyorsabban kell dolgozni – több dolgot kell egyszerre csinálni.



Ábra:
Többmagos processzor. Egyetlen lapkán több, részben önálló CPU-mag dolgozik. A teljesítmény növelésének egyik legfontosabb útja az órajel emelése helyett a párhuzamos feldolgozás lett.

2.6. ARM

Az **ARM** története különösen érdekes, mert jól mutatja, hogy nem mindig a legerősebb processzor győz, hanem az, amelyik az adott feladatra a legjobb kompromisszumot kínálja.

Az ARM architektúra az 1980-as években született a brit Acorn Computersnél. A tervezők egyszerű, gyors és energiatakarékos RISC processzort akartak.

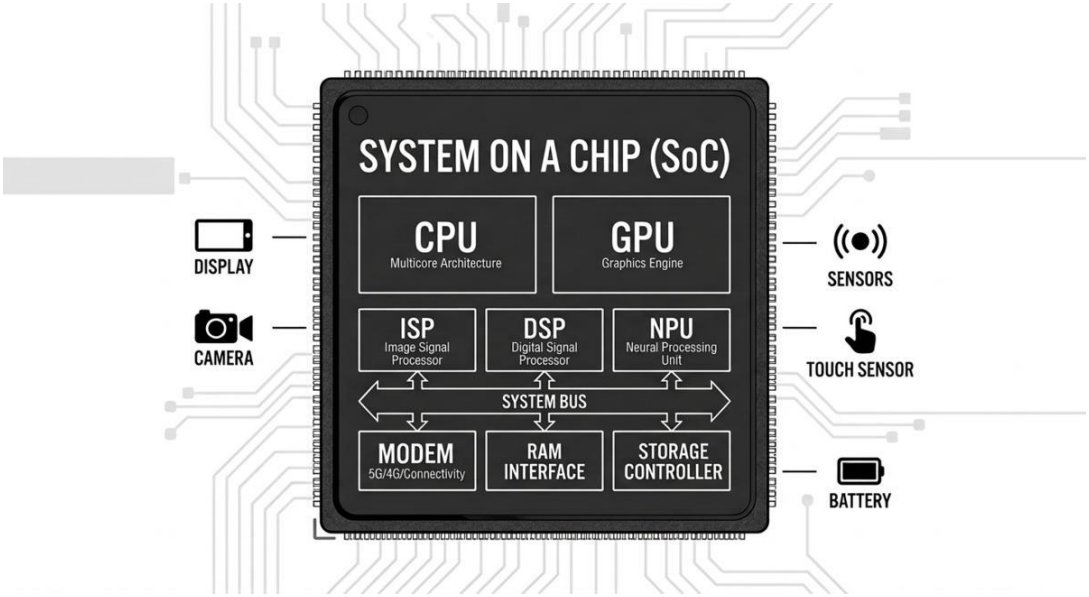
Az energiahatékonyság később döntő előnnyé vált.

Az ARM processzorok elterjedtek a beágyazott rendszerekben, majd a mobiltelefonokban és okostelefonokban. Ma ARM-alapú processzorok milliárdjai működnek telefonokban, táblagépekben, járművekben, hálózati eszközökben és egyre több személyi számítógépben, illetve szerverben.

Az ARM üzleti modellje is szokatlan. Az ARM nem elsősorban saját processzorokat gyárt, hanem **processzorarchitektúrát és terveket licencel**, amelyeket más vállalatok saját chipjeikben használhatnak.

Így ugyanaz az alapvető architektúra rendkívül sokféle készülékben jelenhet meg.

2.7. SoC



A mikroprocesszor annak idején egy egész CPU-t sűrített egyetlen chipre. A fejlődés következő lépése az lett, hogy **szinte az egész számítógépet egyetlen chipre helyezzük**.

Ez a **System on a Chip – SoC**.

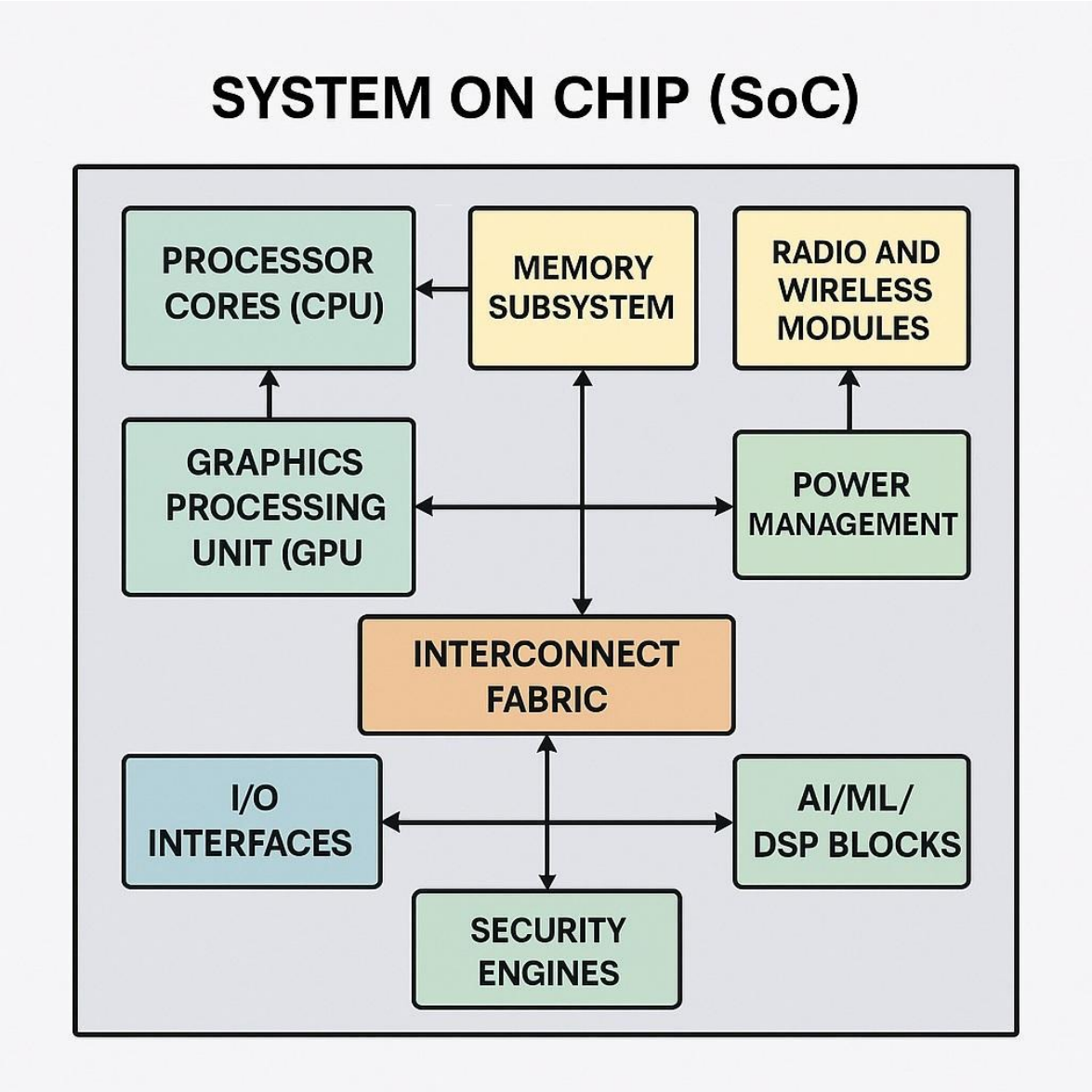
Egy modern SoC tartalmazhat:

CPU-magokat, GPU-t, memóriavezérlőt, kép- és videofeldolgozó egységeket, kommunikációs áramköröket, biztonsági egységeket és mesterséges intelligenciát gyorsító NPU-t.

Az okostelefonok elterjedését nagyrészt ez a rendkívüli integráció tette lehetővé.

A fejlődés érdekes ívet rajzol:

egy teremnyi számítógép → egy szekrény → egy alaplap → egy processzor → szinte egy teljes számítógép egyetlen chipen.



Ábra:
A System on a Chip már nem csupán processzor. Egyetlen lapkán CPU, grafikus egység, memóriavezérlő és számos speciális feldolgozóegység osztozhat.

2.8. CPU kontra GPU

A **CPU** általános célú processzor. Viszonylag kevés, de nagy teljesítményű magja sokféle, egymástól nagyon eltérő feladat gyors végrehajtására alkalmas.

A **GPU – Graphics Processing Unit** eredetileg a számítógépes grafika gyorsítására született. Egy kép előállításakor azonban rengeteg hasonló matematikai műveletet kell egyszerre elvégezni.

Ezért a GPU-k sok párhuzamos számítás elvégzésére alkalmas végrehajtóegységet kaptak.

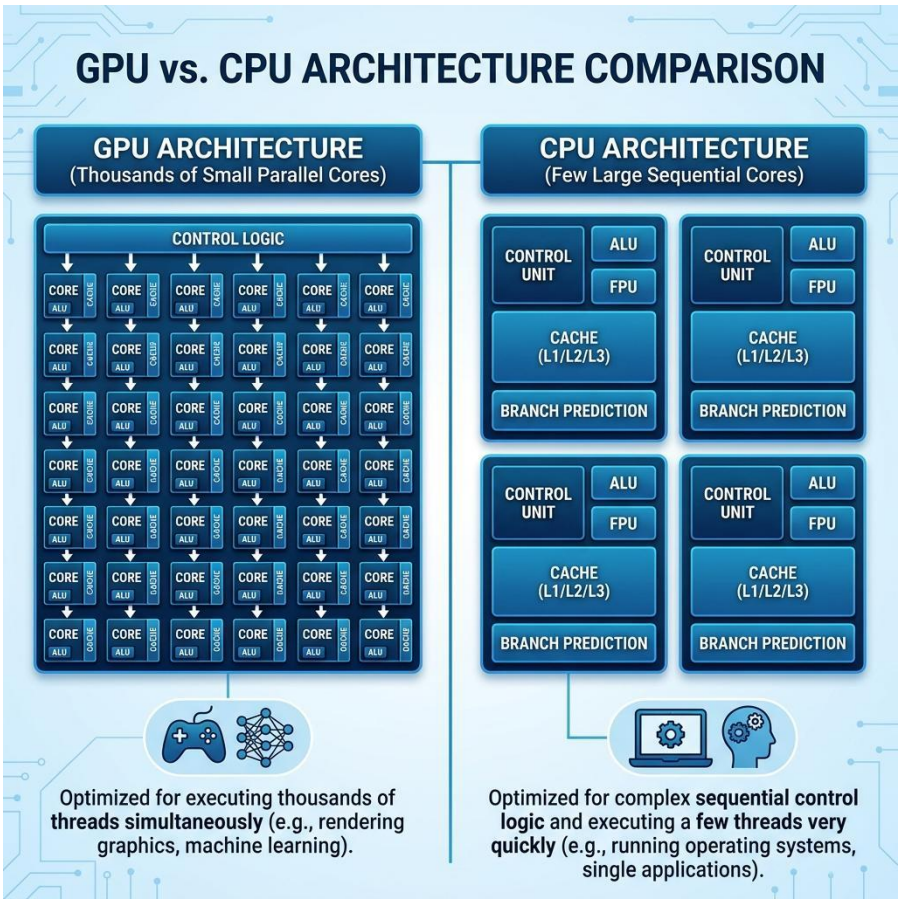
Kiderült, hogy ugyanez a felépítés nemcsak grafikára jó. Tudományos számítások, szimulációk és különösen a neurális hálózatok tanítása is rengeteg párhuzamos műveletet igényel.

A GPU így a grafikus gyorsítóból **általános számítási gyorsítóvá** vált.

Nem arról van szó, hogy a GPU jobb CPU lenne. Más feladatra tervezték.

A CPU inkább néhány rendkívül sokoldalú szakemberhez hasonlítható, a GPU pedig sok száz vagy ezer olyan munkáshoz, akik egyszerre képesek hasonló műveleteket elvégezni.

A mesterséges intelligencia fejlődésének egyik fontos technikai feltétele éppen ez a párhuzamos számítási kapacitás lett.



Ábra:
CPU és GPU eltérő munkamegosztása. A CPU kevés, összetett magja változatos feladatokra alkalmas, míg a GPU nagyszámú párhuzamos végrehajtóegysége sok hasonló számítás egyidejű elvégzésében erős.

Miért nem lehet egyszerűen tovább növelni az órajelet?

Az első mikroprocesszorok órajele még megahertzekben volt mérhető. Az Intel 4004 körülbelül 740 kHz-en működött, az IBM PC 8088 processzora 4,77 MHz-en. Később megjelentek a több száz megahertzes, majd az 1, 2 és 3 GHz-es processzorok.

Logikusnak tűnt, hogy hamarosan 10, 20 vagy akár 100 GHz-es processzoraink lesznek.

Nem így történt.

A tranzisztorok kapcsolgatása **energiát fogyaszt és hőt termel**. Minél magasabb az órajel, annál gyakrabban kell átkapcsolniuk. A fogyasztás és a hőtermelés rohamosan növekszik, különösen azért, mert a nagyobb sebesség eléréséhez gyakran a feszültséget is növelni kell.

Ezt nevezik gyakran **power wallnak**, teljesítmény- vagy fogyasztási falnak.

Van azonban egy még alapvetőbb korlát is.

A jelek terjedési sebessége véges.

Egy 5 GHz-es processzor egy órajelciklusa mindössze:

0,2 nanoszekundum.

A fény vákuumban ezalatt csak körülbelül **6 centimétert** tesz meg – egy valódi chip vezetékeiben az elektromos jel effektív terjedése ennél lassabb, és maguk a tranzisztorok, vezetékek és kapuk is késleltetést okoznak.

A processzor tervezői ezért más utakat kerestek:

pipeline → több művelet átfedése
cache → kevesebb várakozás a memóriára
több végrehajtóegység → több művelet párhuzamosan
több mag → több programszál egyszerre
GPU → tömeges párhuzamos számítás
speciális gyorsítók → adott feladat hatékonyabb végrehajtása

A processzorok fejlődésének egyik legfontosabb tanulsága ezért az, hogy a fejlődés nem állt meg akkor, amikor az órajelek növekedése lelassult.

Megváltozott a kérdés.

Már nem az volt a cél, hogy:

„Hogyan kapcsolgassuk még gyorsabban a tranzisztorokat?”

hanem inkább:

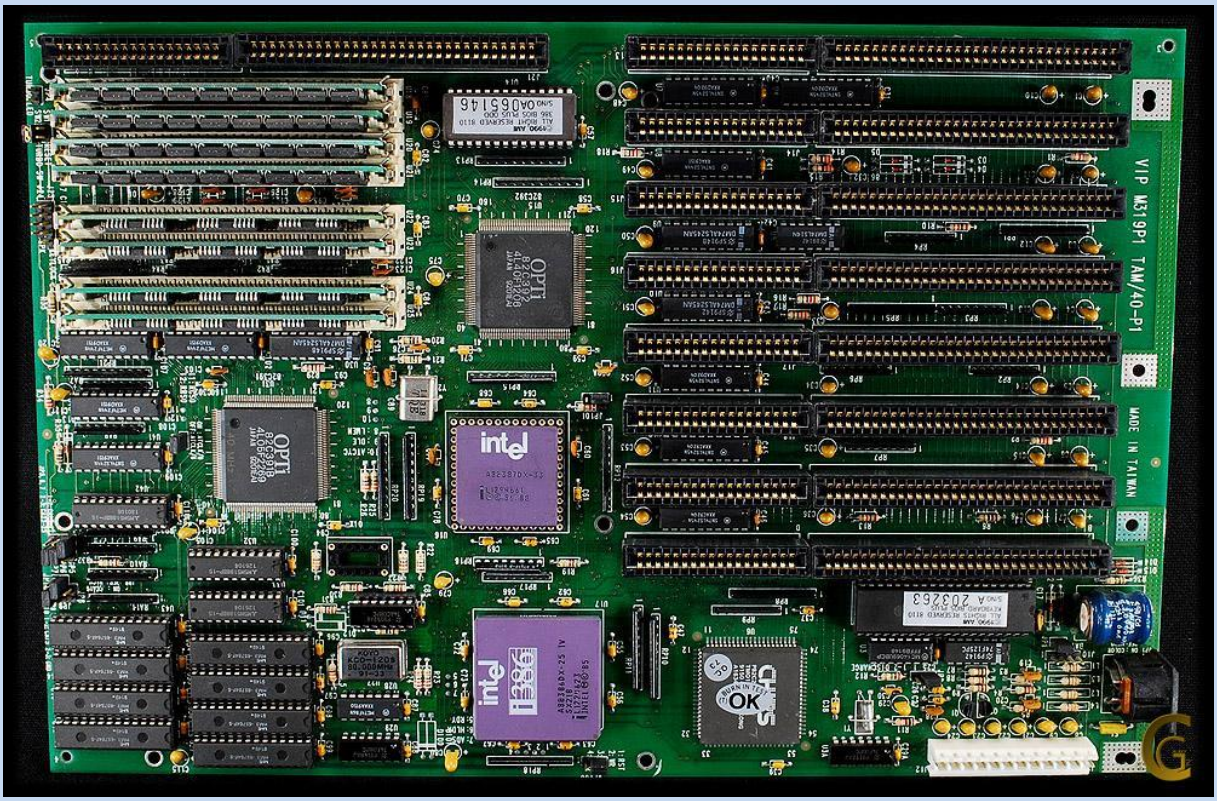
„Hogyan végezzünk több hasznos munkát ugyanannyi idő és energia alatt?”

És tulajdonképpen ez vezetett el a mai többmagos CPU-khoz, GPU-khoz és SoC-khez.

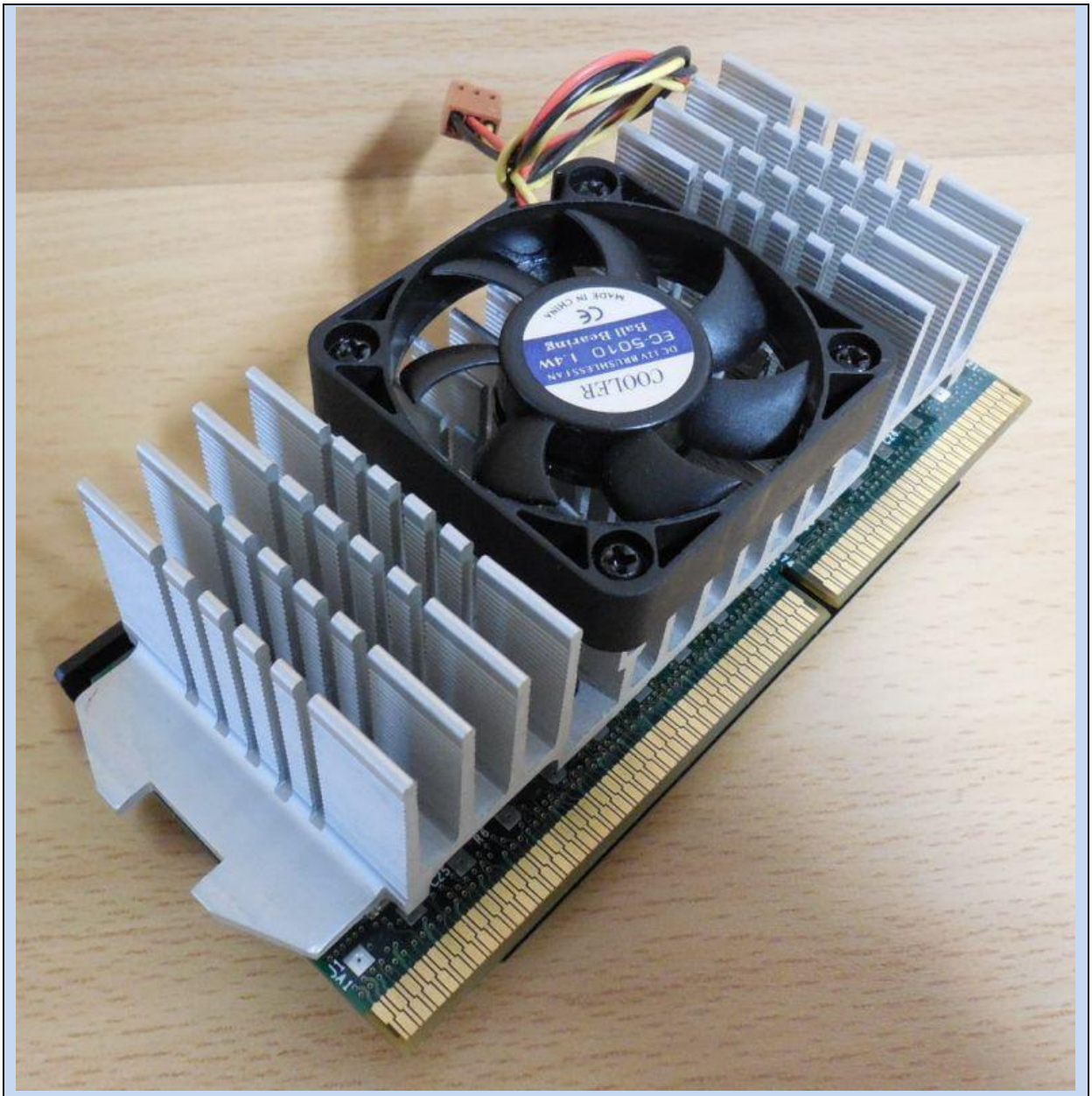
A processzor egyre melegebb lett

A korai mikroprocesszorok olyan kevés energiát fogyasztottak, hogy külön hűtésre alig volt szükségük. Még számos 386-os processzor is egyszerűen, csupasz tokozással működött. A tranzisztorszám és az órajel növekedésével azonban a hőtermelés is egyre komolyabb problémává vált.

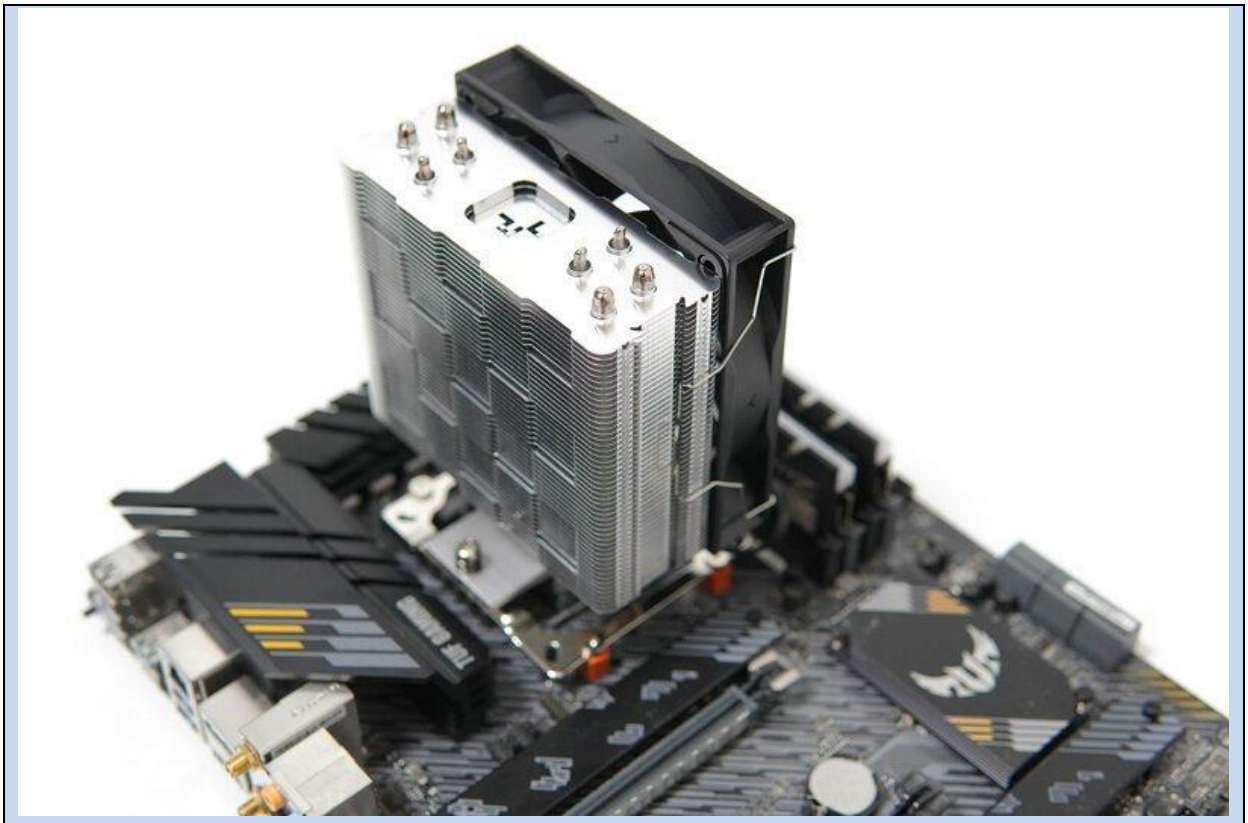
A fejlődés szinte ránézésre követhető: **csupasz processzor → kis hűtőborda → nagyobb hűtőborda → ventilátoros hűtés → hőcsöves hűtés → folyadékhűtés.** A mai nagy teljesítményű processzorok terhelés alatt akár 100–200 wattnál is nagyobb hőteljesítményt adhatnak le, ezért a hűtőrendszer már a számítógép egyik fontos szerkezeti eleme.



Intel 80386 processzor. A 386-os korszakban még gyakori volt, hogy a CPU külön hűtőborda és ventilátor nélkül működött.



Az 1990-es évek végére a hűtőborda mellett a ventilátoros kényszerhűtés is általánossá vált.





Modern processzorhűtés: nagyméretű, hőcsöves toronyhűtő, illetve folyadékhűtés radiátorral.

A hűtők méretének növekedése önmagában elmeséli a CPU-k hőtermelésének történetét.

Ki tervez meg több milliárd tranzisztort?

Az Intel 4004 körülbelül **2300 tranzisztort** tartalmazott. Áramköri felépítését még emberi léptékben is át lehetett tekinteni.

Egy mai nagy processzor vagy rendszerchip ezzel szemben több tízmilliárd tranzisztort tartalmazhat.

Ha egy mérnök másodpercenként egy tranzisztort helyezne el, és megállás nélkül dolgozna, **egymilliárd tranzisztorhoz is több mint harminc évre lenne szüksége.**

Hogyan lehet akkor ilyen áramkört egyáltalán megtervezni?

A titok ugyanaz, amely sok más bonyolult rendszer létrehozását lehetővé teszi:

nem tranzisztorokban gondolkodunk, hanem egyre magasabb szintű építőelemekben.

Nem tranzisztort, hanem funkciót tervezünk

A processzortervező rendszerint nem azt mondja:

„Ide teszek egy tranzisztort, mellé még kettőt, és összekötöm őket."

Hanem például azt: „Szükségem van egy 64 bites összeadóra."

vagy: „Ez az egység hajtsa végre a lebegőpontos szorzást."

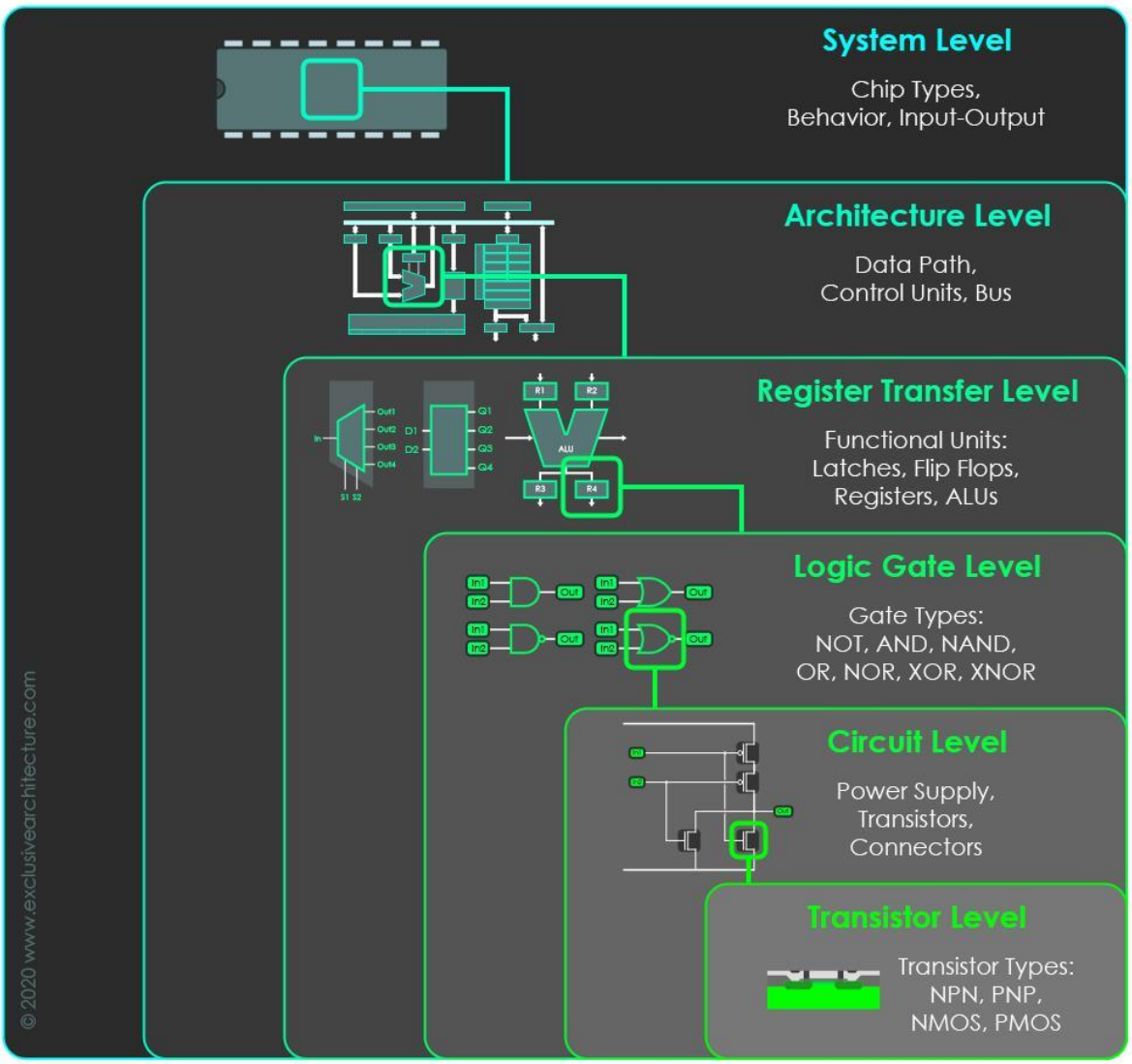
vagy: „Itt legyen egy 32 kilobájtos gyorsítótár."

A tervezés hierarchikus.

Egy tranzisztorcsoportból logikai kapu készül.

tranzisztorok → kapuk → funkcionális egységek → processzormag → többmagos processzor → teljes chip

Ugyanúgy, ahogyan egy építész sem téglánként tervezi meg a felhőkarcolót.



Ábra: *A processzor hierarchikus felépítése. A tervezőnek nem kell egyszerre több milliárd tranzisztort áttekintenie. A tranzisztorokból logikai elemek, azokból funkcionális blokkok, majd processzormagok és teljes rendszerek épülnek.*

A kapcsolási rajz helyett program

A modern digitális áramkörök jelentős részét **hardverleíró nyelveken** írják le.

Ilyen például a **VHDL**, a **Verilog** és a ma széles körben használt **SystemVerilog**.

Ezek első pillantásra programozási nyelvekre hasonlítanak, de fontos különbség van.

Egy hagyományos program azt írja le: **mit csináljon egymás után a processzor.**

A hardverleíró nyelv pedig azt: **milyen áramkör létezzen, és annak részei hogyan működjenek együtt.**

A mérnök például leírhatja egy összeadó, regiszter vagy vezérlőegység működését. A tervezőszoftver ebből logikai hálózatot készít.

Ezt a folyamatot nevezzük **logikai szintézisnek**.

A számítógép megtervezi a kapukat

A szintézis után már létezik az áramkör logikai kapcsolatait leíró hálózat, a **netlist.**

De még mindig nem tudjuk, hogy a chip fizikai felületén pontosan hol legyenek az egyes elemek.

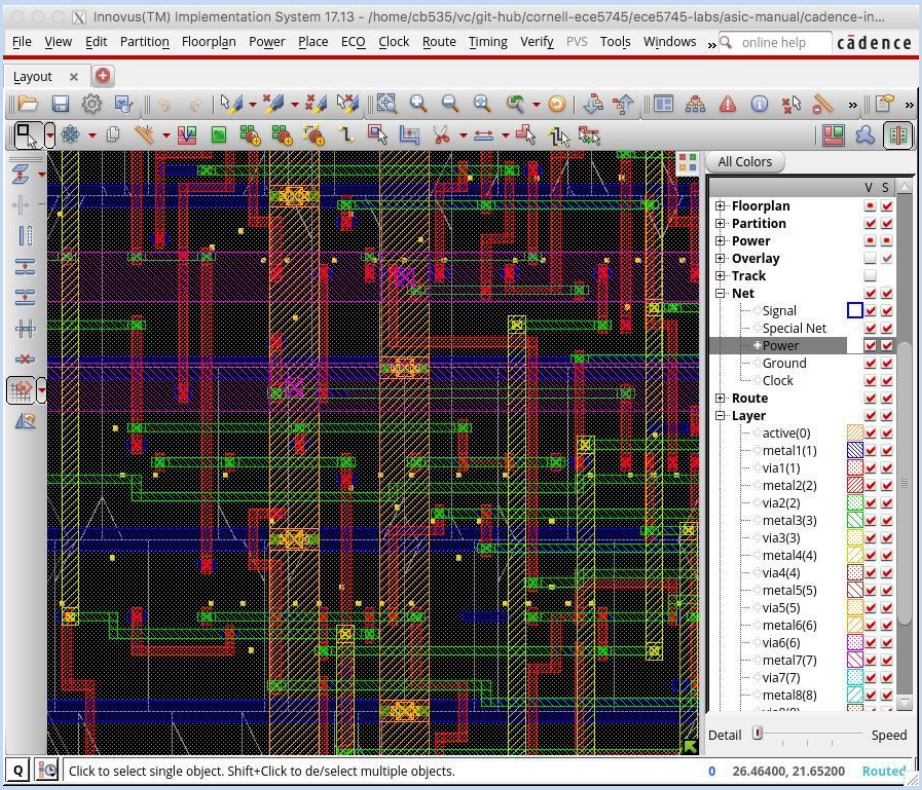
Ekkor következik az automatizált fizikai tervezés.

A tervezőprogram megpróbálja elhelyezni az áramköri elemeket:

placement – elhelyezés

majd létrehozza közöttük a vezetékeket:

routing – huzalozás.



Csakhogy ez elképesztően nehéz optimalizálási feladat.

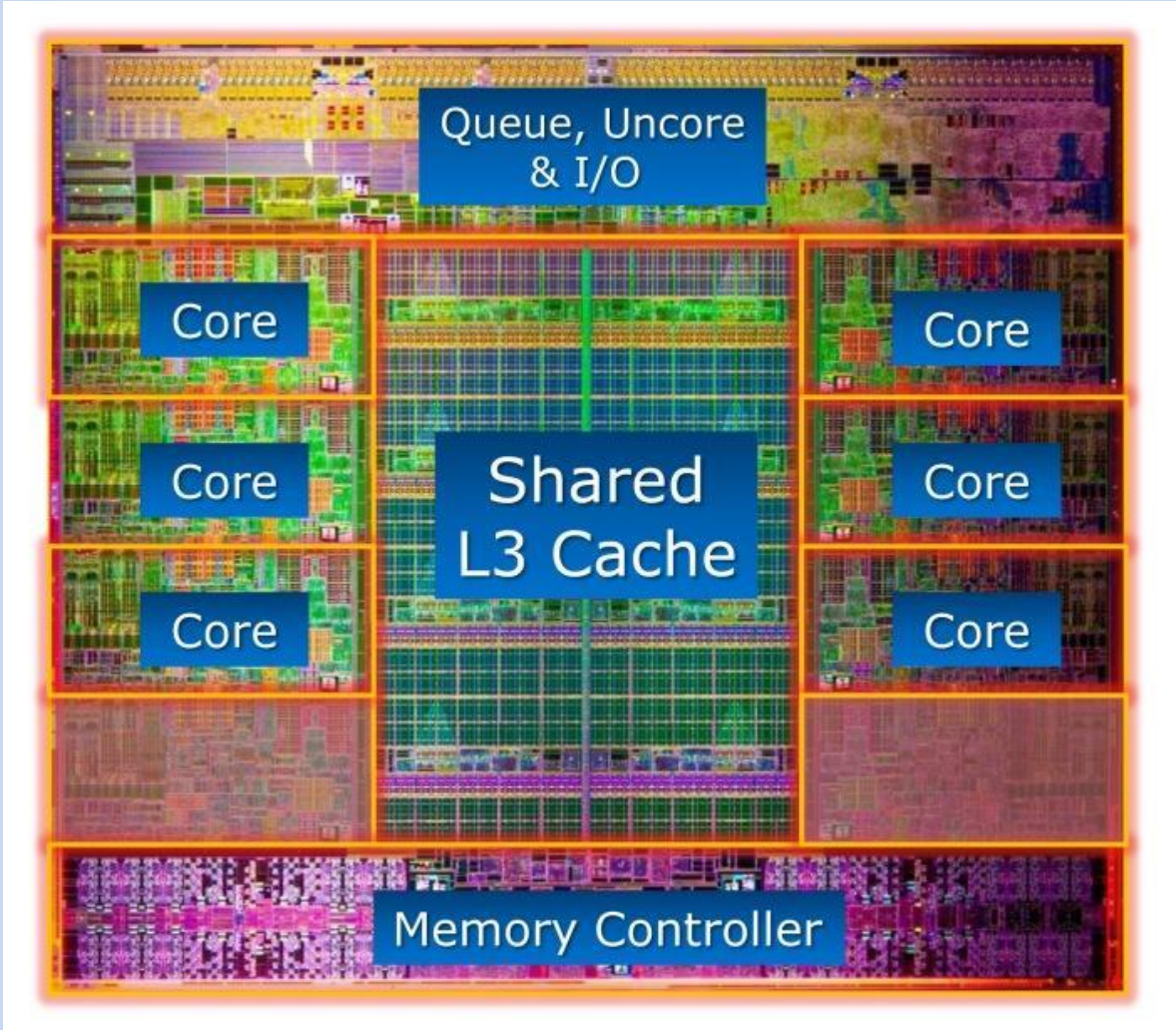
Nem elég, hogy két elem össze legyen kötve. Figyelembe kell venni:

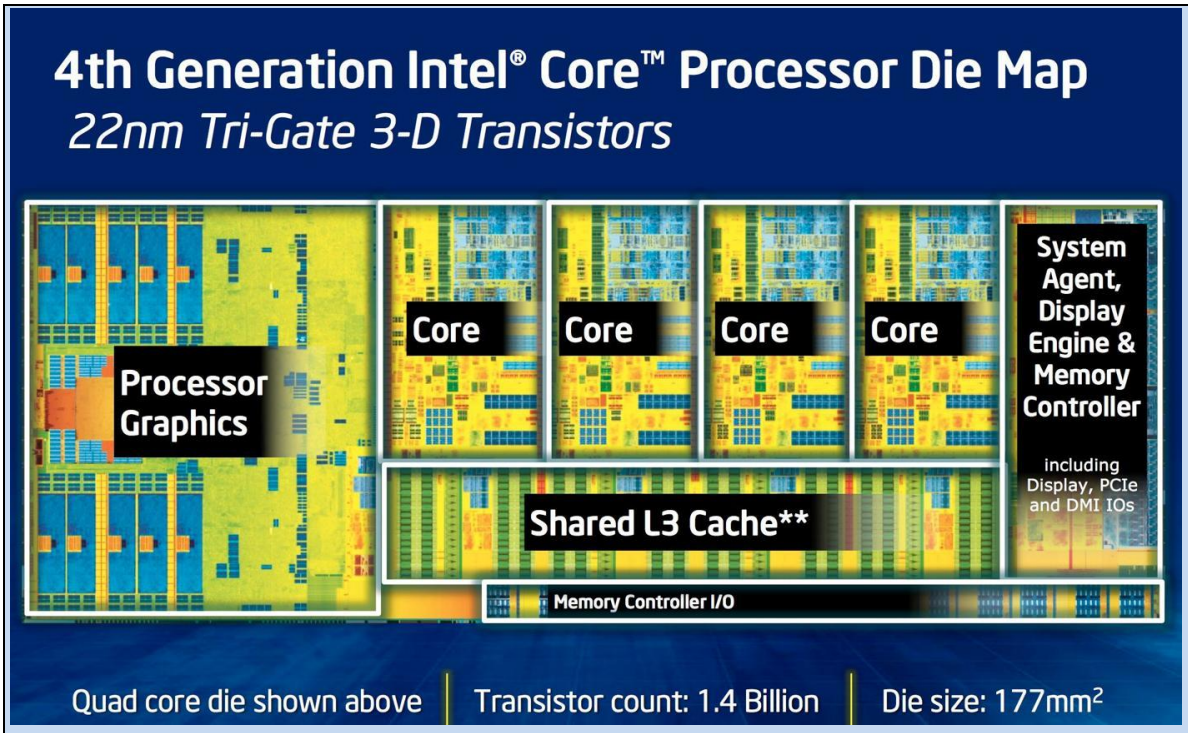
- a jel terjedési idejét,
- az órajelet,
- az energiafogyasztást,
- a hőtermelést,
- a vezetékek kölcsönhatását,
- a tápellátást,
- és azt is, hogy az egész egyáltalán legyártható legyen.

A számítógép tehát nem egyszerűen „összedrótozza” a kapcsolást. **Milliárdnyi elem elhelyezését optimalizálja.**

Először a nagy egységeket helyezik el

Ha megnézzük egy modern processzor lapkájának fényképét, szabályos területeket láthatunk.





Ezek nem véletlenül alakultak ki.

Már a tervezés elején eldőlhet, hogy nagyjából hol legyen:

CPU-mag – cache – memóriavezérlő – grafikus egység – I/O – speciális gyorsító.

Ezt nevezik **floorplanningnek.**

Kicsit olyan, mint egy város megtervezése. Először kijelöljük a lakónegyedeket, ipari területeket és főutakat. Csak ezután tervezzük meg az egyes utcákat és épületeket.

A chip vezetékhálózata pedig valóban emlékeztethet egy elképesztően sűrű város úthálózatára.

De honnan tudjuk, hogy működik?

Talán ez még nagyobb probléma, mint maga a tervezés.

Egy több milliárd tranzisztoros processzor minden lehetséges állapotát **lehetetlen egyenként kipróbálni.**

Ezért a tervezés hatalmas részét nem is maga a tervezés, hanem a **verifikáció,** vagyis az ellenőrzés teszi ki.

A mérnökök szimulálják az áramkört, automatikus teszteket futtatnak, formális matematikai módszerekkel vizsgálják bizonyos tulajdonságait, majd prototípusokon is ellenőrzik.

A chipnek ráadásul nemcsak logikailag kell helyesen működnie.

Azt is ellenőrizni kell, hogy:

minden jel időben odaér-e,
nem fogyaszt-e túl sokat,
nem melegszik-e túl,
megfelelő-e a tápellátás,
és legyárthatók-e a kialakított struktúrák.

Egyetlen hiba rendkívül drága lehet. Egy fejlett processzor gyártásához szükséges maszkkészlet és újragyártás költsége már olyan nagy, hogy a hibát lehetőleg **még a szilícium elkészítése előtt** meg kell találni.

A régi terveket sem dobják el

Van még egy fontos trükk.

A processzor minden részét nem tervezik újra minden generációnál.

A már bevált áramköri blokkok újra felhasználhatók és továbbfejleszthetők. Ezeket gyakran **IP-blokkoknak – Intellectual Property cores –** nevezik**.**

Lehet ilyen: memóriavezérlő, USB-vezérlő, PCIe-egység, processzormag, grafikus egység vagy akár teljes kommunikációs rendszer.

Az ARM üzleti modellje nagyrészt éppen erre épül: más cégek licencelhetnek kész processzorarchitektúrát vagy processzormag-terveket, majd beépíthetik saját chipjeikbe.

Így egy modern SoC tervezése bizonyos szempontból hasonlíthat egy **rendkívül bonyolult LEGO-készlet összeállításához** – azzal a különbséggel, hogy az egyes építőelemek belseje maga is elképesztően összetett.

És már a mesterséges intelligencia is segít

A történet legújabb fordulata, hogy az automatizálás tovább lép.

A chiptervező programok régóta optimalizálnak elhelyezést és huzalozást, de az újabb módszerekben **gépi tanulást is alkalmaznak** bizonyos tervezési problémák megoldásának gyorsítására vagy jobb megoldások keresésére.

Ez különös visszacsatolást hozott létre:

egyre nagyobb számítási teljesítményű processzorokat használunk arra, hogy megtervezzük a következő, még nagyobb számítási teljesítményű processzorokat.

A számítógép tehát bizonyos értelemben már részt vesz **saját következő nemzedékének megtervezésében.**

Egy processzor születése röviden

Ötlet, követelmények
↓
Architektúra
↓
Hardverleíró nyelv – RTL
↓
Szimuláció és verifikáció
↓
Logikai szintézis
↓
Netlist
↓
Floorplan
↓
Elhelyezés és huzalozás
↓
Időzítés, fogyasztás és gyárthatóság ellenőrzése
↓
Fizikai chipterv
↓
Maszkok
↓
Wafergyártás
↓
Tokozás és tesztelés
↓
PROCESSZOR

És van ebben egy nagyon szép történeti ív, amit a keretes írás zárómondatának használnék:

Az első számítógépeket az ember alkatrészről alkatrészre építette. A mai processzor már annyira bonyolult, hogy csak egy másik számítógép segítségével lehet megtervezni.

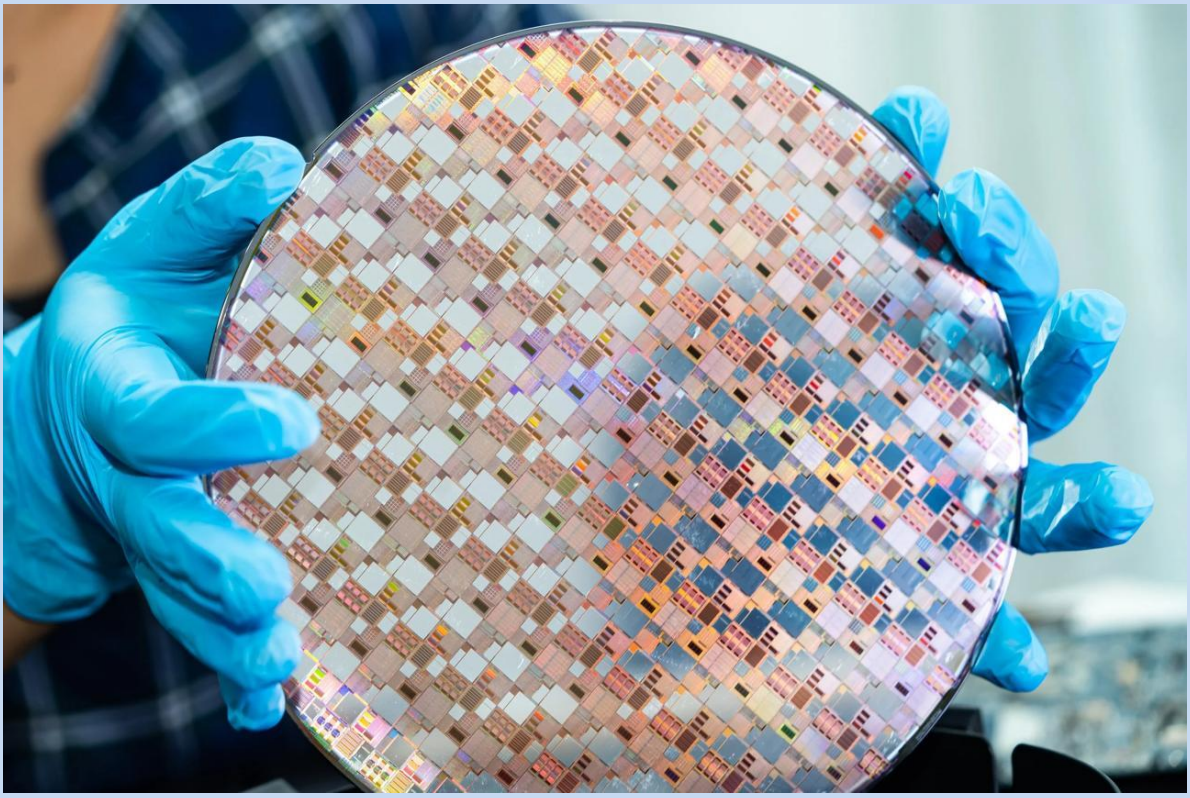
Ez ráadásul remekül előkészíti a következő kérdést is: **ha már megterveztük ezt a több tízmilliárd tranzisztort, hogyan lehet fizikailag legyártani őket egy körömnyi szilíciumdarabon?** Ez szerintem akár egy második keretes írást is megérne.

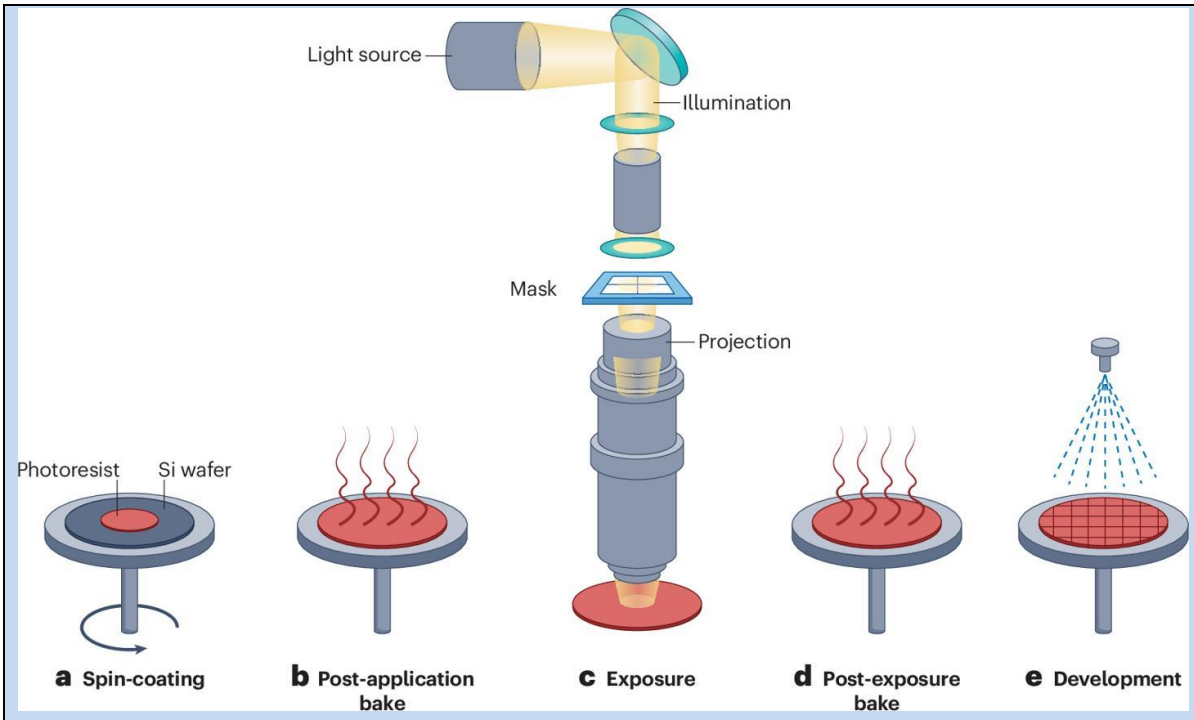
Hogyan készül több milliárd tranzisztor egy körömnyi lapkán?

Egy modern processzor gyártásakor természetesen senki nem helyezi el egyenként a tranzisztorokat. A teljes áramköri mintázatot **rétegről rétegre alakítják ki egy szilíciumkorongon**, a waferen.

Bizonyos értelemben ez a világ legpontosabb és legbonyolultabb nyomdai eljárása.







Az alapanyag: szinte tökéletes szilíciumkristály

A kiindulási anyag rendkívül tiszta, egykristályos szilícium. Ebből hengert növesztenek, majd vékony szeletekre vágják.

A mai nagyüzemi gyártásban jellemzően **300 mm átmérőjű wafereket** használnak.

Egyetlen waferen egyszerre sok processzor készül.

Ábra: *300 mm-es szilíciumwafer. A rajta látható ismétlődő téglalapok az egyes chipek, amelyek a gyártás végén külön lapkákra vághatók.*

Fénnyel rajzolunk

A következő kulcsszó a **fotolitográfia**.

A wafer felületére fényérzékeny réteget, úgynevezett fotorezisztet visznek fel. Ezután rendkívül bonyolult optikai rendszerrel rávetítik az elkészítendő struktúra mintázatát.

A megvilágított és nem megvilágított területek eltérően reagálnak az előhívásra.

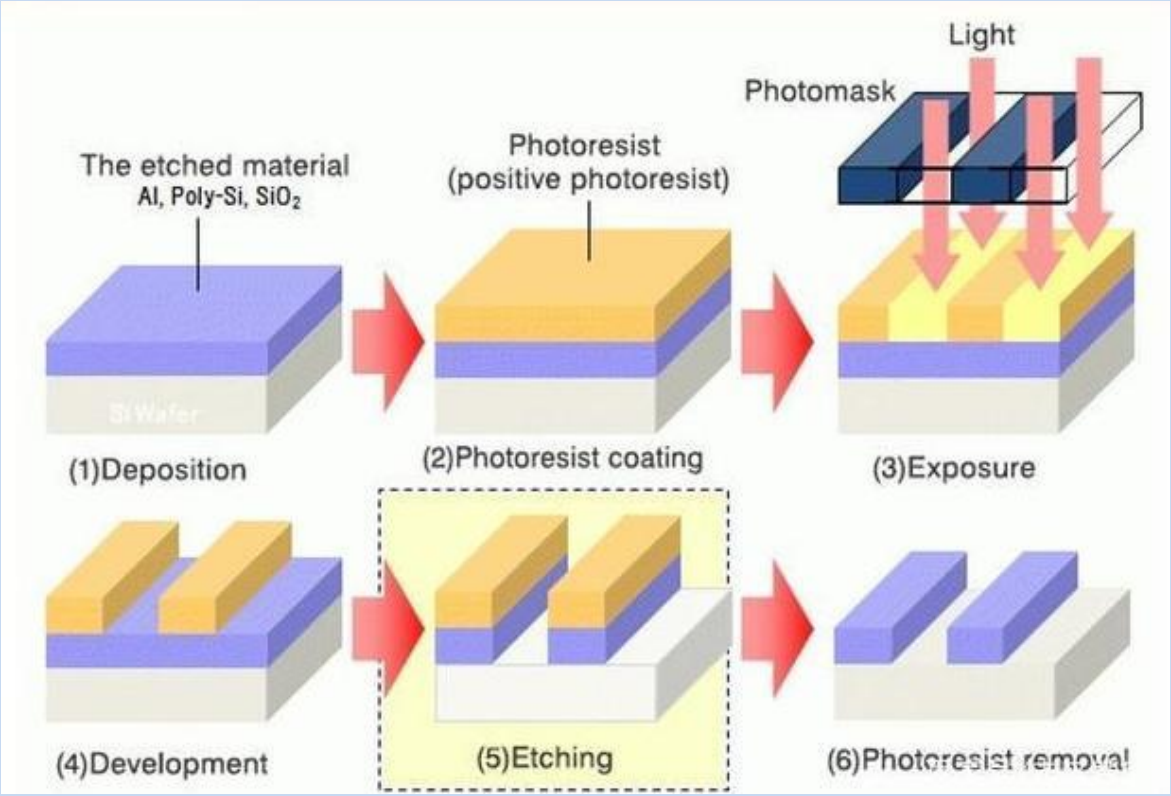
Így létrejön egy mikroszkopikus „sablon”.

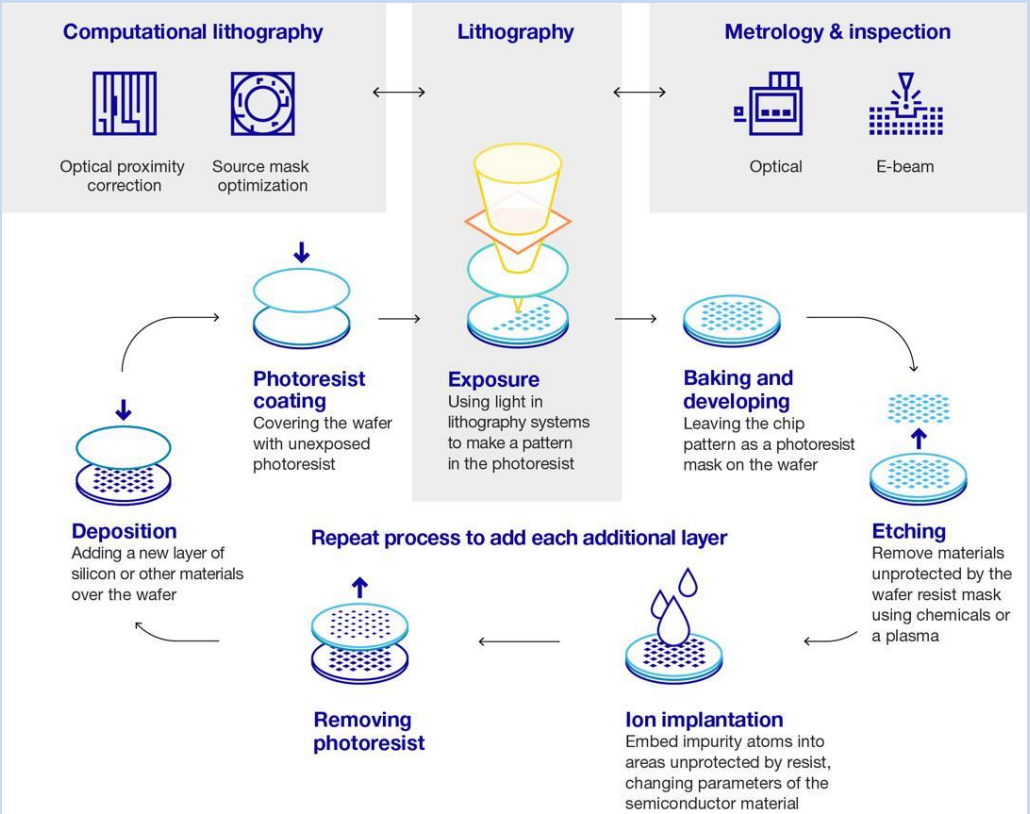
Ezután különféle technológiai műveletek következhetnek: anyagot maratnak le, új anyagréteget visznek fel, ionokat juttatnak a szilíciumba, szigetelő- vagy vezetőrétegeket alakítanak ki.

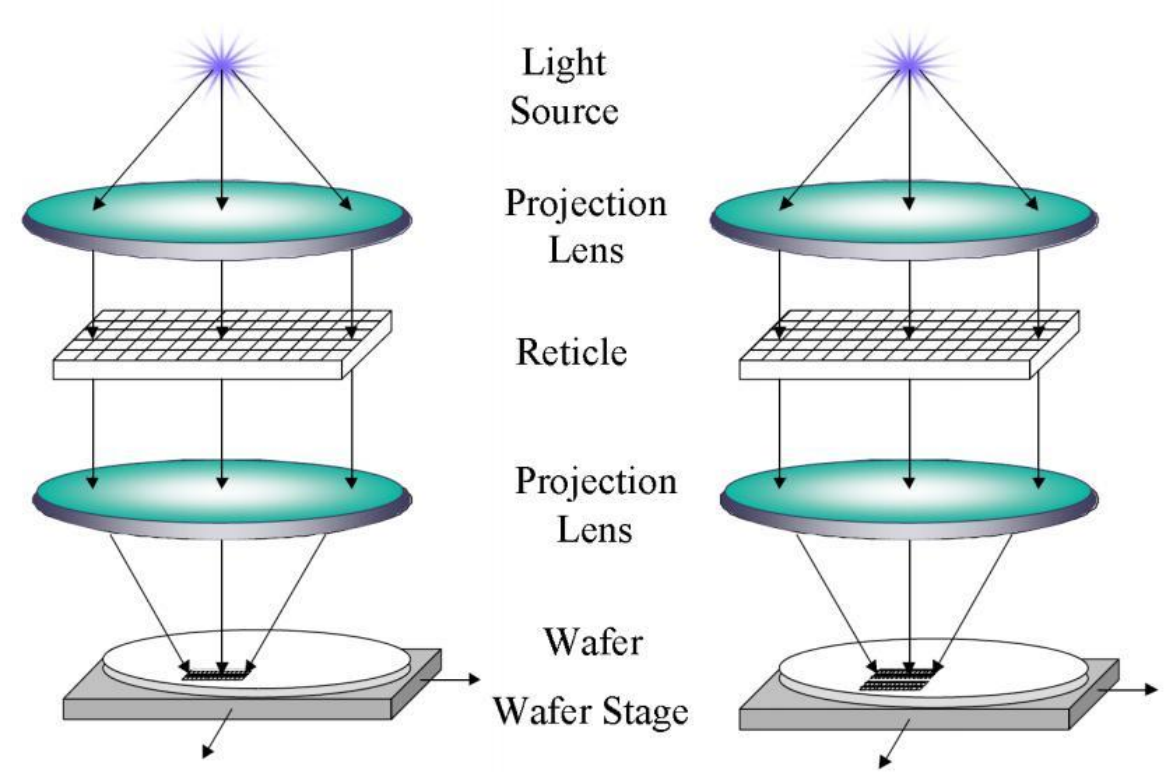
Majd az egészet újrakezdik.

Új réteg. Új minta. Új levilágítás. Új megmunkálás.

Egy modern chip elkészítéséhez ilyen lépések százaira, a teljes gyártási folyamatban pedig akár ezernél is több külön műveletre lehet szükség.





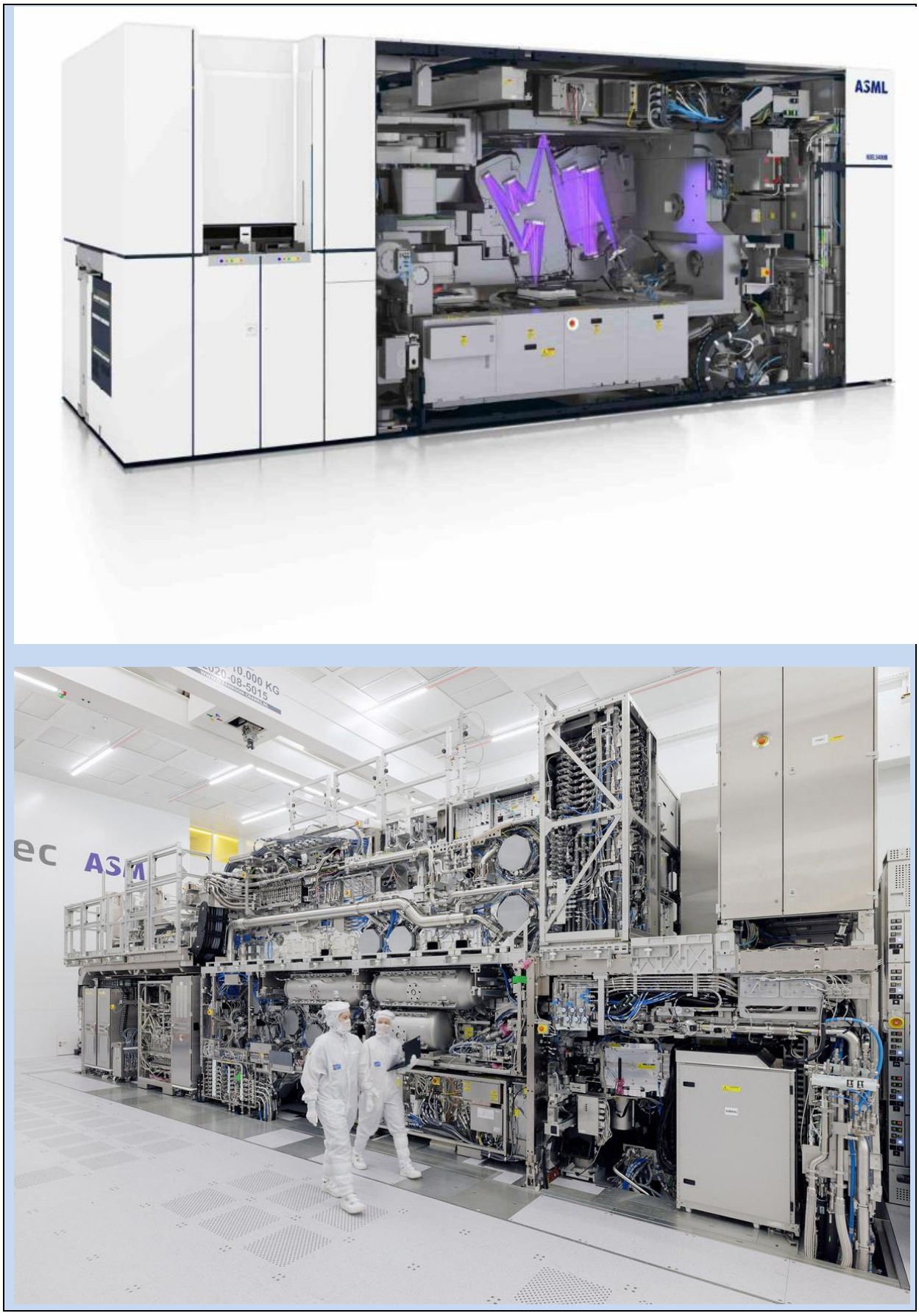


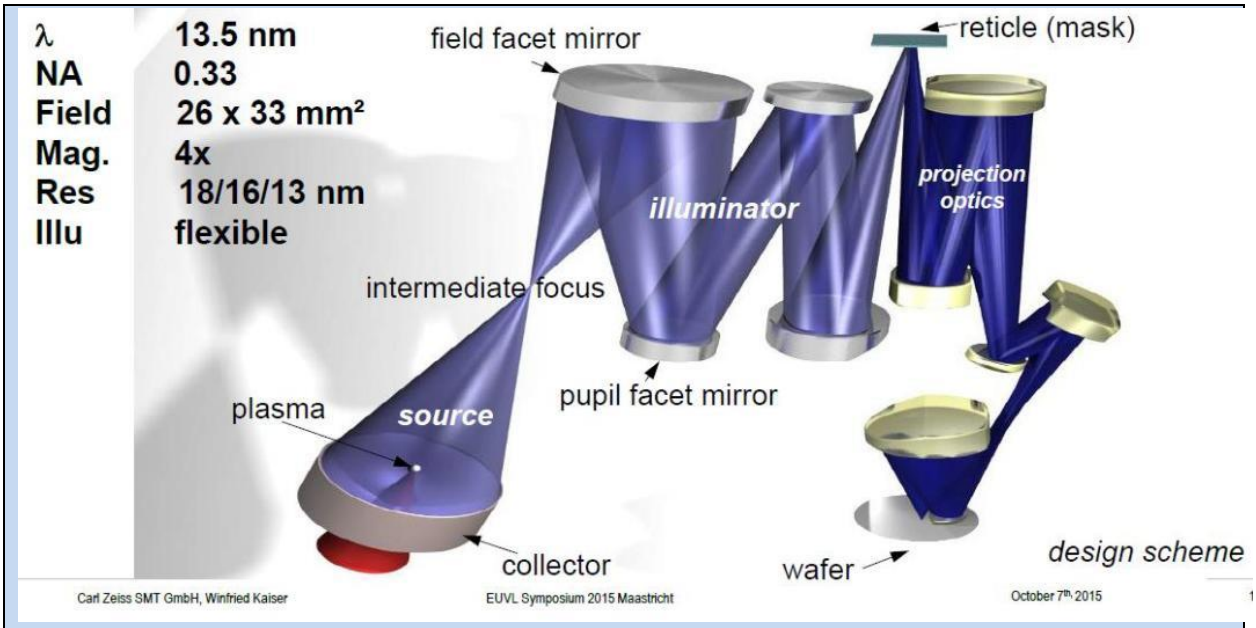
Ábra: *A fotolitográfia alapelve. A fényérzékeny rétegen kialakított mintázat segítségével lehet meghatározni, hol történjen a következő anyagleválasztási, maratási vagy más gyártási művelet.*

De hogyan rajzolunk néhány nanométeres mintát?

Itt válik igazán meghökkentővé a történet.

A legfejlettebb chipeknél **EUV-litográfiát**, vagyis extrém ultraibolya sugárzást alkalmaznak. Ennek hullámhossza körülbelül **13,5 nanométer**.


ASML



Az EUV-fény előállítása önmagában elképesztő technikai teljesítmény. Apró óncseppeket lézerimpulzusokkal plazmává alakítanak, és ebből nyerik a szükséges sugárzást.

A 13,5 nm-es EUV-t a levegő és még az üveg is erősen elnyeli. Ezért a rendszernek vákuumban kell működnie, és hagyományos lencsék helyett rendkívüli pontosságú **tükröket** használ.

Egy ilyen litográfiai berendezés a valaha épített legösszetettebb ipari gépek közé tartozik.

És mindezt azért, hogy a szilíciumra elkészíthessük azokat a parányi struktúrákat, amelyekből a tranzisztorok lesznek.

Mekkora egy mai tranzisztor?

Itt kell óvatosnak lennünk.

Amikor azt olvassuk, hogy egy processzor

7 nm-es, 5 nm-es, 3 nm-es vagy 2 nm-es technológiával

készült, könnyű azt gondolni, hogy egy tranzisztor 2 vagy 3 nanométer hosszú.

Ma ez már nem igaz.

A régebbi félvezető-technológiáknál a csíkszélesség megnevezése még szorosabban kapcsolódott egy konkrét fizikai mérethez, például a tranzisztor kapuhosszához. A modern technológiai „node” elnevezések azonban lényegében **technológiai generációk neveivé váltak**.

Egy „3 nm-es” tranzisztornak nincs szükségképpen 3 nm-es alkatrésze, amelyből egyértelműen levezethető volna a név.

A mai „3 nm” inkább egy gyártástechnológiai generáció neve, mint egy vonalzóval megmérhető tranzisztorméret.

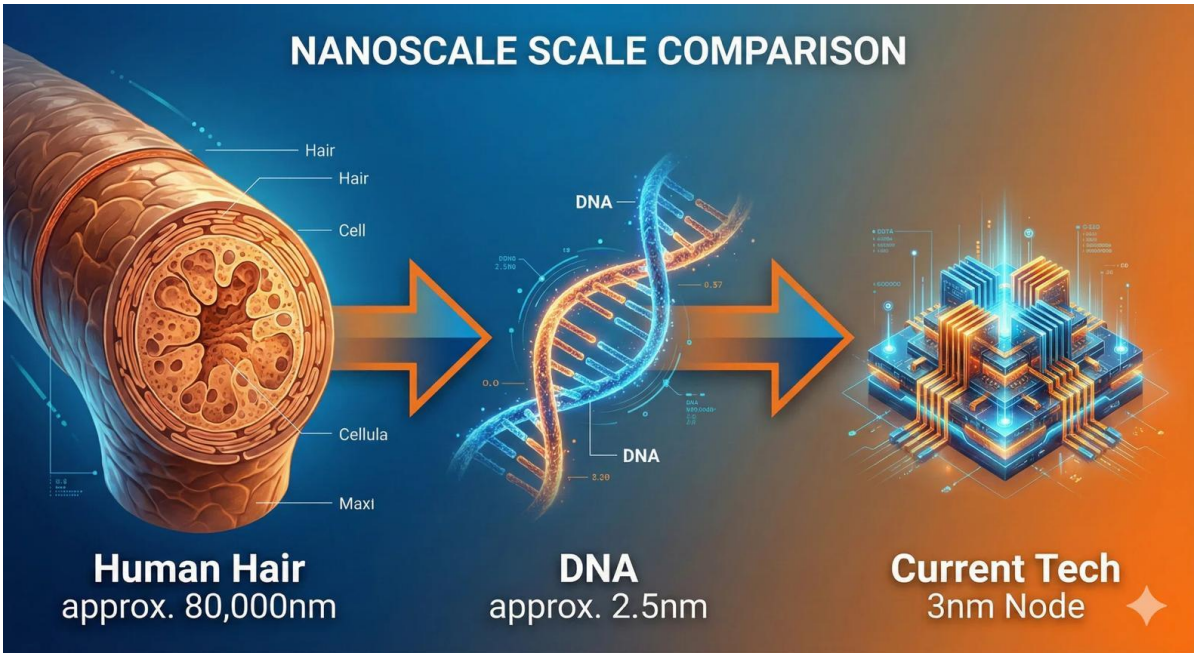
Akkor mégis milyen kicsi?

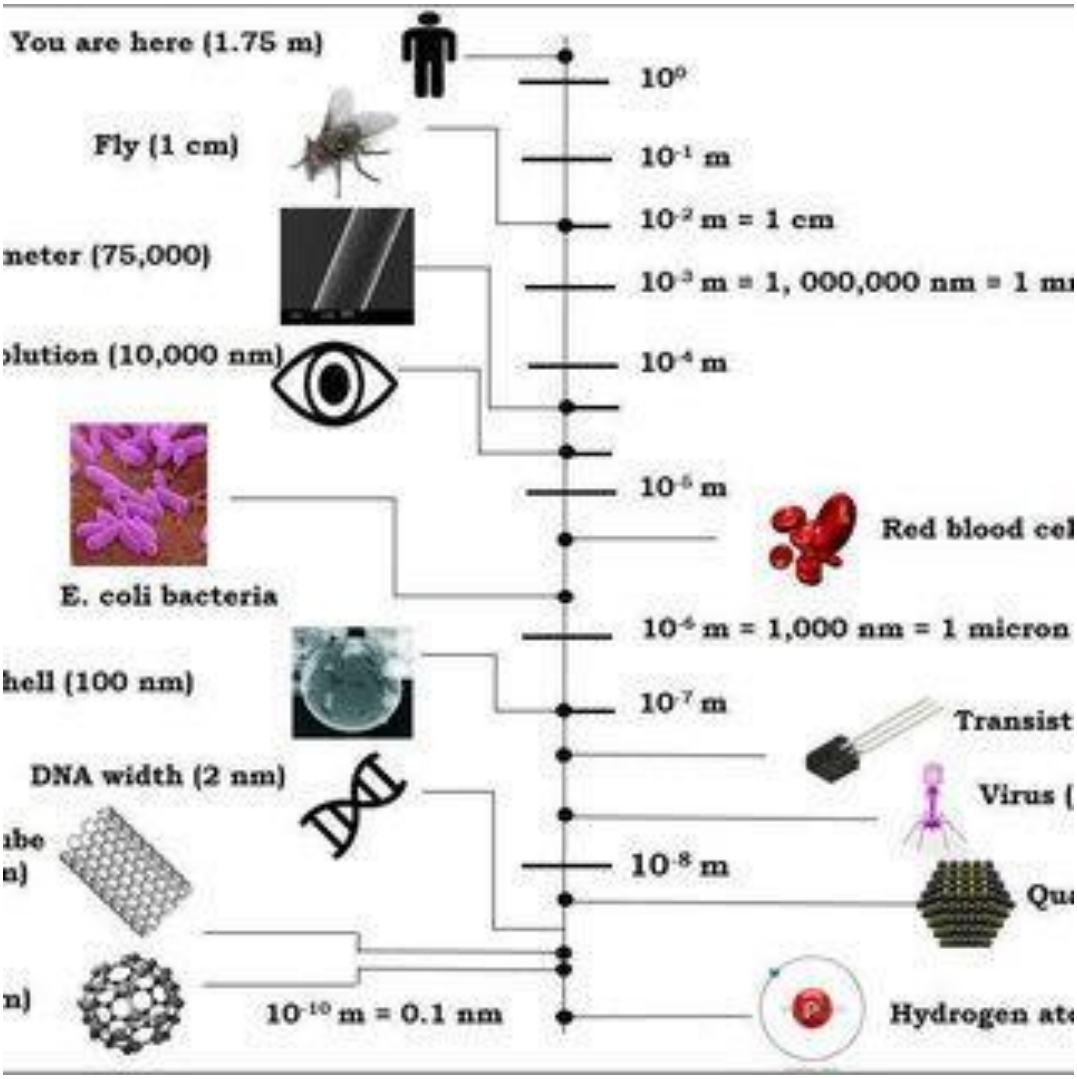
A modern tranzisztor több különböző mérettel rendelkezik. Egyes kritikus szerkezeti méretek **néhány tíz nanométeres**, más részletek pedig valóban **néhány nanométeres** tartományba eshetnek.

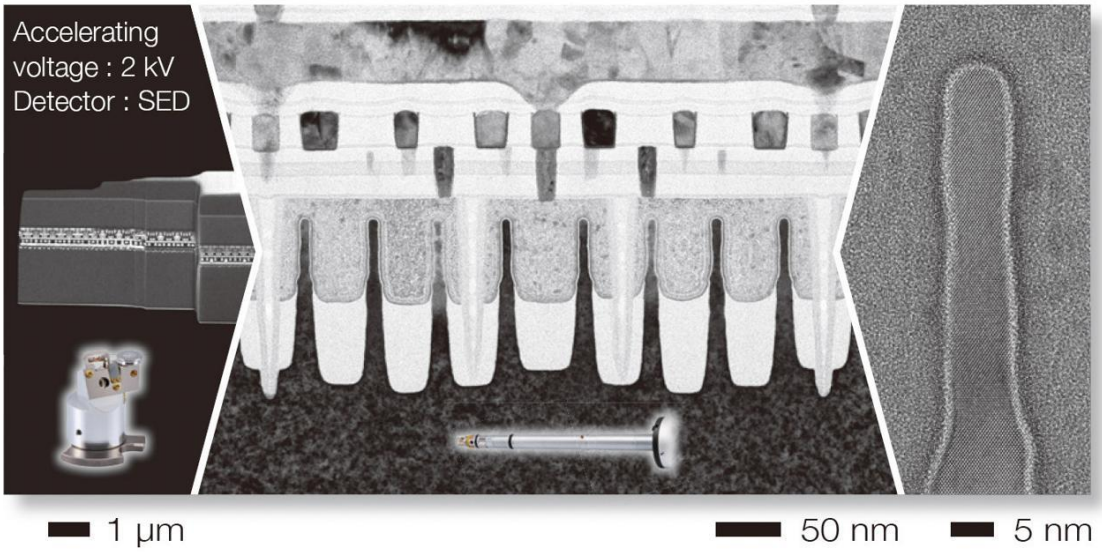
A legvékonyabb rétegek már szinte atomi léptékűek.

Összehasonlításként:

emberi hajszál: kb. 50 000–100 000 nm
vörösvértest: kb. 7 000–8 000 nm
baktérium: gyakran kb. 1 000–5 000 nm
DNS kettős spirál szélessége: kb. 2 nm
szilíciumatom: néhány tized nanométeres lépték







Specimen: FinFET with a process rule of 5 nm

Instrument : JEM-ARM200F NEOARM
Accelerating voltage : 200 kV
Image : TEM

Ezért amikor néhány nanométeres rétegről beszélünk, már **néhány vagy néhány tucat atomnyi méretskálán** dolgozunk.

És már nem is lapos a tranzisztor

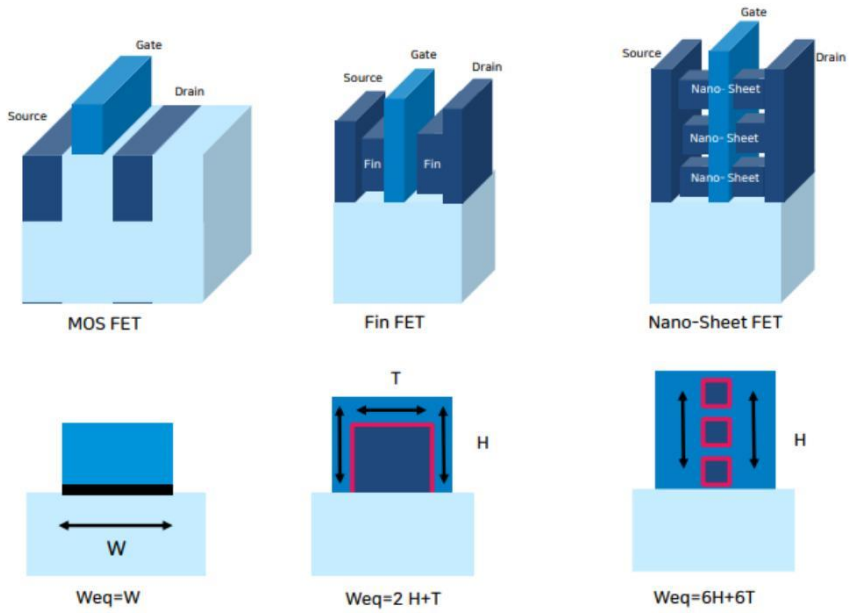
A régi MOSFET lényegében síkbeli szerkezet volt.

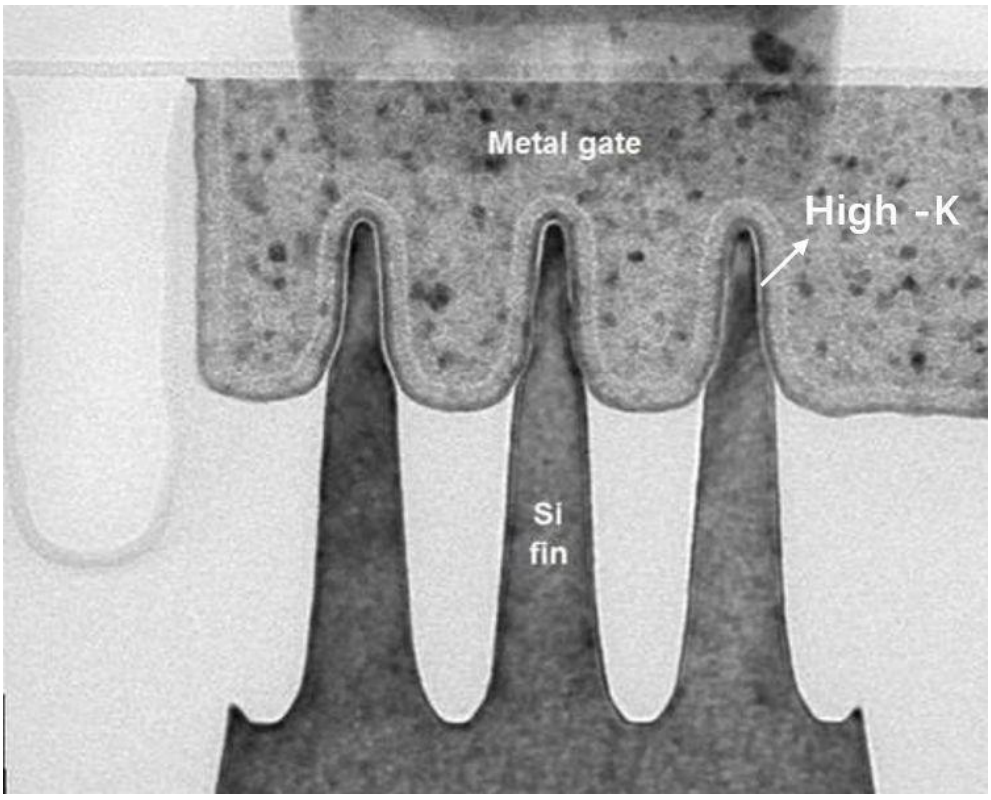
Ahogy azonban egyre kisebb lett, egy komoly probléma jelentkezett: a kapu egyre nehezebben tudta szabályozni az elektronok áramlását.

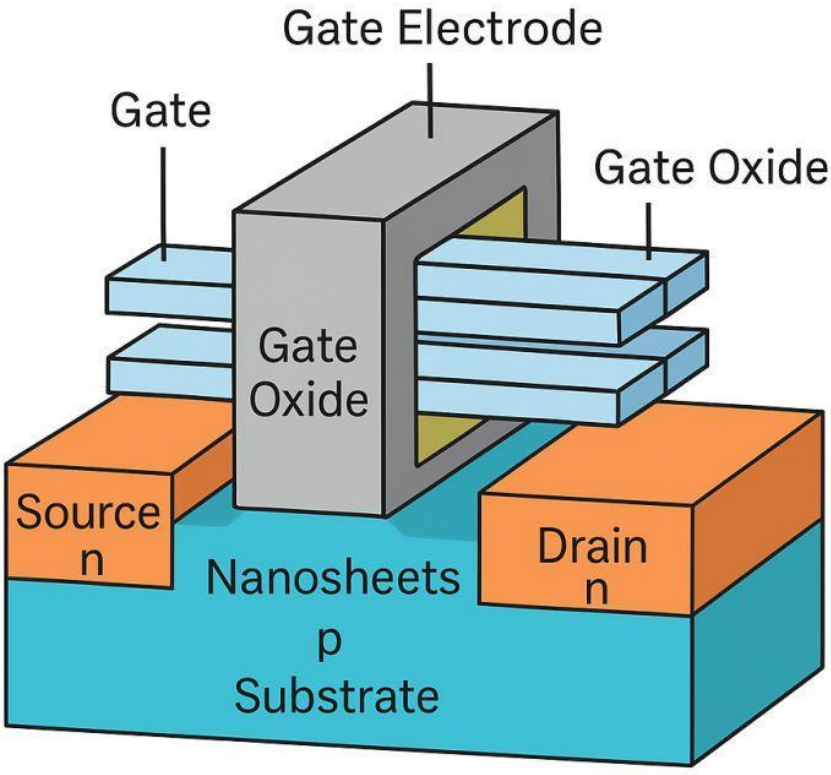
Ezért a tranzisztort elkezdték **háromdimenzióssá tenni**.

A FinFET-nél a vezető csatorna egy keskeny „uszonyként” emelkedik ki a lapka síkjából, a kapu pedig több oldalról körülveszi.

그림 15. Planar FET, MOSFET, GAAFET 구조 비교







Gate-all-Around FET

A következő lépés a **GAAFET – Gate-All-Around FET**, illetve annak nanosheet változata.

Itt a kapu már gyakorlatilag **minden oldalról körülveszi a csatornát.**

Ez jobb elektromos vezérlést tesz lehetővé ilyen parányi méreteknél.

A fejlődés tehát:

Planáris tranzisztor → FinFET → GAAFET/nanosheet

Nem egyszerűen kisebbre rajzoljuk ugyanazt a tranzisztort. **Magát a tranzisztor formáját is megváltoztatjuk.**

A tranzisztorok száma milliószorosára nőtt. Méretük pedig eljutott oda, ahol már az atomok mérete szab határt.

Több milliárd tranzisztor – egyszerre

Talán ez a gyártás legfontosabb gondolata.

A fotolitográfia nem egyenként készíti el a tranzisztorokat.

Amikor egy mintázatot levilágítanak, **óriási számú struktúra készül párhuzamosan.**

Ez ugyanaz az alapvető előny, amely Gutenberg nyomdájában is megjelent: ha egyszer elkészítettük a mintát, sok elemet reprodukálhatunk vele.

Csakhogy itt a „betűk” nanométeresek.

A chipgyártás ezért bizonyos értelemben: **atomi pontosságú nyomdatechnika.**

És mi történik, ha egyetlen tranzisztor hibás?

Ez is érdekes kérdés.

Egy több tízmilliárd tranzisztoros chipnél nem reális elvárás, hogy minden egyes legyártott lapka tökéletes legyen.

A wafereket ezért tesztelik, majd a lapkákat kivágják és osztályozzák.

Egyes hibák miatt a chip használhatatlan lesz. Máskor azonban csak egy processzormag vagy a cache egy része hibás.

Ha a konstrukció lehetővé teszi, ezt a részt letilthatják.

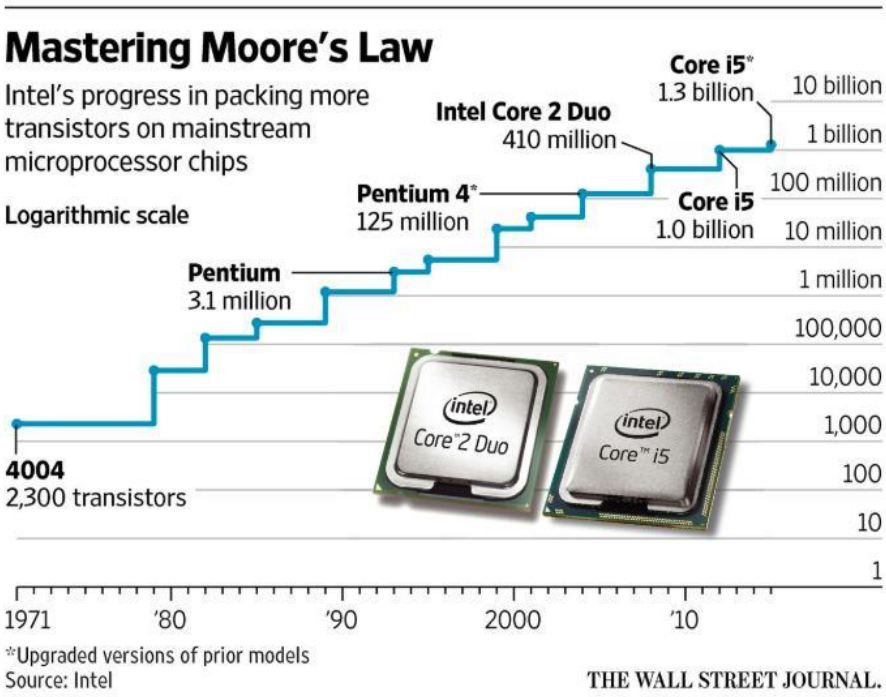
Így előfordulhat, hogy ugyanabból vagy nagyon hasonló fizikai lapkából különböző termékváltozatok készülnek:

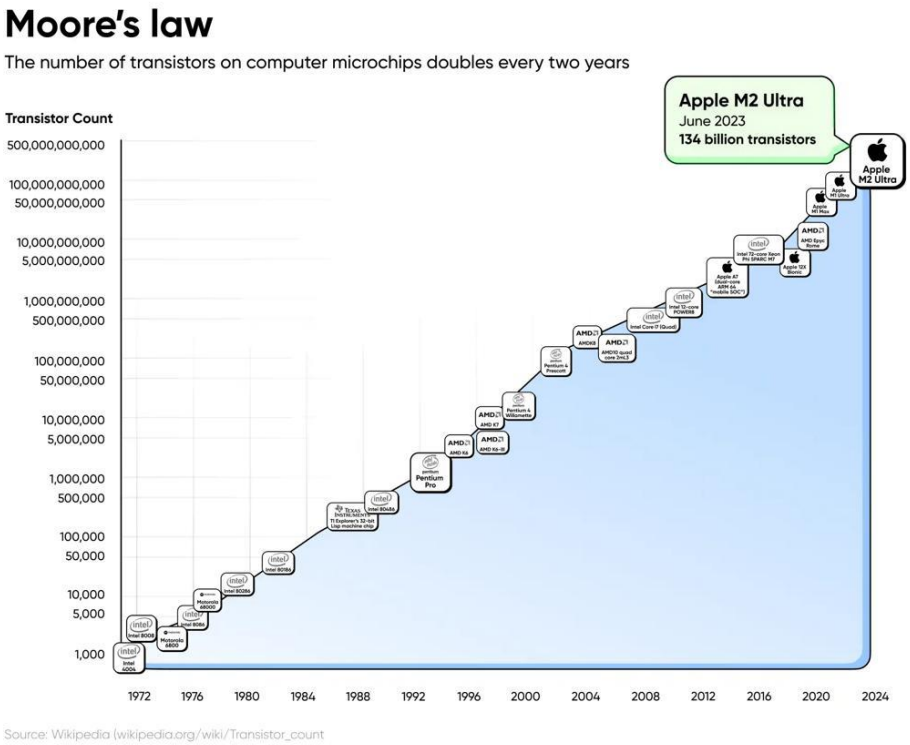
8 működő mag → magasabb kategória

6 működő mag → alacsonyabb kategória

Ez az úgynevezett **binning** tágabb gyártási és minősítési folyamatának egyik érdekes következménye.

Moore törvénye – 2300 tranzisztortól a tízmilliárdokig

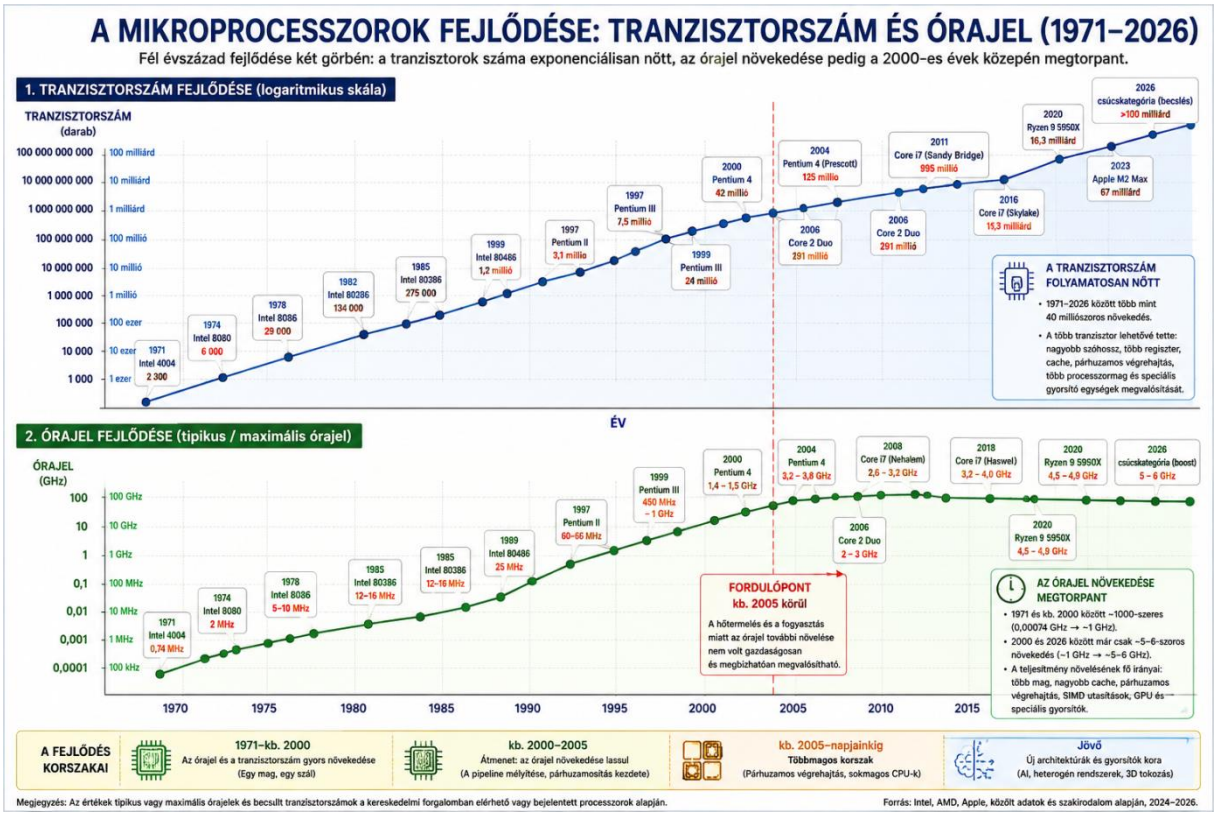




Ábra: *Moore törvénye a mikroprocesszorok történetében. Az Intel 4004 mintegy 2300 tranzisztorától fél évszázad alatt eljutottunk a több tízmilliárd tranzisztort tartalmazó processzorokig és rendszerchipekig. A függőleges tengelyt célszerű logaritmikus skálán ábrázolni.*

Intel 4004 (1971) – 0,74 MHz
Intel 8080 (1974) – 2 MHz
Intel 8086 (1978) – 5–10 MHz
Intel 80286 (1982) – kezdetben 6–8 MHz
Intel 80386 (1985) – kezdetben 12–16 MHz
Intel 80486 (1989) – 25 MHz
Pentium (1993) – 60–66 MHz
Pentium II (1997) – 233 MHz-től
Pentium III (1999) – 450 MHz-től, később 1 GHz fölé
Pentium 4 (2000) – 1,4–1,5 GHz-től, később 3,8 GHz
Core 2 (2006) – jellemzően 2–3 GHz
Core i7 (2008–) – néhány GHz
mai csúcskategóriás CPU-k – boost órajellel nagyjából 5–6 GHz körüli tartomány.

1971–kb. 2005: tranzisztorszám ↑ és órajel ↑↑
kb. 2005 után: tranzisztorszám tovább ↑, de az órajel növekedése erősen lelassul
ezután: több mag, nagyobb cache, párhuzamosítás, SIMD, GPU, speciális gyorsítók.



3. A memória

A számítógép nemcsak számol. **Emlékeznie is kell.** Meg kell őriznie az adatokat, amelyekkel éppen dolgozik, a program utasításait, a részeredményeket, sőt azt is, hogy a műveletsorban éppen hol tart.

A memória története ezért szinte ugyanolyan fontos része a számítógép történetének, mint a processzoré. Ráadásul talán még látványosabban mutatja a fejlődést: volt idő, amikor

egyetlen bit tárolásához szemmel látható alkatrész kellett. Ma egyetlen parányi chipben bitek milliárdjai férnek el.

3.1. Relék és elektroncsövek

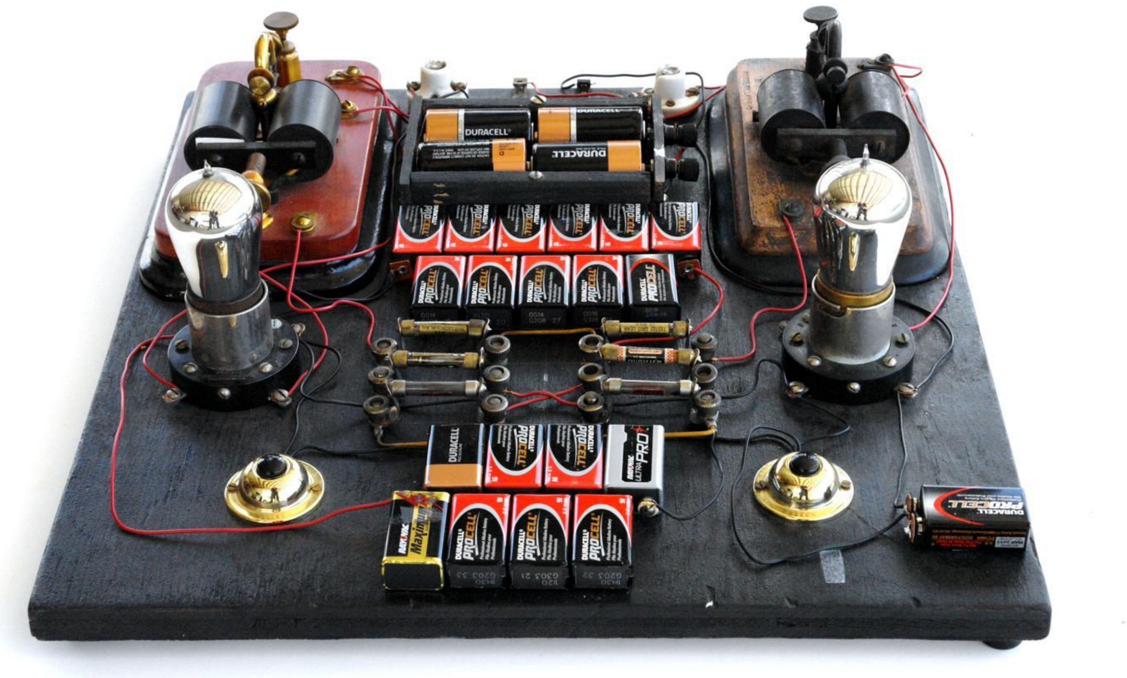
A legkorábbi számítógépekben ugyanazokat az elemeket használták tárolásra is, amelyekből a logikai áramkörök épültek.

A relé két stabil állapotot vehet fel, így elvileg egy bit tárolására is alkalmas. Elektromechanikus szerkezet lévén azonban viszonylag nagy és lassú volt.

Az elektroncső már sokkal gyorsabban működött. Megfelelően összekapcsolt elektroncsövekből **bistabil áramkört**, flip-flopot lehetett készíteni, amely két stabil állapot valamelyikében maradt. Ezzel egy bitet tárolhatott.

Csakhogy egyetlen bithez több elektroncső és további alkatrészek kellettek. Néhány ezer bit tárolása már hatalmas, sok energiát fogyasztó és sok hőt termelő berendezést eredményezett.

A számítógép építői ezért hamarosan más megoldások után kezdtek kutatni.



Ábra: *Az első számítógépekben relékből vagy elektroncsövekből kialakított bistabil áramkörök is tárolhatták a biteket. Megbízható működésükért azonban nagy mérettel és jelentős energiafogyasztással kellett fizetni.*

3.2. Késleltetővonal

Az első elektronikus memóriák egyik legötletesebb megoldása tulajdonképpen nem is „tárolta” az adatot a szó megszokott értelmében.

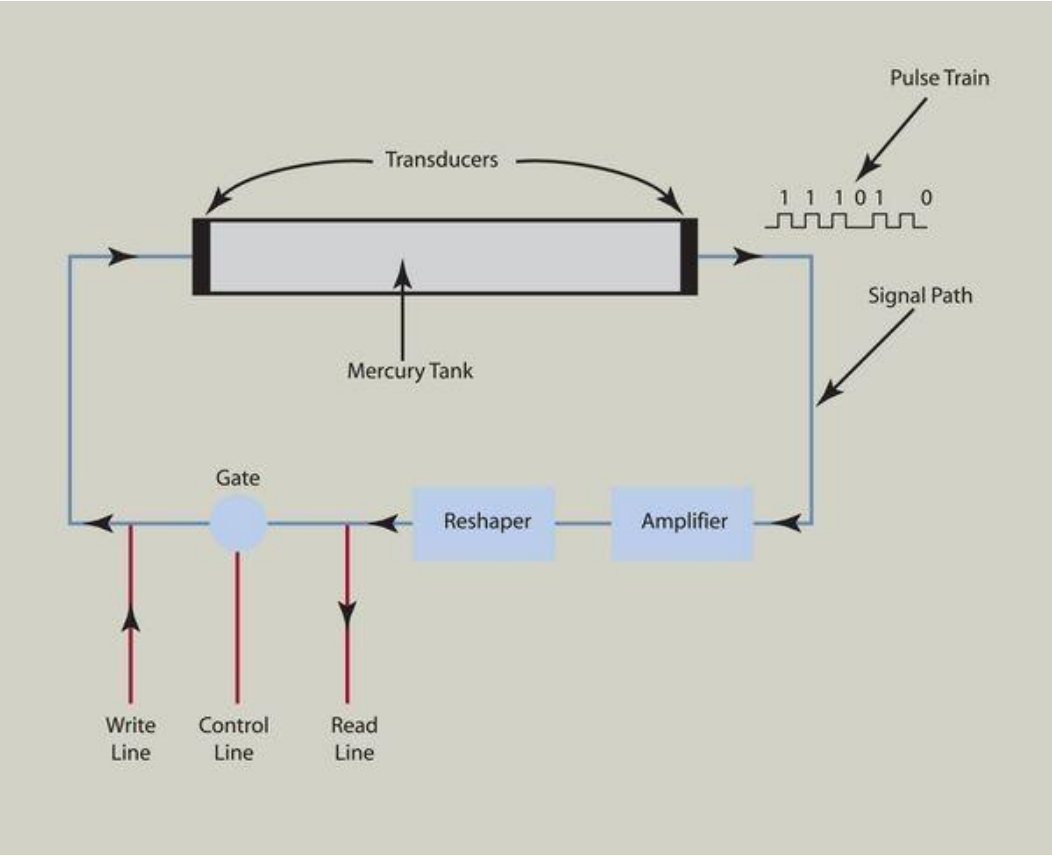
A **késleltetővonalas memória** folyamatosan mozgásban tartotta.

Egy csőben – gyakran higanyban – ultrahangimpulzusokat indítottak el. Az impulzusoknak időre volt szükségük, hogy elérjék a cső másik végét. Ott elektromos jellé alakították őket, felerősítették, majd ismét visszaküldték a cső elejére.

A bitek tehát körbe-körbe jártak.

A módszer egyik nagy hátránya az volt, hogy az adatokat nem lehetett tetszőleges sorrendben azonnal elérni. Meg kellett várni, amíg a kívánt bit „odaér”.

Ezért nevezzük **soros hozzáférésű memóriának.**



Ábra: *Késleltetővonalas memória. Az információ hangimpulzusok formájában haladt végig a közegen, majd a vonal végén visszacsatolták. Az adat tehát szó szerint állandó mozgásban volt.*

3.3. Williams-cső

A másik korai megoldás a televíziózásból ismert katódsugárcsövet használta memóriaként.

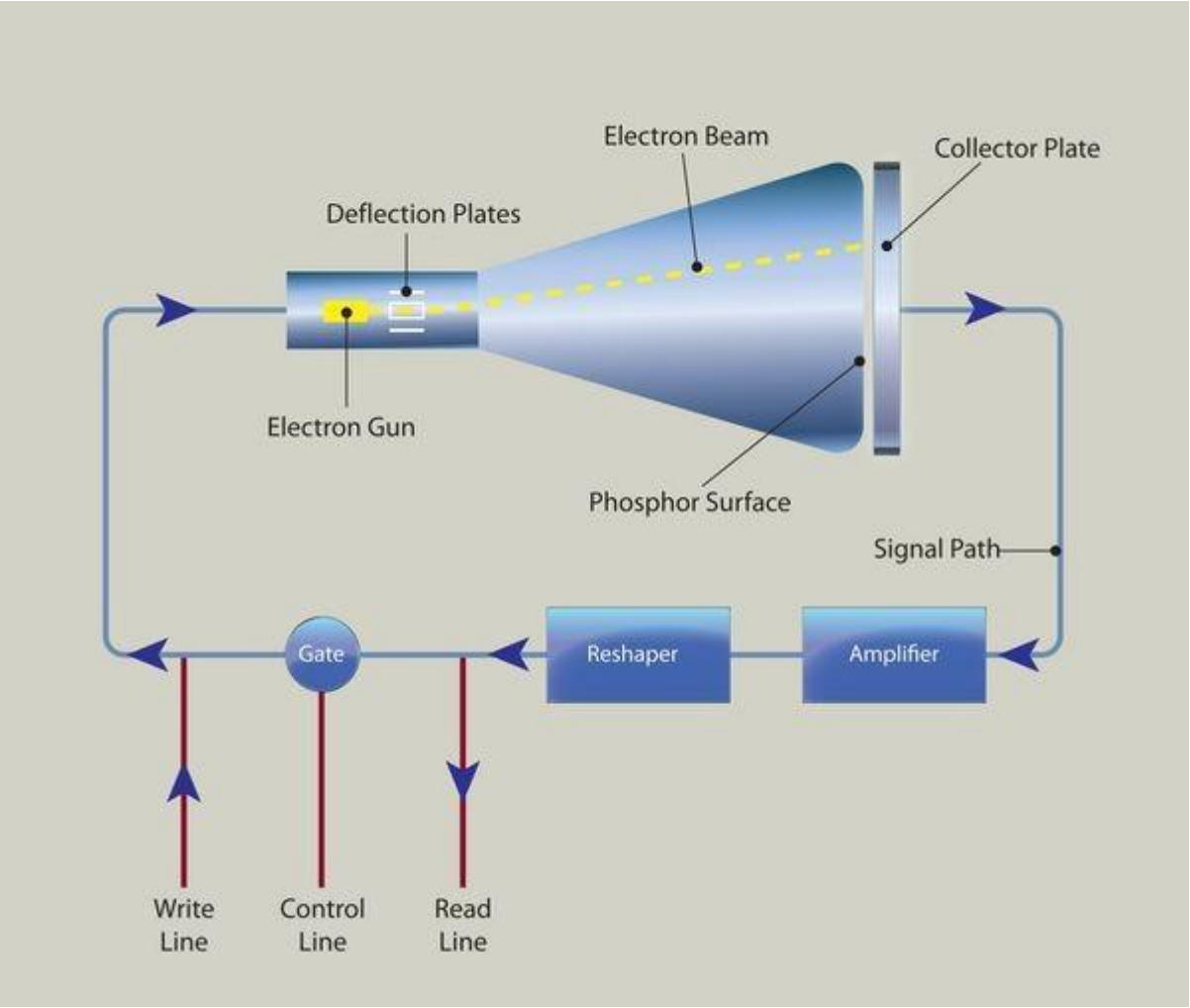
A **Williams–Kilburn-cső**, röviden Williams-cső működése azon alapult, hogy az elektronsugár becsapódása a képernyő felületén rövid ideig megmaradó elektromos töltésmintázatot hoz létre.

Ezek a töltések jelentették a biteket.

A nagy előrelépést az jelentette, hogy az adatokhoz már **közvetlenül hozzá lehetett férni**. Nem kellett megvárni, hogy a keresett adat végighaladjon egy késleltetővonalon.

A Williams-cső azonban kényes szerkezet volt. A tárolt információt rendszeresen frissíteni kellett, és a rendszer érzékeny volt az elektromos zavarokra.

Mégis fontos lépcsőfok lett a valódi, elektronikus, véletlen hozzáférésű memória felé.



Ábra: *Williams–Kilburn-cső. A katódsugárcső képernyőjén létrehozott elektromos töltésmintázatok tárolták a biteket. Az egyik első valódi véletlen hozzáférésű elektronikus memória volt.*

3.4. Ferritmemória

A korai elektronikus memóriák problémáira az 1950-es évektől egy egészen más technológia adott hosszú időre megoldást: a **ferritgyűrűs memória**, angolul *magnetic-core memory*.

A biteket apró, mágneses ferritből készült gyűrűk tárolták.

Minden egyes gyűrű **egy bitnek felelt meg**.

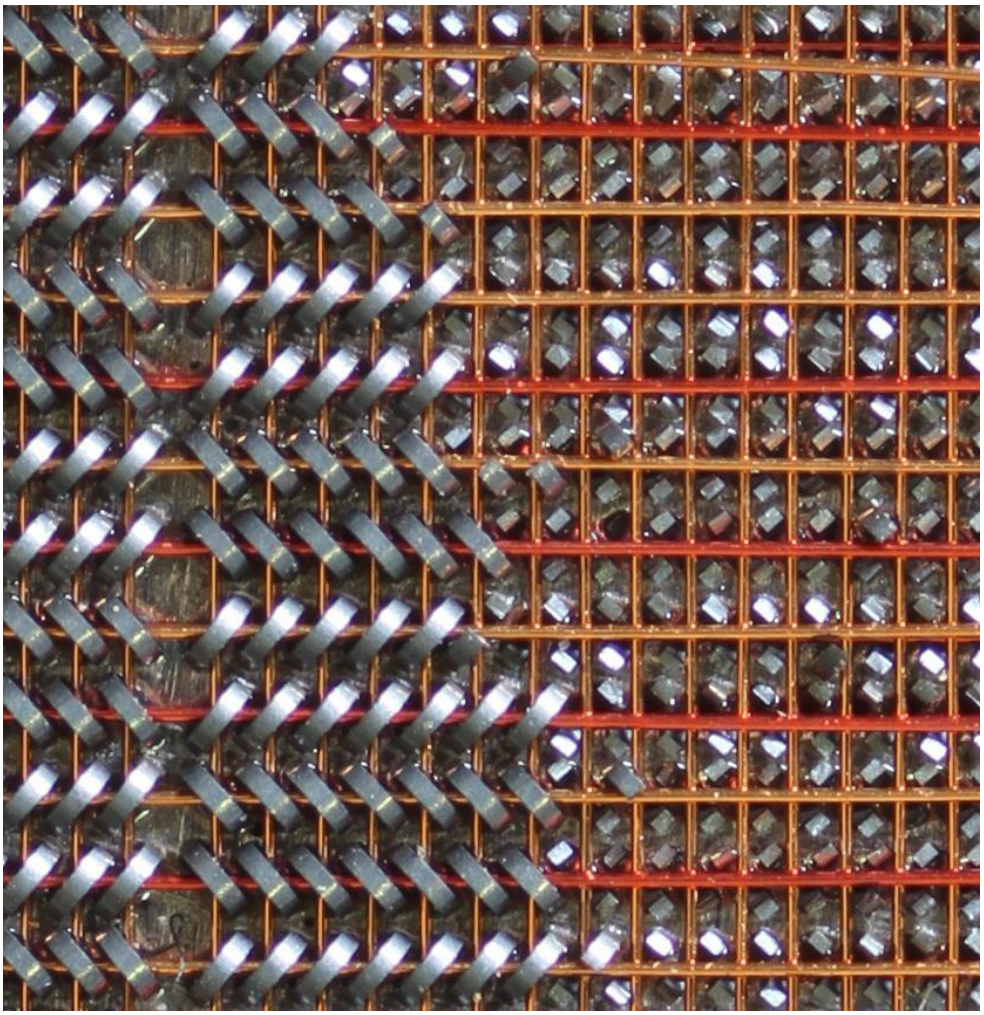
A gyűrű mágnesezettségének egyik iránya jelenthette a 0-t, az ellenkező irány pedig az 1-et. A gyűrűkön vékony vezetékeket fűztek keresztül, amelyek segítségével kiválaszthatták, írni vagy olvasni lehetett az adott bitet.

A ferritmemória nagy előnye volt, hogy az információt az áram kikapcsolása után is megőrizte. Gyors, megbízható és az akkori technológiához képest rendkívül tartós volt.

Az 1950-es évektől az 1970-es évek elejéig a számítógépek egyik meghatározó memóriatechnológiájává vált.

És különleges tulajdonsága miatt ma is lenyűgöző:

a biteket szabad szemmel látni lehet.



Ábra: *Ferritgyűrűs memória közelről. Minden apró gyűrű egyetlen bitet tárolt. A vezetékeket kézzel vagy részben automatizált eljárásokkal fűzték át a gyűrűkön, így a nagyobb memóriák elkészítése rendkívül munkaigényes volt.*

3.5. RAM

A **RAM – Random Access Memory**, vagyis véletlen hozzáférésű memória elnevezés nem egyetlen technológiát jelent.

A név arra utal, hogy a memória bármely címe közvetlenül elérhető, nagyjából azonos hozzáférési idővel.

Ez alapvetően különbözik például a késleltetővonaltól vagy a mágnesszalagtól, ahol az adat helyzete meghatározza, mennyi idő alatt érhető el.

A félvezető-technika fejlődésével a memóriafunkciókat integrált áramkörökbe lehetett költöztetni.

Ez döntő változást hozott.

A ferritgyűrűket és vezetékhálózatukat apró szilíciumlapka váltotta fel. A memóriakapacitás gyorsan növekedett, miközben a méret és az egy bitre jutó ár drámaian csökkent.

A félvezető RAM fokozatosan kiszorította a ferritmemóriát.

3.6. SRAM és DRAM

A félvezető RAM két alapvető típusa az **SRAM** és a **DRAM**.

Az SRAM – *Static Random Access Memory* – egy bitet több tranzisztorból álló bistabil áramkörben tárol. Nagyon gyors, és amíg áramot kap, nem igényli az adatok rendszeres frissítését.

Hátránya, hogy egy bit tárolásához viszonylag sok tranzisztor szükséges, ezért drágább és nagyobb helyet foglal.

A **DRAM – Dynamic Random Access Memory** ezzel szemben egy bitet tipikusan egy tranzisztor és egy parányi kondenzátor segítségével tárol.

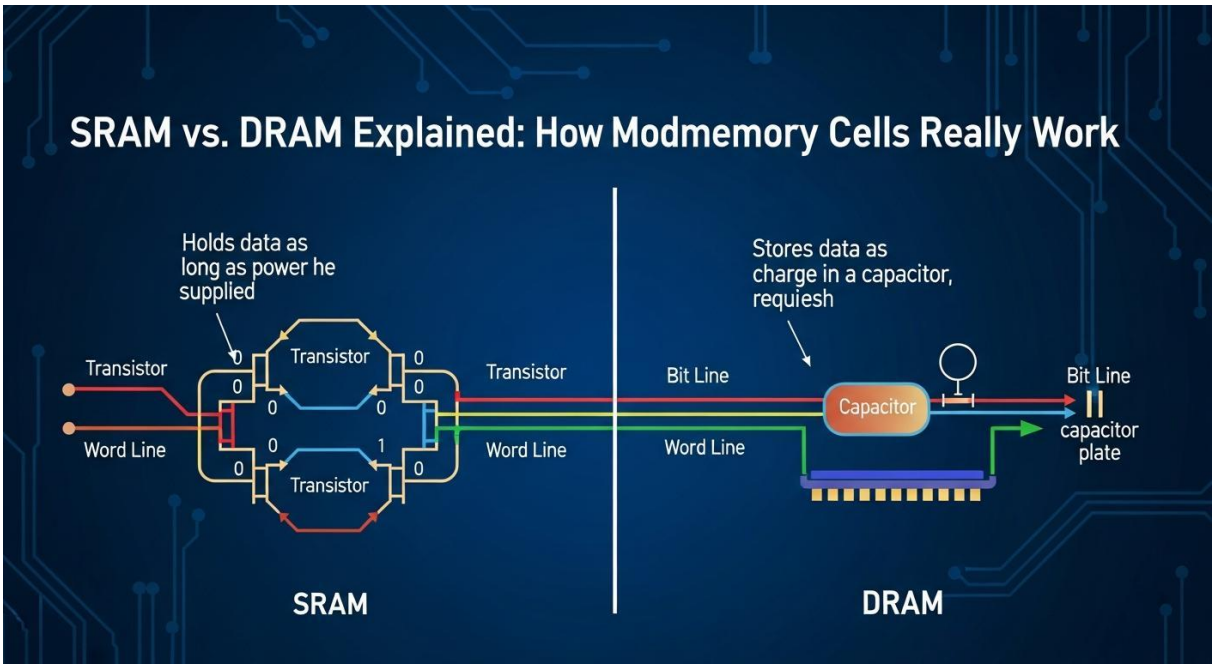
A kondenzátor töltése azonban fokozatosan elszivárog. Ezért a DRAM tartalmát rendszeresen **frissíteni kell.**

Ez első hallásra hátránynak tűnik, de egy DRAM-cella sokkal kisebb lehet, ezért hatalmas mennyiségű memória építhető gazdaságosan.

Így alakult ki a munkamegosztás:

SRAM → nagyon gyors, drága → elsősorban cache

DRAM → lassabb, olcsóbb, nagy kapacitású → főmemória



Ábra: *SRAM és DRAM memóriacella elvi összehasonlítása. Az SRAM több tranzisztor árán nagy sebességet kínál, míg a DRAM egyszerűbb cellája sokkal nagyobb memóriakapacitást tesz lehetővé.*

3.7. ROM, PROM, EPROM, EEPROM

Nem minden memóriának kell folyamatosan változnia.

A számítógép bekapcsolásakor például szükség van olyan programra, amely már akkor is rendelkezésre áll, amikor az operációs rendszer még el sem indult.

Erre szolgált a **ROM – Read Only Memory**, vagyis csak olvasható memória.

A ROM tartalmát eredetileg már a gyártás során meghatározták.

Később megjelent a **PROM**, amelyet a felhasználó egyszer programozhatott be.

Az **EPROM** már törölhető és újraprogramozható volt. Jellegzetes tokján egy kvarcüveg ablak található. Ezen keresztül ultraibolya fénnyel lehetett törölni a memória teljes tartalmát.

A következő lépés az **EEPROM** lett, amelyet már elektromosan lehetett törölni és újraírni.

A fejlődés jól kirajzolódik:

ROM → egyszer rögzített

PROM → egyszer programozható

EPROM → UV-fénnyel törölhető

EEPROM → elektromosan törölhető és újraírható



Ábra: *EPROM memória jellegzetes kvarcüveg ablakkal. A chip tartalmát erős ultraibolya fénnyel lehetett törölni, majd új programot lehetett beleírni. Az ablakot használat közben rendszerint címkével takarták le.*

3.8. Flash

Az EEPROM továbbfejlesztéséből született meg a **flash memória.**

A flash szintén elektromosan írható és törölhető, de felépítése lehetővé teszi nagy mennyiségű adat gazdaságos tárolását.

Ráadásul **nem felejt az áram kikapcsolásakor.**

Ez tette alkalmassá arra, hogy rengeteg korábbi adattárolási technológiát kiváltson.

Flash memória található a pendrive-ban, memóriakártyában, SSD-ben, telefonban, fényképezőgépben és számtalan beágyazott elektronikus eszközben.

A flash különösen jól példázza, hogy a memória és a háttértár közötti egykori éles technológiai határ részben elmosódott.

A számítógép munkamemóriája továbbra is jellemzően DRAM, miközben az SSD ugyanúgy félvezetőket használ – de az adatokat kikapcsolás után is megőrzi.



Ábra: *Flash memória. Ugyanennek a technológiai családnak a leszármazottai őrzik adatainkat a memóriakártyákban, pendrive-okban és SSD-kben.*

3.9. Cache

A cache-sel már a processzoroknál találkoztunk, de a memória történetében is fontos helye van.

A számítógép különböző memóriái ugyanis érdekes kompromisszumot alkotnak:

minél gyorsabb egy memória, általában annál drágább és kisebb; minél nagyobb és olcsóbb, annál lassabb.

Ezért alakult ki a **memóriahierarchia**.

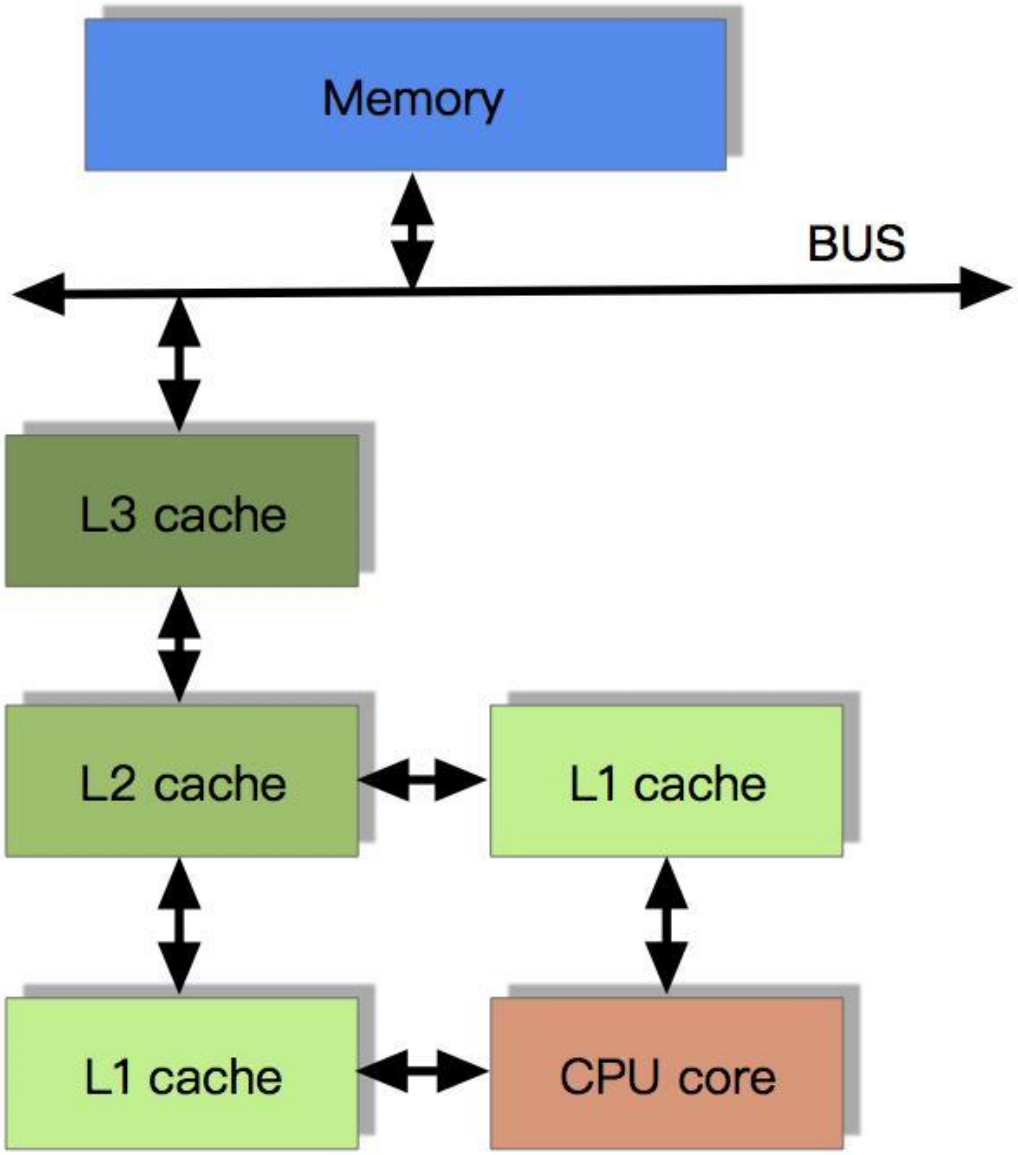
A processzorhoz legközelebb vannak a regiszterek. Ezeket követi az L1, L2 és L3 cache, majd a DRAM főmemória, végül pedig a háttértár.

Szemléletesen:

Regiszter → L1 → L2 → L3 → RAM → SSD

Balról jobbra haladva általában nő a kapacitás és csökken az egy bitre jutó költség – ugyanakkor nő az elérési idő is.

A modern számítógép egyik legfontosabb trükkje éppen az, hogy megpróbálja **a várhatóan szükséges adatot még azelőtt a gyorsabb memória közelébe hozni, hogy a processzor kérné**.



Ábra: *A memóriahierarchia. A processzorhoz közeli memóriák rendkívül gyorsak, de kicsik és drágák. Távolodva egyre nagyobb, olcsóbb, ugyanakkor lassabb tárolókat találunk.*

Mekkora volt egy bit? – a ferritmemória

Ma egyetlen bit olyan kicsi, hogy szabad szemmel természetesen nem láthatjuk.

A ferritmemória korszakában azonban egészen más volt a helyzet.

Egy bitnek saját, kézzelfogható mágneses gyűrűje volt.

A korai ferritmagok átmérője akár milliméteres nagyságrendű is lehetett, a technológia fejlődésével pedig egyre kisebb gyűrűket készítettek. A memóriasíkra pillantva ezért szó szerint **láthatjuk az egyes biteket**.

Egy byte nyolc bitből áll.

Vagyis egyetlen byte tárolásához nyolc ferritgyűrű kellett.

Egy kilobyte:

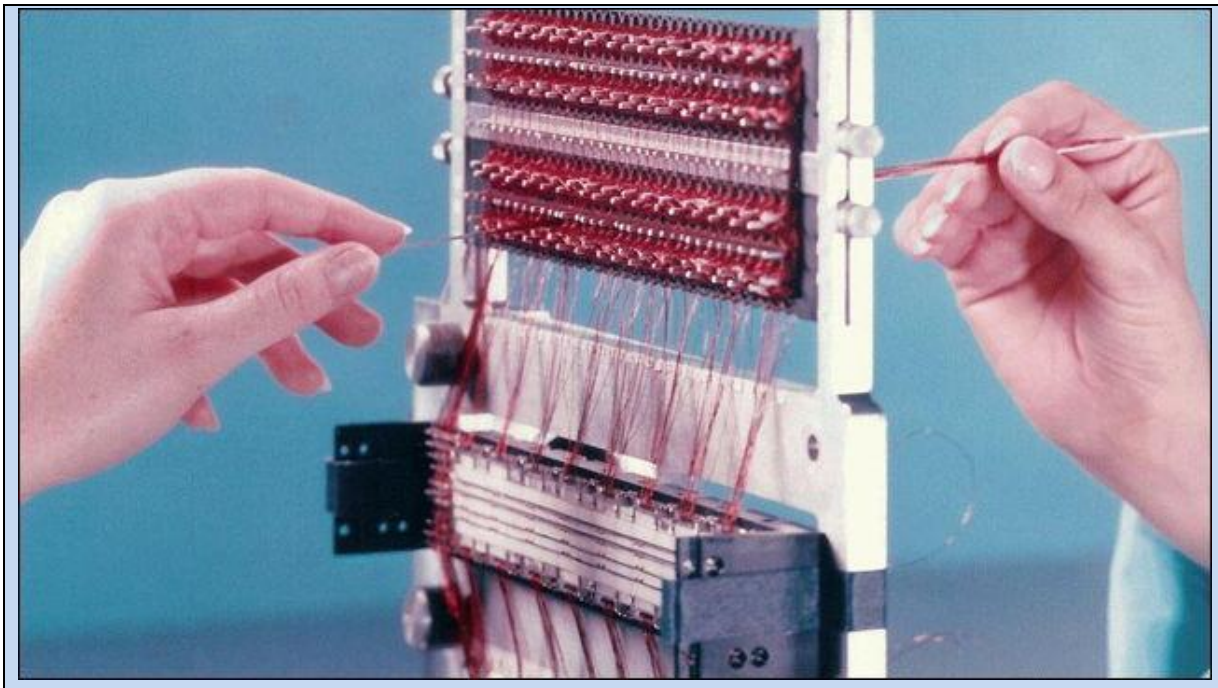
8192 bit → 8192 gyűrű.

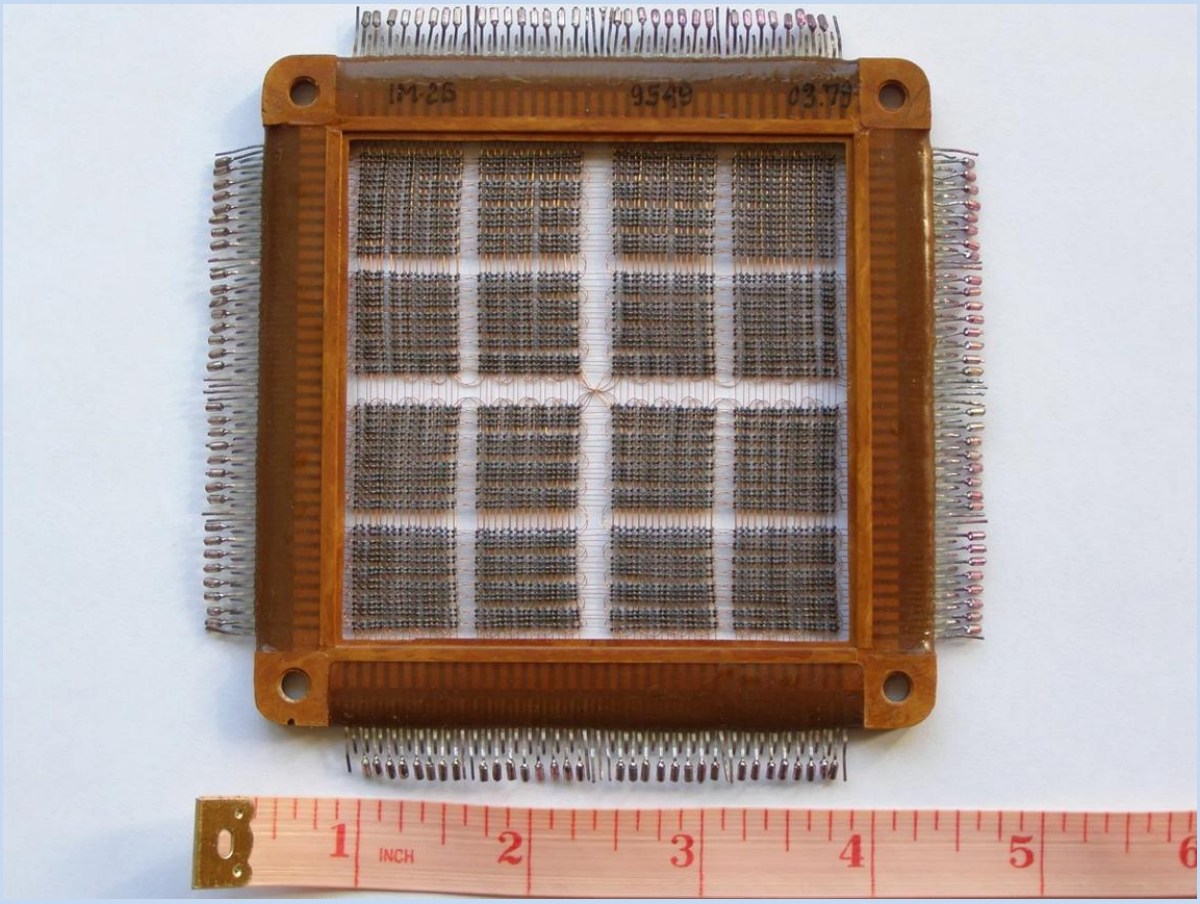
Egy megabyte pedig:

8 388 608 bit.

- 1 kB = 1000 bájt;
- 1 KiB = 1024 bájt = 8192 bit;
- 1 MiB = 1 048 576 bájt = 8 388 608 bit.

Ha egyetlen ferritgyűrűvel akarnánk minden bitet tárolni, több mint **nyolcmillió apró mágneses magra** lenne szükségünk.





Ábra: *Amikor még látni lehetett a biteket. A ferritmemória minden apró mágneses gyűrűje egyetlen bitet tárolt. Talán nincs még egy számítógép-alkatrész, amely ilyen szemléletesen mutatná meg a memória miniatürizálásának elképesztő mértékét.*

A gyűrűkön több hajszálvékony vezetéket kellett átvezetni. A memóriasíkok gyártása ezért különleges precizitást igényelt, és jelentős részben kézi munkával történt.

A ferritmemória egyik legérdekesebb tulajdonsága az volt, hogy **nem felejtett az áram kikapcsolásakor**. A mágneses állapot megmaradt.

Ez néha egészen különös történelmi dokumentumokat hagyott ránk: régi számítógépek ferritmemóriájából évtizedekkel később is kiolvasható lehetett az utoljára tárolt információ.

A fejlődés nagyságrendjét pedig talán semmi sem érzékelteti jobban ennél:

A ferritmemóriában egy bit még egy megfogható tárgy volt. Ma egy körömnyi félvezetőlapkán bitek milliárdjai férnek el.

4. Az adattárolás

A memória és az adattárolás között alapvető különbség van. A memória elsősorban azt őrzi, **amivel a számítógép éppen dolgozik**. Az adattároló feladata ezzel szemben az, hogy az információt akkor is megőrizze, amikor a gépet kikapcsoljuk.

Az adattárolás története különösen változatos. Az adatokat lyukak, mágneses állapotok, mikroszkopikus gödröcskék, elektromos töltések, sőt akár **DNS-molekulák** is őrizhetik.

A technológiák változtak, a cél ugyanaz maradt:

minél több információt minél kisebb helyen, minél gyorsabban, olcsóbban és hosszabb ideig megőrizni.

4.1. Lyukkártya

A lyukkártyával már találkoztunk a számítástechnika történetében. Jacquard szövőszékétől Hollerith népszámlálási gépein át vezetett az út a számítógépekig.

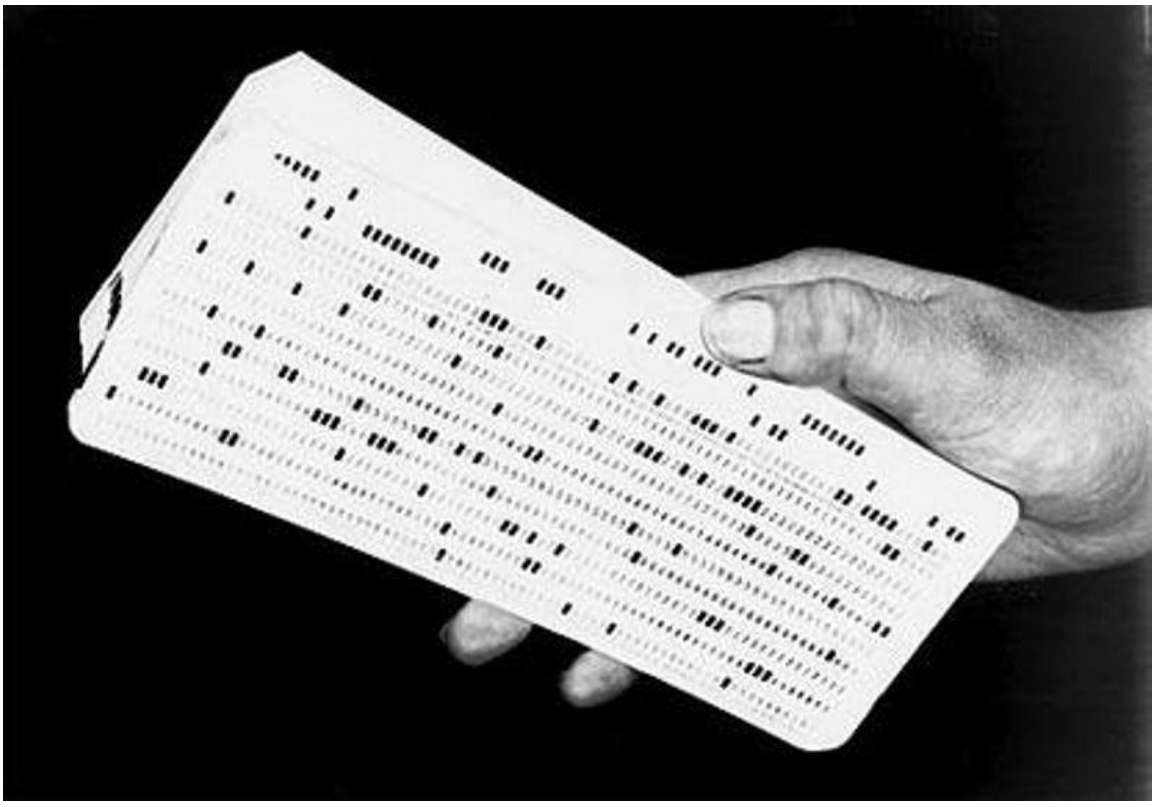
A lyukkártya működési elve rendkívül egyszerű: a kártya meghatározott helyein **van lyuk vagy nincs lyuk**. Ez a két állapot alkalmas információ kódolására.

A számítógépes korszak legismertebb IBM-kártyája 80 oszlopot tartalmazott. Egy kártyára jellemzően egy programsor vagy 80 karakter adat került.

A programozó elkészítette a kártyacsomagot, átadta a számítóközpontnak, majd várta az eredményt. Egy elejtett, összekevert kártyacsomag pedig valódi katasztrófa lehetett.

A lyukkártya ugyanakkor meglepően tartós adathordozó volt. Nem igényelt energiát, tartalma szemrevételezhető volt, és megfelelő körülmények között évtizedekig megőrizhette az adatot.





Ábra:
80 oszlopos lyukkártyák. Évtizedeken keresztül programok és adatok millióit tárolták ilyen kartonlapokon. A lyuk jelenléte vagy hiánya hordozta az információt.

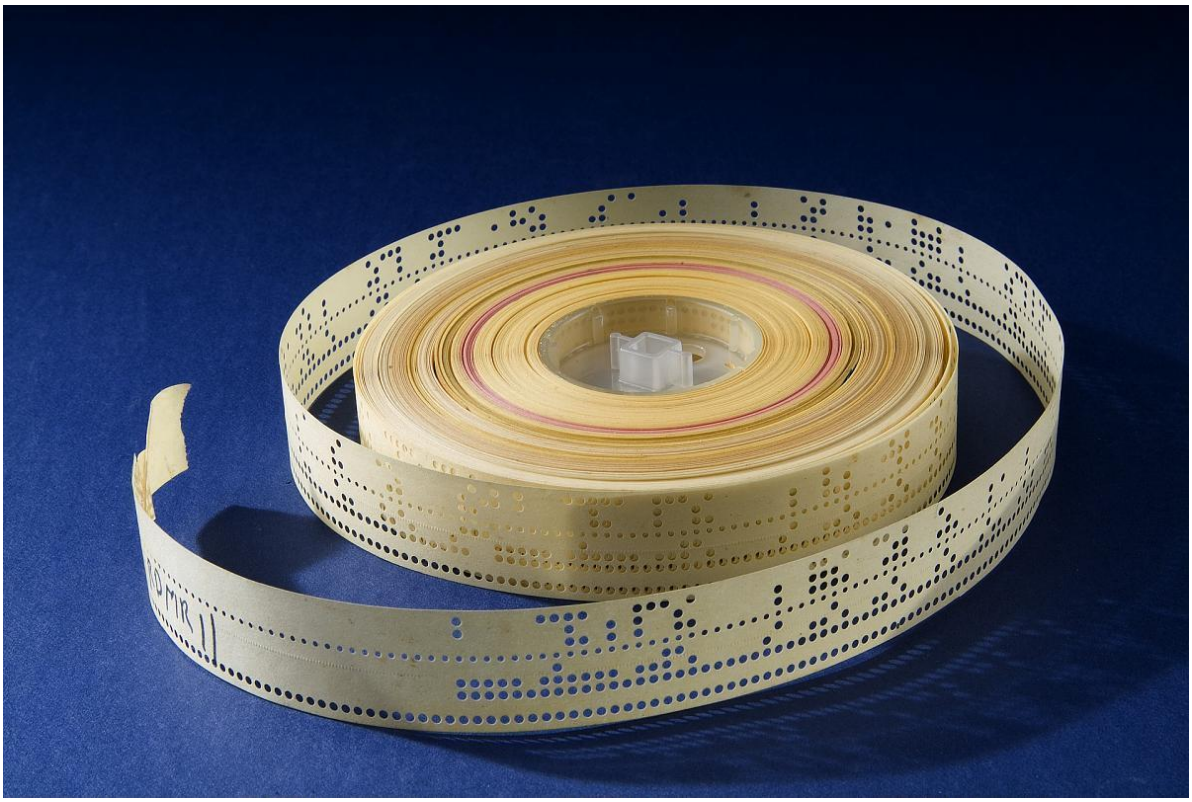
4.2. Lyukszalag

A lyukszalag ugyanazt az alapelvet használta, mint a lyukkártya, de az információ nem különálló lapokon, hanem hosszú papírszalagon sorakozott.

Előnye az egyszerűség és az olcsóság volt. Távírók, telexgépek, számítógépek és numerikus vezérlésű szerszámgépek egyaránt használták.

A lyukszalag ráadásul bizonyos értelemben önmagát dokumentálta: ha ismertük a kódolást, megfelelő olvasóval – szükség esetén akár mechanikus eszközzel – vissza lehetett nyerni az adatokat.

Hátránya a kis kapacitás, a lassúság és a szekvenciális hozzáférés volt.



Ábra:
Lyukszalag. A lyukak sorozata kódolta az adatokat. Egyszerűsége miatt a számítástechnika mellett a távírótechnikában és az ipari gépek vezérlésében is sokáig használták.

4.3. Mágnesdob

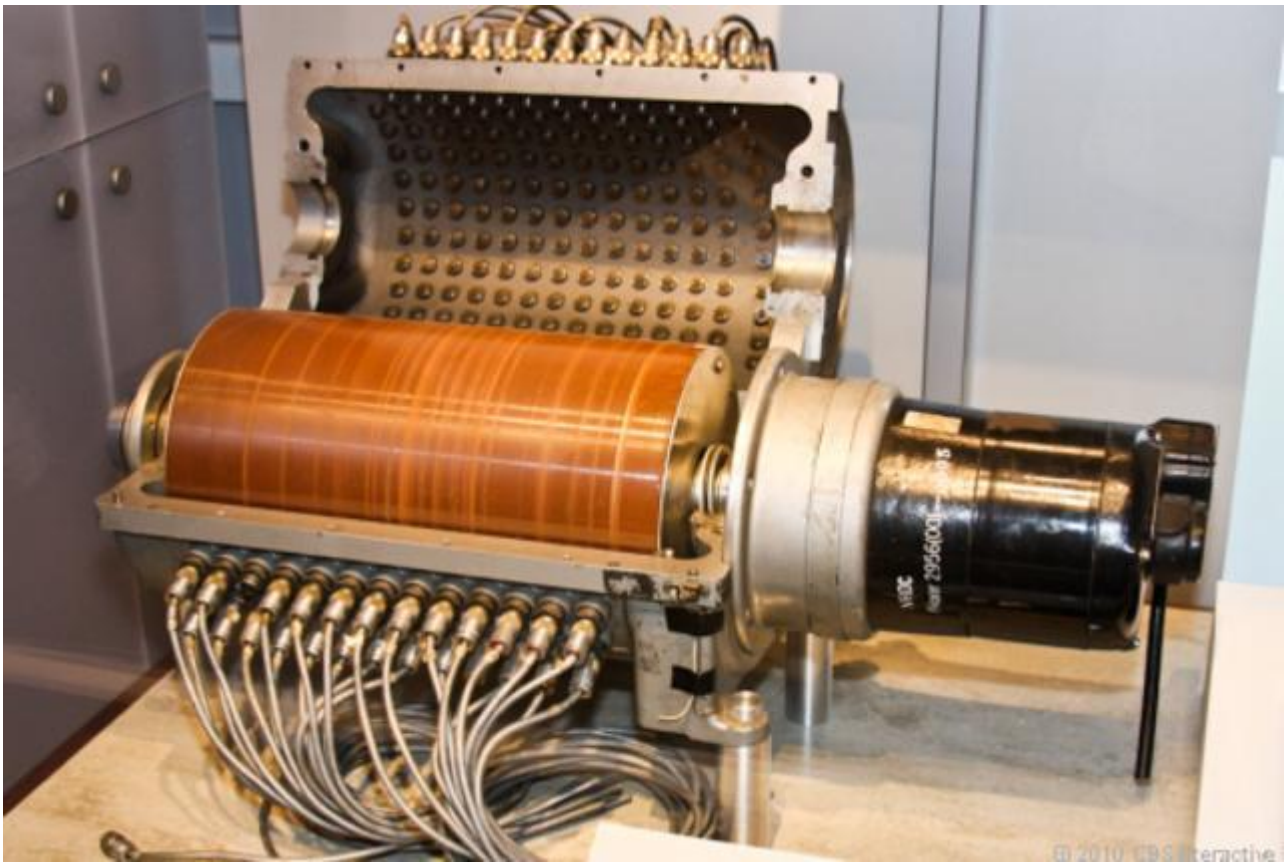
A mágnesdob már az elektronikus számítógépek korszakának terméke.

Egy forgó fémhenger felületét mágneses anyaggal vonták be. A henger mellett elhelyezett író-olvasó fejek mágneses állapotok formájában rögzítették és olvasták az adatokat.

A dob folyamatosan forgott, ezért az adat eléréséhez meg kellett várni, amíg a megfelelő terület az olvasófejhez ér.

A mágnesdobokat nemcsak háttértárként, hanem egyes korai számítógépek **főmemóriájaként** is használták.

A programozók még azt is figyelembe vehették, hogy a következő utasítás a dob melyik pontján helyezkedjen el, hogy mire szükség lesz rá, éppen az olvasófejhez érkezzen.



Ábra:
Mágnesdob. A forgó henger mágneses felületén tárolták az adatokat. A merevlemez egyik fontos technikai előfutárának tekinthető.

4.4. Mágnesszalag

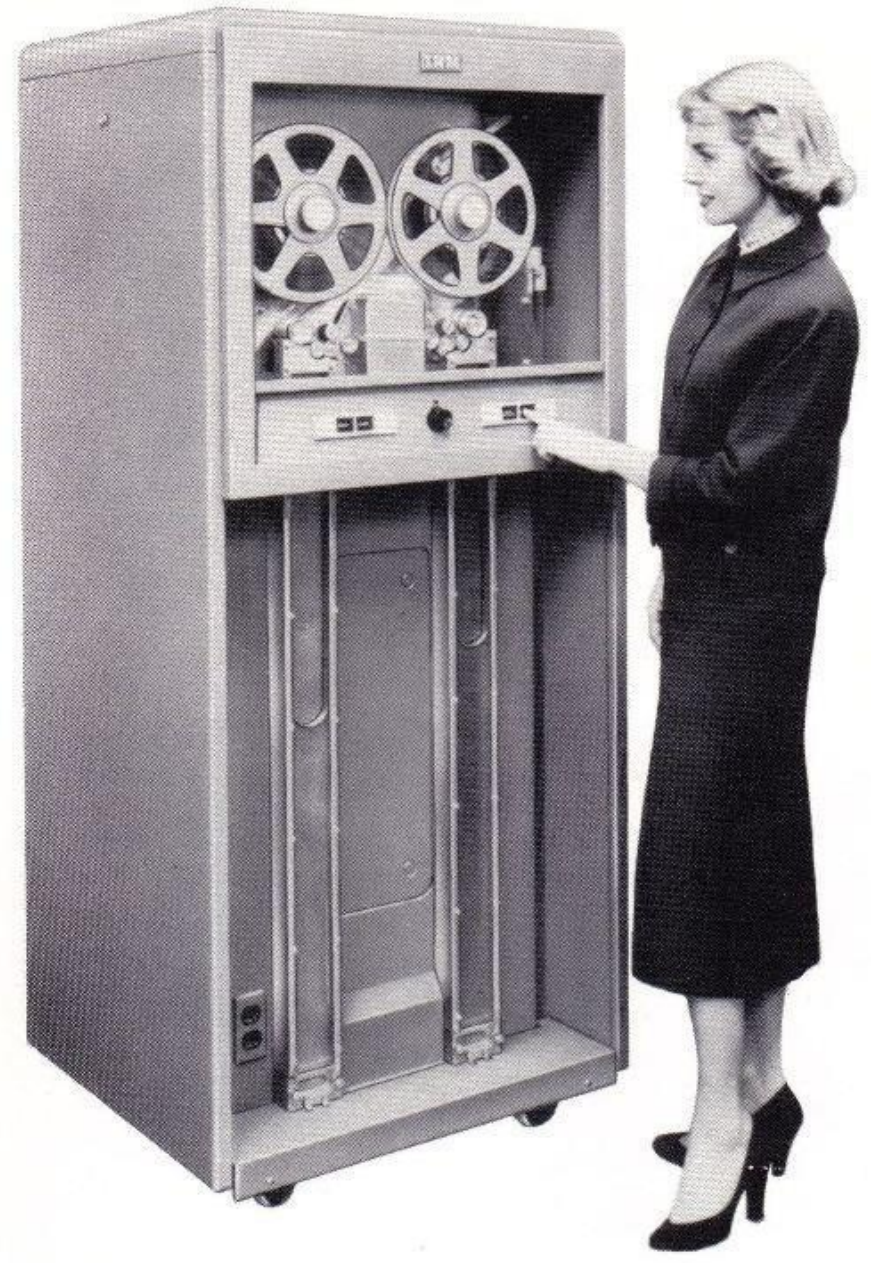
A mágnesszalag működési elve hasonló a hangrögzítésnél alkalmazott magnetofonszalagéhoz. A műanyag hordozórétegre felvitt mágneses bevonat őrzi az információt.

Legfontosabb tulajdonsága, hogy **szekvenciális adathordozó**. Ha a szalag végén lévő adatot szeretnénk elérni, előbb oda kell tekerni.

Ezért nem alkalmas arra, hogy a számítógép folyamatosan véletlenszerűen olvasson róla fájlokat. Hatalmas adatmennyiségek archiválására és biztonsági mentésére viszont ma is kiváló.

A mágnesszalag tehát nem tűnt el.

A modern szalagos rendszerek kazettái már **terabájtos nagyságrendű** adatot tárolhatnak, és nagy adatközpontokban ma is használják őket.





Ábra: *A mágnesszalag a számítástechnika egyik meglepően hosszú életű adathordozója. A hatalmas orsókat kazetták váltották fel, de az alapelv több mint fél évszázad után is használatban van.*

4.5. Merevlemez

1956-ban az IBM bemutatta a RAMAC rendszert, amelynek 350-es lemezegysége az első kereskedelmi merevlemezes adattárolók egyike, és a merevlemez történetének jelképes kiindulópontja.

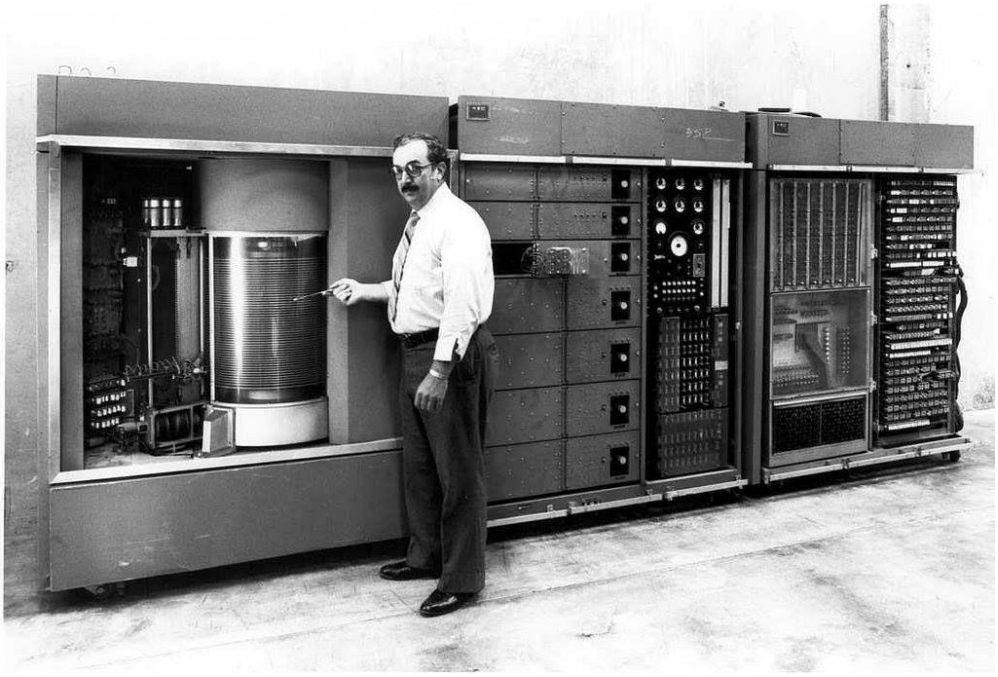
A berendezés **5 millió karakter**, nagyjából 5 MB adat tárolására volt képes. Ehhez ötven, 24 hüvelykes mágneses lemezt használt, és maga az egység egy nagy szekrény méretét érte el.

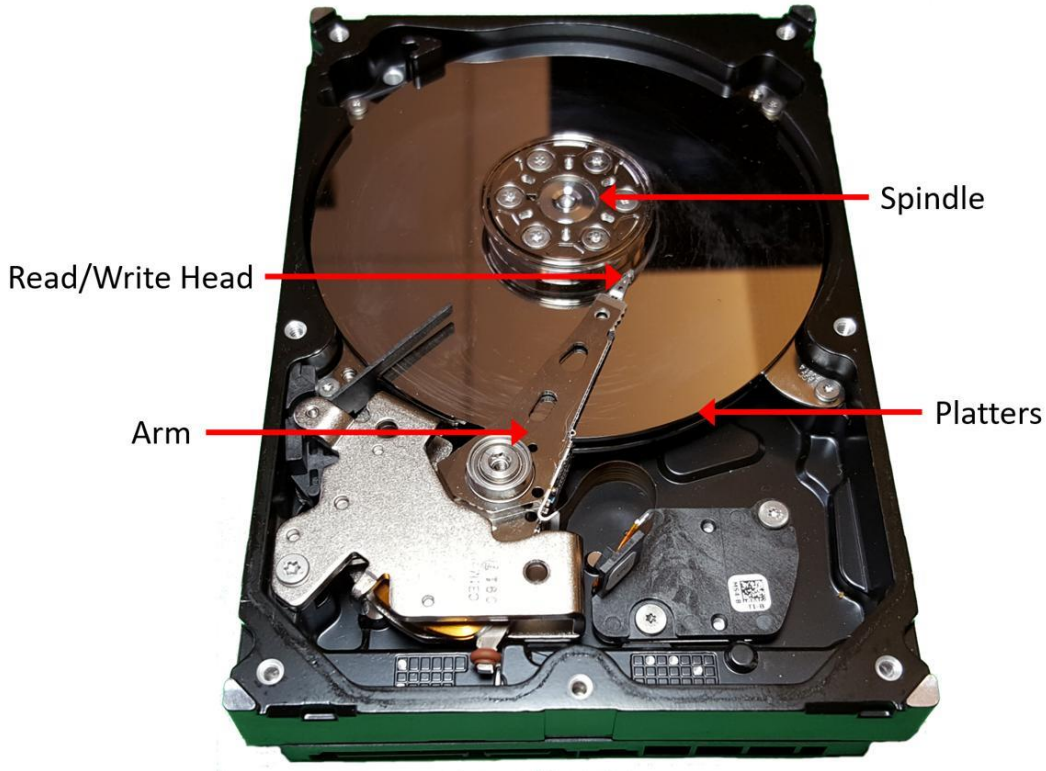
A merevlemez alapelve évtizedeken át lényegében változatlan maradt.

Gyorsan forgó, mágneses bevonatú lemezek fölött parányi távolságban mozog az író-olvasó fej. Az adatokat koncentrikus sávokban tárolják.

A fejlődés azonban elképesztő volt.

A szekrényméretű, néhány megabájtos berendezésből tenyérnyi eszköz lett, amely **több terabájtot** képes tárolni.





Ábra: *Az IBM 350 lemezegységtől a modern merevlemezig. Az első rendszer néhány megabájtjához egy szekrénynyi berendezés kellett; ma ugyanennek sokmilliószorosa elfér egyetlen meghajtóban.*

4.6. Floppy

A mágneslemez hordozható változata a **floppy disk**, magyarul hajlékonylemez lett.

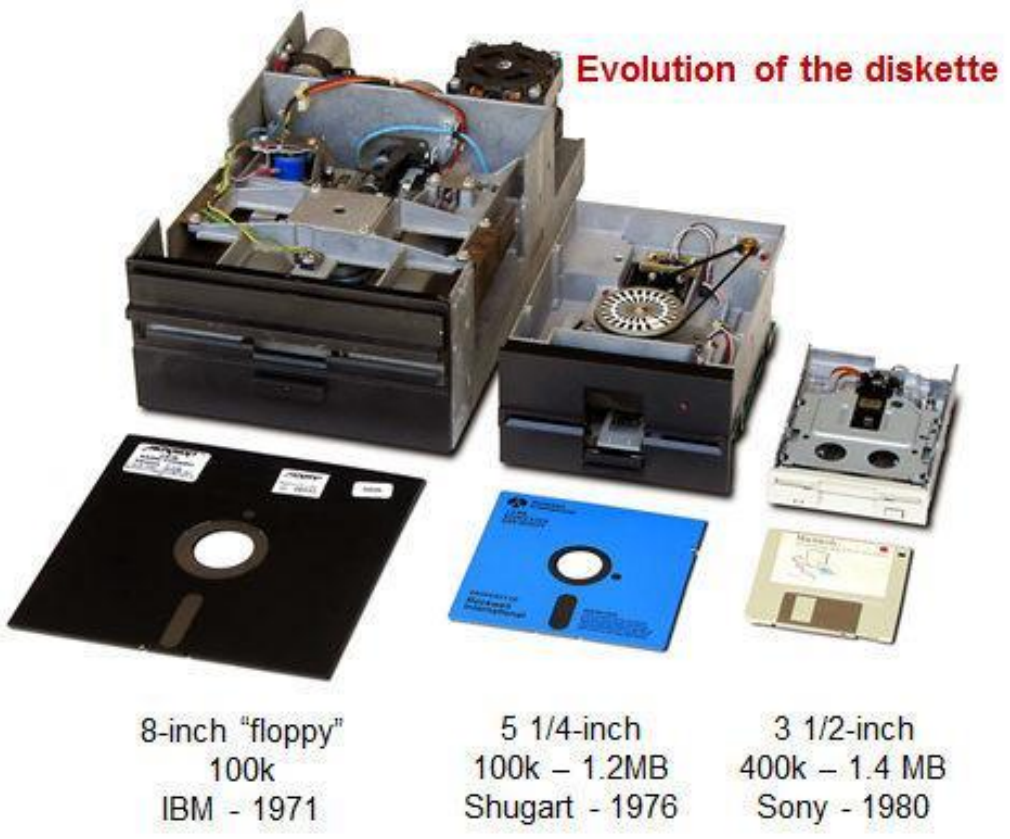
Az IBM az 1970-es évek elején kezdte használni a 8 hüvelykes változatot. Ezt követte az 5¼ hüvelykes, majd a legismertebb 3½ hüvelykes lemez.

A méret csökkent, miközben a kapacitás nőtt.

A PC-korszakban a floppy lett a programok és dokumentumok cseréjének egyik legfontosabb eszköze. A 3½ hüvelykes, 1,44 MB-os lemez egy egész korszak jelképévé vált.

Ma 1,44 MB nevetségesen kevésnek tűnik. Egyetlen modern fénykép gyakran több adatot tartalmaz, mint amennyi több floppy lemezre ráférne.

A floppy mégis tovább él – **a mentés ikonjaként**. Sok programban ma is egy olyan tárgy képe jelenti a „Mentés” parancsot, amelyet a fiatalabb felhasználók jelentős része már soha nem használt.



Ábra: *A floppy három nemzedéke: 8, 5¼ és 3½ hüvelykes lemezek. A technológia eltűnt a számítógépekből, alakja azonban a „Mentés” ikonban tovább él.*

4.7. CD, DVD, Blu-ray

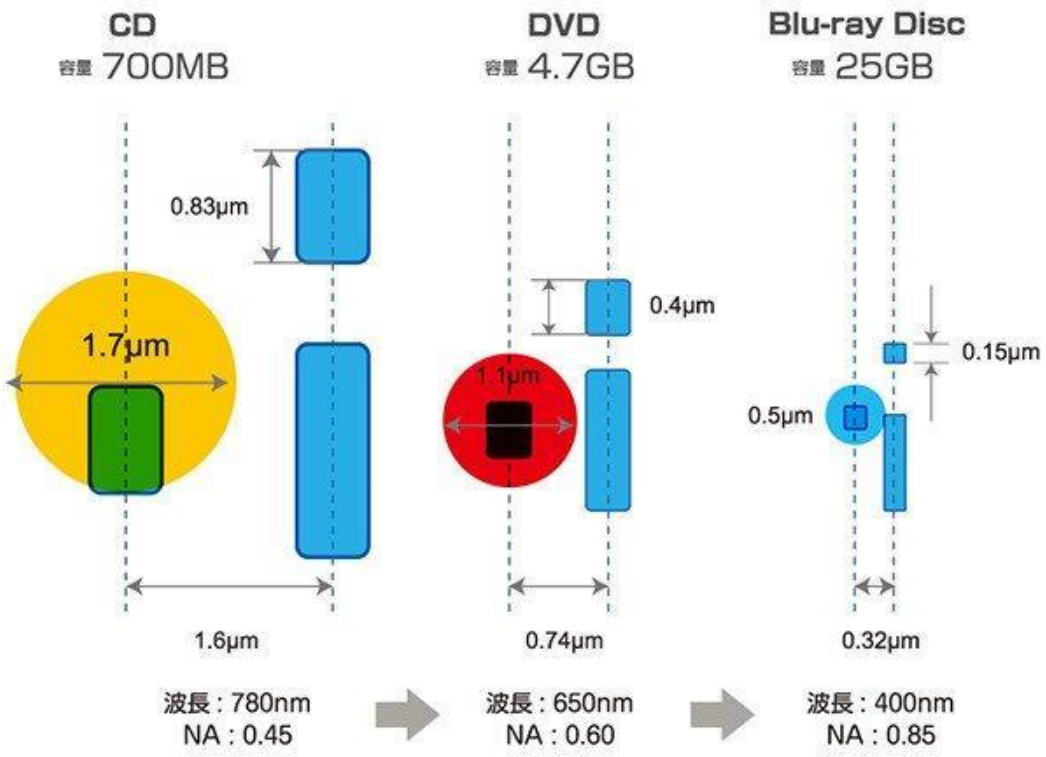
Az optikai adattárolásnál az információt már nem mágneses állapotok jelentik.

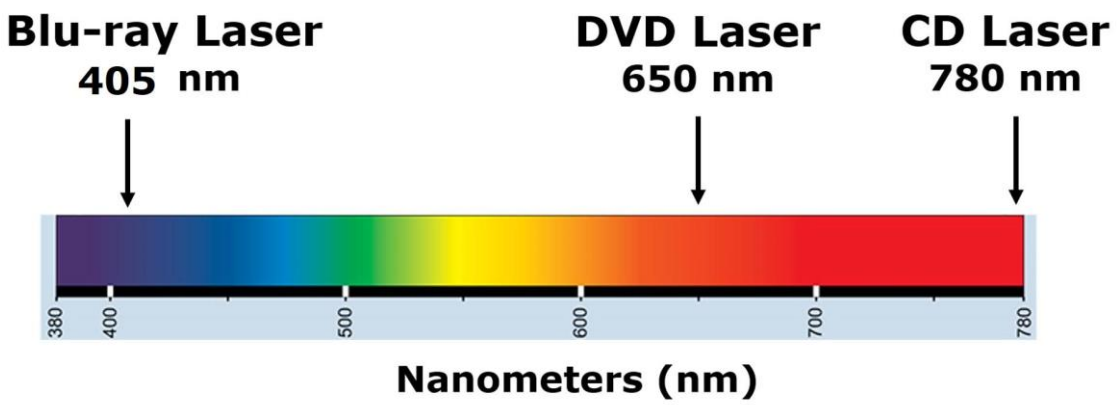
A lemez felületén kialakított vagy optikailag érzékelhető mikroszkopikus struktúrát lézersugár olvassa.

A **CD** eredetileg digitális hanghordozóként terjedt el, majd számítógépes adattárolóvá is vált. Tipikus kapacitása 650–700 MB.

A **DVD** néhány gigabájtra növelte ezt, a **Blu-ray** pedig rétegenként több tíz gigabájt tárolását tette lehetővé.

A kapacitás növelésének egyik kulcsa az egyre rövidebb hullámhosszú fény alkalmazása volt. Minél kisebb fényfoltot lehetett létrehozni, annál sűrűbben lehetett az információt a lemezen elhelyezni.





Ábra: *CD, DVD és Blu-ray. Az egyre rövidebb hullámhosszú lézer és a kisebb adattároló struktúrák tették lehetővé, hogy ugyanakkora lemezre egyre több információ kerüljön.*

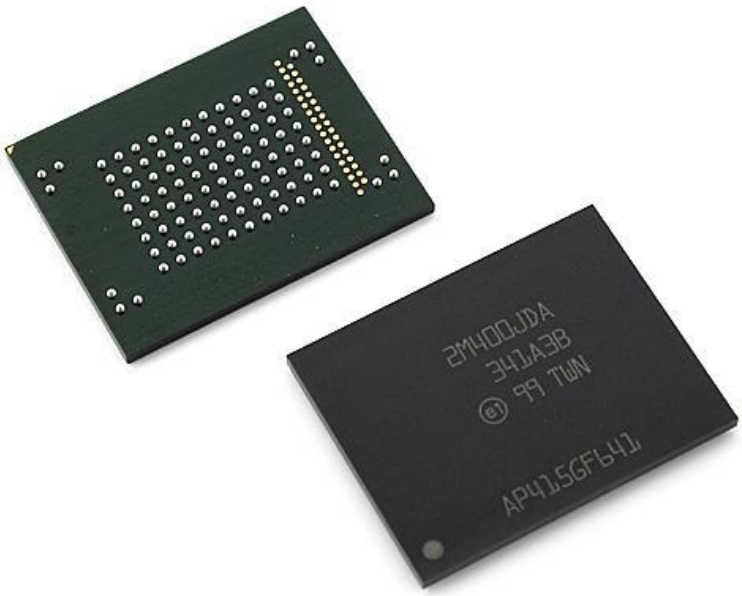
4.8. Flash memória

A flash memóriával az előző fejezetben már találkoztunk. Adattárolási jelentősége azonban akkora, hogy itt is helye van.

A flash egyik legfontosabb tulajdonsága, hogy **nincsenek mozgó alkatrészei**, és áram nélkül is megőrzi az adatot.

Ez tette lehetővé a pendrive-okat, memóriakártyákat, telefonok belső tárhelyét és végül az SSD-ket.

A mechanikus szerkezet eltűnésével az adattároló kisebb, ütésállóbb és gyorsabb lett.



Ábra: *A flash memória megszüntette a mozgó alkatrészek szükségességét. Ugyanez a technológiai alap található a pendrive-okban, memóriakártyákban és SSD-kben.*

4.9. SSD

Az **SSD – Solid State Drive** ugyanazt a szerepet tölti be, mint korábban a merevlemez, de forgó lemezek helyett félvezető flash memóriában tárolja az adatokat.

Nincs motor.

Nincs forgó lemez.

Nincs mozgó író-olvasó fej.

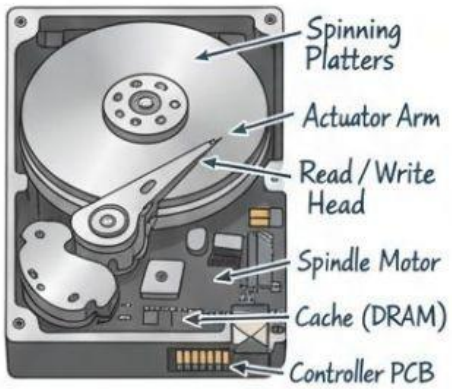
Ennek következtében az SSD gyorsabban éri el az adatokat, csendes, kevésbé érzékeny a mechanikai rázkódásra, és kis méretben is elkészíthető.

A merevlemez és az SSD közötti különbséget jól mutatja a hozzáférés módja. A merevlemeznél fizikailag meg kell várni, hogy a fej a megfelelő sáv fölé kerüljön és a keresett szektor odaérjen. Az SSD-ben nincs ilyen mechanikai várakozás.

Ezért az SSD megjelenése a számítógépek **érzékelhető sebességét** is drámaian növelte.

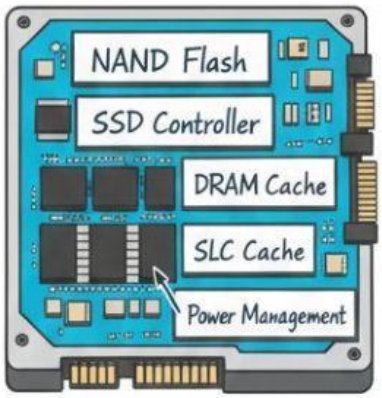
HDD vs SSD

HDD (Hard Disk Drive)



- Mechanical moving parts
- Data stored magnetically
- Slower due to seek time & rotation
- More power consumption
- Noise & vibration present

SSD (Solid State Drive)



- No moving parts.
- Data stored electrically.
- Very fast access time.
- Low power consumption.
- Silent operation.

Feature	HDD	SSD
Storage Technology	Magnetic	NAND Flash
Speed	Slow	Very Fast
Moving Parts	Yes	No
Durability	Low	High
Power Usage	High	Low
Noise	Yes	Silent
Cost per GB	Cheaper	Costlier

Data Access Flow:

CPU → Controller → Head → Platter → HDD



Ábra: *Merevlemez és SSD. Az egyik mechanikai, a másik teljesen elektronikus adattároló. A mozgó alkatrészek elhagyása különösen az adatok elérési idejét csökkentette drasztikusan.*

4.10. Felhő

A felhő első pillantásra teljesen új adattárolási módnak tűnik.

Pedig fizikailag **nincs felhő**.

Ha egy fényképet, dokumentumot vagy levelet a felhőben tárolunk, az valójában valahol egy vagy több adatközpont fizikai adattárolóin található.

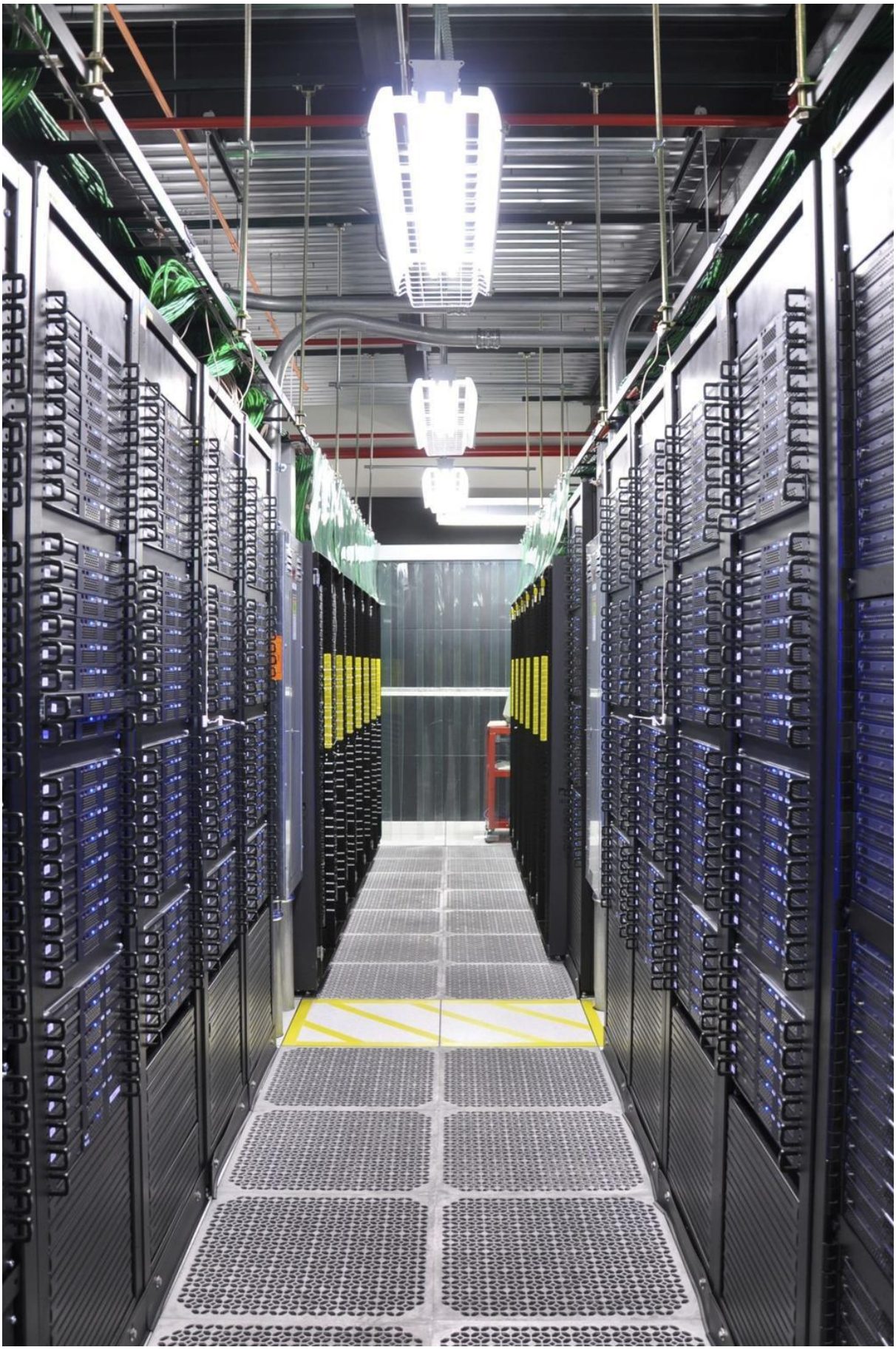
Az újdonság nem az adathordozó, hanem az, hogy **nem kell tudnunk, pontosan hol van az adat**.

Az interneten keresztül elérhető, több példányban tárolható, automatikusan másolható egyik gépről a másikra, és akár kontinenseken átívelő rendszerek gondoskodhatnak róla.

Ez óriási kényelmet jelent, ugyanakkor új kérdést vet fel:

Valóban a miénk-e az adat, ha csak egy szolgáltatáson keresztül férünk hozzá?

A felhő tehát technikai fejlődés mellett az információ birtoklásának fogalmát is megváltoztatta.





Ábra: *A „felhő” valójában nagyon is földi. Hatalmas adatközpontok szervereinek és háttértárainak sokasága őrzi az interneten tárolt adatainkat.*

4.11. DNS és kísérleti adattárolók

Meddig lehet még növelni az adatsűrűséget?

Az egyik legizgalmasabb lehetőséget maga az élet mutatja.

A **DNS** rendkívül sűrű információtároló rendszer. Négyféle építőeleme – A, C, G és T – megfelelő kódolással digitális információ hordozására is használható.

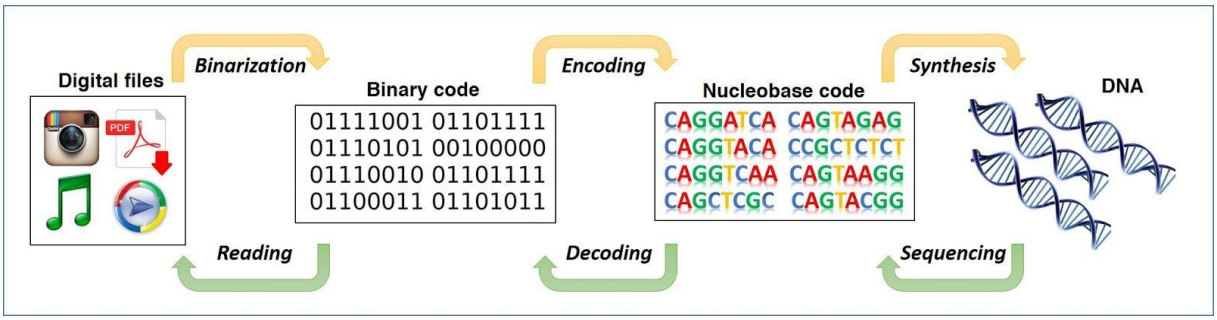
A számítógépes adatokat először DNS-szekvenciává alakítják, majd mesterséges DNS-molekulákat állítanak elő. Az információ visszaolvasásához a DNS-t szekvenálják, majd a kapott bázissorrendet ismét digitális adattá alakítják.

A módszer egyelőre drága és lassú, ezért nem fogja kiváltani az SSD-t a számítógépünkben.

Hosszú távú archiválásra azonban rendkívül érdekes lehet: nagyon kis térfogatban óriási mennyiségű információ tárolható, és megfelelő körülmények között a DNS rendkívül hosszú ideig fennmaradhat.

Más kísérleti technológiák üvegben, kristályszerkezetekben vagy más nanoméretű fizikai állapotokban próbálják évezredekre megőrizni az információt.





Ábra: *Az adattárolás lehetséges jövője. Digitális információ DNS-molekulák bázissorrendjébe is kódolható. Más kísérletek különleges üvegekben és mikroszkopikus struktúrákban próbálják rendkívül hosszú időre megőrizni adatainkat.*

Miért a papír az egyik legtartósabb adathordozónk?

Van egy több mint kétezer éves papiruszunk.

Van több száz éves nyomtatott könyvünk.

De vajon **el tudjuk-e olvasni száz év múlva a mai SSD-n tárolt fájljainkat?**

Meglepő módon a digitális korszak egyik legnagyobb problémája éppen az információ hosszú távú megőrzése.

A jó minőségű, savmentes papír megfelelő környezetben **évszázadokig fennmaradhat.** Ráadásul különleges tulajdonsága van:

az információ és az olvasásához szükséges „technológia" szinte ugyanaz a tárgy.

Egy könyvhöz nem kell operációs rendszer.

Nem kell hozzá USB-csatlakozó.

Nem kell tudnunk, milyen fájlformátumban készült.

Nem kell hozzá villamos energia.

És nincs szükségünk egy száz évvel korábbi készülék működő példányára sem.

Elég hozzá **a szemünk és a nyelv ismerete.**

A digitális adathordozó egészen más. Egy régi mágnesszalag tartalma hiába maradt fizikailag sértetlen, ha nincs hozzá működő szalagmeghajtó. Egy floppyhoz meghajtó és megfelelő interfész kell. Egy fájlhoz pedig olyan program, amely még ismeri annak formátumát.

A problémának ezért legalább három rétege van:

Megmaradt-e az adathordozó?
Van-e még eszköz, amely el tudja olvasni?
Van-e még program, amely értelmezni tudja a fájlt?





Ez a digitális archiválás egyik paradoxona.

Egy modern adattároló **nagyságrendekkel több információt** képes megőrizni, mint egy könyv, de fennmaradásához folyamatos gondoskodás szükséges. Az adatokat időről időre új adathordozóra kell másolni, új formátumba kell konvertálni, és gondoskodni kell arról is, hogy az értelmezésükhöz szükséges műszaki ismeretek fennmaradjanak.

Egy könyv ezzel szemben száz évig nyugodtan állhat egy polcon.

És száz év múlva – ha nem pusztította el víz, tűz, penész vagy ember – egyszerűen **levesszük, kinyitjuk és elolvassuk**.

Ezért a digitális megőrzés korában különös módon ismét felértékelődhet az egyik legrégebbi technológiánk:

A hosszú távra szánt legfontosabb információk esetében néha még mindig a jó minőségű papír az egyik legbiztonságosabb adathordozó.

II. rész – Ember és gép találkozása

5. Hogyan adunk parancsot?

A számítógép önmagában csak utasításokat végrehajtó szerkezet. Ahhoz, hogy hasznos eszközzé váljon, valamilyen módon közölnünk kell vele, **mit szeretnénk tenni.**

Ez kezdetben kapcsolók átállítását és kábelek átkötését jelentette. Később lyukkártyákon vitték be a programot, majd megjelentek az írógépszerű terminálok, a billentyűzet, az egér és az érintőképernyő. Ma már beszéddel, kézmozdulatokkal, sőt tekintetünkkel is irányíthatunk számítógépeket.

A fejlődés iránya jól felismerhető:

a gép nyelvéhez alkalmazkodó embertől az ember természetes kommunikációjához alkalmazkodó gépig.

5.1. Kapcsolók

Az első elektronikus számítógépeken gyakran nem volt billentyűzet vagy képernyő. A kezelő a gép előlapján elhelyezett kapcsolókkal állította be a számokat, memóriacímeket és utasításokat.

Egy kapcsoló két állapota könnyen megfeleltethető volt a bináris számrendszer két számjegyének:

kikapcsolva = 0
bekapcsolva = 1

A kezelő kapcsolók sorozatával állított be egy bináris értéket, majd egy külön gombbal vagy kapcsolóval betöltötte azt a számítógép memóriájába. Egy rövid program bevitele is sok aprólékos műveletet igényelt.

A korai mikroszámítógépek egy része ugyanezt a módszert örökölte. Az Altair 8800 előlapján például kapcsolók és lámpák segítségével lehetett adatokat bevinni, illetve a gép állapotát megfigyelni.

A villogó lámpákkal teli kezelőpult később a számítógép egyik filmes jelképévé vált. A valóságban azonban használata lassú, fárasztó és könnyen elhibázható volt.



Ábra:
Amikor a programot még kapcsolónként kellett bevinni. Minden kapcsoló egy bináris számjegyet képviselt, a lámpák pedig a memória vagy a processzor pillanatnyi állapotát jelezték.

5.2. Lyukkártya

A lyukkártya nemcsak adatokat tárolt, hanem a számítógépnek adott parancsok és programok bevitelére is szolgált.

A programozó rendszerint először papírra írta a programot, majd egy lyukasztógép billentyűzetén begépelte. A gép minden programsorhoz elkészített egy lyukkártyát. Az IBM legelterjedtebb formátuma 80 oszlopot és 12 sort használt; egy kártya jellemzően legfeljebb 80 karaktert hordozott.

A kártyacsomagot ezután beolvasták a számítógépbe. A felhasználó azonban többnyire nem beszélgetett közvetlenül a géppel. Leadta a programot, majd percekkel vagy akár órákkal később kapta meg a nyomtatott eredményt.

Ez volt a **kötegelt adatfeldolgozás** világa.

A lyukkártya egyben sajátos fegyelmet is követelt. Egyetlen rossz helyre ütött lyuk hibás utasítást eredményezhetett. Ha pedig a kártyacsomag kiesett a kézből és összekeveredett, a program sorrendje is felborulhatott.



Ábra:
Lyukkártyás programbevitel. A program minden sora külön kártyára kerülhetett, a teljes programot pedig egy rendezett kártyacsomag alkotta.

5.3. Teletype

A teletype, magyarul távgépíró vagy géptávíró eredetileg a távíróhálózatok számára készült. Billentyűzetből, nyomtatóműből és gyakran papírszalag-olvasóból, illetve -lyukasztóból állt.

A számítógépekhez kapcsolva egyszerre vált **beviteli és kimeneti eszközzé.**

A felhasználó a billentyűzeten begépelte a parancsot. A számítógép válasza nem képernyőn jelent meg, hanem a gép papírra nyomtatta.

Innen ered több ma is használt informatikai kifejezés. A parancssoros felület „terminálja” eredetileg valóban egy fizikai végberendezés volt. A Unix és Linux rendszerekben máig találkozhatunk a **TTY** rövidítéssel, amely a *teletypewriter* nevéből származik.

A teletype fontos átmenetet képviselt: a számítógéppel folytatott munka már hasonlított az írásbeli párbeszédhez, de a válasz még maradandó papírszalagon vagy papírlapon jelent meg.



Ábra:
A teletype volt az első széles körben használt interaktív számítógépes terminálok egyike. A parancsot billentyűzettel írták be, a számítógép válaszát pedig papírra nyomtatta.

5.4. Billentyűzet

A számítógép billentyűzete az írógép közvetlen leszármazottja. A betűk elrendezése, a szóköz, a Shift és több más billentyű eredete is az írógépes korszakra vezethető vissza.

A billentyűzet azonban hamarosan többre vált képessé, mint a szöveg bevitele.

Megjelentek a funkcióbillentyűk, a kurzormozgató nyilak, a vezérlőbillentyűk, majd a numerikus rész és a különféle parancskombinációk. Egy billentyű lenyomása nem feltétlenül

egy betűt jelentett: programot indíthatott el, menüt nyithatott meg vagy megváltoztathatta egy másik billentyű jelentését.

A számítógépes billentyűzetek elrendezése sokáig egyáltalán nem volt egységes. A személyi számítógépek elterjedésével azonban néhány kialakítás – különösen az IBM PC billentyűzeteinek leszármazottai – iparági szabvánnyá vált.

A billentyűzet különösen jól illik a **pontos, szimbolikus kommunikációhoz**. Hosszú szöveg, programkód vagy számsor bevitelére ma is hatékonyabb, mint az érintőképernyő vagy a beszédvezérlés.



Ábra:
Az írógéptől a számítógépes billentyűzetig. Az alapvető betűelrendezés alig változott, de a billentyűzet a szövegbeviteli eszközből általános parancsadó felületté vált.

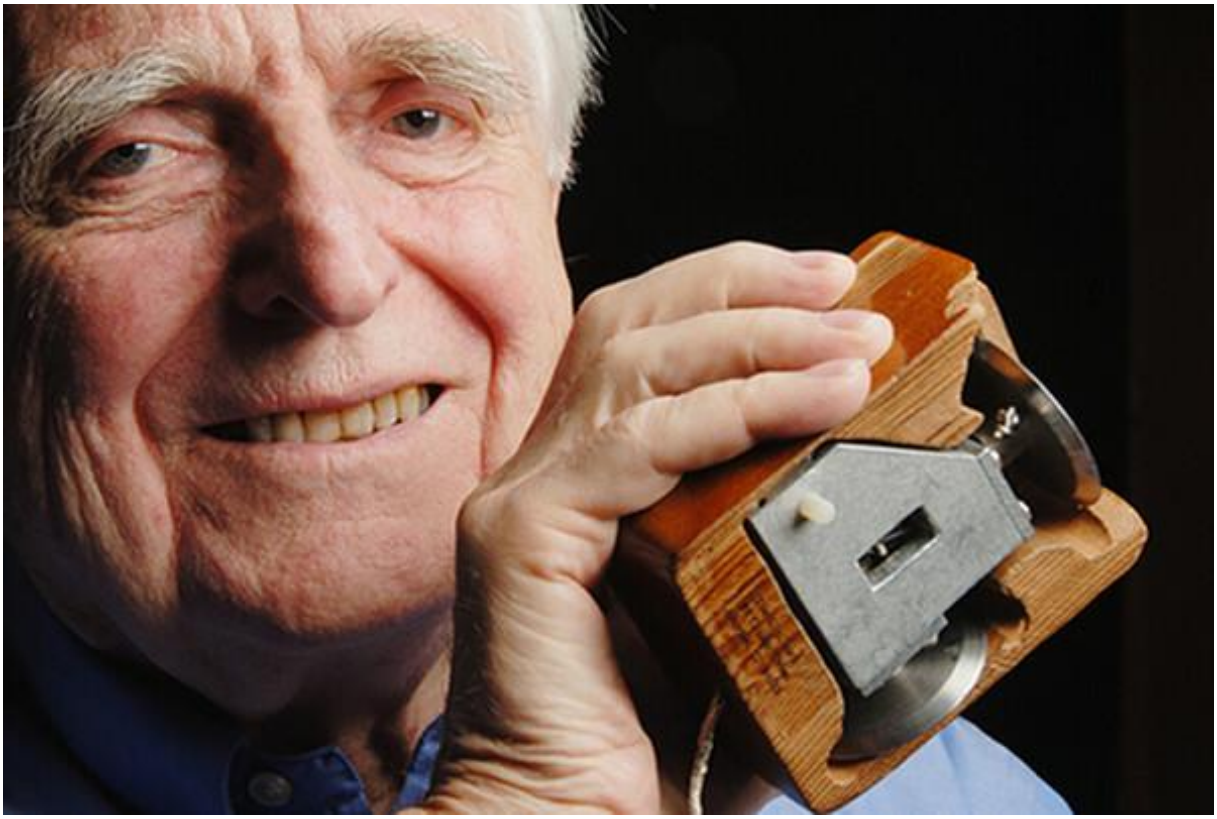
5.5. Douglas Engelbart és az egér

Az egér gondolata az 1960-as években született Douglas Engelbart kutatócsoportjában, az amerikai Stanford Research Institute-ban. Engelbart célja nem egyszerűen egy kényelmes mutatóeszköz megalkotása volt: olyan rendszert akart létrehozni, amely **kiterjeszti az ember gondolkodási és együttműködési képességeit**.

Az első egér fából készült kis doboz volt. Alján két, egymásra merőleges irányban forgó kerék érzékelte az asztalon végzett mozgást. A szerkezetet Bill English építette meg Engelbart elképzelése alapján.

Az egér 1968. december 9-én vált szélesebb körben ismertté, amikor Engelbart San Franciscóban bemutatta az NLS rendszert. A később **„minden bemutatók anyjának”** nevezett előadáson az egér mellett ablakokat, hipertextet, közös dokumentumszerkesztést és élő videokapcsolatot is láthatott a közönség.

Az egér igazi jelentőségét azonban csak a grafikus felhasználói felület mutatta meg. Amíg a számítógépet főleg begépelt parancsokkal irányították, az egér csupán különös kiegészítőnek tűnt. Amikor azonban ikonokra, menükre és ablakokra kellett mutatni, nélkülözhetetlenné vált.



Ábra:
Douglas Engelbart egyik korai egere. A fa burkolat alatt két egymásra merőleges kerék mérte a mozgást. Az eszköz nevét valószínűleg a dobozból kilógó, farokra emlékeztető vezeték ihlette.

5.6. Joystick

A joystick története régebbi a személyi számítógépeknél. Hasonló botkormányokat repülőgépek és különféle katonai vezérlőrendszerek irányítására is használtak.

A számítógépes joystick működése egyszerű: a kar kitérítésének irányát – és később annak mértékét is – elektromos jelekké alakítja. A gombok további parancsokat adnak.

A joystick különösen a számítógépes játékokban vált fontossá. Egy mozgó alak, repülőgép vagy űrhajó irányítása sokkal természetesebb volt vele, mint billentyűzettel.

A korai digitális joystickok gyakran csak négy vagy nyolc irányt érzékeltek. Az analóg változatok már azt is meg tudták állapítani, mennyire térítette ki a játékos a kart.

Később a joystickot részben felváltotta a két analóg karral, gombokkal, rezgéssel és mozgásérzékelőkkel felszerelt játékvezérlő.



Ábra:
A klasszikus joystick a számítógépes játékok egyik jelképe lett. A kapcsolókkal érzékelt néhány iránytól hosszú út vezetett a mai analóg, mozgásérzékelős vezérlőkig.

5.7. Érintőképernyő

Az érintőképernyő különleges eszköz, mert egyszerre **kimeneti és beviteli felület**. Nem egy különálló eszközt mozgatunk, hanem közvetlenül azt a tárgyat érintjük meg, amelyet ki szeretnénk választani.

A korai érintőképernyős rendszerek már az 1960-as és 1970-es években megjelentek. Az Illinois-i Egyetemen fejlesztett PLATO oktatási rendszer plazmakijelzős, érintésérzékeny termináljai több későbbi technológiát is megelőlegeztek.

Többféle érzékelési elv alakult ki.

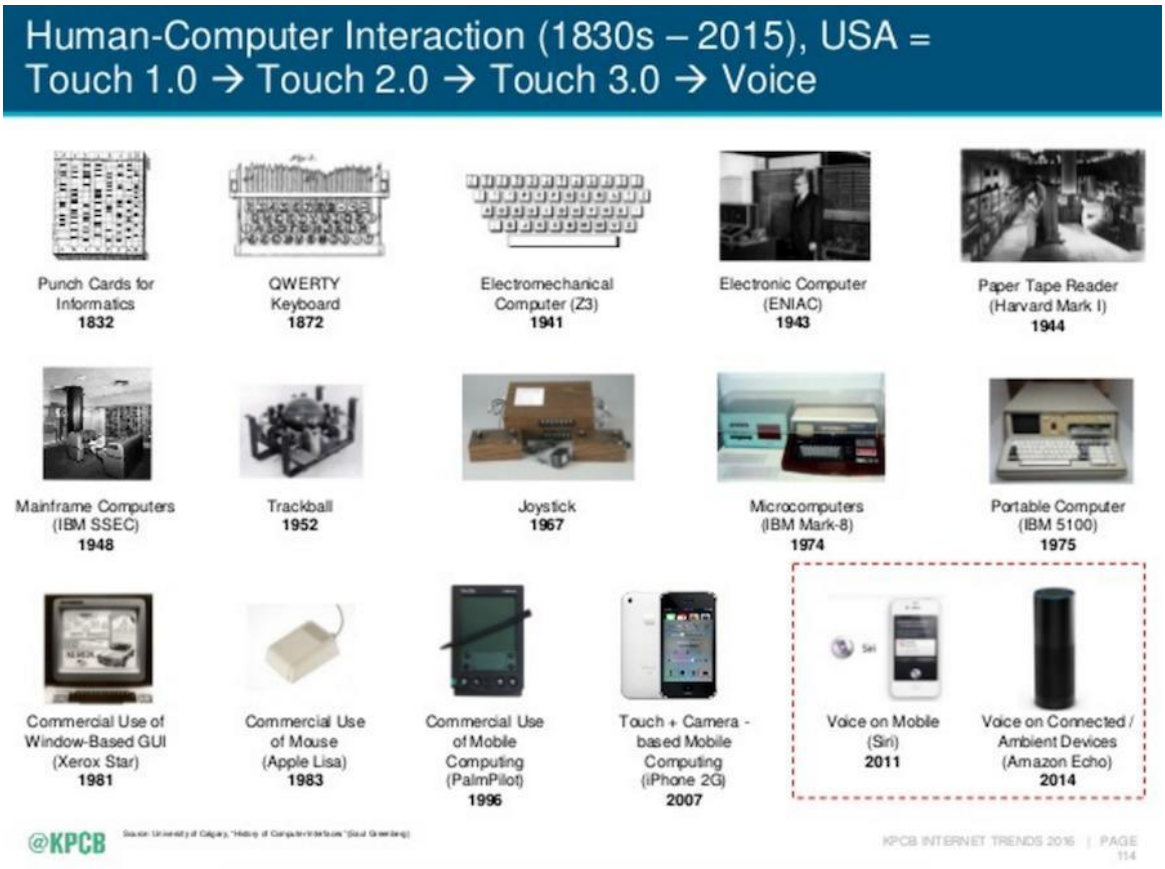
A **rezisztív** képernyő nyomás hatására érintkező vezető rétegekkel érzékeli a helyet. Ujjal, kesztyűben vagy érintőceruzával is működhet.

A **kapacitív** képernyő az emberi test elektromos hatását érzékeli. Ez teszi lehetővé a könnyű érintést és a több ujjal végzett mozdulatokat.

Az érintőképernyő igazi áttörését az okostelefon hozta el. Eltűnhetett a külön billentyűzet, az egér és a legtöbb nyomógomb. Ugyanaz a felület mindig az éppen szükséges kezelőszerveket jelenítheti meg.



Ábra:
Az érintőképernyőn a kijelző és a kezelőfelület egyetlen felületté vált. A korai PLATO-terminálóktól a mai okostelefonokig ugyanaz az alapgondolat érvényesül: azt érintjük meg, amit irányítani szeretnénk.



5.8. Beszédvezérlés

A beszéd talán a legtermészetesebb emberi parancsadási mód. Egy számítógép számára azonban rendkívül nehéz felismerni.

Ugyanazt a szót minden ember kissé másképpen ejti ki. Különbözőek a hangok, a hangsúlyok, a beszédtempók és a nyelvjárások. A háttérzaj, a visszhang és a hasonló hangzású szavak tovább nehezítik a feladatot.

A korai beszédfelismerő rendszerek csak néhány szót vagy számjegyet értettek meg, gyakran egyetlen betanított beszélőtől. A számítási teljesítmény növekedése, majd a statisztikai és neurális hálózati módszerek fejlődése azonban egyre nagyobb szókincs és folyamatos beszéd felismerését tette lehetővé.

A mai rendszer már nem feltétlenül pusztán hangokat azonosít. A mondat környezetéből megpróbálja kikövetkeztetni, melyik szó lehet a legvalószínűbb.

A beszédvezérlés különösen hasznos lehet vezetés közben, látássérült felhasználóknál, valamint olyan helyzetekben, amikor a kezünk foglalt. Mégsem váltja ki teljesen a billentyűzetet és az érintést: nyilvános helyen például nem mindig szeretnénk hangosan elmondani a parancsainkat.



Ábra:
A beszédvezérlés fejlődése. A néhány betanított szót felismerő laboratóriumi rendszerektől eljutottunk a folyamatos beszédet és összetett utasításokat értelmező digitális asszisztensekig.

5.9. Gesztusvezérlés

A gesztusvezérlésnél a számítógép testünk mozgásából próbálja megérteni a szándékunkat.

A legegyszerűbb gesztusokkal már az érintőképernyőn is találkozunk:

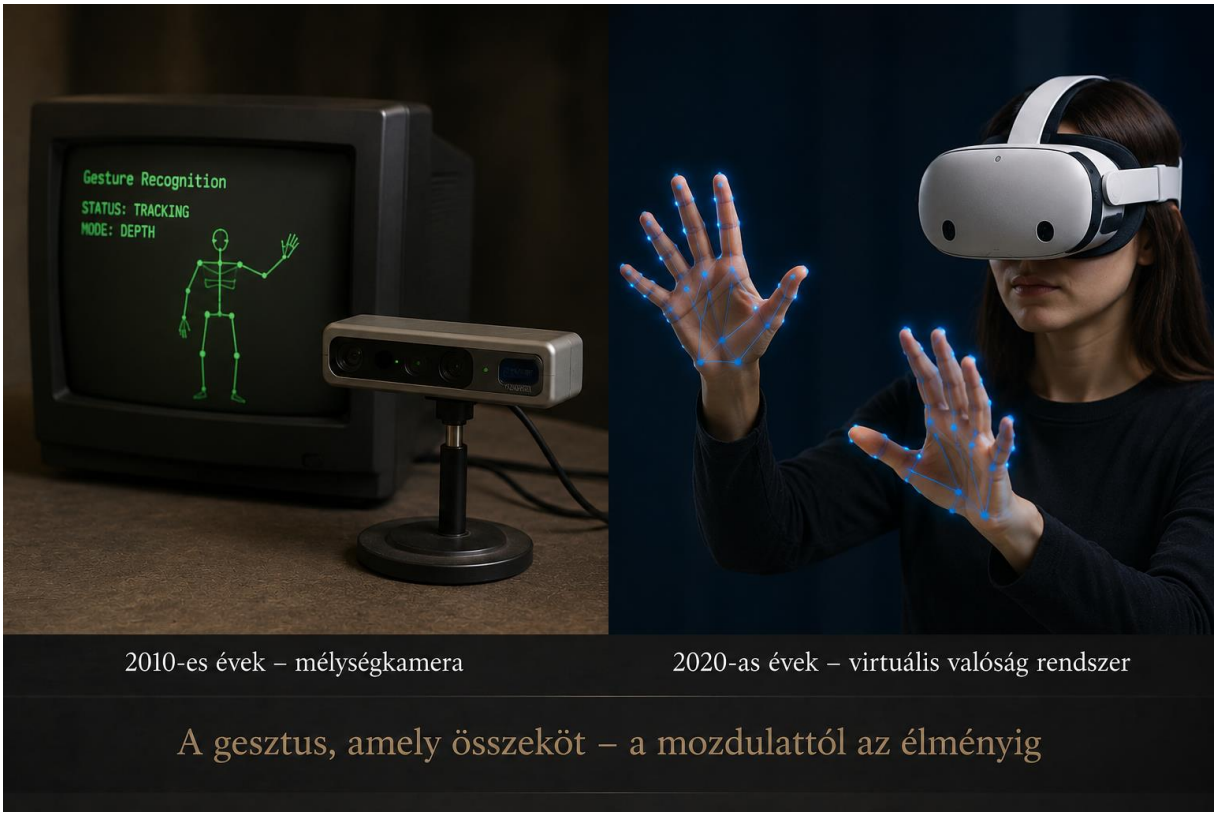
koppintás, húzás, csippentés, kétujjas nagyítás vagy forgatás.

A kamerás rendszerek ennél tovább mennek. Felismerhetik a kéz helyzetét, a kar mozdulatát, a testtartást vagy akár az arckifejezést is. A mélységérzékelő kamerák azt is megállapíthatják, milyen távol van tőlük egy testrész.

A gesztusvezérlés játékokban, virtuális és kiterjesztett valóságban, ipari rendszerekben és akadálymentesítő megoldásokban is használható.

Van azonban egy alapvető nehézsége: a természetes emberi mozdulatok nem mindig egyértelműek. A gépnek különbséget kell tennie a szándékos parancs és a véletlen kézmozdulat között.

A jó gesztusvezérlés ezért nem feltétlenül azt jelenti, hogy minél több mozdulatot ismerünk fel. Inkább azt, hogy **néhány egyszerű és könnyen megjegyezhető gesztust megbízhatóan értelmezünk.**



Ábra:
Gesztusvezérlés. A kamera vagy más érzékelő a kéz és a test mozgását digitális paranccsá alakítja. Az ember ilyenkor már nem közvetlenül egy eszközt, hanem saját mozdulatait használja kezelőfelületként.

Az egér, amelyet évekig senki sem akart használni

Douglas Engelbart és munkatársai már az 1960-as években megalkották és bemutatták az egeret. Az 1968-as előadáson a közönség láthatta, ahogy Engelbart a képernyőn mozgatja a mutatót, szövegrészeket választ ki, dokumentumok között navigál és ablakokat kezel.

Mégsem rohant minden számítógépgyártó azonnal egeret szerelni a gépeihez.

Ennek egyszerű oka volt:

az akkori számítógépek többségéhez nem kellett egér.

A programokat lyukkártyán vitték be, vagy a felhasználó begépelt parancsokkal dolgozott. A képernyőn többnyire karakterek sorakoztak. Nem voltak ikonok, ablakok és lehúzható menük, amelyekre rá lehetett volna mutatni.

Az egér tehát nem önmagában volt forradalmi. Egy újfajta számítógép-használat részeként vált értelmessé.

Engelbart ráadásul nem is egy egyszerűbb írógépet akart létrehozni. Olyan rendszert képzelt el, amelyben az ember gyorsan mozoghat összekapcsolt dokumentumok között, szövegeket szerkeszthet, információkat rendezhet, és távoli munkatársaival közösen dolgozhat.

Az egér csak egyetlen eleme volt ennek a nagyobb elképzelésnek.

A Xerox kutatóközpontjában később az egér grafikus felhasználói felülethez kapcsolódott. A Xerox Star kereskedelmi rendszerében már ikonokra, mappákra és ablakokra lehetett vele mutatni, de a gép drága volt, és nem vált tömegtermékké.

Az Apple Lisa, majd különösen az 1984-es Macintosh tette az egeret szélesebb körben ismertté. A grafikus felület és a mutatóeszköz együtt már könnyen megérthető rendszert alkotott.

A felhasználók kezdetben még ekkor is idegenkedhettek tőle. Meg kellett tanulniuk, hogyan mozgassák a kezükben az egeret úgy, hogy közben a képernyőt nézzék. Új fogalmak jelentek meg:

kattintás, dupla kattintás, húzás, kijelölés.

Ma ezek magától értetődő mozdulatoknak tűnnek, de valójában tanult viselkedések.

Az egér története fontos tanulságot hordoz:

Egy találmány önmagában még nem feltétlenül változtatja meg a világot. Meg kell születnie annak a teljes rendszernek is, amelyben valóban szükség van rá.

Engelbart egerét nem azért mellőzték sokáig, mert rossz találmány volt, hanem mert **megelőzte a korát**. A számítógépeknek kellett előbb olyanná válniuk, hogy használni tudjuk.

Ábra:
Az egér az 1968-as „minden bemutatók anyján”. A szerkezet már működött, de az ikonokra és ablakokra épülő grafikus számítógépnek még évekre volt szüksége ahhoz, hogy az egér nélkülözhetetlenné váljon.

---

6. Hogyan válaszol a gép?

A számítógéppel folytatott kapcsolat kétirányú. Mi adatokat és parancsokat adunk neki, a gép pedig valamilyen érzékelhető formában közli velünk az eredményt.

Ezt nevezzük **kimenetnek**.

A legkorábbi számítógépek néhány lámpa felvillantásával vagy papírra nyomtatott számsorokkal „válaszoltak”. Később megjelent a képernyő, majd a színes grafika, a hang, a háromdimenziós kép és a virtuális valóság.

A fejlődés iránya itt is jól látható:

jelzés → szöveg → rajz → kép → hang → térbeli élmény.

A számítógép válasza egyre inkább hasonlít arra, ahogyan a valóságos világot érzékeljük.

6.1. Kontroll-lámpák

A legkorábbi számítógépeken gyakran még nem volt képernyő. A gép állapotát a kezelőpulton elhelyezett lámpák mutatták.

Egy-egy lámpa rendszerint egy bithez tartozott:

nem világít = 0
világít = 1

A lámpasorok jelezhették egy regiszter tartalmát, a memória egy kiválasztott részét, az aktuális címet vagy a processzor működési állapotát. A kezelő a villogásból olykor azt is felismerhette, ha a program végtelen ciklusba került vagy a gép működése megállt.

A lámpák azonban nem elsősorban az eredmények kényelmes megjelenítésére szolgáltak. Inkább a gép belső működésébe engedtek betekintést.

A korai mikroszámítógépek, például az Altair 8800 is kapcsolókkal és lámpákkal kommunikált a felhasználóval. A kezelő beállította a bináris számot a kapcsolókon, a gép pedig lámpák segítségével mutatta a választ.

Ez volt a számítógép és az ember közötti kommunikáció egyik legegyszerűbb formája: **a gép szó szerint felvillantotta a biteket.**



Ábra:
Kontroll-lámpák egy korai számítógép kezelőpultján. A lámpák egyes bitek, regiszterek és belső állapotok értékét mutatták. A számítógép működése ekkor még részben szabad szemmel követhető volt.

6.2. Nyomtatók

A számítógépek első széles körben alkalmazott kimeneti eszköze nem a képernyő, hanem a **nyomtató** volt.

A korai gépek eredményeit gyakran papírszalagra, különálló lapokra vagy leporellószerű, perforált szélű papírra nyomtatták. A nyomtatott eredmény megmaradt, ellenőrizhető, továbbadható és archiválható volt.

A nagy számítóközpontokban különösen elterjedtek a **sornyomtatók.** Ezek nem karakterenként, hanem egyszerre egy teljes sort nyomtattak ki. Rendkívül gyorsak, ugyanakkor nagyok és zajosak voltak.

A személyi számítógépek korszakában többféle technológia terjedt el.

A **mátrixnyomtató** apró tűkkel ütötte a festékszalagot a papírhoz. A karaktereket pontokból állította össze, ezért egyszerű grafikát is tudott nyomtatni. Előnye volt, hogy indigós, többpéldányos nyomtatványokat is készíthetett.

A **tintasugaras nyomtató** mikroszkopikus tintacseppeket juttat a papírra. Olcsó, jó minőségű színes nyomtatást tett lehetővé.

A **lézernyomtató** elektrosztatikusan feltöltött fényérzékeny dobra rajzolja a képet, majd festékport visz át a papírra. Működési elve rokonságban áll a fénymásolóéval.

A nyomtató különleges kimeneti eszköz, mert a számítógép válaszát **fizikai tárggyá alakítja**. A képernyő képe eltűnik a gép kikapcsolásakor; a papírlap viszont megmarad.



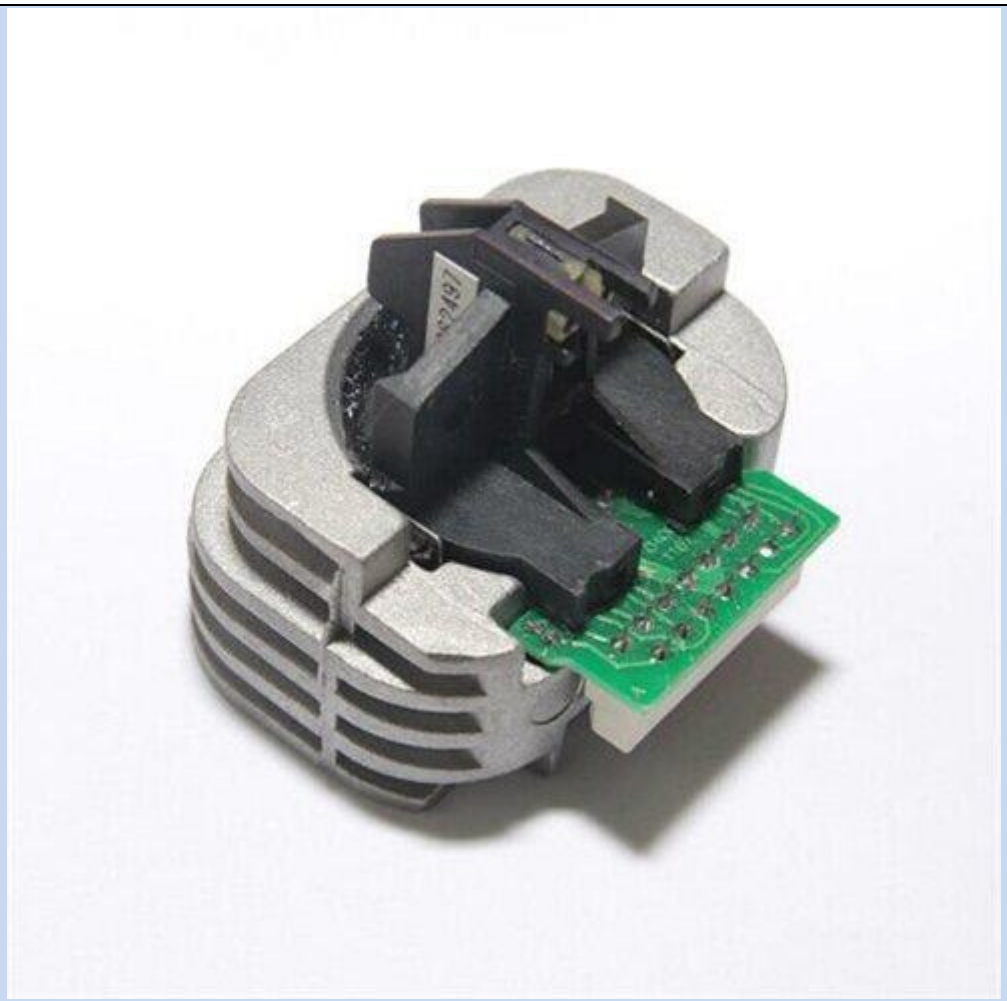
Ábra:
A számítógépes nyomtatók fejlődése. A zajos sornyomtatóktól és mátrixnyomtatóktól a nagy felbontású tintasugaras és lézernyomtatókig a gép válasza papíron is megőrizhetővé vált.

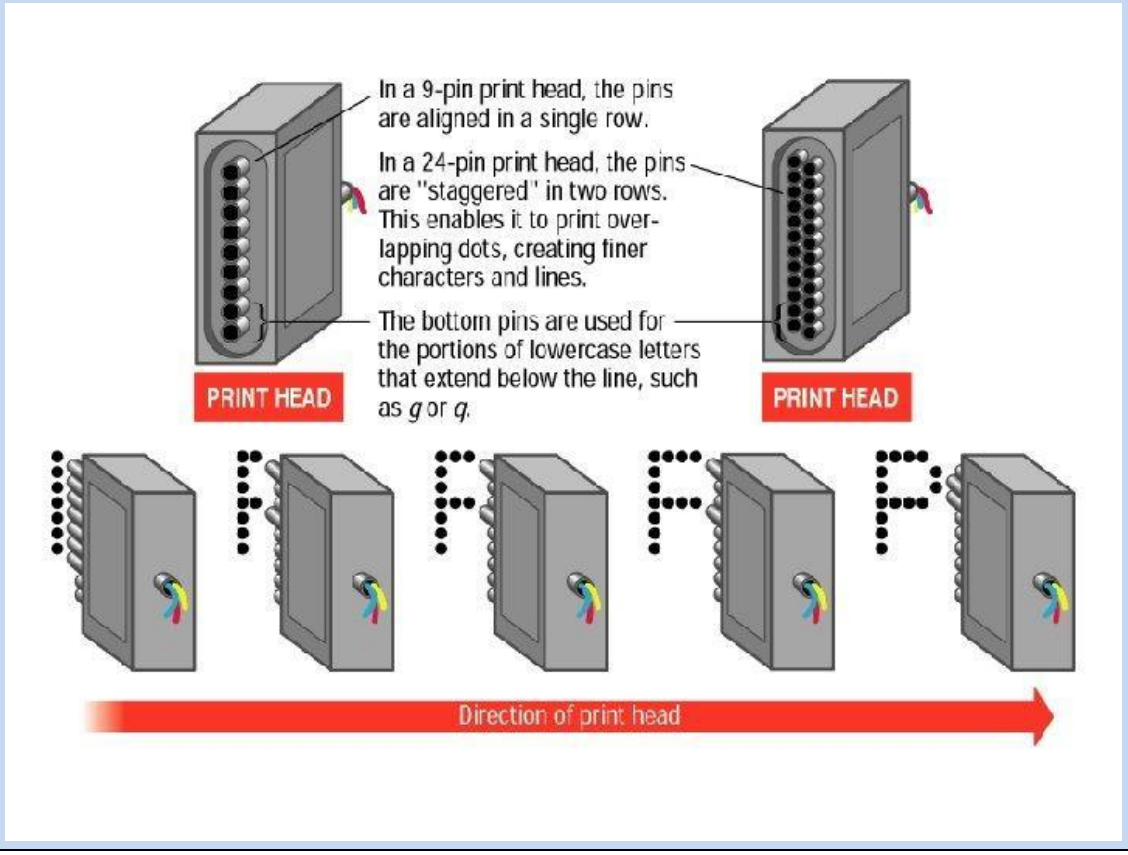
Hogyan működik a tűs nyomtató?

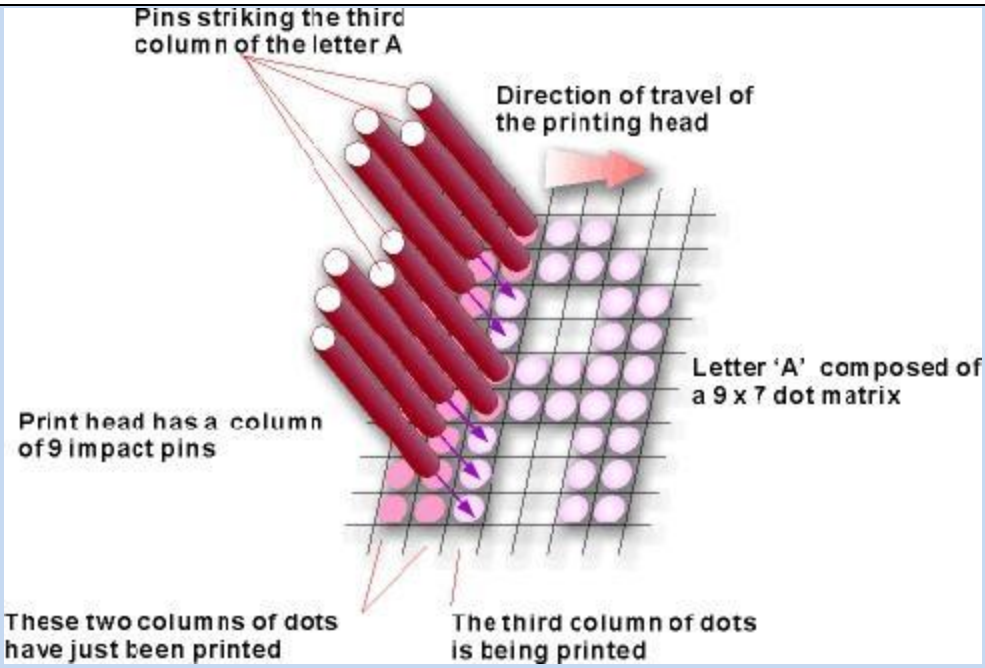
A tűs vagy mátrixnyomtató működése meglepően közel áll a hagyományos írógépéhez. Mindkettő **mechanikai ütéssel** viszi át a festéket a papírra.

Az írógépben azonban minden betűhöz előre kialakított karakter tartozik. A tűs nyomtatóban nincsenek betűformák. Helyettük apró fémrudak, **tűk** találhatók a nyomtatófejben.

A számítógép ezekből a pontokból építi fel a betűt, számot vagy akár a képet.




In a 9-pin print head, the pins are aligned in a single row.
In a 24-pin print head, the pins are "staggered" in two rows. This enables it to print over-lapping dots, creating finer characters and lines.
The bottom pins are used for the portions of lowercase letters that extend below the line, such as g or q.
PRINT HEAD
PRINT HEAD
Direction of print head



Ábra: *A mátrixnyomtató feje. Az elektromágnesek által mozgatott apró tűk a festékszalagot a papírhoz ütve pontokat hoznak létre. A fej oldalirányú mozgásából és a papír továbbításából áll össze a teljes kép.*

Pontokból lesz a betű

A nyomtatófej nagy sebességgel halad a papír előtt. A fej és a papír között festékkel átitatott szalag fut – hasonlóan az írógép festékszalagjához.

Amikor egy tű működésbe lép, előreugrik:

tű → festékszalag → papír

A tű hozzáüti a festékszalagot a papírhoz, és egy apró pont keletkezik.

Ezután a tű visszahúzódik, a fej kissé továbbmozdul, majd más tűk csapnak le.

Mindez másodpercenként akár több százszor vagy ezerszer történik.

A papíron végül pontok mátrixából rajzolódik ki:

A B C 1 2 3, egy diagram, sőt akár egy egyszerű fénykép is.

Innen származik a **mátrixnyomtató** elnevezés.

9 tű – a klasszikus mátrixnyomtató



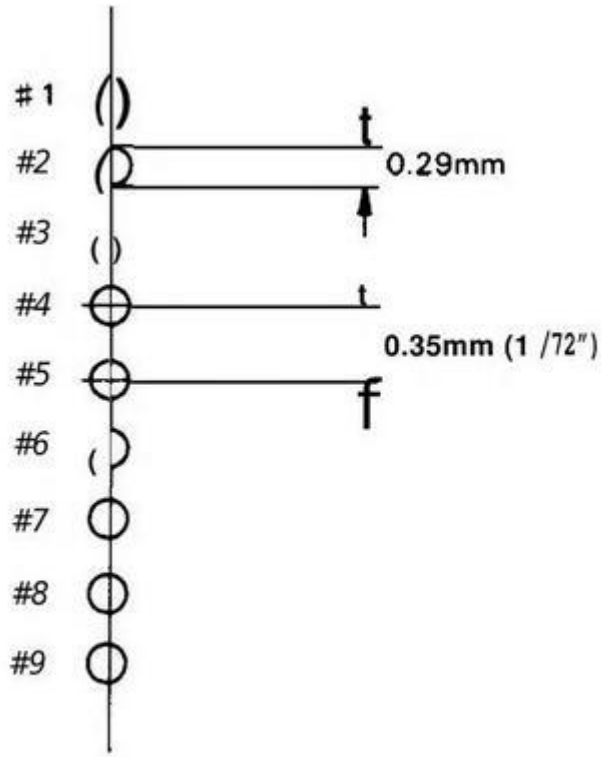
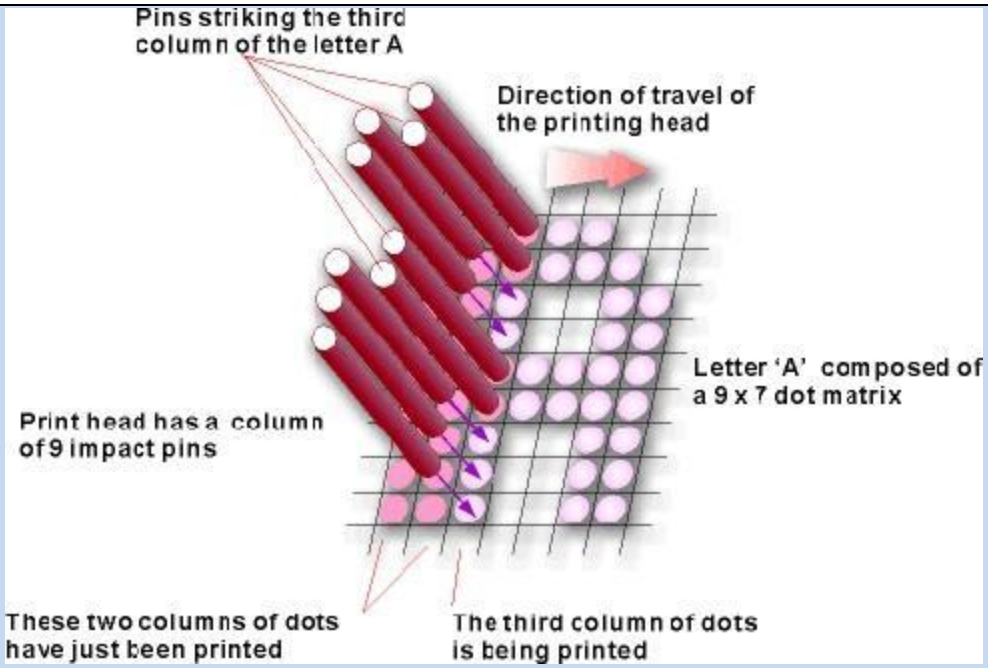


Figure 1-2. Pin Configuration



Ábra: *9 tűs nyomtatófej és az általa létrehozott karakter. A függőleges tűsor vízszintes mozgatásával kétdimenziós pontmátrix rajzolható a papírra.*

A klasszikus PC-korszak első elterjedt mátrixnyomtatói között sok **9 tűs** modell volt.

A fejben kilenc tű helyezkedett el függőleges sorban. A fej vízszintes mozgása közben a megfelelő pillanatban működtetett tűk alakították ki a karaktereket.

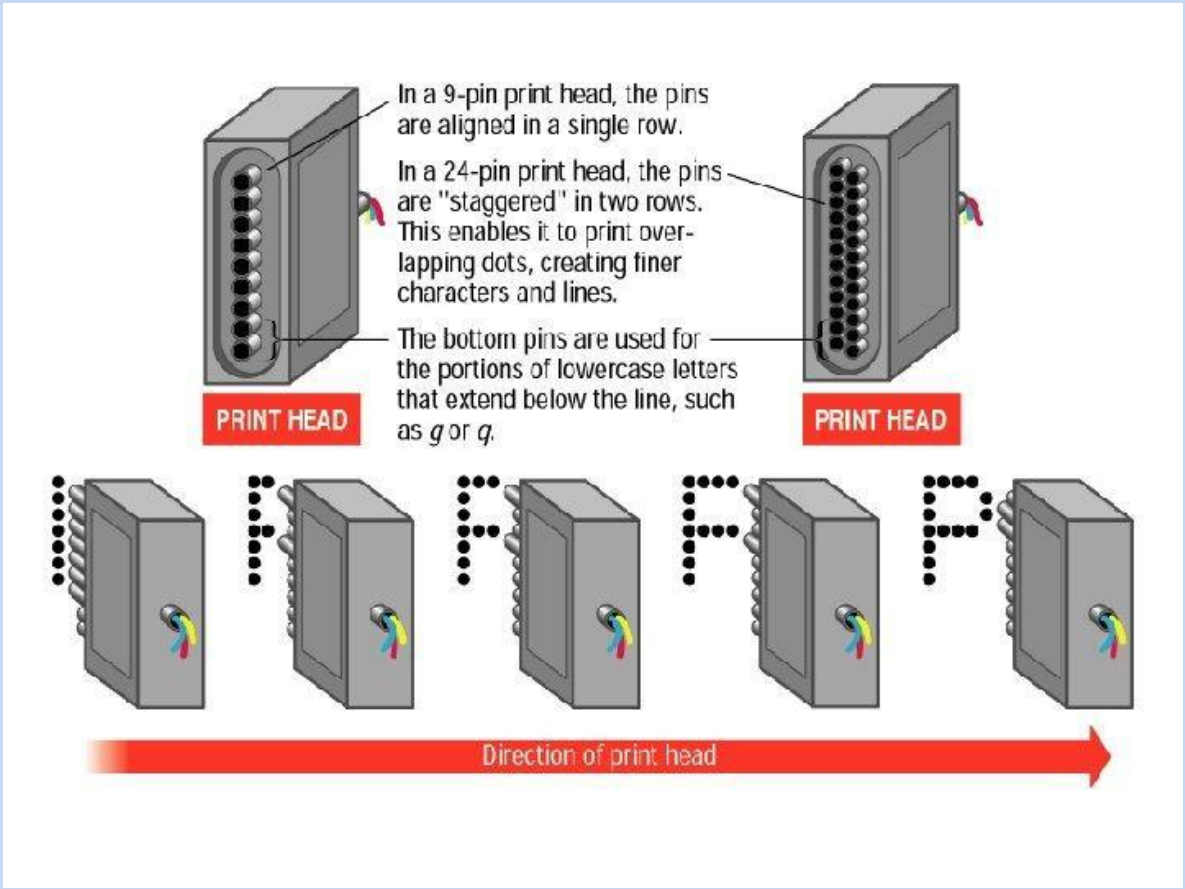
A betűk pontszerkezete jól látható volt. Ha közelről megnéztünk egy kinyomtatott karaktert, könnyen felismerhettük az egyes pontokat.

Ez azonban nem feltétlenül jelentett nagyon rossz minőséget. A fej egy sor nyomtatásakor többször is végighaladhatott ugyanazon a területen, kissé eltolt pontokat elhelyezve. Így jobb minőségű, úgynevezett **NLQ – Near Letter Quality** nyomtatást lehetett készíteni.

Ennek ára természetesen a sebesség csökkenése volt.

24 tű – amikor szebb lett a betű





A következő nagy lépést a **24 tűs nyomtatófej** jelentette.

A több tű jóval sűrűbb pontelhelyezést tett lehetővé. A betűk körvonalai simábbak lettek, az átlós és ívelt vonalak kevésbé mutatták a jellegzetes „lépcsőzetes” formát.

A 24 tűs nyomtatók ezért már lényegesen szebb leveleket és dokumentumokat készítettek. Az Epson **LQ** sorozatának neve is erre utalt: *Letter Quality*.

A különbséget legegyszerűbben ugyanannak a nagyított betűnek a képével lehet megmutatni:

9 tű → ritkább pontmátrix → durvább karakter

24 tű → sűrűbb pontmátrix → finomabb karakter

Miért volt olyan rettenetesen hangos?

Aki használt mátrixnyomtatót, valószínűleg a hangjára is emlékszik.

rrrrrr–zzzz–rrrrrr–zzzz...

Ennek egyszerű fizikai oka volt.

A nyomtatófejben lévő elektromágnesek nagy sebességgel lökték előre a tűket, amelyek mechanikusan nekicsapódtak a festékszalagnak és a papírnak. Eközben maga a viszonylag nehéz nyomtatófej is gyorsan mozgott jobbra-balra.

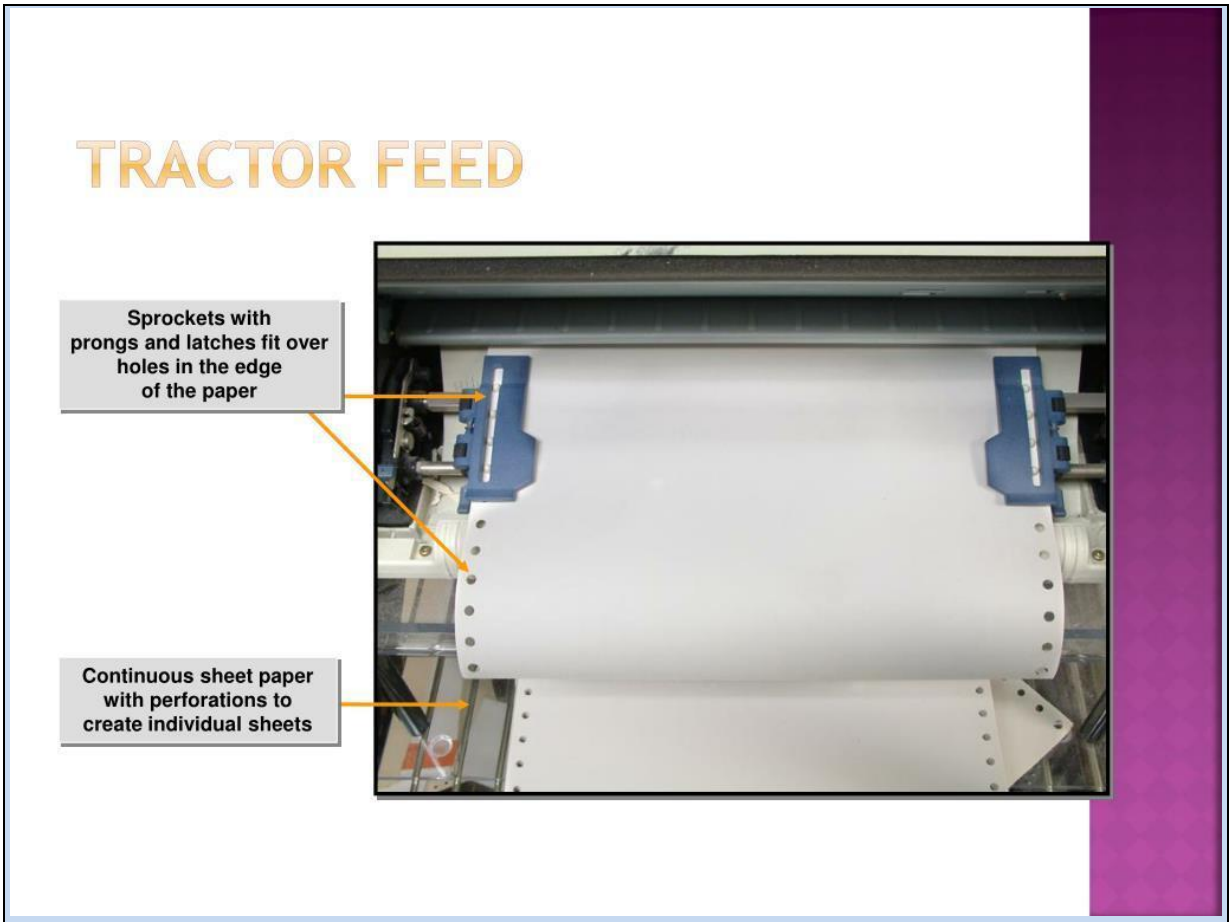
A mátrixnyomtató tehát valójában **sok apró kalapácsütéssel nyomtatott**.

A hang nem mellékes hiba volt, hanem közvetlen következménye a működési elvnek.

A leporelló és a traktor


EPSON
LQ-630K





A mátrixnyomtatók másik jellegzetessége a két szélén lyukasztott, harmonikaszerűen összehajtott **leporellópapír** volt.

A nyomtató két fogazott keréksora belekapaszkodott a papír szélén lévő lyukakba, és pontosan továbbította azt.

Ezt nevezzük **traktoros papírtovábbításnak**.

Hosszú listák, könyvelési kimutatások vagy programlisták akár több száz oldalon keresztül folyamatosan nyomtathatók voltak anélkül, hogy valakinek új papírt kellett volna behelyeznie.

Nyomtatás után a perforációnál szét lehetett választani a lapokat, a lyukasztott szélső vezetősávokat pedig letéptük.

Miért használják még ma is?

A tűs nyomtató egyik tulajdonságát a lézer- és tintasugaras nyomtató sem tudja igazán utánozni:

üt.

Ez hátrány, ha csendes nyomtatót szeretnénk – de óriási előny, ha egyszerre több példányt kell készíteni.

A tű ütése ugyanis átvihető az egymás mögött elhelyezett önátírós papírrétegekre. Így egyetlen nyomtatási művelettel több példányos számla, szállítólevél vagy más bizonylat készülhet.

Ezért a mátrixnyomtató még a lézernyomtatók korszakában sem tűnt el teljesen. Raktárakban, logisztikai rendszerekben, ipari környezetben és bizonyos ügyviteli feladatoknál továbbra is találkozhatunk vele.

A tűs nyomtató lényegében egy számítógéppel vezérelt, rendkívül gyors miniatűr kalapácssor: nem betűket nyomtat, hanem pontokat – a betű csak a pontokból áll össze.

Ábra a keretes írás fő képéhez:
9 és 24 tűs mátrixnyomtatás. A több tű sűrűbb pontmátrixot és finomabb karakterrajzolatot tett lehetővé. A technológia mechanikus ütése miatt zajos volt, ugyanakkor többpéldányos nyomtatványok készítésére különösen alkalmas – ezért teljesen még ma sem tűnt el.

6.3. Plotterek

A korai nyomtatók elsősorban betűk és számok megjelenítésére készültek. Mérnöki rajzok, térképek és diagramok elkészítéséhez másféle berendezésre volt szükség.

Ez lett a **plotter**, magyarul rajzgép.

A klasszikus tollas plotter valódi írószerszámmal rajzolt a papírra. A számítógép vezérelte a toll mozgását, illetve azt, hogy mikor érjen hozzá a papírhoz.

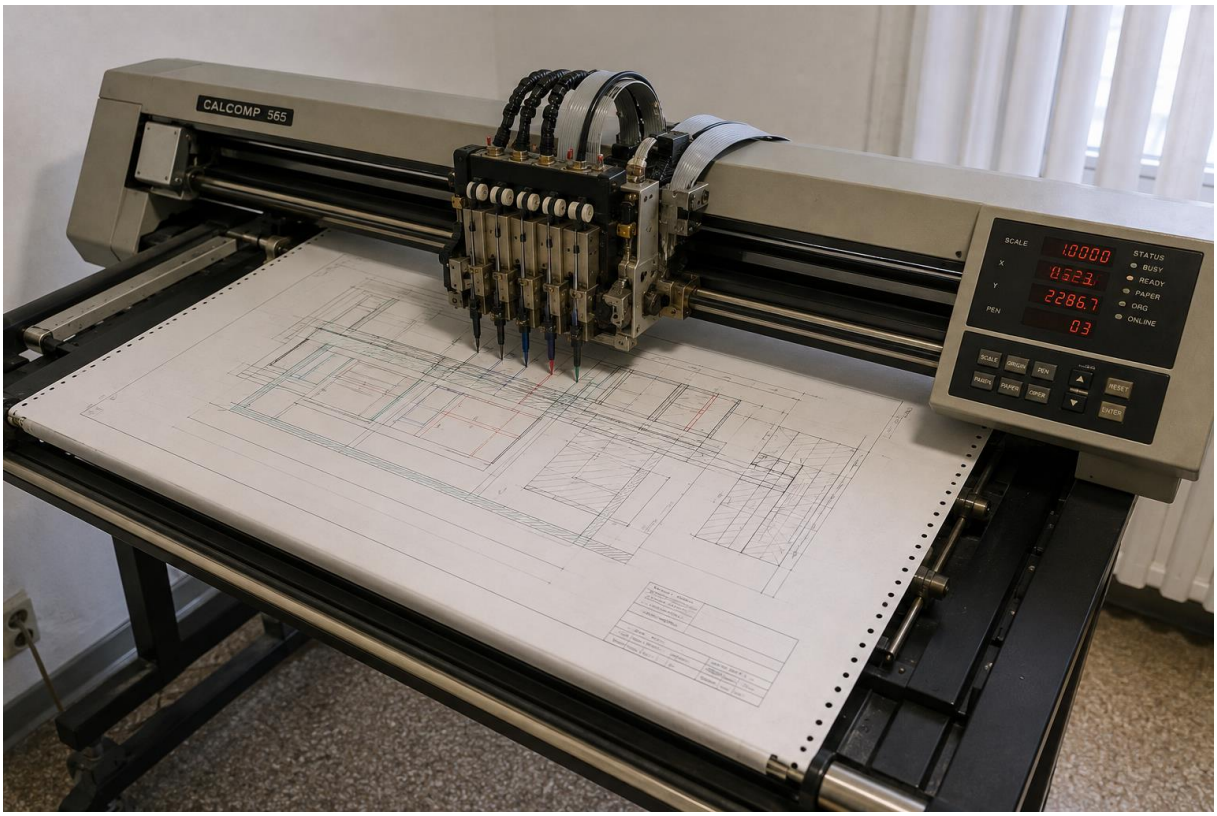
A síkágyas plotternél a papír mozdulatlan maradt, a toll pedig két irányban mozgott fölötte. A dobplotternél a papír is előre-hátra mozgott egy henger segítségével.

A plotter különösen jól tudott folytonos vonalakat rajzolni. Ezért műszaki rajzokhoz, kapcsolási rajzokhoz, építészeti tervekhez és térképekhez használták.

A színes ábrákat több különböző színű tollal készítették el. A gép az egyik tollat visszatette a tartójába, majd felvette a következőt.

A plotter működésének megfigyelése sajátos élményt nyújtott: a számítógép láthatóan és hallhatóan **megrajzolta a gondolatot.**

A hagyományos tollas plottereket később nagyméretű tintasugaras nyomtatók váltották fel, de a „plotter” elnevezés részben ezekre is áthagyományozódott.



Ábra:
Tollas plotter munka közben. A számítógép mechanikusan mozgatta a valódi írószerszámot, így mérnöki rajzokat, térképeket és diagramokat készíthetett.

6.4. CRT

A képernyős megjelenítés első meghatározó technológiája a **katódsugárcső**, angol rövidítéssel CRT volt.

A CRT eredetileg nem számítógépekhez született. Az oszcilloszkópok és televíziók már korábban alkalmazták. A számítógépes terminálok azonban hamar felismerték benne az interaktív kijelzés lehetőségét.

A katódsugárcső belsejében egy elektronágyú elektronsugarat hoz létre. Elektromágneses vagy elektrosztatikus eltérítés irányítja a sugarat a képernyő megfelelő pontjára. A belső felület fluoreszkáló anyaga az elektronok becsapódásakor felvillan.

A kép nem marad meg tartósan, ezért folyamatosan újra kell rajzolni. Ezt nevezzük **frissítésnek**.

A korai terminálok főként szöveget jelenítettek meg. A képernyőn többnyire zöld, borostyánsárga vagy fehér karakterek világítottak sötét háttér előtt.

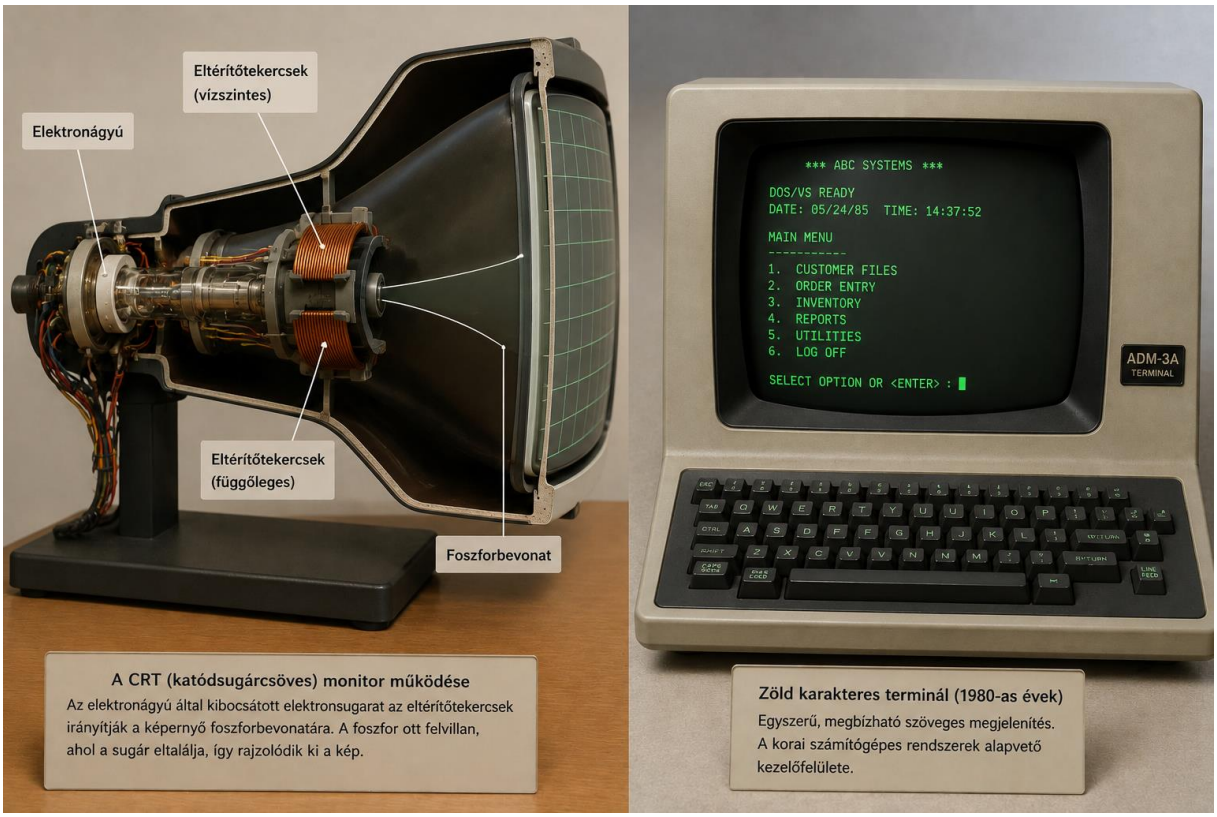
A grafikus kijelzőknél két fő megoldás alakult ki.

A **vektoros kijelző** közvetlenül a megjelenítendő vonalakat rajzolta meg az elektronsugárral. Különösen éles vonalakat adott, ezért műszaki és katonai rendszerekben is használták.

A **raszteres kijelző** soronként pásztázta végig a képernyőt, és apró képpontokból állította össze a képet. Ez lett a televíziók és számítógép-monitorok uralkodó működési elve.

A színes CRT-k képpontjai vörös, zöld és kék fényt kibocsátó részekből álltak. Ezek különböző erősségű felvillanása hozta létre a többi színt.

A CRT-monitorok évtizedeken át meghatározták a számítógépek külsejét. Jó képminőséget és gyors válaszidőt nyújtottak, ugyanakkor nagyok, nehezek és jelentős energiafogyasztásúak voltak.



Ábra:
A katódsugárcsőben vékony elektronsugár pásztázza végig a fluoreszkáló képernyőt. A kép folyamatos újrarajzolása kelti az állandó látvány érzetét.

6.5. LCD

A CRT leváltásában a **folyadékkristályos kijelző**, az LCD játszotta a legfontosabb szerepet.

A folyadékkristály különleges anyag: molekuláinak rendeződése elektromos térrel megváltoztatható, ez pedig befolyásolja a rajta áthaladó fényt.

Az LCD képpontjai önmagukban általában nem világítanak. A kijelző mögött háttérvilágítás található, a folyadékkristályos réteg pedig képpontonként szabályozza, mennyi fény jusson át.

A színes LCD minden képpontját három alpixel alkotja:

vörös, zöld és kék.

A három alpixel fényerejének változtatásával színek milliói állíthatók elő.

A korai LCD-k kis méretűek és lassúak voltak. Számológépekben, digitális órákban és műszerekben terjedtek el. A technológia fejlődésével egyre nagyobb, gyorsabb és jobb színvisszaadású kijelzőket lehetett készíteni.

A **TFT**, vagyis vékonyréteg-tranzisztoros LCD-nél minden képponthoz saját vezérlőtranzisztor tartozik. Ez pontosabb és gyorsabb képszabályozást tett lehetővé.

Az LCD egyik legnagyobb előnye, hogy lapos és könnyű. Nem igényel hosszú katódsugárcsövet, ezért a képernyő mélysége néhány centiméterre, majd milliméteres nagyságrendre csökkent.

Ez tette lehetővé a hordozható számítógépek fejlődését, majd a lapos monitorok és televíziók általános elterjedését.



Ábra:
Az LCD nem elektronsugárral rajzol képet. A folyadékkristályos cellák képpontonként szabályozzák a háttérvilágításból áthaladó fény mennyiségét.

6.6. LED és OLED

A LED, vagyis fénykibocsátó dióda elektromos áram hatására közvetlenül fényt bocsát ki.

A korai LED-kijelzők egyszerű számjegyek és jelzések megjelenítésére szolgáltak. Számológépekben, órákban, mérőműszerekben és kezelőpaneleken találkozhattunk velük.

A nagyméretű LED-kijelzők sok különálló vörös, zöld és kék fényforrásból épülnek fel. Stadionokban, hirdetőtáblákon és szabadtéri képernyőkön alkalmazzák őket.

A „LED-televízió" elnevezés azonban kissé megtévesztő. Ezek többnyire továbbra is LCD-kijelzők, csak a háttérvilágításukat biztosítják fénycsövek helyett LED-ek.

Az **OLED – Organic Light-Emitting Diode** másképp működik. Minden egyes képpont saját fényt bocsát ki, ezért nincs szükség külön háttérvilágításra.

Ha egy OLED-képpont fekete, egyszerűen kikapcsolható. Ez nagyon mély feketéket és nagy kontrasztot eredményez.

Az OLED-rétegek vékonyak lehetnek, sőt hajlítható hordozóra is felvihetők. Így ívelt, összehajtható vagy feltekerhető kijelzők is készíthetők.

A technológiának ugyanakkor hátrányai is vannak. A szerves fénykibocsátó anyagok idővel öregedhetnek, és tartósan megjelenített képelemek nyoma bizonyos esetekben visszamaradhat.

A kijelző fejlődésének iránya mégis jól látható:

a nagy és nehéz képcsőtől a falra akasztható, majd összehajtható fénykibocsátó felületig.



Ábra:
A LED közvetlenül fényt bocsát ki. Az OLED-kijelzőben minden képpont önálló fényforrás, ezért nincs szükség háttérvilágításra, és rendkívül vékony, akár hajlítható kijelző is készíthető.

Amikor a monitorkártyából számítógép lett

A mai videokártyák önálló processzorral, több gigabájt saját memóriával és elképesztő számítási teljesítménnyel rendelkeznek. Az első PC-k videokártyáinak feladata ehhez képest nagyon egyszerű volt:

jeleket kellett előállítaniuk a monitor számára.

Innen ered a régebbi magyar elnevezés is: **monitorvezérlő kártya**.

Az IBM PC megjelenése után azonban alig másfél évtized alatt óriási fejlődés zajlott le:

MDA, CGA → Hercules → EGA → VGA → SVGA → 2D gyorsító → 3D gyorsító → GPU

MDA – amikor a betű volt a fontos





Az IBM az 1981-es PC-hez két alapvetően eltérő megjelenítési rendszert kínált.

Az egyik az **MDA – Monochrome Display Adapter** volt.

Az MDA kiváló minőségű, 80×25 karakteres szöveget jelenített meg. A karakterek 9×14 képpontos cellákban jelentek meg, ezért a kép az akkori viszonyok között rendkívül éles volt.

Grafikát azonban az eredeti MDA nem tudott.

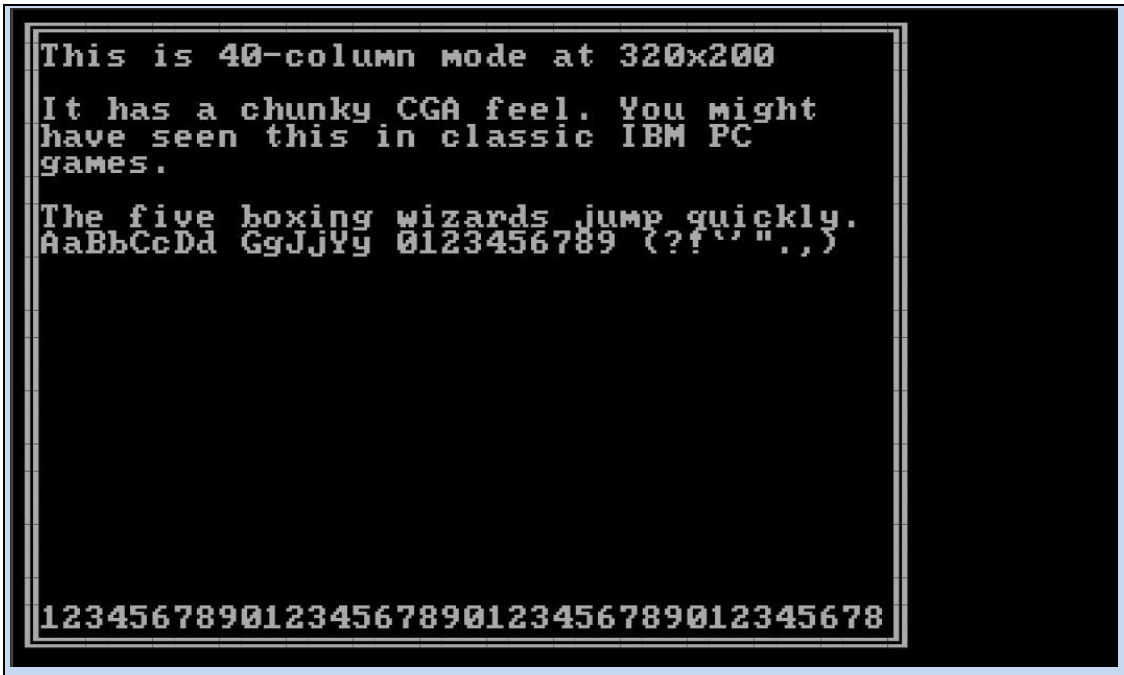
Üzleti felhasználásra ez nem feltétlenül jelentett problémát. Szövegszerkesztéshez, táblázatkezeléshez és adatbázisokhoz sokkal fontosabb volt a jól olvasható szöveg.

Ábra:
IBM MDA monitorvezérlő. Kiváló minőségű monokróm szöveget jelenített meg, grafikus üzemmódja azonban nem volt.

CGA – megjelennek a színek







Ábra:
CGA-grafika. A 320×200 képpontos, négy színt használó kép a korai PC-játékok egyik jellegzetes látványvilágává vált.

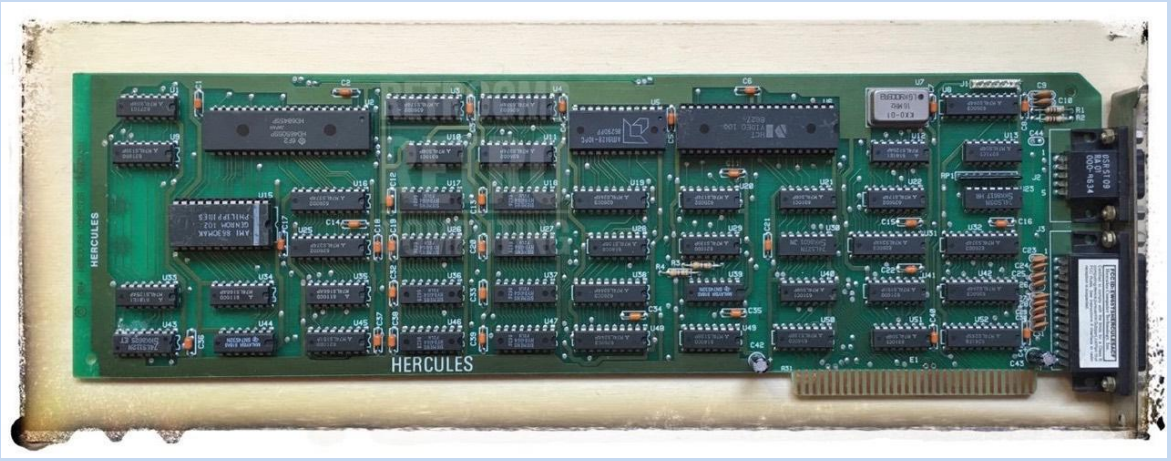
A másik IBM-megoldás a **CGA – Color Graphics Adapter** volt.

Ez már grafikát és színeket is meg tudott jeleníteni. Legismertebb grafikus üzemmódja 320×200 képpontot kínált négy egyidejű színnel. 640×200-as felbontásban alapkiépítésben két szín állt rendelkezésre.

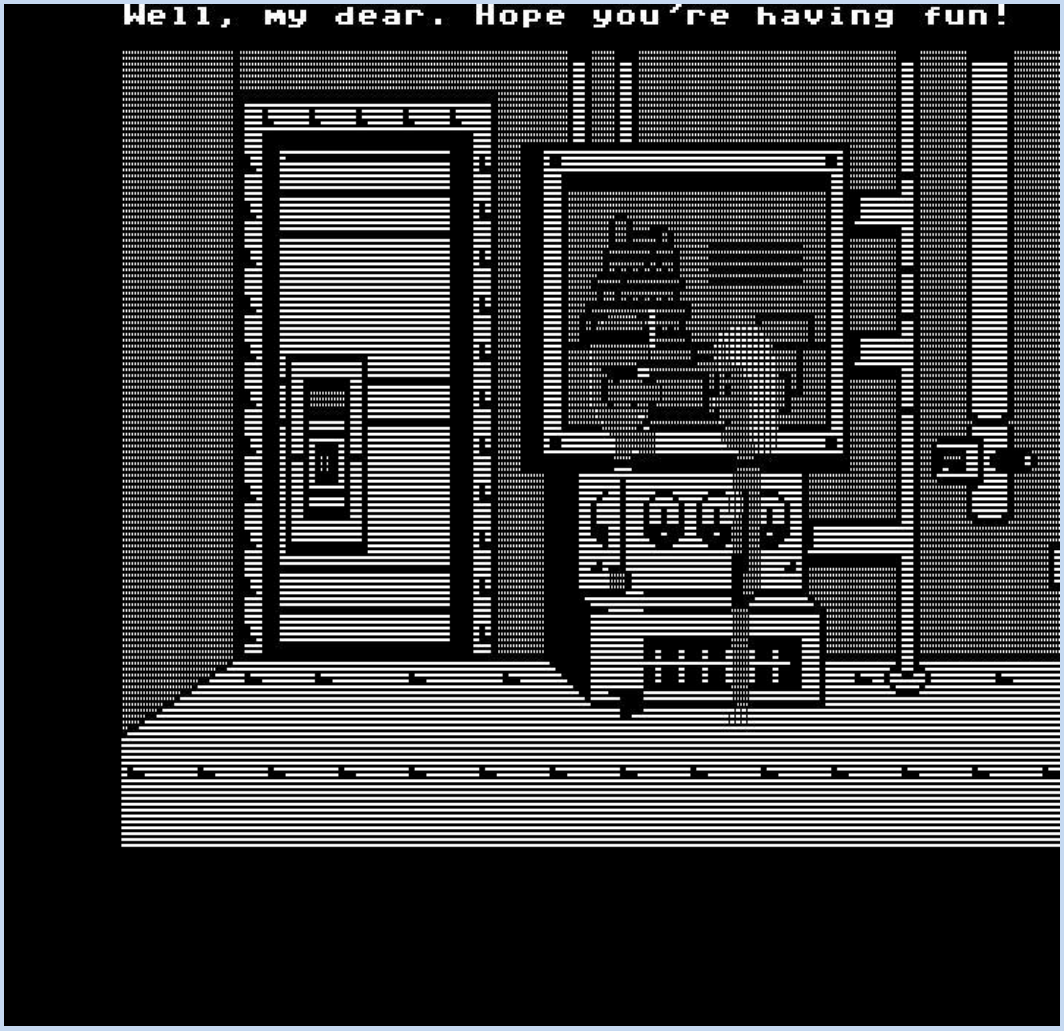
Mai szemmel kezdetlegesnek tűnik, de a PC ezzel valódi grafikus számítógéppé vált.

A CGA különösen a játékok fejlődésében lett fontos.

Hercules – a monokróm monitor megtanul rajzolni







Ábra:
Hercules Graphics Card. Az MDA-kompatibilis, jól olvasható szöveg mellé nagy felbontású, 720×348 képpontos monokróm grafikát kínált.

1982-ben jelent meg a **Hercules Graphics Card**, amely érdekes kompromisszumot kínált.

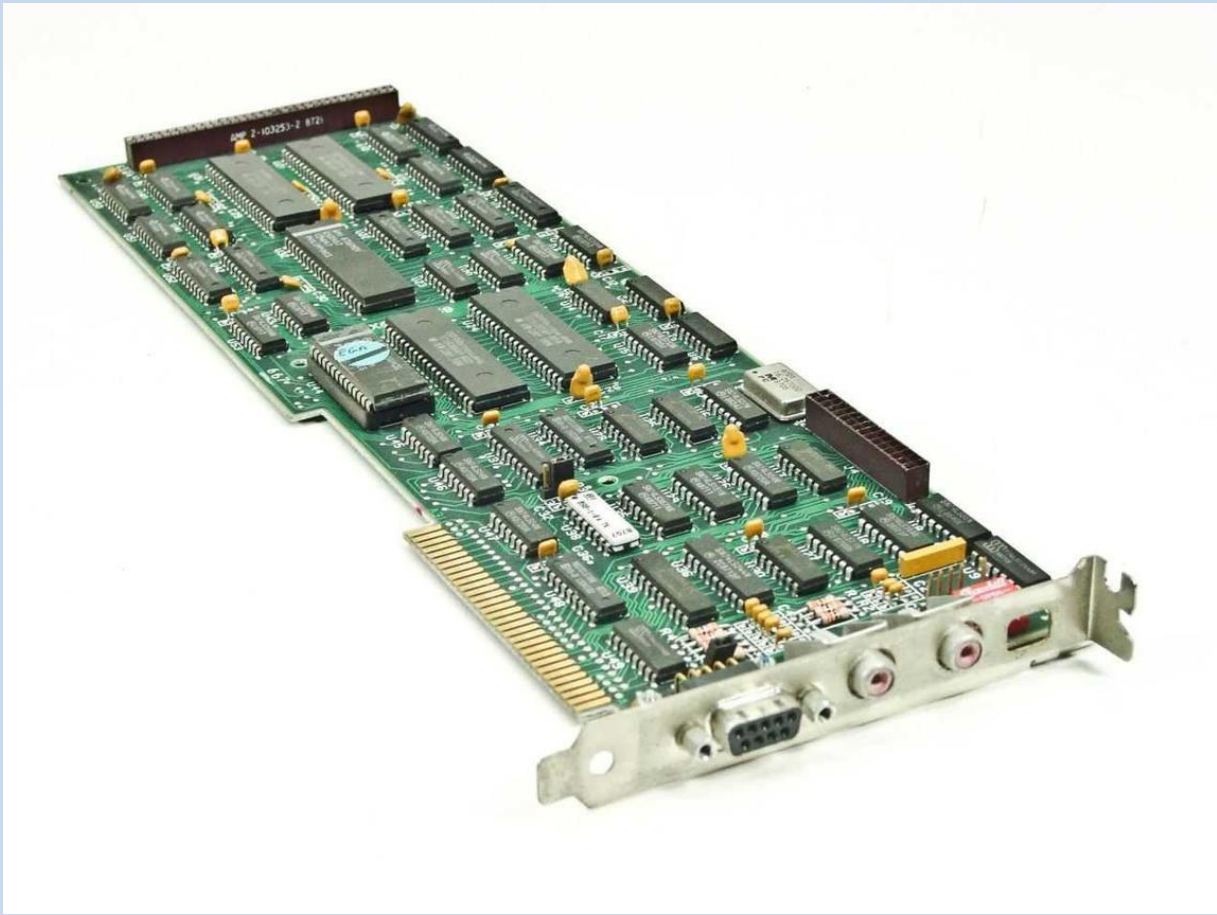
Megőrizte az MDA kiváló minőségű szövegét, ugyanakkor **720×348 képpontos monokróm grafikus üzemmódot** is biztosított.

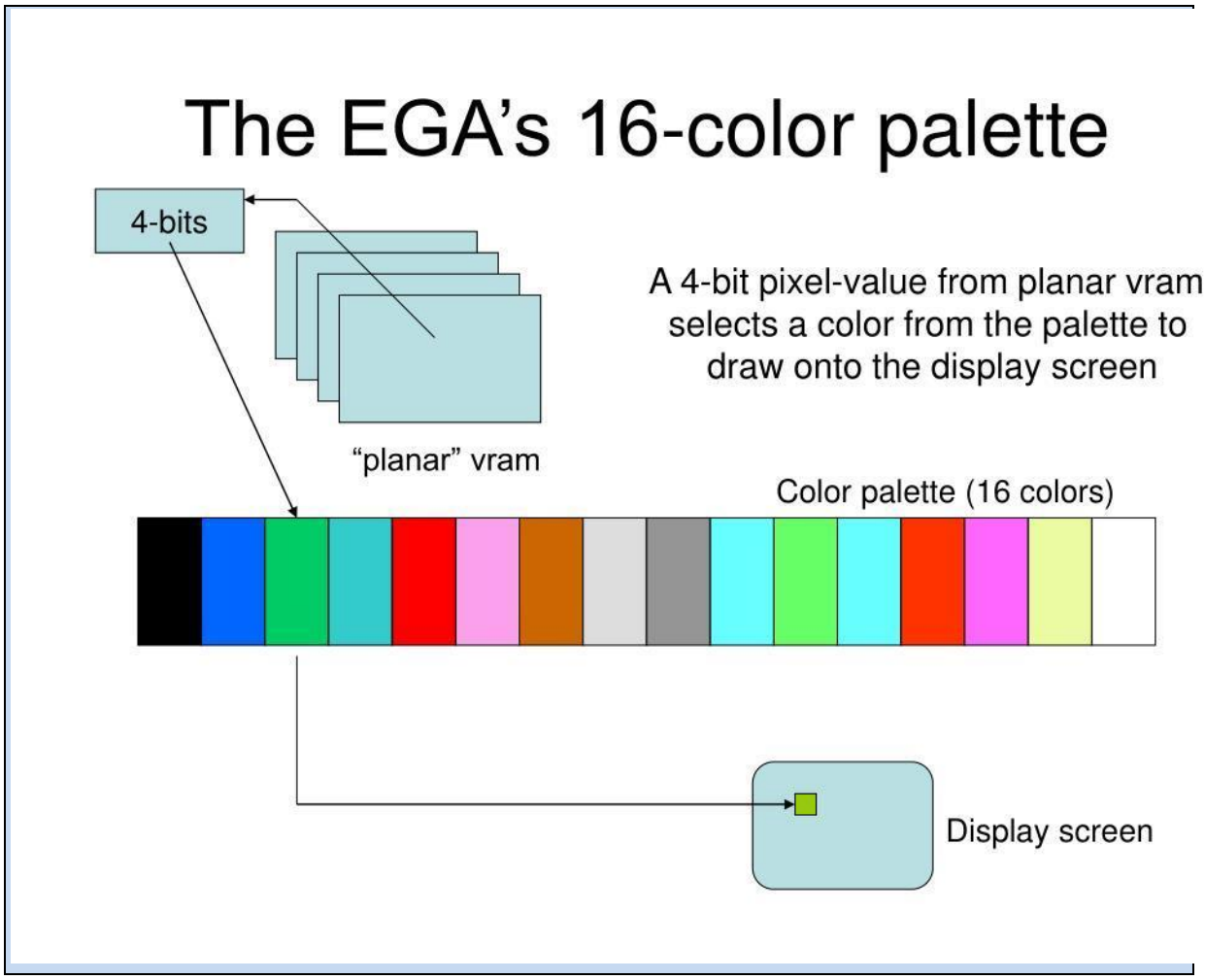
Ez a CGA-nál lényegesen nagyobb grafikai felbontást jelentett.

A Hercules különösen az üzleti világban vált népszerűvé. Táblázatok, diagramok, CAD-programok és grafikus alkalmazások használói úgy kaptak nagy felbontású grafikát, hogy nem kellett lemondaniuk az éles monokróm szövegről.

Az IBM nem ezt a szabványt találta ki – a piac mégis széles körben elfogadta.

EGA – 16 szín egyszerre




The EGA's 16-color palette
4-bits
A 4-bit pixel-value from planar vram selects a color from the palette to draw onto the display screen
"planar" vram
Color palette (16 colors)
Display screen



1984-ben az IBM bemutatta az **EGA – Enhanced Graphics Adaptert**.

Legjobb üzemmódjában 640×350 képpontos felbontást és egyszerre **16 színt** kínált egy 64 színű palettából.

A különbség látványos volt. A játékok, grafikai programok és üzleti alkalmazások sokkal részletesebb és színesebb képet jeleníthettek meg.

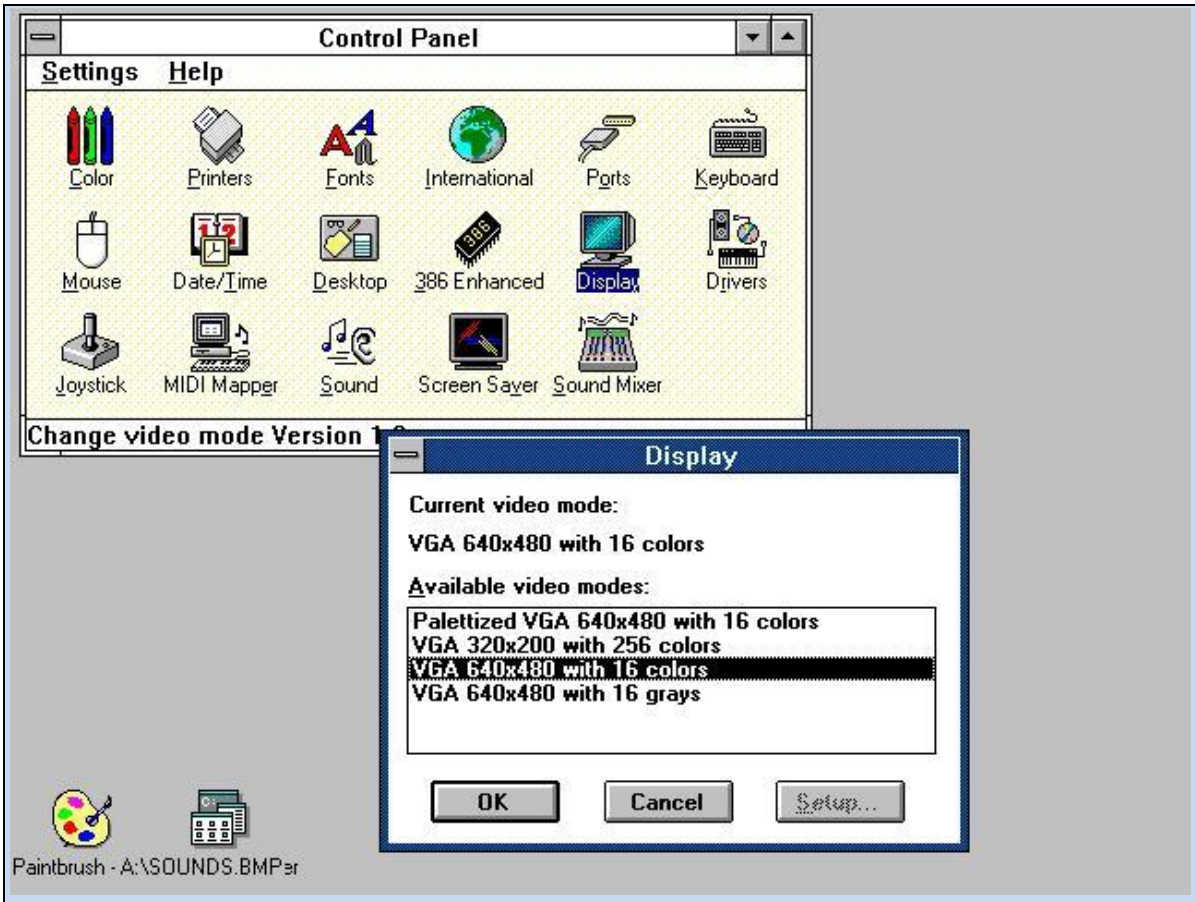
Az EGA azonban csak néhány évig uralkodott.

A következő lépés olyan sikeres lett, hogy neve máig fennmaradt.

VGA – a szabvány, amely túlélte önmagát




FLOOR
10
SCORE
112300
LIVES
5
HEALTH
94%
AMMO
43
thandor.net



Az IBM 1987-ben, a PS/2 számítógépcsaláddal vezette be a **VGA – Video Graphics Array** rendszert.

Két üzemmódja különösen fontossá vált:

640×480 képpont, 16 szín

és

320×200 képpont, 256 szín.

Utóbbi óriási hatással volt a PC-játékokra. A 256 egyidejű szín már lényegesen gazdagabb képi világot tett lehetővé.

A VGA másik nagy öröksége a **15 pólusú analóg VGA-csatlakozó** lett, amely még évtizedekkel később is ott volt a számítógépeken és monitorokon.

Az IBM azonban ekkor már kezdte elveszíteni a PC-szabványok feletti ellenőrzését. Más gyártók VGA-kompatibilis, de annál nagyobb felbontású és több színt kezelő kártyákat készítettek.

Megszületett a gyűjtőnéven **SVGA-nak** nevezett világ.

Amikor a videokártya már számolni kezdett

A nagyobb felbontás és színmélység egyre több adat mozgatását igényelte.

A processzornak például ablakokat kellett mozgatnia, területeket kitöltenie, vonalakat rajzolnia és képrészleteket másolnia.

Miért végezze mindezt a CPU?

A videokártyák ezért saját áramköröket kaptak bizonyos grafikai műveletek gyorsítására.

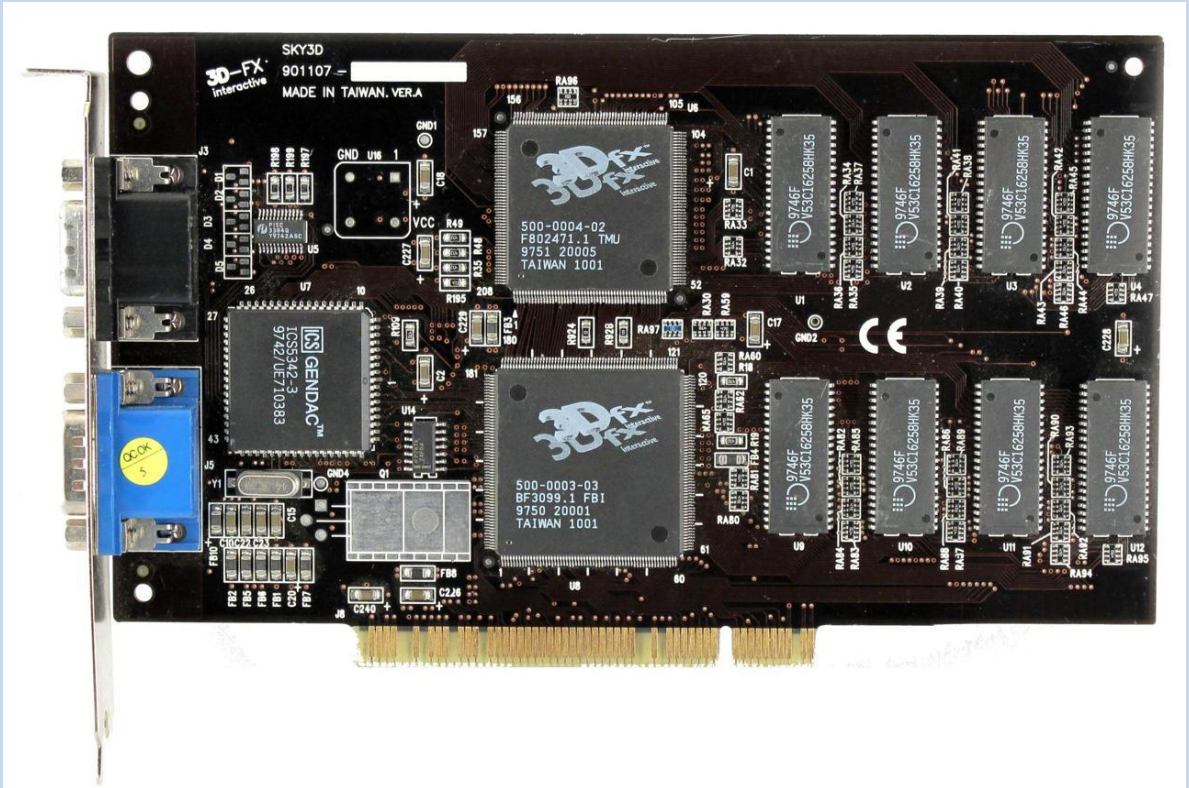
Megjelentek a **2D grafikus gyorsítók**.

Az olyan gyártók, mint az S3, ATI, Matrox vagy Tseng Labs egyre intelligensebb grafikus vezérlőket készítettek.

A monitorkártya már nem egyszerűen képpontokat tárolt és küldött a monitorra.

Elkezdte maga előállítani a képet.

3dfx Voodoo – a 3D forradalom



Ábra:
3dfx Voodoo Graphics. A kártya csak a háromdimenziós grafika gyorsításával foglalkozott, ezért egy hagyományos 2D videokártya mellett működött. Megjelenése mérföldkő volt a PC-s játékok történetében.

Az 1990-es évek közepén a háromdimenziós játékok új problémát jelentettek.

Egy 3D-s világ megjelenítéséhez perspektivikus transzformációkat, textúrázást, mélységvizsgálatot, színkeverést és rengeteg más műveletet kellett nagyon gyorsan elvégezni.

1996-ban jelent meg a **3dfx Voodoo Graphics**.

A Voodoo különlegessége, hogy eredetileg **csak a 3D grafikát gyorsította**. A számítógépben továbbra is szükség volt egy hagyományos 2D videokártyára. A kettőt egy külső átvezető kábellel kapcsolták össze.

Ha Windowsban dolgoztunk, a 2D kártya készítette a képet.

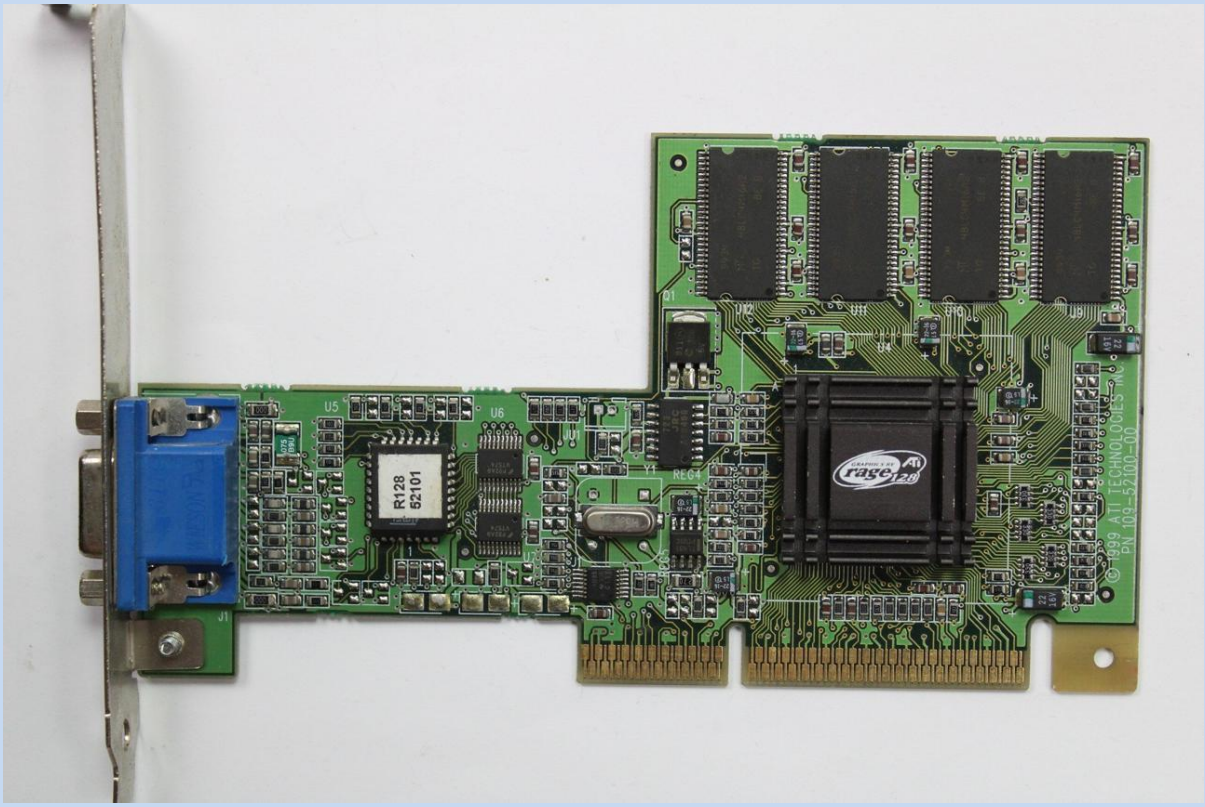
Ha elindítottunk egy támogatott 3D-s játékot, átvette a munkát a Voodoo.

A különbség megdöbbentő lehetett: nagyobb felbontás, simább mozgás, textúrázott felületek és látványos fényhatások jelentek meg.

A 3dfx saját **Glide** grafikus API-ja pedig a korszak számos játékában rendkívül jó teljesítményt nyújtott.

A későbbi **Voodoo2** még egy különlegességet hozott: két kártyát lehetett összekapcsolni **SLI** módban a teljesítmény növelésére.

ATI, NVIDIA – megszületik a GPU





A 3D gyorsítás sikere után megindult a verseny.

Az ATI, NVIDIA, Matrox, S3, 3dfx és más gyártók egyre nagyobb teljesítményű grafikus chipeket készítettek. A külön 2D és 3D egységek fokozatosan egyetlen kártyában egyesültek.

1999-ben az NVIDIA a **GeForce 256** bemutatásakor kezdte hangsúlyosan használni a **GPU – Graphics Processing Unit** elnevezést.

A grafikus processzor ekkor már olyan feladatokat is átvett a CPU-tól, mint a geometriai transzformációk és a megvilágítás számítása.

A fejlődés ezután felgyorsult.

A fix funkciójú grafikus áramköröket programozható **shader egységek** egészítették ki. Ezekből később nagyszámú, párhuzamosan működő számolóegység fejlődött ki.

És itt következett a történet váratlan fordulata.

A grafikus kártyától a mesterséges intelligenciáig

A háromdimenziós grafika előállításához rengeteg hasonló matematikai műveletet kell egyszerre elvégezni.

Kiderült, hogy ugyanez a tulajdonság más feladatoknál is rendkívül hasznos.

Tudományos számítások.

Szimulációk.

Képfeldolgozás.

Kriptográfia.

És végül **neurális hálózatok.**

A GPU így fokozatosan általános célú párhuzamos számítási eszközzé vált.

A fejlődés különös kört írt le:

Az első monitorkártya csak megmutatta, amit a CPU kiszámolt. A mai GPU már maga végzi a számítás jelentős részét – és néha sokkal nagyobb számítási teljesítménnyel rendelkezik, mint a mellette dolgozó CPU.

Egy korszak egyetlen sorban

MDA – szöveg
→ CGA – szín és grafika
→ Hercules – nagy felbontás
→ EGA – több szín
→ VGA – általános grafikus szabvány
→ SVGA – nagyobb felbontás
→ 2D gyorsító – a kártya rajzolni kezd
→ 3dfx Voodoo – 3D gyorsítás
→ GPU – programozható párhuzamos processzor
→ GPGPU/AI – a grafikus kártya már nemcsak grafikát számol

Ez azért különösen jó keretes történet, mert **összeköti a számítástechnika két korszakát**: az IBM PC monitorvezérlőjét és a mai AI-gyorsítókat. A történet elején a kártya feladata néhány karakter kirajzolása; a végén ugyanennek a technológiai ágnak a leszármazottai neurális hálózatokat futtatnak.

6.7. Hang

A számítógép válasza nemcsak látható, hanem hallható is lehet.

A legkorábbi gépek hangja többnyire nem szándékos kimenet volt. Relék kattogtak, ventilátorok zúgtak, nyomtatók csattogtak, mágnesszalagok és lemezegységek mozogtak.

A kezelők azonban hamar felfedezték, hogy a gép működése hanggá alakítható. Egyes korai számítógépeknél a számítások elektromágneses zavarait rádióvevővel tették hallhatóvá, sőt egyszerű dallamokat is előállítottak velük.

A személyi számítógépek kezdetben egyszerű hangszórót kaptak. Ez többnyire csak egyetlen hangmagasságú vagy négyszögjelszerű csipogást tudott létrehozni. Mégis elegendő volt figyelmeztetésekhez és egyszerű játékhangokhoz.

Később megjelentek a több hangcsatornás hanggenerátorok és a hangkártyák. A számítógép már nemcsak előre meghatározott elektronikus hangokat, hanem digitalizált beszédet, zenét és hangeffektusokat is lejátszhatott.

A hang digitalizálásakor a levegő nyomásváltozását rendszeres időközönként megmérjük. Ezt nevezzük **mintavételezésnek**. Minél gyakrabban és minél pontosabban mérjük meg a hangot, annál hűebben lehet visszaállítani.

A hangkimenet szerepe sokrétű:

- jelzi a rendszer állapotát;
- zenét és beszédet közvetít;
- játékok és filmek térhatását erősíti;
- felolvassa a képernyő tartalmát;
- akadálymentesíti a számítógép használatát.

A hang különösen fontos a látássérült felhasználók számára. A képernyőolvasó programok beszéddé alakítják a feliratokat, menüket és dokumentumokat.

A gép ekkor már nemcsak megmutatja, hanem **el is mondja a választ.**



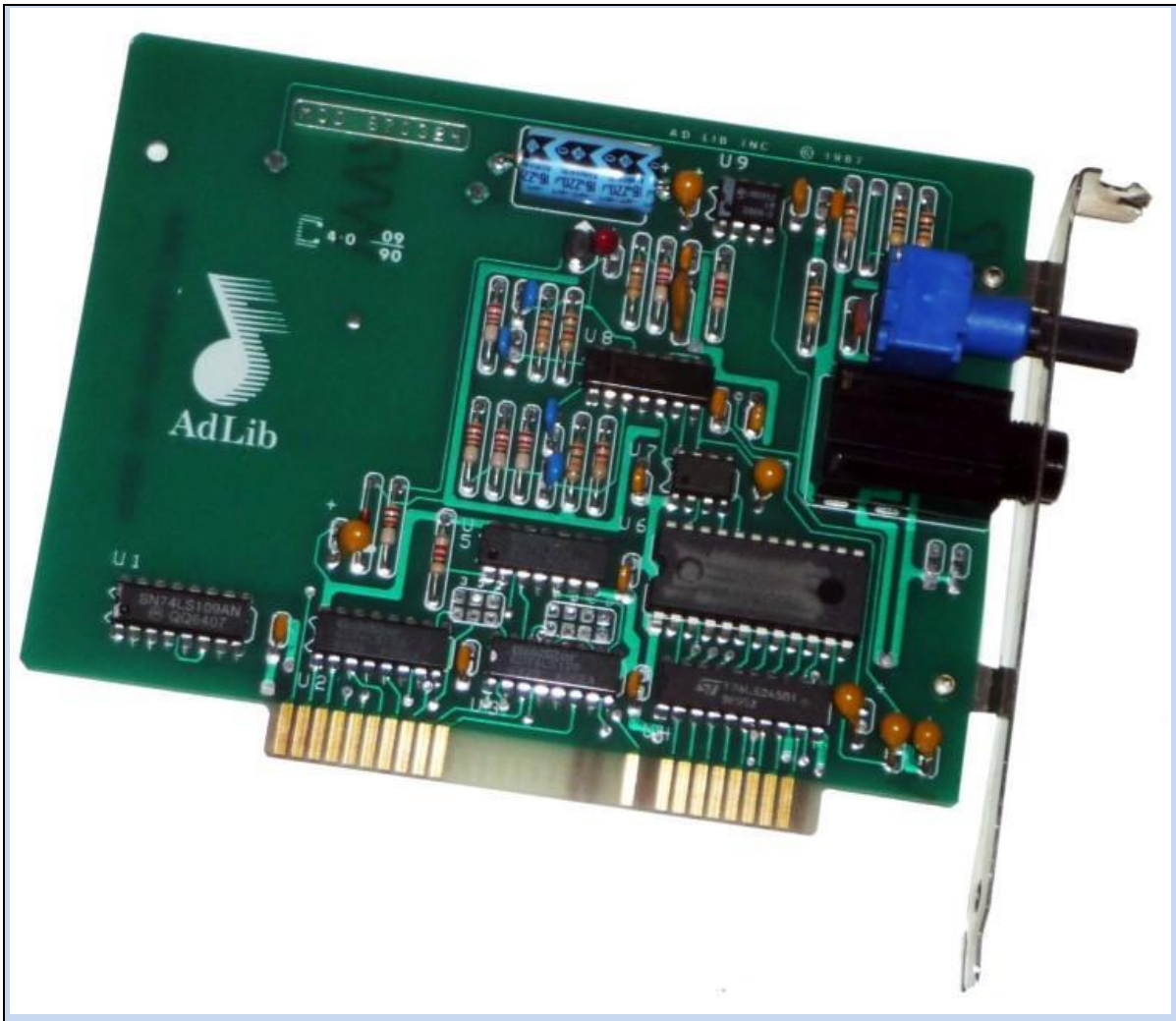
Ábra:
A számítógépes hang fejlődése az egyszerű csipogástól a digitalizált beszédig, zenéig és térhatású hangzásig vezetett. A hang ma már nemcsak szórakoztat, hanem fontos kezelő- és akadálymentesítő felület is.

Amikor a PC megtanult zenélni

Az első IBM PC-k hangképessége finoman szólva szerény volt. A gépbe épített apró hangszóró, a **PC Speaker** alapvetően egyszerű négyszögjelek megszólaltatására volt alkalmas. Csipogott, sípolt, és ügyes programozással még dallamokat is ki lehetett csalni belőle – de valódi zenei hangzásra nem tervezték.

Az 1980-as évek végén azonban megjelentek a külön megvásárolható hangkártyák, és néhány év alatt teljesen megváltoztatták a PC-k világát.

AdLib – amikor a PC zenélni kezdett



Ábra:
AdLib Music Synthesizer Card. A Yamaha OPL2 chip FM-szintézissel állította elő a zenét. Az 1980-as évek végének PC-játékai ezzel kaptak először igazán jellegzetes zenei hangzást.

Az egyik első igazán sikeres PC-s hangkártya az **AdLib Music Synthesizer Card** volt, amely 1987-ben jelent meg.

Lelke a Yamaha **YM3812**, ismertebb nevén **OPL2** hanggenerátor volt. Ez nem előre rögzített hangmintákat játszott vissza, hanem **FM-szintézissel** állította elő a hangokat.

A módszerrel különböző hangszerekhez hasonló hangszíneket lehetett létrehozni. Mai füllel az AdLib jellegzetes elektronikus hangzása könnyen felismerhető, de a PC Speakerhez képest óriási ugrást jelentett.

A játékfejlesztők gyorsan felfedezték, és az AdLib hamarosan gyakorlatilag szabvánnyá vált a PC-s játékzenében.

Sound Blaster – zene és valódi hang



Ábra:
Sound Blaster 16. Az AdLib-kompatibilis FM-zene mellett már jó minőségű digitalizált hangot is kezelhetett. A Sound Blaster család hosszú időre a PC-s hang de facto szabványává vált.

1989-ben a Creative Labs bemutatta az első **Sound Blastert.**

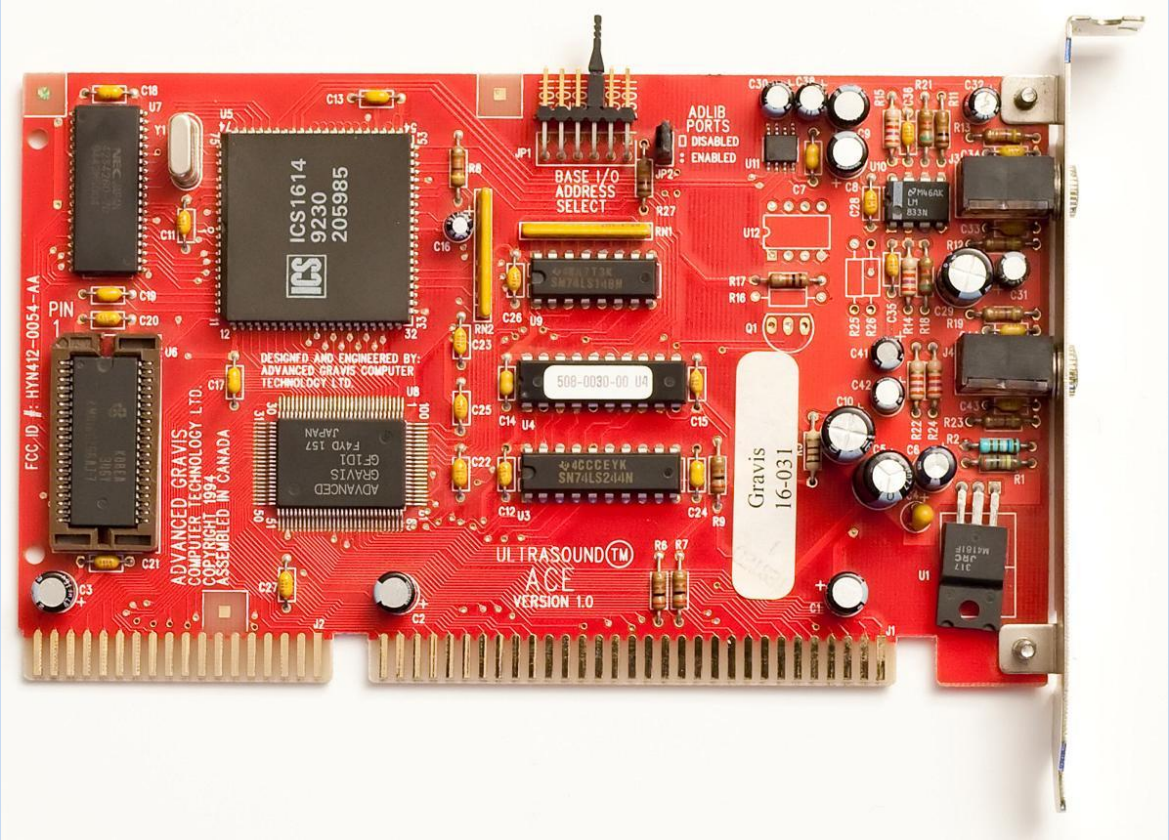
A siker egyik titka az volt, hogy az AdLibbel kompatibilis FM-hanggenerátort is tartalmazott, ezért a már elkészült játékok zenéjét is meg tudta szólaltatni.

De ennél többet kínált: **digitalizált hangot is képes volt lejátszani.**

Ez alapvető változás volt. A számítógép már nemcsak elektronikusan előállított hangszíneket játszhatott, hanem digitalizált emberi beszédet, zajokat és valódi hangfelvételeket is.

A későbbi Sound Blaster Pro, majd különösen az 1992-es **Sound Blaster 16** tovább javította a hangminőséget. A Sound Blaster név annyira meghatározóvá vált, hogy a „Sound Blaster kompatibilis” hosszú ideig szinte a PC-s hangkártya szinonimája lett.

Gravis UltraSound – egy másik út



Ábra:
Gravis UltraSound. A kártya FM-szintézis helyett memóriában tárolt digitális hangmintákból építette fel a zenét. Különösen a demoscene és a tracker zenészek körében vált legendássá.

1992-ben jelent meg a **Gravis UltraSound**, röviden GUS.

Működési filozófiája alapvetően eltért az AdLib és a Sound Blaster FM-szintézisétől. A GUS **hangmintákból, wavetable jelleggel** állította elő a zenét.

A hangszerek valódi hangjáról készített digitális mintákat tárolhatott a kártya saját memóriájában, majd ezeket különböző hangmagasságokon játszhatta le és keverhette össze.

Az eredmény sok esetben lényegesen természetesebb volt az FM-szintézis jellegzetesen elektronikus hangjánál.

A Gravis UltraSound különösen a **demoscene** és a tracker zenék világában vált legendássá. Hardveresen sok hangcsatornát tudott keverni anélkül, hogy ezzel jelentősen terhelte volna a processzort.

Mégsem tudta kiszorítani a Sound Blastert.

A PC-s világban ugyanis nem mindig a technikailag legjobb megoldás győz. A Sound Blaster addigra olyan széles körben támogatott szabvánnyá vált, hogy a játékfejlesztők elsősorban arra optimalizáltak. A kompatibilitás erősebbnek bizonyult a technikai előnynél.

Három korszak, három megoldás

A három kártya szépen bemutatja a PC-hang fejlődését:

AdLib → a hangot kiszámítjuk
FM-szintézis segítségével elektronikus hangszíneket hozunk létre.

Sound Blaster → a hangot digitalizáljuk
A szintetizált zene mellett már felvett hangot, beszédet és hangeffektusokat is lejátszunk.

Gravis UltraSound → hangmintákból zenélünk
Valódi hangszerek digitalizált mintáiból építjük fel a zenét.

A fejlődés következő állomása végül különös módon a hangkártya **eltűnése** lett.

Ahogy az alaplapok egyre több funkciót integráltak, a hangfeldolgozás is beköltözött az alaplapra. A legtöbb felhasználónak már nem volt szüksége külön bővítőkártyára.

Ami az 1990-es években még drága és büszkén mutogatott ISA-bővítőkártya volt, ma néhány apró chip az alaplapon – vagy részben szoftveresen végzett feladat.

A PC először csak csipogott. Aztán megtanult zenélni, beszélni és élethű hangokat megszólaltatni. Végül a hangkártya olyan sikeres lett, hogy szinte láthatatlanná vált.

Ehhez még egy apró történeti érdekességet is betennék: **a MIDI-t és a Roland MT-32/Sound Canvas vonalat**. Ez az AdLib–Sound Blaster–GUS történettel párhuzamosan futott, és a korabeli játékok csúcskategóriás hangzását képviselte.

Akkor ezt a hangkártyás keret végére tenném egy rövid **„A luxus hangzás: MIDI és Roland”** alfejezetként. Azért is érdekes, mert megmutatja: ugyanaz a játék egészen másként szólhatott attól függően, milyen hardver volt a géphez kapcsolva.

A luxus hangzás: MIDI és Roland





Ábra:
Roland MT-32 és Sound Canvas: a PC-játékok „prémium hangrendszerei”. A számítógép MIDI-parancsokat küldött a külső szintetizátornak, amely saját hanggenerátorával szólaltatta meg a zenét.

Az 1980-as évek végén a PC-játékok legjobb zenéjét nem feltétlenül egy hangkártya szolgáltatta. A számítógép **MIDI-üzenetekkel külső szintetizátort is vezérelhetett.**

A MIDI nem magát a digitalizált hangot továbbítja. Inkább egy elektronikus kotta: megmondja, melyik hang szólaljon meg, milyen hangszeren, milyen erősen és meddig. A tényleges hangot a hozzákapcsolt szintetizátor állítja elő.

A játékosok számára legendássá vált az 1987-ben megjelent **Roland MT-32**. A készüléket eredetileg zenészeknek szánták, de több játékfejlesztő – különösen a Sierra – felismerte, milyen látványos, pontosabban hallványos ugrást jelenthet a PC-játékok zenéjében.

Az MT-32 részben mintákra épülő szintézist alkalmazott, ezért az AdLib FM-hangzásánál sokkal gazdagabb, természetesebb hangszíneket tudott előállítani.

Az 1991-es **Roland Sound Canvas SC-55** pedig a General MIDI korszak egyik meghatározó eszközévé vált. A szabványosított hangszerkiosztásnak köszönhetően ugyanaz a MIDI-zene különböző kompatibilis eszközökön is nagyjából a szerző szándékának megfelelően szólalhatott meg.

A különbség néha meghökkentő volt. Ugyanaz a játék PC Speakerrel csipogott, AdLibbel már zenélt, Sound Blasterrel digitalizált hanghatásokat adott – **Roland modullal pedig szinte egy kis zenekarrá változott.**

PC Speaker → AdLib → Sound Blaster → Gravis UltraSound / MIDI → integrált digitális hang

6.8. Virtuális és kiterjesztett valóság

A hagyományos képernyő olyan, mint egy ablak: rajta keresztül nézzük a számítógép által létrehozott világot.

A virtuális valóságban azonban úgy érezhetjük, mintha **belépnénk ebbe a világba.** A **virtuális valóság**, röviden VR, a felhasználó látóterét számítógép által előállított képpel tölti ki. A fejre helyezett kijelző mindkét szemnek kissé eltérő képet mutat, így térhatás jön létre.

A rendszer érzékeli a fej mozgását is. Amikor a felhasználó elfordítja a fejét, a számítógép ennek megfelelően változtatja meg a képet. Ha ez elég gyorsan történik, az agy úgy értelmezi, mintha valóban körülnéznénk egy térben.

A VR nem egyszerűen háromdimenziós képernyő. A jelenlét érzetét több érzékszerv összehangolt ingerlése teremti meg: **látvány, térhatású hang, mozgásérzékelés és esetenként tapintási visszajelzés.**

A **kiterjesztett valóság**, röviden AR, nem helyettesíti a való világ képét, hanem digitális elemeket helyez rá.

Egy szemüveg vagy telefon kameraképén például megjelenhet:

- navigációs nyíl;
- egy gép szerelési utasítása;
- egy múzeumi tárgy magyarázata;
- egy épület még meg nem épült része;
- egy játék virtuális szereplője.

A VR tehát egy másik világot mutat, az AR pedig **kiegészíti a valóságosat**.

A kettő között számos átmeneti forma létezik. Ezeket gyakran kevert valóságnak nevezik, amikor a virtuális tárgyak már a környezet térbeli szerkezetéhez is igazodnak. Egy digitális figura például eltűnhet egy valódi asztal mögött, vagy látszólag ráállhat annak lapjára.

A virtuális és kiterjesztett valóság nemcsak játékra alkalmas. Használható oktatásban, orvosi képzésben, tervezésben, katonai és ipari szimulációkban, távoli együttműködésben és kulturális örökség bemutatásában is.

A fejlődés itt új szintre jutott. A számítógép válasza már nem feltétlenül egy előttünk álló tárgyon jelenik meg.

Körülvesz bennünket, és a tér részévé válik.



Ábra:
A virtuális valóság számítógéppel létrehozott környezetbe helyezi a felhasználót, a kiterjesztett valóság pedig digitális információval egészíti ki a valódi világ látványát.

A gép válaszának útja

A számítógépes kimenet története egyben az ember és gép közötti távolság csökkenésének története is.

Kezdetben a kezelőnek meg kellett tanulnia, melyik lámpa melyik bitet jelenti. Később a gép már szöveges választ nyomtatott, majd képet rajzolt és hangot adott.

Ma pedig olyan környezetet hozhat létre, amelyben körülnézhetünk, mozoghatunk és tárgyakat foghatunk meg.

Kontroll-lámpa → nyomtatott szöveg → képernyő → színes kép → hang → virtuális tér

A számítógép belsejében mindeközben továbbra is csak bitek mozognak.

A kimeneti eszközök feladata, hogy ezeket a biteket az ember érzékei számára értelmezhető formává alakítsák:

fénnyé, tintává, vonallá, képpé, hanggá vagy térélménnyé.



III. rész – A láthatatlan gép: a szoftver

7. A programozás története

A számítógép önmagában nem tudja, mit szeretnénk tőle. A processzor áramkörei ugyan képesek műveletek milliárdjait elvégezni másodpercenként, de ehhez pontosan meg kell mondanunk, milyen műveleteket, milyen sorrendben hajtsanak végre.

Ezt a feladatot végzi a **program.**

A programozás története tulajdonképpen annak története, hogyan alakítottunk ki egyre kényelmesebb közvetítő nyelveket az emberi gondolkodás és a számítógép rendkívül egyszerű utasításokat végrehajtó processzora között.

A fejlődés különös irányt követett. Kezdetben az embernek kellett alkalmazkodnia a géphez. Ma egyre inkább a gép alkalmazkodik az emberhez.

7.1. Ada Lovelace

A programozás története különös módon korábban kezdődött, mint az elektronikus számítógépek története.

Charles Babbage a 19. században tervezte meg Analytical Engine nevű általános célú mechanikus számítógépét. A gép teljes egészében sohasem készült el, működési elve azonban már meglepően hasonlított a későbbi számítógépekére.

Babbage munkatársa, **Ada Lovelace** felismerte, hogy egy ilyen gép nem csupán számításokra használható. 1843-ban Luigi Menabrea Babbage gépéről szóló tanulmányának fordításához olyan részletes jegyzeteket fűzött, amelyek terjedelme jóval meghaladta az eredeti írását.

1. Ada Lovelace



Ábra:
Ada Lovelace (1815–1852) és az Analytical Engine számára készített algoritmusának részlete. A Bernoulli-számok kiszámítására leírt eljárását gyakran a világ első számítógépes programjának tekintik.

Ezek között szerepelt egy eljárás a Bernoulli-számok kiszámítására. Ezt gyakran a történelem első számítógépes programjának nevezik.

Még fontosabb azonban Lovelace felismerése: megértette, hogy ha valamit megfelelő szabályok szerint szimbólumokkal lehet ábrázolni, akkor egy általános célú gép elvileg nemcsak számokkal dolgozhat. Akár zenét is létrehozhatna.

Ez a gondolat csak több mint száz évvel később vált mindennapi valósággá.

7.2. Gépi kód

Az első elektronikus számítógépek programozása még nagyon közel állt magához a hardverhez.

A processzor csak meghatározott **gépi utasításokat** képes közvetlenül végrehajtani. Ezeket bináris számok – nullák és egyesek – formájában tároljuk.

Egy utasítás például azt jelentheti:

- tölts egy számot egy regiszterbe,
- adj hozzá egy másik számot,
- írd az eredményt a memóriába,
- ugorj a program egy másik pontjára.

A gépi kód az ember számára nehezen olvasható. Egy program hosszú számsorokból áll, amelyekben már egyetlen hibás bit is egészen más utasítást eredményezhet.

Ráadásul minden processzortípusnak saját gépi nyelve van.

A gépi kód tehát tökéletesen megfelel a processzornak – de meglehetősen rosszul az embernek.



Ábra:
A korai programozás világa. A programozónak kezdetben közvetlenül a gép utasításkészletéhez kellett alkalmazkodnia. Az assembly már szimbolikus nevekkel helyettesítette a nehezen megjegyezhető számkódokat.

7.3. Assembly

A következő nagy lépés az volt, amikor a számok helyett rövid, megjegyezhető neveket kezdtek használni az utasításokra.

Így született meg az **assembly nyelv**.

Ahelyett például, hogy egy művelet bináris kódját kellett volna megjegyezni, a programozó olyan rövidítéseket írhatott, mint:

MOV – adat mozgatása
ADD – összeadás
SUB – kivonás
JMP – ugrás

Egy külön program, az **assembler** fordítja ezeket a szimbolikus utasításokat gépi kódra.

Az assembly óriási könnyebbséget jelentett, de továbbra is szorosan kötődött az adott processzorhoz. A programozónak ismernie kellett a regisztereket, memóriacímeket és a processzor működését.

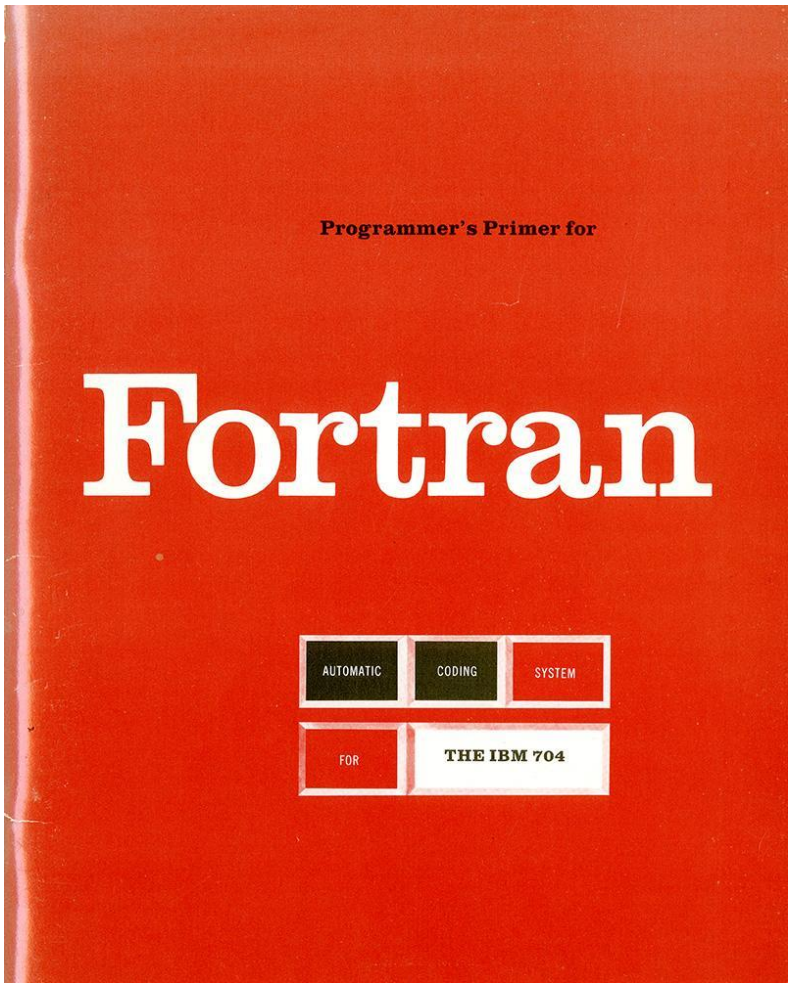
Még mindig az ember alkalmazkodott a géphez.

7.4. FORTRAN

Az 1950-es években merült fel komolyan a gondolat: miért ne írhatnánk a programokat az emberi gondolkodáshoz közelebb álló formában, és bízhatnánk a számítógépre azok gépi kódra fordítását?

Az IBM-nél **John Backus** vezetésével fejlesztették ki a FORTRAN nyelvet. Neve a *FORmula TRANslation*, vagyis „képletfordítás” kifejezésből származik.

Az első FORTRAN-fordító 1957-ben jelent meg.



A mérnökök és tudósok immár matematikai képletekhez hasonló utasításokat írhattak. A **compiler**, vagyis fordítóprogram alakította ezeket gépi kóddá.

Ez alapvető változás volt.

C = FOR COMMENT	CONTINUATION	FORTRAN STATEMENT	IDENTIFICATION
STATEMENT NUMBER 1 5	6	7 72	73 80
C		PROGRAM FOR FINDING THE LARGEST VALUE	
C	X	ATTAINED BY A SET OF NUMBERS	
		DIMENSION A(999)	
		FREQUENCY 30(2,1,10), 5(100)	
		READ 1, N,.(A(I), I = 1,N)	
1		FORMAT (I3/(12F6.2))	
		BIGA = A(1)	
5		DO 20 I = 2,N	
30		IF (BIGA-A(I)) 10,20,20	
10		BIGA = A(I)	
20		CONTINUE	
		PRINT 2, N, BIGA	
2		FORMAT (22H1THE LARGEST OF THESE I3, 12H NUMBERS IS F7.2)	
		STOP 77777	

Ábra:
John Backus és a FORTRAN. Az IBM munkatársa által vezetett csoport fejlesztette ki az első igazán sikeres magas szintű programozási nyelvek egyikét. A programozó immár képletekhez hasonló utasításokat írhatott, amelyeket a fordítóprogram alakított gépi kóddá.

A programozónak egyre kevésbé kellett azzal foglalkoznia, hogyan működik a processzor. Egyre inkább arra koncentrálhatott, **milyen problémát szeretne megoldani.**

A FORTRAN különösen tudományos és műszaki számításokban vált sikeressé, és továbbfejlesztett változatait ma is használják.

7.5. COBOL

A számítógépek hamarosan nemcsak tudományos számításokat végeztek, hanem bankokban, biztosítóknál, állami intézményekben és nagyvállalatoknál is megjelentek.

Ehhez másfajta programozási nyelvre volt szükség.

1959-ben kezdődött a **COBOL – Common Business-Oriented Language** kialakítása. Fejlesztésében fontos szerepet játszott Grace Hopper és az általa korábban kidolgozott programozási nyelvek tapasztalata.



Ábra:
Grace Hopper (1906–1992), a programozás történetének egyik legfontosabb alakja. Úttörő szerepet játszott a fordítóprogramok fejlesztésében, és munkássága jelentősen hozzájárult a COBOL létrejöttéhez.

A COBOL utasításait szándékosan az angol nyelvhez hasonlóan alakították ki. A cél az volt, hogy az üzleti programok könnyebben olvashatók és karbantarthatók legyenek.

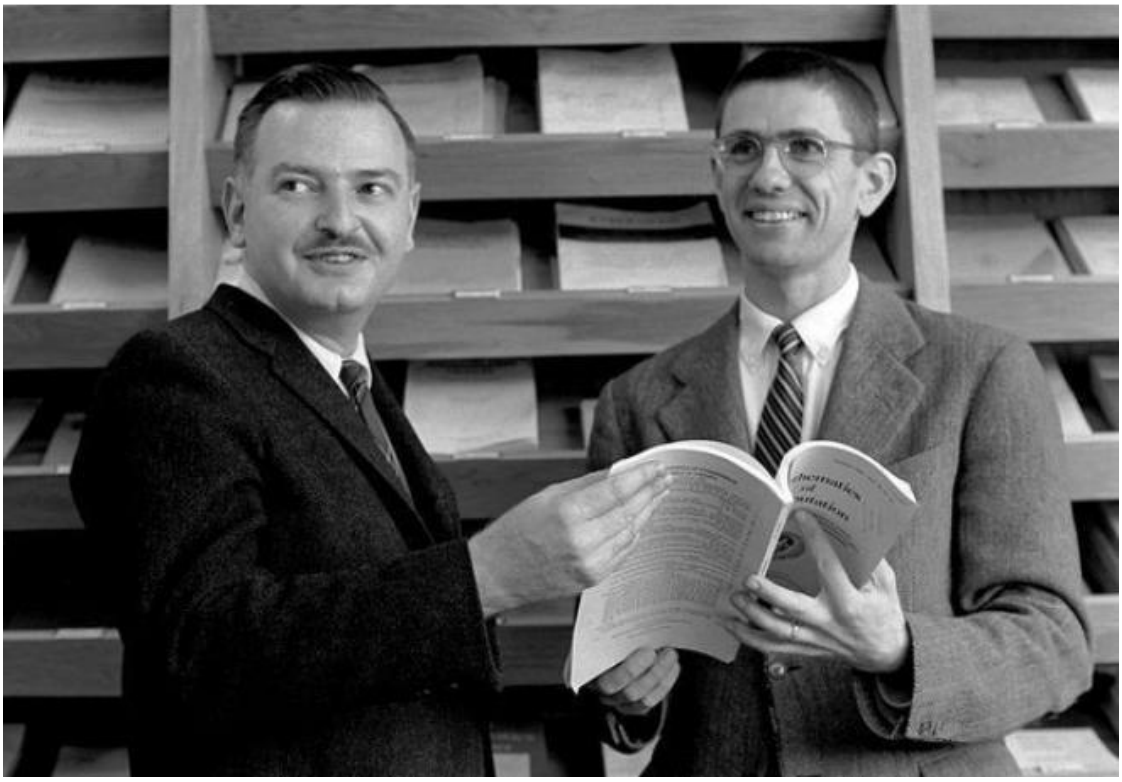
A COBOL rendkívül tartósnak bizonyult. Évtizedekkel később is hatalmas mennyiségű banki, biztosítási és államigazgatási rendszer működik COBOL-programokkal.

A programozási nyelvek történetének egyik tanulsága éppen ez: ami egyszer egy nagy rendszer alapjává válik, azt sokszor rendkívül nehéz lecserélni.

7.6. BASIC

Az 1960-as évek elején a számítógép még drága és ritka eszköz volt. Programozásával elsősorban matematikusok, mérnökök és szakemberek foglalkoztak.

John G. Kemeny – Kemény János – és Thomas E. Kurtz azonban úgy gondolta, hogy a programozásnak minden egyetemi hallgató számára hozzáférhetőnek kell lennie.



Ábra:
Kemény János és Thomas Kurtz, a BASIC megalkotói. A Dartmouth College-ban nem csupán új programozási nyelvet készítettek: céljuk az volt, hogy a számítógép használata ne csak szakemberek kiváltsága legyen.

A Dartmouth College-ban ezért egy egyszerűen megtanulható programozási nyelvet dolgoztak ki.

1964-ben megszületett a **BASIC – Beginner's All-purpose Symbolic Instruction Code**.

A BASIC később különösen a személyi számítógépek világában vált fontossá. Az 1970-es és 1980-as évek számos otthoni számítógépének bekapcsolásakor rögtön egy BASIC értelmezőprogram várta a felhasználót.

Sok millió ember számára a BASIC jelentette az első találkozást a programozással.

7.7. Pascal

Az egyre nagyobb programok új problémát okoztak: hogyan lehet őket áttekinthetően és megbízhatóan felépíteni?

Niklaus Wirth svájci informatikus 1970 körül alkotta meg a Pascal nyelvet, amelyet Blaise Pascalról nevezett el.

A Pascal egyik fő célja a **strukturált programozás** tanítása volt.

A programokat jól elkülöníthető részekre lehetett bontani, és a vezérlési szerkezetek világosabbá tették a program működését.

A Pascal ezért hosszú időn keresztül az informatikaoktatás egyik legfontosabb nyelve lett.

Sok programozó ezen keresztül tanulta meg, hogy egy programnak nemcsak működnie kell – **érthetőnek is kell lennie**.

7.8. C és C++

A C programozási nyelvet **Dennis Ritchie** fejlesztette ki a Bell Labs laboratóriumaiban az 1970-es évek elején.

Sikerének egyik kulcsa különleges egyensúlya volt.

Elég magas szintű ahhoz, hogy összetett programokat lehessen vele készíteni, ugyanakkor elég közel marad a hardverhez ahhoz, hogy rendkívül hatékony programok születhessenek benne.

A Unix operációs rendszer jelentős részét C-ben írták.

A C később szinte egész programnyelv-család ősévé vált.

Bjarne Stroustrup az 1980-as években a C nyelvet új lehetőségekkel egészítette ki. Ebből alakult ki a **C++.**

A C++ egyik legfontosabb újdonsága az objektumorientált programozás széles körű elterjesztése volt.

7.9. Objektumorientált programozás

A programok növekedésével egyre nehezebbé vált több millió sornyi programkód kezelése.

Az objektumorientált programozás egyik alapgondolata, hogy a programot egymással kapcsolatban álló **objektumokból** építjük fel.

Egy objektum egyszerre tartalmazhat adatokat és az ezeken végrehajtható műveleteket.

Egy grafikus programban például egy „ablak” objektumnak lehet:

- mérete,
- helyzete,
- címe,

és lehetnek műveletei:

- megnyitás,
- bezárás,
- átméretezés,
- mozgatás.

Az objektumorientált gondolkodás korai, meghatározó nyelve a norvég **Simula** volt az 1960-as években, majd a Xerox PARC-ban fejlesztett **Smalltalk** vitte tovább az elképzelést.

Később a C++, a Java és számos más nyelv tette általánossá.

7.10. Java

Az 1990-es években az internet terjedése új problémát állított a programozók elé: hogyan lehet ugyanazt a programot különböző számítógépeken futtatni?

A Sun Microsystems munkatársa, **James Gosling** vezetésével létrehozott Java egyik híres jelszava lett:

„Write once, run anywhere.” – Írd meg egyszer, futtasd bárhol.

A Java programokat nem közvetlenül egy adott processzor gépi kódjára fordítják. Egy köztes formátum, a bytecode jön létre, amelyet a **Java Virtual Machine**, a JVM hajt végre.

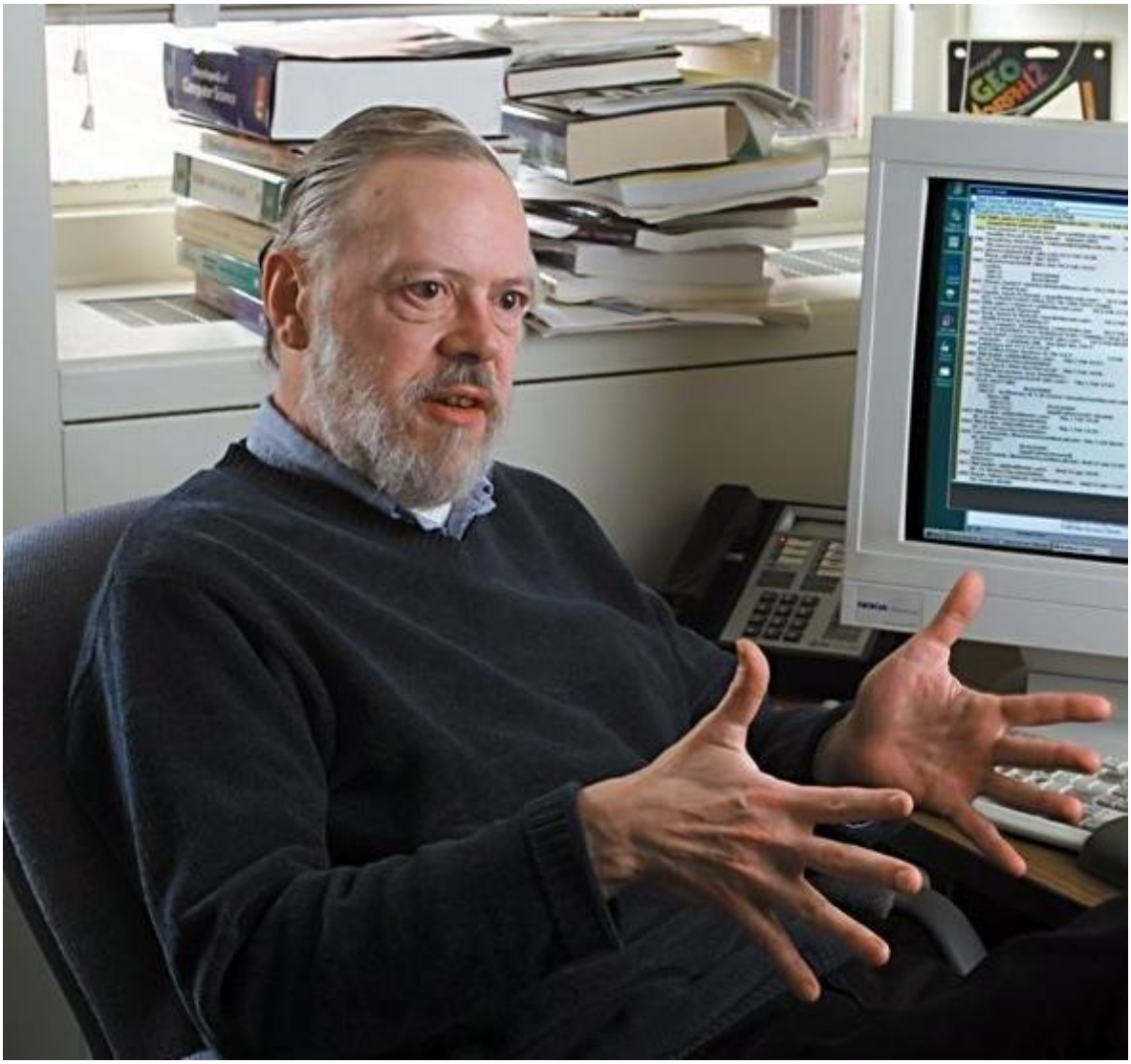
Ha egy gépen rendelkezésre áll megfelelő JVM, ugyanaz a Java-program különböző hardvereken és operációs rendszereken is működhet.

Ez a hordozhatóság a Java egyik legnagyobb sikerének alapja lett.

7.11. Python

Guido van Rossum az 1980-as évek végén kezdett dolgozni egy új programozási nyelven. A Python első nyilvános változata 1991-ben jelent meg.

A név nem a kígyóból származik: Van Rossum a brit **Monty Python** komédiacsoport rajongója volt.



Ábra:
Guido van Rossum, a Python megalkotója. Az 1991-ben nyilvánosságra hozott nyelv egyik legfontosabb célja a programkód egyszerűsége és olvashatósága volt.

A Python egyik alapelve az olvashatóság.

Ugyanazt a feladatot gyakran lényegesen kevesebb programkóddal lehet megoldani, mint alacsonyabb szintű nyelveken.

A Python idővel az egyik legnépszerűbb programozási nyelvvé vált. Használják:

- oktatásban,
- tudományos számításokban,
- adatfeldolgozásban,
- webes alkalmazásokban,
- automatizálásban,
- mesterséges intelligenciában.

Sikere jól mutatja a programozás történetének általános irányát: a programozó egyre kevésbé foglalkozik a számítógép belső működésével, és egyre inkább magával a megoldandó problémával.

7.12. Vizuális programozás

A programozás következő lépése az volt, hogy bizonyos esetekben még a programszöveget is megpróbálták kiváltani.

A **vizuális programozásban** a program elemei grafikus objektumokként jelenhetnek meg. A programozó blokkokat, folyamatokat vagy kapcsolatokat állít össze.

Az egyik legismertebb példa az MIT Media Lab által fejlesztett **Scratch**, amelyben színes blokkok összeillesztésével lehet programokat készíteni.

A vizuális programozás különösen az oktatásban, az ipari vezérlésben és különféle speciális fejlesztőrendszerekben terjedt el.

A számítógéppel folytatott kommunikáció ezzel ismét közelebb került az ember természetes gondolkodásához.

Kemény János és a BASIC – magyar a programozás demokratizálásában

A BASIC történetének egyik megalkotója Magyarországon született.

Kemény János – John George Kemeny – 1926-ban Budapesten született. Családjával 1940-ben emigrált az Egyesült Államokba.

Fiatal matematikusként részt vett a Manhattan-tervben, később pedig Albert Einstein matematikai asszisztense is volt Princetonban.

Pályájának egyik legfontosabb állomása azonban a Dartmouth College lett.

Kemény és kollégája, **Thomas Kurtz** úgy gondolta, hogy a számítógép nem maradhat néhány kiválasztott szakember eszköze. A hallgatóknak – nemcsak a matematikusoknak és mérnököknek – közvetlenül hozzá kell férniük a számítógéphez.

Ehhez két dolgot hoztak létre.

Az egyik a **Dartmouth Time-Sharing System**, amely lehetővé tette, hogy egyszerre több felhasználó dolgozzon terminálokon keresztül ugyanazzal a számítógéppel.

A másik a könnyen megtanulható **BASIC programozási nyelv** volt.

1964. május 1-jén hajnalban futott le az első BASIC-program a Dartmouth rendszerén.

A nyelvet tudatosan kezdőknek tervezték. Egyszerű utasításokkal rövid idő alatt működő programokat lehetett készíteni:

PRINT
INPUT
IF...THEN
FOR...NEXT
GOTO

Kemény elképzelése lényegében az volt, hogy a számítógép használatának ugyanolyan természetes eszközzé kell válnia az egyetemen, mint a könyvtárnak.

A BASIC később messze túlnőtt Dartmouth falain. Amikor az 1970-es évek végén és az 1980-as években megjelentek az otthoni számítógépek, számos gép BASIC-kel együtt került forgalomba. A Commodore, Sinclair, Apple, TRS-80 és sok más számítógép felhasználói közül milliók ezen a nyelven írták életük első programját.

Kemény János munkája ezért nem egyszerűen egy új programozási nyelv megszületéséhez kapcsolódik.

Segített abban, hogy a programozás a szakemberek kiváltságából mindenki számára megtanulható tevékenységgé váljon.

A programozás útja

A programozás történetét egyetlen sorban is össze lehet foglalni:

Gépi kód → Assembly → magas szintű nyelvek → strukturált programozás → objektumok → hordozható nyelvek → vizuális programozás

A fejlődés minden lépcsője ugyanabba az irányba vezetett:

egyre nagyobb távolságra a processzor belső működésétől, és egyre közelebb az emberi gondolkodáshoz.

És a történet ezzel még nem ért véget.

A következő lépcsőn már nem feltétlenül programozási nyelven mondjuk meg a számítógépnek, mit tegyen. Elég lehet hétköznapi nyelven leírnunk a problémát, és egy mesterséges intelligencia elkészítheti hozzá a programot.

Ezzel különös módon visszaérkezünk Ada Lovelace közel kétszáz éves gondolatához: az általános célú számítógép igazi ereje nem egyetlen számításban rejlik, hanem abban, hogy **szimbólumokkal leírható műveletek sorozatát képes végrehajtani.**



Ha nem romlott el, ne javítsd meg!

„If it ain't broke, don't fix it."
Ha nem romlott el, ne javítsd meg.

A régi amerikai mondás akár a mérnöki gondolkodás egyik alaptörvénye is lehetne. A számítástechnika azonban különös világot teremtett: egy jól működő programot is rendszeresen módosítani kell. Új hardverek jelennek meg, változnak az operációs rendszerek, új biztonsági résekre derül fény, és a felhasználók is új szolgáltatásokat várnak.

Ezért a frissítés szükséges. Csakhogy minden módosítás új hibák lehetőségét is magában hordozza.

Innen ered a programozók keserű tréfája:

„A frissítés az ismert hibákat ismeretlen hibákra cseréli."

A modern szoftverek ráadásul olyan összetettek, hogy szinte lehetetlen minden lehetséges hardver-, operációsrendszer- és felhasználói környezetben kipróbálni őket. Egy javítás, amely egymillió számítógépen tökéletesen működik, a következő ezren váratlan problémát okozhat. A fejlesztő kijavít egy hibát – és közben létrehozhat egy másikat.

Így alakult ki a modern számítástechnika egyik furcsa körforgása:

Működik → frissítjük → valami elromlik → kijavítjuk → újra működik → ismét frissítjük…

Mindez persze nem érv a frissítések ellen. Különösen a biztonsági javítások elhagyása sokkal nagyobb kockázat lehet, mint maga a frissítés. Inkább arra emlékeztet, hogy a szoftver – ellentétben egy elkészült géppel – **szinte soha nincs teljesen kész.**

Talán ezért lehetne a digitális kor mérnöki bölcsességét így módosítani:

„Ha nem romlott el, ne javítsd meg – kivéve, ha szoftver."

8. Operációs rendszerek

Amikor bekapcsolunk egy mai számítógépet, néhány másodperc múlva megjelenik az asztal, működik az egér, csatlakozik a hálózat, elérhetjük a fájljainkat, és elindíthatjuk programjainkat.

Mindezt természetesnek vesszük.

Pedig a háttérben egy rendkívül összetett programrendszer dolgozik: az **operációs rendszer.**

Az operációs rendszer közvetítő a számítógép hardvere, az alkalmazások és a felhasználó között. Gondoskodik arról, hogy a programok használhassák a processzort, a memóriát, a háttértárakat, a képernyőt, a billentyűzetet, az egeret, a nyomtatót és a hálózatot.

Nélküle a számítógép ugyan működőképes elektronikus szerkezet lenne, de minden programnak magának kellene tudnia, hogyan kezelje a gép valamennyi alkatrészét.

Az operációs rendszer ezt a terhet veszi le a programozó és a felhasználó válláról.

8.1. Miért kell operációs rendszer?

A legkorábbi számítógépeknek még nem volt a mai értelemben vett operációs rendszerük.

A programozó gyakorlatilag közvetlenül használta a gépet. A programot kapcsolókkal, lyukkártyával vagy lyukszalaggal vitték be, majd a számítógépet előkészítették a feladat végrehajtására.

Ahogy azonban a gépek egyre bonyolultabbá váltak, ez mind kevésbé volt célszerű.

Valakinek – pontosabban valaminek – el kellett döntenie:

- melyik program mikor használhatja a processzort,
- hol helyezkedjen el a memóriában,
- hogyan olvasson és írjon fájlokat,
- hogyan érje el a háttértárat,
- hogyan kommunikáljon a perifériákkal,
- mi történjen, ha több program egyszerre szeretné ugyanazt az erőforrást használni.

Ezeket a feladatokat fokozatosan az operációs rendszer vette át.

Kialakult egy új réteg:

felhasználó → alkalmazás → operációs rendszer → hardver

Az operációs rendszer így a számítógép láthatatlan „forgalomirányítójává” vált.

8.2. Batch rendszerek

Az 1950-es évek nagyszámítógépei rendkívül drágák voltak. Megengedhetetlen pazarlásnak számított, ha a gép azért állt, mert a következő programot készítették elő.

Ezért a feladatokat **kötegekbe – batch-ekbe – rendezték.**

A programozó elkészítette programját és adatait, gyakran lyukkártyákon, majd átadta azokat az operátornak. Az operátor több felhasználó munkáját összegyűjtötte, és egymás után futtatta őket.

A felhasználó nem ült a számítógép előtt.

Leadta a feladatát – és várta az eredményt.

Ha a programban egyetlen apró hiba volt, előfordulhatott, hogy órák múlva kapta vissza a nyomtatott hibaüzenetet. Kijavította a programot, majd újra beadta.

A batch operációs rendszerek legfontosabb feladata az volt, hogy automatikusan egymás után indítsák a feladatokat, és minél jobban kihasználják a drága számítógépet.

A hangsúly tehát még nem a felhasználó kényelmén, hanem **a gépidő maximális kihasználásán** volt.

8.3. Unix

Az 1960-as évek végén a Bell Laboratories munkatársai, elsősorban **Ken Thompson és Dennis Ritchie**, egy új operációs rendszert kezdtek fejleszteni.

Ez lett a **Unix**.

Első változata 1969 körül született meg.

A Unix egyik nagy újítása nem egyetlen technikai megoldás volt, hanem egy egész szemlélet.

A rendszer sok kis, jól meghatározott feladatot végző programból épült fel. Ezeket egymással összekapcsolva bonyolult műveleteket lehetett létrehozni.

A Unix többfelhasználós és többfeladatos rendszer volt. Egy számítógépet egyszerre több ember használhatott terminálokon keresztül, miközben több program futhatott párhuzamosan.

Különösen fontossá vált, hogy a Unix jelentős részét C nyelven írták. Emiatt sokkal könnyebben lehetett más számítógéptípusokra átültetni, mint a korábbi, nagyrészt assemblyben készült operációs rendszereket.



Ábra:
Ken Thompson és Dennis Ritchie a Bell Labsban. A Unix operációs rendszer és a C programozási nyelv alapvetően meghatározta a modern számítástechnika fejlődését. A C hatása számos későbbi programozási nyelvben ma is felismerhető.

A Unix hatása óriási.

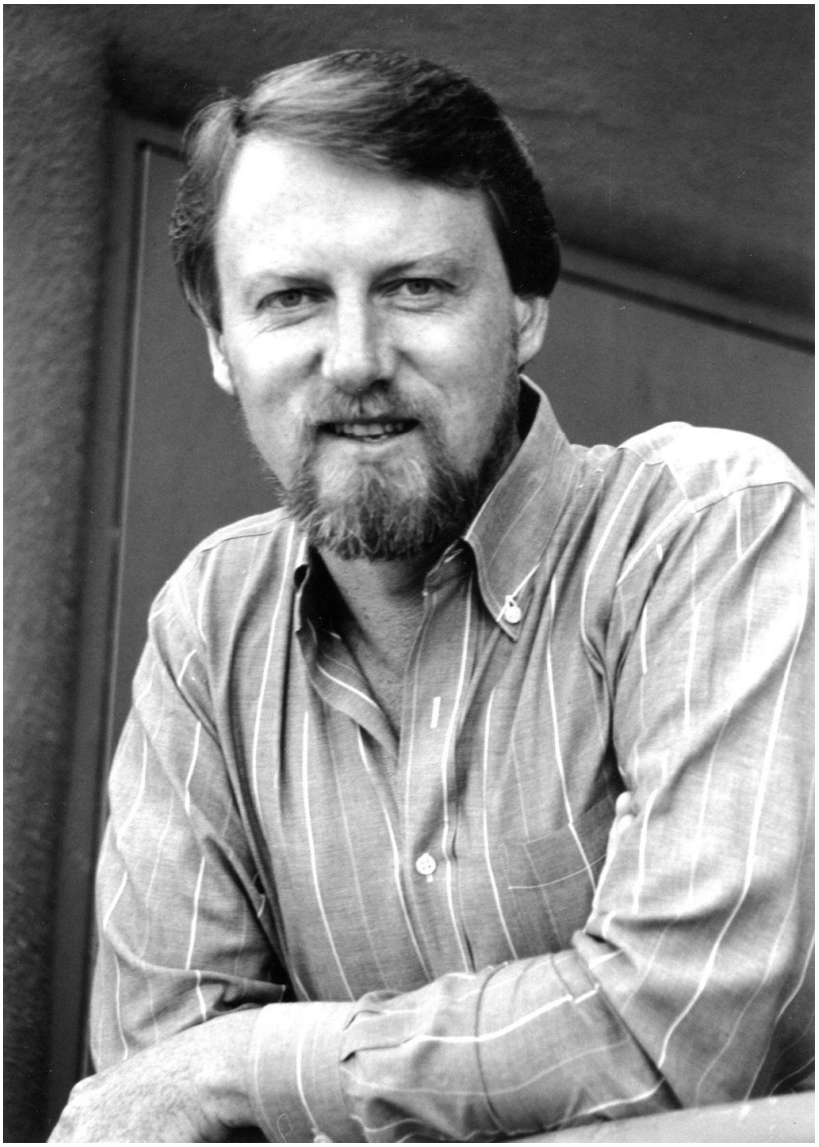
Leszármazottai és alapelvei ma is megtalálhatók szervereken, szuperszámítógépeken, munkaállomásokon és személyi eszközökön.

Még az Android telefonok mélyén is Unixból származó gondolatok élnek tovább.

8.4. CP/M

Amikor az 1970-es években megjelentek a mikroprocesszorok és az első mikroszámítógépek, ezeknek is szükségük lett operációs rendszerre.

Az egyik első igazán sikeres rendszer a **CP/M – Control Program for Microcomputers** volt, amelyet Gary Kildall fejlesztett ki.



Ábra:
CP/M egy korabeli mikroszámítógépen. Gary Kildall operációs rendszere az 1970-es évek végének és az 1980-as évek elejének egyik legfontosabb mikroszámítógépes szabványává vált.

A CP/M elsősorban az Intel 8080 és Zilog Z80 processzorokra épülő gépeken terjedt el.

Jelentősége abban állt, hogy valamiféle közös szoftveres környezetet teremtett a sokféle, egymástól eltérő mikroszámítógép között.

Ha egy gép képes volt CP/M-et futtatni, akkor a CP/M-re készült programok jelentős része használhatóvá vált rajta.

Szövegszerkesztők, adatbázis-kezelők, programozási nyelvek és üzleti programok egész világa alakult ki körülötte.

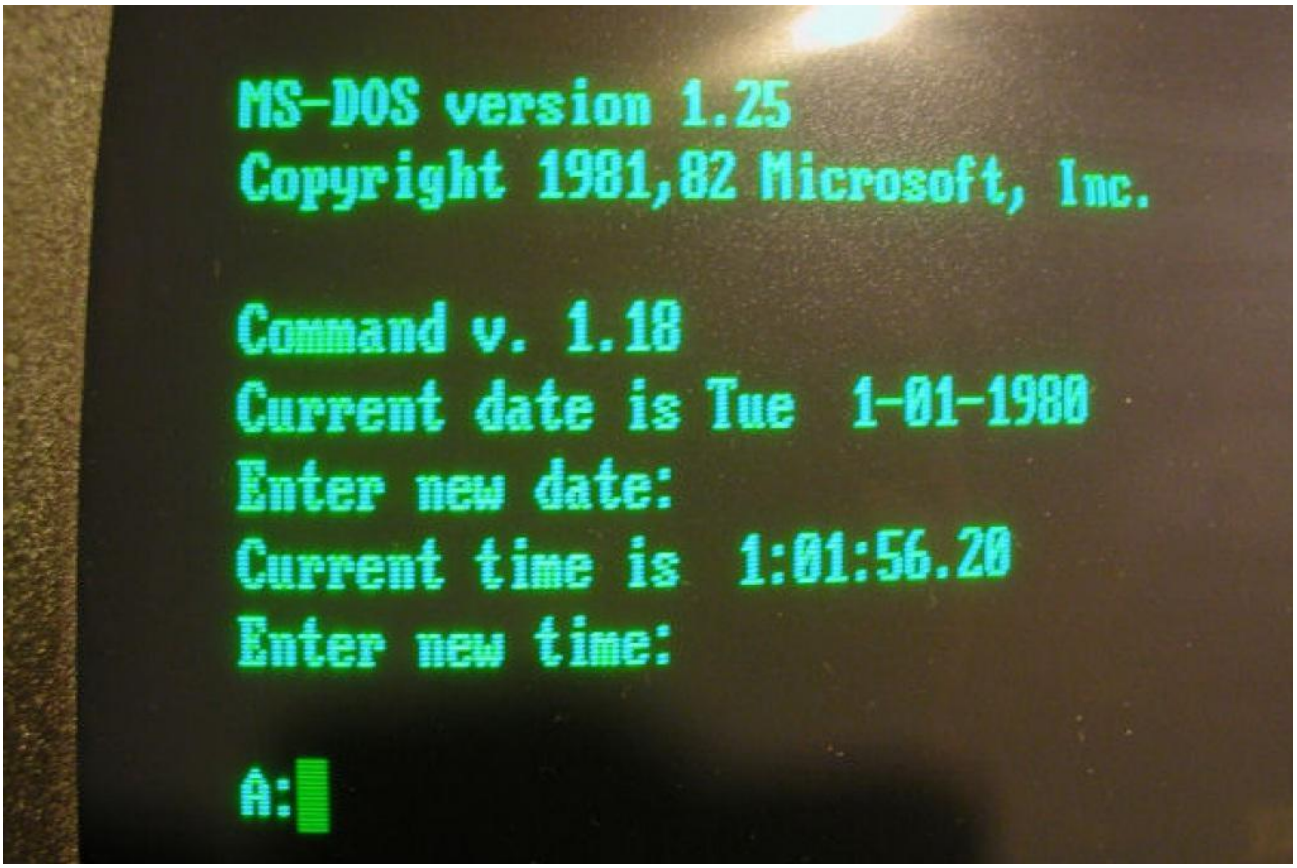
Egy ideig úgy tűnt, hogy a CP/M lehet a személyi számítógépek általános operációs rendszere.

Aztán megjelent az IBM PC.

8.5. DOS

Az IBM 1981-ben mutatta be személyi számítógépét, az IBM PC-t.

Operációs rendszerként a Microsoft által szállított **PC DOS** került rá. A Microsoft más gyártóknak saját változatát, az **MS-DOS-t** adta.



Ábra:
Az MS-DOS parancssoros felülete. Az IBM PC és kompatibiliseinek elterjedésével a DOS az 1980-as évek személyi számítógépeinek meghatározó operációs rendszerévé vált.

A DOS parancssoros operációs rendszer volt.

A felhasználót például ilyen prompt fogadta:

C:\>

A programokat és műveleteket begépelt parancsokkal lehetett elindítani:

DIR
COPY
DEL
CD

A DOS mai szemmel egyszerű rendszernek tűnik. Korai változatai lényegében egyetlen program futtatására készültek, és a memória kezelése is komoly korlátokat jelentett.

Előnye viszont éppen egyszerűsége és kis erőforrásigénye volt.

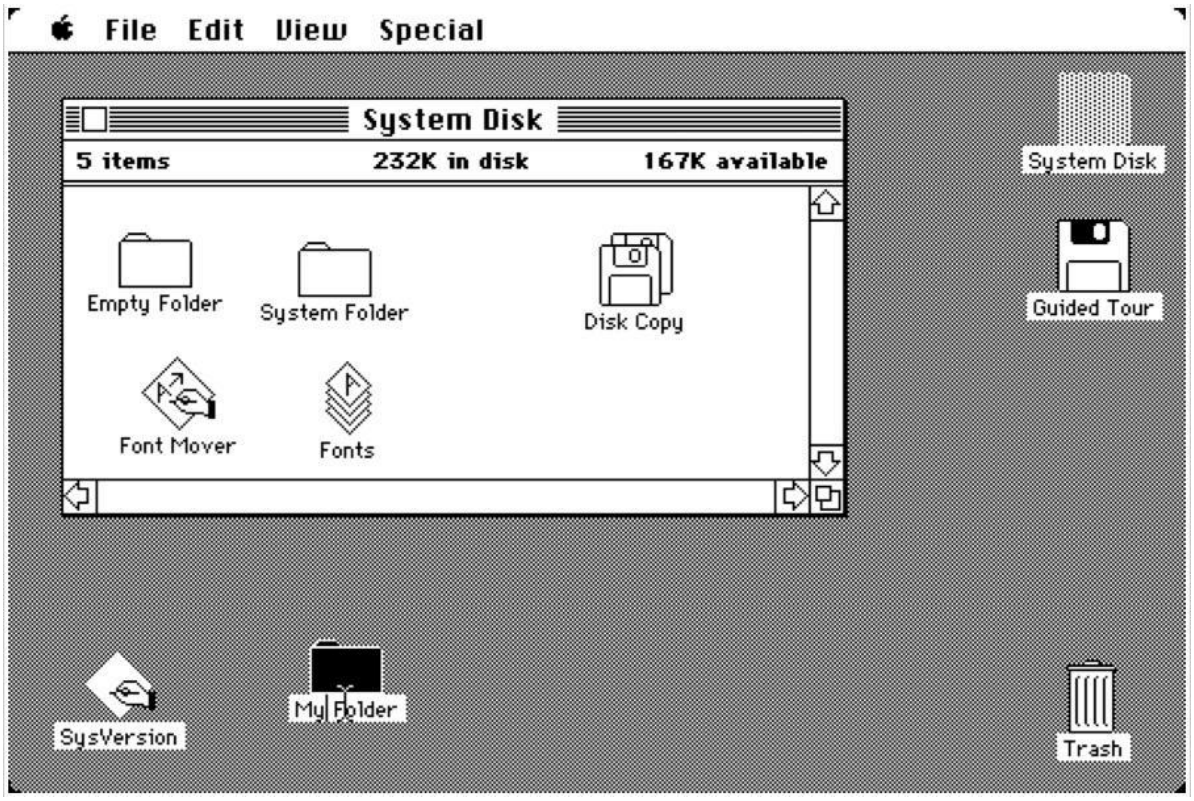
Az IBM PC és főként a kompatibilis gépek robbanásszerű terjedésével az MS-DOS az 1980-as évek személyi számítógépeinek meghatározó operációs rendszerévé vált.

8.6. Macintosh

1984-ben az Apple bemutatta a Macintosh számítógépet.

A Macintosh operációs rendszerének legfeltűnőbb újdonsága a **grafikus felhasználói felület** volt.

A felhasználónak többé nem kellett minden parancsot begépelnie.



Ábra:
Az 1984-es Macintosh grafikus felülete. Ablakok, ikonok, menük és egér: a számítógép kezeléséhez többé nem kellett minden parancsot begépelni.

Megjelentek az ablakok, ikonok, menük és az egérrel kezelhető grafikus elemek.

A dokumentumokat, fájlokat jelképező ikonok képviselték. A törléshez a fájlt akár egy képernyőn látható szemetesbe lehetett húzni.

Ezek az ötletek nem mind az Apple-nél születtek. A grafikus felület fontos előzményeit a Xerox PARC kutatóközpontban dolgozták ki. Az Apple azonban felismerte, hogy ezekből nem csupán érdekes kísérlet, hanem **emberek milliói számára használható termék** készíthető.

A számítógép használata ezzel ismét közelebb került az ember hétköznapi gondolkodásához.

8.7. Windows

A Microsoft 1985-ben jelentette meg a Windows első változatát.

A korai Windows még a DOS-ra épülő grafikus környezet volt, és kezdetben korántsem aratott akkora sikert, mint későbbi változatai.

A fordulat fokozatosan következett be.

A Windows 3.x már széles körben elterjedt, majd az 1995-ben megjelent **Windows 95** a személyi számítógépek történetének egyik meghatározó operációs rendszerévé vált.



Ábra:
Windows 95. A Start menüvel és tálcával kialakított kezelési mód hosszú időre meghatározta, hogyan használjuk a személyi számítógépeket.

Megjelent a Start menü, a tálca és számos olyan kezelési megoldás, amely hosszú időre meghatározta a PC használatát.

Közben a Microsoft egy technikailag új rendszer fejlesztésébe is belekezdett. Ez volt a **Windows NT**, amelynek technológiai vonala később a fogyasztói Windows rendszerek alapjává is vált.

A Windows sikerének egyik kulcsa nem pusztán maga az operációs rendszer volt.

A PC-kompatibilis gépek óriási választéka, a rengeteg alkalmazás és játék, valamint a hardvergyártók támogatása együtt hatalmas ökoszisztémát hozott létre.

A Windows így évtizedekre a személyi számítógépek legelterjedtebb operációs rendszerévé vált.

8.8. Linux

1991-ben egy finn egyetemista, **Linus Torvalds** egy saját operációsrendszer-kernel fejlesztésébe kezdett.



Ábra:
Linus Torvalds, aki 1991-ben kezdte fejleszteni a Linux kernelt. Az interneten együtt dolgozó fejlesztők közössége később a számítástechnika egyik legsikeresebb nyílt forráskódú rendszerét hozta létre.

A projektből hamarosan valami sokkal nagyobb lett.

Torvalds az interneten keresztül más programozókat is bevont a fejlesztésbe. A Linux kernel a GNU projekt eszközeivel és számos más nyílt forráskódú programmal együtt teljes operációs rendszerek alapjává vált.

Megszülettek a különféle **Linux-disztribúciók.**

A Linux különlegessége nemcsak technikai tulajdonságaiban rejlik, hanem fejlesztési modelljében is.

A forráskód hozzáférhető. Fejlesztők ezrei dolgozhatnak rajta, javíthatják és továbbfejleszthetik.

Asztali számítógépeken a Linux sohasem vált olyan dominánssá, mint a Windows.

Máshol azonban elképesztő sikert ért el.

Linux működteti a világ szervereinek jelentős részét, szuperszámítógépeket, hálózati eszközöket, televíziókat, beágyazott rendszereket – és közvetve Android telefonok milliárdjait.

A Linux tehát sokszor ott van körülöttünk akkor is, amikor nem látjuk.

8.9. Mobil operációs rendszerek

A mobiltelefonok fejlődésével új operációsrendszer-kategória alakult ki.

A korai okostelefonokon több rendszer versenyzett egymással. Fontos szerepet játszott például a **Symbian**, a Palm OS, a Windows Mobile és a BlackBerry rendszere.

A nagy fordulat 2007–2008-ban következett be.

Az Apple iPhone operációs rendszere – a később **iOS** névre keresztelt rendszer – az érintőképernyőt helyezte a kezelés középpontjába.

Nem az egeret próbálta lekicsinyíteni.

Az ujj lett a mutatóeszköz.



Ábra:
A mobil operációs rendszerek fejlődése. A Symbian és más korai rendszerek után az iPhone, majd az Android tette az érintőképernyős okostelefont általános célú zsebszámítógéppé.

2008-ban megjelentek az első **Android** rendszerű telefonok is. Az Android Linux kernelre épült, és számos gyártó készülékein megjelent.

A mobil operációs rendszernek új feladatokat kellett megoldania.

Kezelnie kellett:

- az érintőképernyőt,
- a mobilhálózatot,
- a Wi-Fi és Bluetooth kapcsolatokat,
- a kamerákat,
- a GPS-t,
- a különféle érzékelőket,
- az akkumulátor energiafogyasztását,
- valamint az alkalmazások jogosultságait és biztonságát.

A mobiltelefon ezzel valódi általános célú számítógéppé vált.

Csak éppen már nem számítógépnek nevezzük.

Mi történik a bekapcsoló gomb megnyomása után?

Megnyomjuk a számítógép bekapcsológombját.

Néhány másodperc múlva megjelenik a Windows, Linux vagy más operációs rendszer képernyője.

De mi történt közben?

Meglepően sok minden.

1. Megérkezik az áram

A tápegység biztosítja a számítógép alkatrészeinek szükséges feszültségeket. Amikor ezek stabilak, a processzor megkezdheti a működését.

Csakhogy a RAM ekkor még üres.

Az operációs rendszer sincs benne.

2. A processzor egy előre meghatározott helyről indul

A processzort úgy tervezték, hogy bekapcsolás vagy reset után egy meghatározott címen keresse az első végrehajtandó utasítást.

Itt olyan programkódhoz jut, amely nem az üres RAM-ban, hanem nem felejtő memóriában található.

Ez a gép firmware-ének része.

A PC-kben ezt hagyományosan **BIOS-nak**, az újabb gépeken **UEFI-nek** nevezzük.

3. A firmware ellenőrzi és előkészíti a gépet

A BIOS vagy UEFI megkezdi a hardver inicializálását.

Ellenőrzi és előkészíti többek között:

a processzort,

a memóriát,

a billentyűzetet,

a háttértárakat,

a grafikus rendszert,

valamint más alapvető hardverelemeket.

A régi PC-k bekapcsolásakor ezt a folyamatot jól lehetett követni a képernyőn.

Ezután meg kell találni azt az eszközt, amelyről az operációs rendszer elindítható.

4. Megkeresi a bootolható eszközt

Ez lehet:

SSD → merevlemez → USB-meghajtó → hálózat

a gép beállításaitól függően.

A firmware azonban még nem tölti be önmagában az egész operációs rendszert.

Először egy kisebb programot indít el.

Ez a **bootloader**, vagyis rendszerbetöltő.

5. Elindul a rendszerbetöltő

A bootloader már tudja, hol található az operációs rendszer legfontosabb része.

Betölti azt a háttértárról a RAM-ba.

Ez a rész az operációs rendszer **kernelje**, vagyis rendszermagja.

6. Életre kel a kernel

A kernel átveszi a számítógép irányítását.

Beállítja a memóriakezelést, elindítja az eszközmeghajtókat, felismeri a hardvereket, hozzáfér a fájlrendszerekhez, és létrehozza azokat az alapvető folyamatokat, amelyekre a rendszernek szüksége van.

Most már valóban az operációs rendszer vezeti a gépet.

7. Elindulnak a szolgáltatások

A háttérben sorra életre kelnek a rendszer szolgáltatásai.

Hálózat.

Hang.

Grafikus rendszer.

Biztonsági szolgáltatások.

Bejelentkezés.

És még számos olyan folyamat, amelyet a felhasználó soha nem lát.

8. Megjelenik az asztal

Végül elindul a grafikus felhasználói környezet.

Megjelenik az asztal, a tálca vagy Dock, az ikonok és az egérmutató.

A gép használatra kész.

Az egész folyamat leegyszerűsítve:

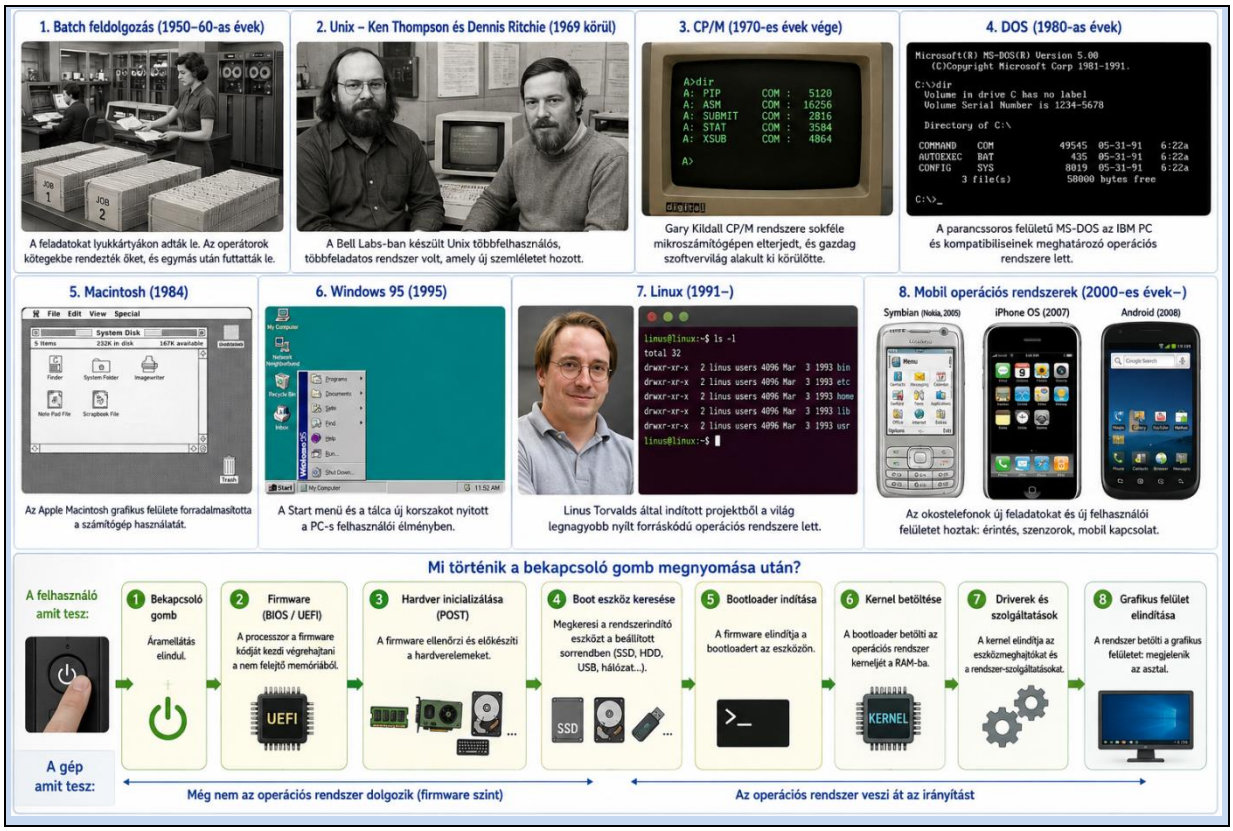
Bekapcsolás → firmware (BIOS/UEFI) → hardver inicializálása → bootloader → kernel → eszközmeghajtók és szolgáltatások → grafikus felület → felhasználó

Mindez egy modern számítógépen gyakran néhány másodperc alatt lezajlik.

A legérdekesebb talán az első lépés: **hogyan indulhat el egy program olyan gépen, amelyen még semmilyen program nem fut?**

A válasz az, hogy valaminek mindig már ott kell lennie.

Ez a nem felejtő memóriában tárolt firmware indítja el azt a láncreakciót, amelynek végén egy üres memóriájú elektronikus szerkezetből használatra kész számítógép lesz.


1. Batch feldolgozás (1950–60-as évek)
A feladatokat lyukkártyákon adták le. Az operátorok kötegekbe rendezték őket, és egymás után futtatták le.
2. Unix – Ken Thompson és Dennis Ritchie (1969 körül)
A Bell Labs-ban készült Unix többfelhasználós, többfeladatos rendszer volt, amely új szemléletet hozott.
3. CP/M (1970-es évek vége)
A>dir
A: PIP COM : 5120
A: ASM COM : 16256
A: SUBMIT COM : 2816
A: STAT COM : 3584
A: XSUB COM : 4864
A>
Gary Kildall CP/M rendszere sokféle mikroszámítógépen elterjedt, és gazdag szoftvervilág alakult ki körülötte.
4. DOS (1980-as évek)
Microsoft(R) MS-DOS(R) Version 5.00
(C)Copyright Microsoft Corp 1981-1991.
C:\>dir
Volume in drive C has no label
Volume Serial Number is 1234-5678
Directory of C:\
COMMAND COM 49545 05-31-91 6:22a
AUTOEXEC BAT 435 05-31-91 6:22a
CONFIG SYS 8019 05-31-91 6:22a
3 file(s) 58000 bytes free
C:\>_
A parancssoros felületű MS-DOS az IBM PC és kompatibiliseinek meghatározó operációs rendszere lett.
5. Macintosh (1984)
Az Apple Macintosh grafikus felülete forradalmasította a számítógép használatát.
6. Windows 95 (1995)
A Start menü és a tálca új korszakot nyitott a PC-s felhasználói élményben.
7. Linux (1991–)
linus@linux:~$ ls -l
total 32
drwxr-xr-x 2 linus users 4096 Mar 3 1993 bin
drwxr-xr-x 2 linus users 4096 Mar 3 1993 etc
drwxr-xr-x 2 linus users 4096 Mar 3 1993 home
drwxr-xr-x 2 linus users 4096 Mar 3 1993 lib
drwxr-xr-x 2 linus users 4096 Mar 3 1993 usr
linus@linux:~$
Linus Torvalds által indított projektből a világ legnagyobb nyílt forráskódú operációs rendszere lett.
8. Mobil operációs rendszerek (2000-es évek–)
Symbian (Nokia, 2005)
iPhone OS (2007)
Android (2008)
Az okostelefonok új feladatokat és új felhasználói felületet hoztak: érintés, szenzorok, mobil kapcsolat.
Mi történik a bekapcsoló gomb megnyomása után?
A felhasználó amit tesz:
A gép amit tesz:
1 Bekapcsoló gomb
Áramellátás elindul.
2 Firmware (BIOS / UEFI)
A processzor a firmware kódját kezdi végrehajtani a nem felejtő memóriából.
UEFI
3 Hardver inicializálása (POST)
A firmware ellenőrzi és előkészíti a hardverelemeket.
4 Boot eszköz keresése
Megkeresi a rendszerindító eszközt a beállított sorrendben (SSD, HDD, USB, hálózat...).
SSD
5 Bootloader indítása
A firmware elindítja a bootloadert az eszközön.
6 Kernel betöltése
A bootloader betölti az operációs rendszer kerneljét a RAM-ba.
KERNEL
7 Driverek és szolgáltatások
A kernel elindítja az eszközmeghajtókat és a rendszer-szolgáltatásokat.
8 Grafikus felület elindítása
A rendszer betölti a grafikus felületet: megjelenik az asztal.
Még nem az operációs rendszer dolgozik (firmware szint)
Az operációs rendszer veszi át az irányítást

IV. rész – Mire használjuk a számítógépet?

9. A számítógép, mint iroda

Az első számítógépeket elsősorban számításokra építették. Tudományos képleteket oldottak meg, ballisztikai táblázatokat készítettek, népszámlálási és üzleti adatokat dolgoztak fel.

A személyi számítógép azonban egészen más környezetbe is beköltözött: **az irodába**.



Ábra:
Az iroda átalakulása az 1980-as években. Az írógép, számológép, kartotékszekrény és papíralapú nyilvántartások feladatait fokozatosan egyetlen eszköz, a személyi számítógép vette át.

Ennek jelentőségét ma már nehéz érzékelni. Egyetlen számítógép olyan eszközöket és munkafolyamatokat egyesített, amelyek korábban külön gépeket, külön szakmákat, sőt egész helyiségeket igényeltek.

A számítógép fokozatosan **univerzális irodai munkaeszközzé** vált.

9.1. Szövegszerkesztés – az intelligens írógép

Az írógépnek volt egy nagyon kellemetlen tulajdonsága: amit egyszer leírtunk, az ott maradt a papíron.

Ha hibáztunk, javítani kellett. Ha egy bekezdést át akartunk helyezni, gyakran újra kellett gépelni az egész oldalt. Ha egy hosszabb dokumentum elején változtattunk valamit, az oldaltörések miatt akár több oldalt is újra kellett készíteni.

A szövegszerkesztő mindezt megváltoztatta.

A szöveg először már nem papírra, hanem **memóriába került.**

Ez látszólag apró különbség, valójában forradalom volt.

A mondatokat át lehetett írni. Bekezdéseket lehetett mozgatni. Szavakat lehetett keresni és cserélni. Ugyanazt a dokumentumot újra és újra ki lehetett nyomtatni.

A szöveg **szerkeszthető adattá** vált.

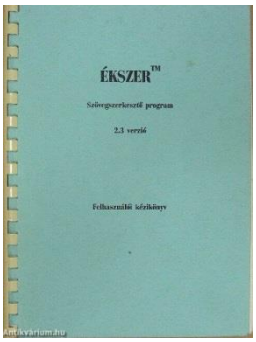
WordStar – a korai szabvány

Az 1978-ban megjelent **WordStar** a mikroszámítógépek egyik első széles körben elterjedt szövegszerkesztője volt. Eredetileg CP/M alatt futott, később DOS-változata is megjelent.



Ábra:
WordStar, a mikroszámítógépes korszak egyik első sikeres szövegszerkesztője. A mai grafikus programoktól nagyon eltérő képernyő ellenére már lehetővé tette a dokumentum javítását, átrendezését és ismételt kinyomtatását.

Ékszer – magyar szövegszerkesztő



A magyar számítástechnika történetéhez hozzátartozik az **Ékszer** szövegszerkesztő is.

Különösen fontos volt, hogy a magyar nyelvű szövegkezelés saját problémákat vetett fel. Az ékezetes karakterek kezelése a korai számítógépeken korántsem volt olyan természetes, mint ma.

Ábra:
Ékszer – magyar szövegszerkesztő. A hazai fejlesztésű programoknak olyan problémákat is meg kellett oldaniuk, mint a magyar ékezetes betűk bevitele, megjelenítése és nyomtatása.

A szövegszerkesztők fejlődésének következő nagy állomása a grafikus felület lett. A képernyőn egyre inkább azt lehetett látni, ami később a papírra került.

Megszületett a **WYSIWYG – What You See Is What You Get**, vagyis „azt kapod, amit látsz” elve.

A Microsoft Word és társai végül gyakorlatilag teljesen kiszorították az írógépet az irodákból.

Amikor a számítógép megtanult olvasni – az OCR

A számítógép sokáig csak azt a szöveget ismerte, amelyet billentyűzeten begépeltek vagy valamilyen más módon kódolt formában vittek be. Egy könyv lapjáról vagy egy gépelt dokumentumról készült kép számára csupán képpontok sokasága volt. A következő nagy lépés az volt, amikor a gép megtanulta felismerni, hogy a képen látható vonalak és foltok valójában betűket és számokat alkotnak.

Ezt nevezzük **optikai karakterfelismerésnek**, angolul *Optical Character Recognitionnek*, röviden **OCR-nek**.

Az ötlet meglepően régi. Már a 20. század elején születtek olyan szerkezetek, amelyek nyomtatott jeleket próbáltak elektromos úton felismerni. Az 1950-es és 1960-as évektől bankokban, postai rendszerekben és nagyvállalatoknál jelentek meg gyakorlati karakterfelismerő berendezések. Ezek azonban még erősen korlátozottak voltak: többnyire meghatározott betűtípusokat és szabályosan nyomtatott karaktereket tudtak elolvasni.

A személyi számítógépek, a lapolvasók és az egyre nagyobb számítási teljesítmény az 1980-as és 1990-es években új korszakot nyitott. Egy dokumentumot már egyszerűen be lehetett szkennelni, az OCR-program pedig megpróbálta felismerni a betűket, szavakat, majd később a teljes oldal szerkezetét is.

Ez alapvetően megváltoztatta a papíron tárolt információ kezelését. Régi könyvek, újságok és iratok millióit lehetett kereshető, szerkeszthető digitális szöveggé alakítani. A digitalizálás többé nem jelentette feltétlenül azt, hogy valakinek oldalról oldalra újra be kellett gépelnie a dokumentumot.

Az OCR azonban sokáig könnyen zavarba jött. A rossz minőségű nyomat, a megsárgult papír, a foltok, a ferdén beolvasott oldal, a különleges betűtípus vagy akár egy egyszerű „rn” betűkapcsolat is megtéveszthette. A régi könyvek digitalizálásánál ezért ma is gyakran találkozunk furcsa felismerési hibákkal.

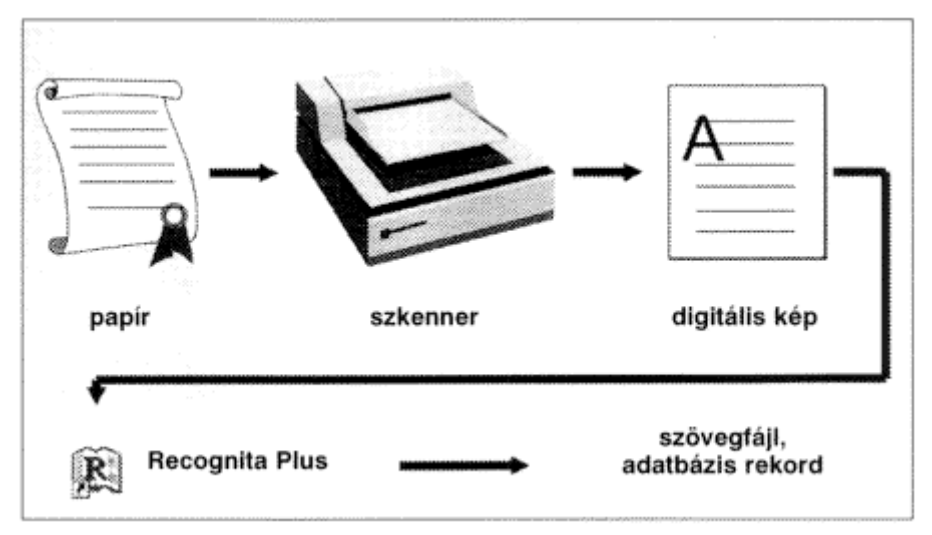
A mai rendszerek már nem egyszerűen egy-egy betű alakját hasonlítják össze mintákkal. A karakterek mellett figyelembe veszik a szavakat, a mondat környezetét és az oldal szerkezetét is. A gépi tanulás és a neurális hálózatok pedig tovább javították a felismerést. A modern rendszerek már kézírást, táblázatokat, többhasábos oldalakat, fényképezett dokumentumokat, sőt történelmi nyomtatványokat is képesek feldolgozni.

Az OCR így érdekes átmenetet jelent a hagyományos számítástechnika és a mesterséges intelligencia között. A gép már nem egyszerűen tárolja a betűket: **megpróbálja felismerni őket.**

MAGYAR SZÁL – Recognita: magyar program, amely megtanította olvasni a számítógépet

A magyar számítástechnika egyik kevésbé ismert, pedig nemzetközileg is kiemelkedő sikertörténete a **Recognita optikai karakterfelismerő program**.

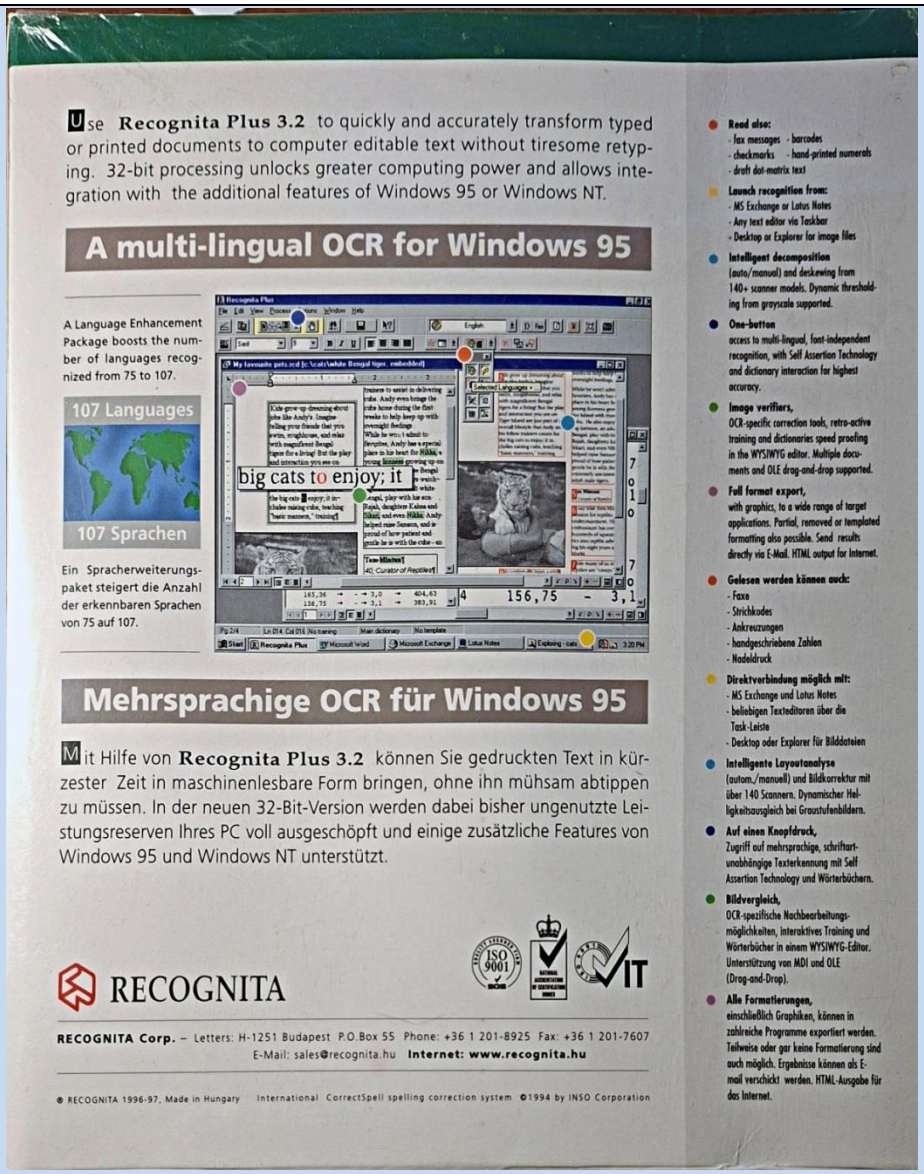
A történet az 1980-as évek közepén, a **Számítástechnikai Koordinációs Intézetben (SZKI)** kezdődött. **Kovács Emőke** 1984-ben kezdett karakterfelismeréssel foglalkozni, majd 1985-ben **Marosi István** is bekapcsolódott a fejlesztésbe. A cél egyszerűen megfogalmazható, de az akkori számítógépekkel rendkívül nehezen megvalósítható volt: a számítógép egy beszkennelt dokumentum képéből ismerje fel a betűket, és készítsen belőle szerkeszthető szöveget.



1987-re elkészült a Recognita 1.0, amely már nemzetközi terjesztésre is alkalmas volt. A magyar program hamarosan a világ élvonalába került: 1988-ra már az európai OCR-piac egyik vezető termékévé vált.

A Recognita egyik nagy erőssége a sokoldalúsága volt. Nem csupán néhány előre meghatározott betűtípust tudott felismerni, hanem különféle nyomtatott szövegekkel is megbirkózott. Fejlesztése során egyre több írásrendszert és nyelvet támogatott; végül 114 latin, görög és cirill betűs nyelv karakterkészletének kezelésére volt alkalmas.

A siker különösen figyelemre méltó, ha belegondolunk, milyen számítógépeken kellett mindezt megvalósítani. Az 1980-as évek személyi számítógépeinek teljesítménye a mai gépekéhez képest elenyésző volt. A fejlesztők ezért nem támaszkodhattak egyszerűen hatalmas számítási kapacitásra: **jó algoritmusokra volt szükségük**.



Recognita Plus 3.2, 1996–97. A Windows 95 és Windows NT rendszerre készült változat már 107 nyelv felismerését kínálta. A dobozon büszkén szerepelt: **„Made in Hungary”**. *A szerző gyűjteményéből.*

1996-ra már **28 országban mintegy 140 000 Recognita-program működött.** Ebben az évben az amerikai **Caere Corporation**, az OmniPage OCR-program fejlesztője megvásárolta a Recognita céget. A magyar fejlesztés ezzel nem tűnt el. Technológiája és szakemberei bekapcsolódtak az OmniPage további fejlesztésébe: Kovács Emőke az OmniPage OCR-algoritmusain dolgozott, Marosi István pedig később az OmniPage Capture SDK fejlesztésében játszott fontos szerepet.

A Recognita története ezért nem egyszerűen egy sikeres magyar program története. Azt mutatja, hogy egy magyar fejlesztőcsapat olyan technológiát hozott létre, amely **nemzetközi versenyben is az élvonalba került, majd beépült a világ egyik legismertebb OCR-termékének fejlődésébe.**



Recognita Plus OCR-program. A magyar fejlesztésű karakterfelismerő program egyik korabeli kiadása. A találó „Falja a betűket” reklámgrafika jól érzékelteti az OCR lényegét: a számítógép a beszkennelt dokumentum képéből szerkeszthető szöveget készít. *A szerző gyűjteményéből.*

A szövegszerkesztő megtanította a számítógépet írni.
A Recognita segített megtanítani olvasni.

9.2. Táblázatkezelés – az elektronikus kockás papír

Az irodák másik nélkülözhetetlen eszköze a számolótábla volt.

Sorok, oszlopok, számok, összegek, százalékok.

Mi történik azonban, ha egy korábbi adat megváltozik?

Papíron sok mindent újra kellett számolni.

Az elektronikus táblázat egyik zseniális ötlete az volt, hogy a cellákba nemcsak számokat, hanem **képleteket** is lehetett írni.

Ha egy adat megváltozott, a számítógép automatikusan újraszámolta az eredményt.

VisiCalc

1979-ben jelent meg az Apple II számítógépre a **VisiCalc**, Dan Bricklin és Bob Frankston programja.





Ábra:
VisiCalc az Apple II-n. Az első széles körben sikeres elektronikus táblázatkezelő megmutatta, hogy a személyi számítógép nemcsak érdekes technikai újdonság, hanem komoly üzleti munkaeszköz is lehet.

A VisiCalc után következett többek között a Lotus 1-2-3, majd a grafikus környezetben működő Microsoft Excel.

B1: "Principal MENU

Worksheet Range Copy Move File Print Graph Data Quit

Global, Insert, Delete, Column-Width, Erase, Titles, Window, Status

	A	B	C	D	E
1		Principal	$50,000		
2		Rate	13.0%		
3		Years	5		
4		Payment	1,137.65		
5					
6	Year	Begin Bal.	End Bal.	Total Paid	Interest
7	1	50,000.00	42,406.26	13,651.84	6,058.10
8	2	42,406.26	33,764.33	13,651.84	5,009.92
9	3	33,764.33	23,929.53	13,651.84	3,817.05
10	4	23,929.53	12,737.22	13,651.84	2,459.53
11	5	12,737.22	0.00	13,651.84	914.63
12					
13					
14					
15					
16					
17					
18					
19					
20					



Ábra:
A táblázatkezelők következő nemzedéke. A Lotus 1-2-3 az IBM PC korszakának meghatározó üzleti alkalmazása lett, az Excel pedig a grafikus felület lehetőségeit kihasználva vitte tovább az elektronikus táblázat ötletét.

9.3. Adatbázisok – a kartotékszekrény beköltözik a gépbe

Egy iroda nemcsak szövegeket ír és számol.

Adatokat is tárol.

Ügyfelek neve.

Címek.

Termékek.

Megrendelések.

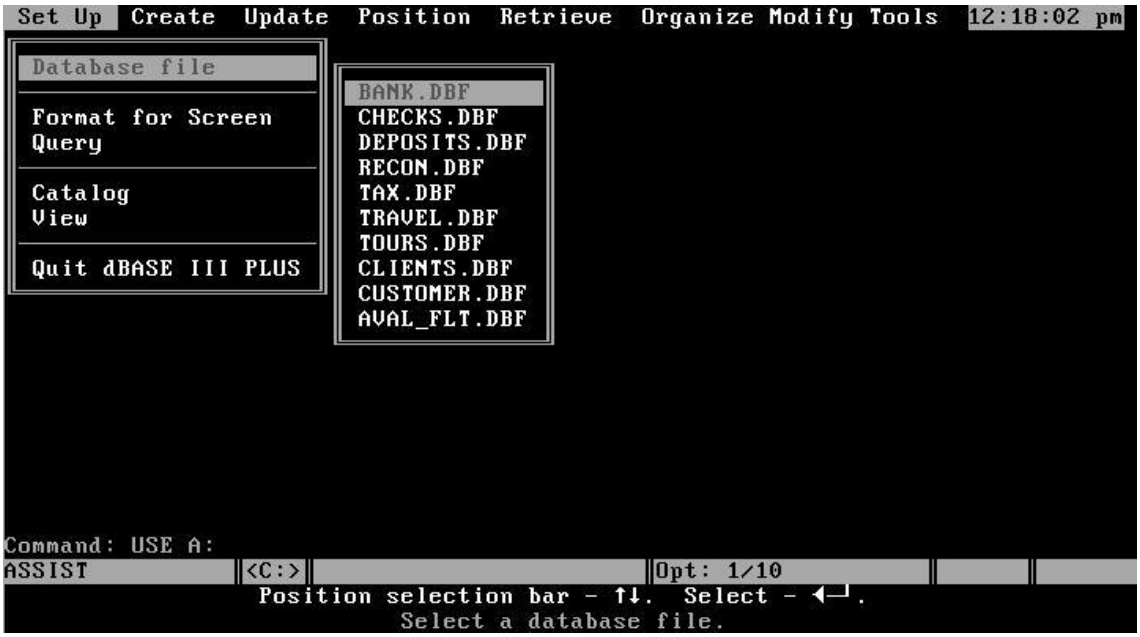
Számlák.

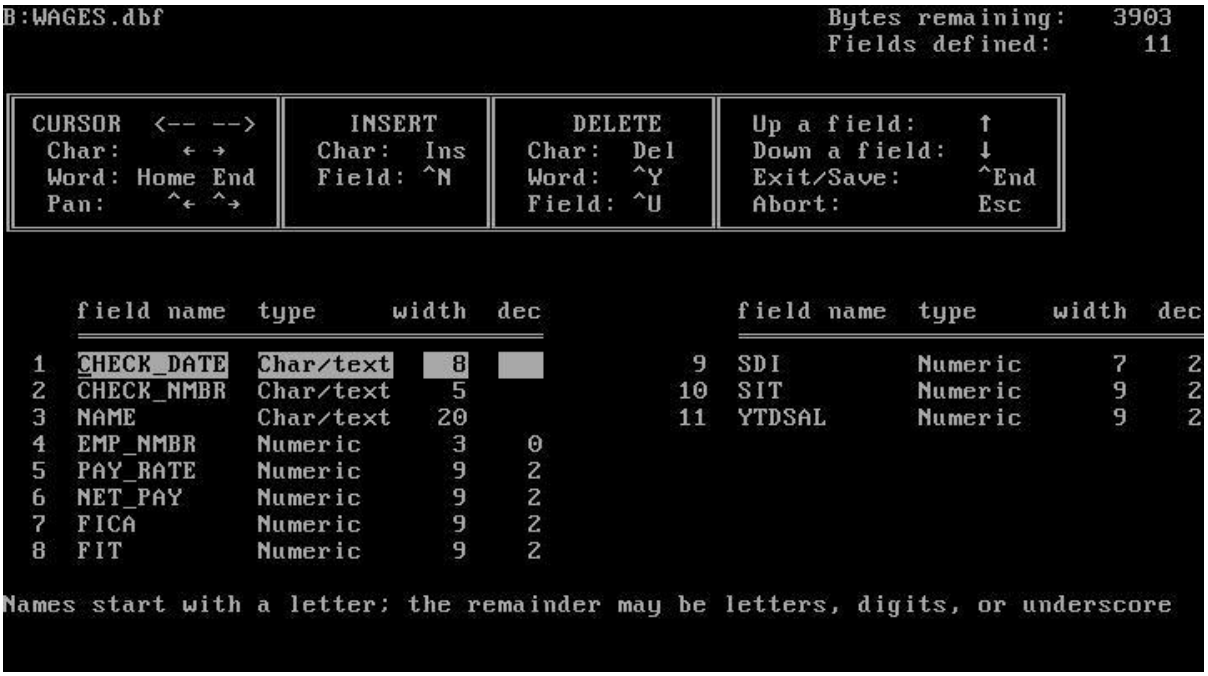
Alkalmazottak.

Korábban mindez kartotékokon, könyvekben és irattartókban élt.

Az adatbázis-kezelők ezeket az információkat tették elektronikusan tárolhatóvá, kereshetővé és összekapcsolhatóvá.

A korai személyi számítógépes korszak egyik legismertebb adatbázis-kezelője a **dBASE** lett.





Ábra:
dBASE, a személyi számítógépes adatbázis-kezelés egyik meghatározó rendszere. Segítségével már egy kisebb vállalkozás is saját elektronikus nyilvántartásokat és ügyviteli rendszereket készíthetett.

A valódi áttörés azonban nem egyszerűen az volt, hogy a kartotékok bekerültek a számítógépbe.

Az elektronikus adatbázis **kereshető** volt.

Több ezer vagy több millió rekord közül pillanatok alatt ki lehetett választani azokat, amelyek megfeleltek bizonyos feltételeknek.

A kartotékszekrényből információs rendszer lett.

9.4. Kiadványszerkesztés – a nyomda beköltözik az irodába

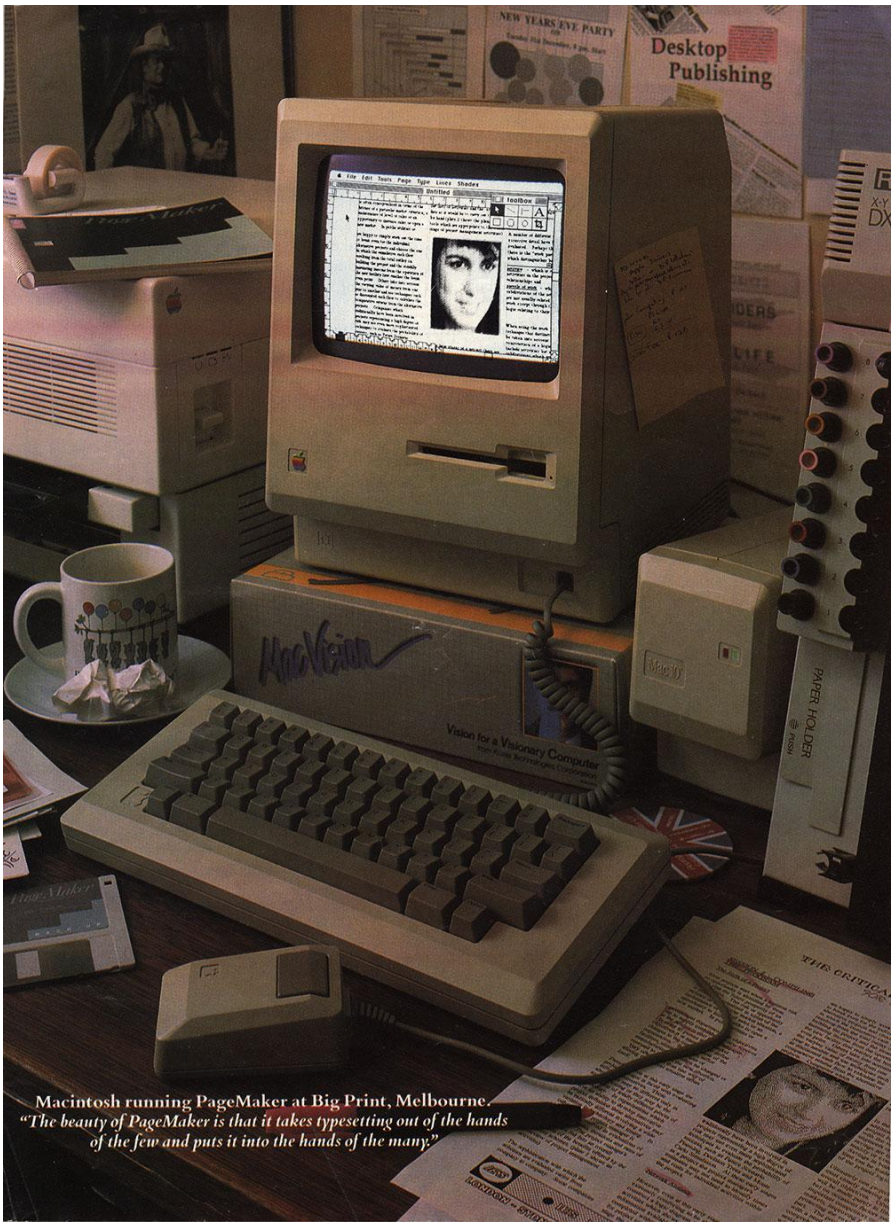
A személyi számítógép következő meglepő hódítása a nyomdai előkészítés volt.

Ehhez három technológia találkozására volt szükség:

grafikus számítógép + lézernyomtató + kiadványszerkesztő program.

Az 1980-as évek közepén az Apple Macintosh grafikus felülete, az Apple LaserWriter lézernyomtató és az Aldus PageMaker program együtt új fogalmat teremtett:

Desktop Publishing – DTP, vagyis asztali kiadványszerkesztés.



Ábra:
Az asztali kiadványszerkesztés megszületése. A Macintosh, a lézernyomtató és az Aldus PageMaker segítségével olyan tördelési feladatokat lehetett egy íróasztalon elvégezni, amelyek korábban professzionális nyomdai berendezéseket igényeltek.

Szöveget és képet lehetett egy oldalon elhelyezni, betűtípust és betűméretet választani, hasábokat kialakítani, képeket méretezni, majd az eredményt nagy felbontásban kinyomtatni.

Egy kis szerkesztőség, vállalkozás vagy akár egyetlen ember is professzionális külsejű kiadványokat készíthetett.

A személyi számítógép ezzel már nemcsak írógép, számológép és kartotékszekrény lett.

Kis nyomdává is változott.

VisiCalc – a program, amely miatt számítógépet vásároltak

1978-ban a Harvard Business School hallgatója, **Dan Bricklin** egy előadáson figyelte, ahogy tanára egy táblára felírt pénzügyi modellt módosít.

Egyetlen szám megváltoztatása miatt több kapcsolódó eredményt újra kellett számolni.

Bricklin fejében megszületett egy kérdés:

Mi lenne, ha a tábla maga számolná újra az eredményeket?

Bob Frankstonnal együtt elkészítettek egy programot az Apple II számítógépre.

Ez lett az 1979-ben megjelent **VisiCalc**, nevében a *visible calculator* – „látható számológép” – rövidítése.



A képernyő egy papíralapú táblázatra emlékeztetett.

Sorok és oszlopok metszéspontjában cellák helyezkedtek el. Egy cellába számot, szöveget vagy képletet lehetett írni.

Az igazi varázslat akkor történt, amikor a felhasználó megváltoztatott egy adatot.

A VisiCalc automatikusan újraszámolta azokat a cellákat, amelyek ettől függtek.

Ma ez magától értetődőnek tűnik.

1979-ben azonban elképesztő lehetőség volt.

Egy pénzügyi terv különböző változatait percek alatt ki lehetett próbálni:

Mi történik, ha 10%-kal nő az ár?
Mi történik, ha csökken a forgalom?
Mi történik, ha emelkedik a kamat?

A számítógép ezzel nem egyszerűen gyorsabban számolt.

Lehetővé tette a „mi lenne, ha?” gondolkodást.

A VisiCalc üzletemberek számára is értelmet adott az addig sokszor hobbinak vagy technikai érdekességnek tekintett személyi számítógépnek.

Az Apple II iránti keresletre olyan erős hatással volt, hogy a VisiCalcot gyakran a személyi számítógép történetének első **„killer application”**, vagyis „gépvásárlásra késztető alkalmazásaként” emlegetik.

Korábban többnyire azért vásároltak programot, mert volt számítógépük.

A VisiCalc megfordította a logikát:

számítógépet vásároltak azért, hogy használhassák a programot.

Ez a szoftveripar történetének egyik legfontosabb fordulata.

Prezi – magyar ötlet, amely újragondolta a prezentációt

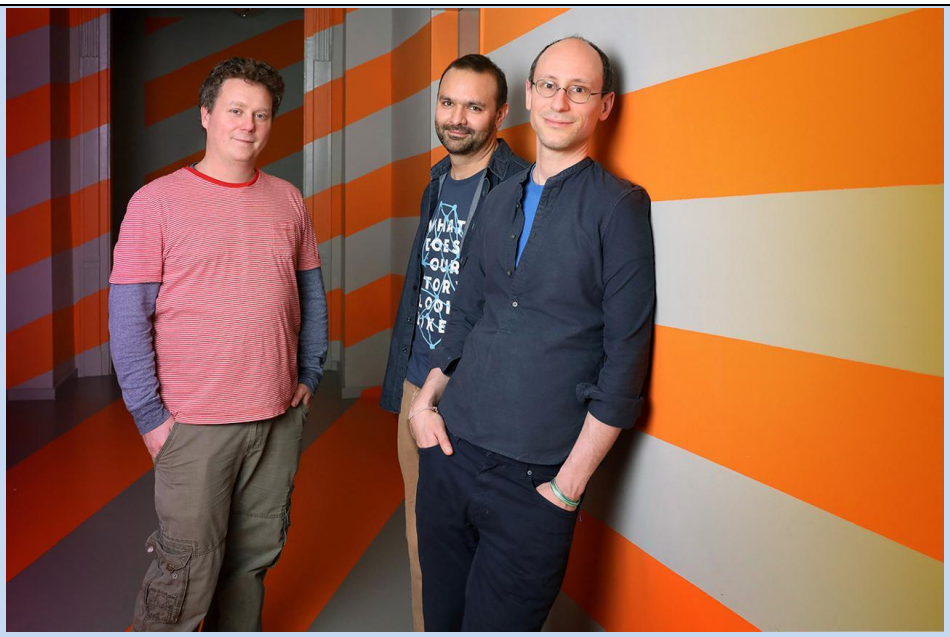
A legtöbb prezentációs program diák sorozatából áll. Az előadó az első diától az utolsóig halad, mintha egy vetített diaképsort mutatna be.

A **Prezi** szakított ezzel a több évtizedes szemlélettel.

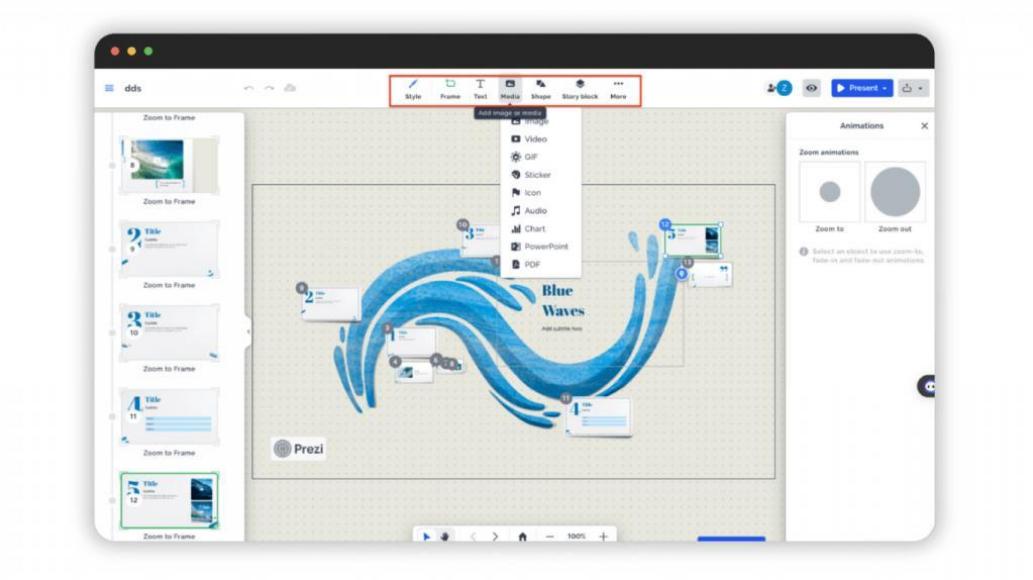
A program alapötlete az volt, hogy az egész előadás **egyetlen nagy vásznon** helyezkedik el. Az előadó ezen a vásznon nagyíthat, kicsinyíthet, elfordulhat, és tetszőleges útvonalon vezetheti végig a hallgatóságot.

A rendszer egyszerre mutathatta az összképet és a részleteket.

A Prezit **2009-ben** alapította Árvai Péter, Halácsy Péter és Somlai-Fischer Ádám.



A vállalkozás rövid idő alatt nemzetközi sikert aratott. Felhasználói száma néhány év alatt milliókra nőtt, és a programot iskolákban, egyetemeken, vállalatoknál és konferenciákon világszerte használni kezdték.



Ábra: A Prezi új szemléletet hozott a prezentációk világába. A hagyományos diasor helyett egyetlen nagy virtuális vásznon helyezkednek el a témák, amelyek között a bemutató szabadon mozoghat.

A Prezi nem csupán új szoftver volt.

Új gondolkodásmódot vezetett be.

Nem egymás után következő diákat mutatott, hanem **egy gondolattérképet**, amelyben a témák egymással kapcsolatban álltak.

Ez sok esetben jobban tükrözte az emberi gondolkodás szerkezetét is.

A Prezi története azt bizonyítja, hogy egy kis országból induló vállalkozás is képes lehet olyan ötlettel előállni, amely világszerte megváltoztatja emberek millióinak munkáját.

A magyar informatika nemcsak kiváló mérnököket adott a világnak, hanem olyan szoftvereket is, amelyeket naponta emberek milliói használnak.

Hová tűntek a tegnap sztárprogramjai?

Az 1980-as és 1990-es évek számítógép-használója egészen más programneveket ismert, mint a mai. A **WordStar** az egyik legismertebb szövegszerkesztő volt. Később a **WordPerfect** uralta a piac jelentős részét. Táblázatkezelésben a **Lotus 1-2-3** számított etalonnak, mellette ott volt a **Quattro Pro**. Adatbázis-kezelésre milliók használták a **dBASE**, majd a **FoxBase/FoxPro** programokat. A kiadványszerkesztés úttörője, a **Ventura Publisher** szintén fogalom volt. Segédprogramok királya, a **Norton Commander** és **Utilities**. stb. stb.

Ma ezek közül sokat már csak az informatika története iránt érdeklődők ismernek.

Mi történt velük? Rosszak lettek?

Többnyire nem. Éppen ellenkezőleg: gyakran annyira sikeresek voltak, hogy fejlesztőik nehezen merték feladni azt, ami sikeressé tette őket.

A DOS csapdája

A nyolcvanas évek vezető alkalmazásai karakteres képernyőre, billentyűzetes vezérlésre és DOS-ra készültek. Felhasználóik megtanulták a parancsaikat és billentyűkombinációikat, a vállalatok pedig hatalmas mennyiségű dokumentumot és adatot hoztak létre velük.

Amikor megjelent a grafikus felület, az egér és a Windows, nem volt elegendő néhány új menüt hozzáadni a régi programhoz. Sok esetben szinte újra kellett gondolni az egész alkalmazást.

A korábbi siker így furcsa módon teherré válhatott.

Nem mindig a jobb program győz

A szoftverpiacon önmagában a technikai minőség ritkán elegendő. Fontossá vált az operációs rendszerrel való együttműködés, a fájlformátumok elterjedtsége, a felhasználói megszokás, a vállalati beszerzés és az is, hogy milyen más programokkal együtt lehetett megvásárolni az alkalmazást.

A Microsoft ebben különösen erős helyzetbe került. A Windows elterjedésével a **Word, Excel, PowerPoint és később az Access** egységes irodai környezetet alkotott. A Microsoft Office sikere már nem négy különálló program sikere volt, hanem egy **ökoszisztémáé.**

A versenytársaknak ettől kezdve nem egyszerűen egy jobb szövegszerkesztőt vagy táblázatkezelőt kellett készíteniük. Egy egész rendszerrel kellett versenyezniük.

A piacvezető veszélyes helyzetben van

A WordPerfect története jól mutatja a problémát. DOS alatt rendkívül sikeres volt, de a grafikus Windows-világba való átmenet nehézkesen zajlott. Mire megfelelő Windows-változatai megjelentek, a Microsoft Word már jelentős előnyre tett szert.

A WordStar még korábban veszített lendületéből.

A Lotus 1-2-3 a DOS-korszak táblázatkezelőjének szinte szinonimája volt, de Windows alatt az Excel fokozatosan átvette a helyét.

A dBASE körül pedig egy egész programozási kultúra alakult ki. A formátum és a programozási nyelv olyan sikeres lett, hogy számos kompatibilis rendszer – köztük a FoxBase – jelent meg. A Microsoft később megszerezte a Fox Software-t, a FoxPro pedig Visual FoxPro néven még sokáig tovább élt, de végül annak fejlesztése is megszűnt.

A Ventura Publisher különösen érdekes eset. Az asztali kiadványszerkesztés egyik korai nagy sikere volt, tulajdonosai azonban többször változtak, miközben a DTP-piac is átalakult. Végül más rendszerek kerültek előtérbe.

És eltűntek a fejlesztők is

A programokkal együtt sokszor az őket létrehozó vállalatok neve is kikopott a köztudatból:

MicroPro → WordStar
WordPerfect Corporation → WordPerfect
Lotus Development Corporation → Lotus 1-2-3
Borland → Quattro Pro
Ashton-Tate → dBASE
Fox Software → FoxBase, FoxPro
Ventura Software → Ventura Publisher

Egyes cégeket felvásároltak, mások átalakultak vagy eltűntek. Maguk a programok pedig olykor új tulajdonosnál még évekig tovább éltek.

Miért bukhat el egy sikeres program?

A történetekből néhány közös ok rajzolódik ki:

technológiai platformváltás – DOS → Windows → web → mobil → felhő;

a régi siker terhe – a meglévő felhasználók és kompatibilitási követelmények akadályozhatják a radikális megújulást;

késői alkalmazkodás – mire elkészül az új változat, a versenytárs már elfoglalta a piacot;

ökoszisztémák kialakulása – egy önálló programnak egy komplett programcsomaggal kell versenyeznie;

felvásárlások és vállalati döntések – egy kiváló termék sorsát nem feltétlenül a fejlesztői határozzák meg;

hálózati hatás – minél többen használnak egy fájlformátumot vagy programot, annál nagyobb előnyt jelent ugyanazt választani.

A számítástechnika történetének egyik fontos tanulsága ezért az, hogy **a piacvezető helyzet sohasem végleges**.

A WordStar, a WordPerfect vagy a Lotus 1-2-3 (a sor még hosszan folytatható lenne) a maga korában nem jelentéktelen program volt. Éppen ellenkezőleg: olyan sikeresek voltak, hogy egy időben nehéznek tűnt elképzelni nélkülük a személyi számítógépet.

Néhány évtizeddel később pedig már történelmet tanulunk róluk.

A szoftvervilágban nem elég egyszer jól megírni egy programot. Minden nagy technológiai váltásnál szinte újra meg kell nyerni a piacot.

Az örök túlélők – hogyan maradhat életben egy program negyven évig?

A számítástechnika története tele van egykori sztárprogramokkal, amelyek néhány év vagy évtized után eltűntek. Még érdekesebbek azonban azok, amelyek túlélték a DOS-t, a Windows megjelenését, az internetet, a mobilforradalmat és a felhő korszakát is.

Az **AutoCAD első változata 1982-ben** jelent meg. A program azóta természetesen szinte teljesen átalakult, de a név és az alapvető feladat – a számítógéppel segített tervezés – megmaradt.

A **Microsoft Excel** elődjeinek története az 1980-as évek elejére nyúlik vissza, maga az Excel pedig 1985-ben jelent meg Macintoshra, majd 1987-ben Windowsra. Nemcsak túlélte a nála egykor sokkal erősebb Lotus 1-2-3-at, hanem gyakorlatilag a táblázatkezelő fogalmának szinonimájává vált.

Az **Adobe Photoshop** 1990-ben került kereskedelmi forgalomba. Azóta a számítógépes képfeldolgozás világa alapjaiban változott meg, de a Photoshop folyamatosan együtt változott vele.

Hasonlóan hosszú életű számos műszaki, grafikai és professzionális alkalmazás is.

Mi a túlélés titka?

Elsősorban az, hogy valójában nem ugyanaz a program él negyven éve.

A mai AutoCAD, Excel vagy Photoshop belülről és képességeit tekintve alig hasonlít első változatára. A fejlesztők új hardverekhez, grafikus felületekhez, operációs rendszerekhez, fájlformátumokhoz, internetes szolgáltatásokhoz és újabban a mesterséges intelligenciához igazították őket.

A sikeres programnak tehát különös kettősséget kell megvalósítania:

meg kell változnia úgy, hogy közben önmaga maradjon.

A felhasználó felismeri a nevét, megtalálja régi munkamódszereinek legalább egy részét, és lehetőség szerint továbbra is hozzáfér korábbi dokumentumaihoz. Közben azonban a program mögött szinte minden kicserélődhet.

A hosszú élet másik titka a **felhasználói közösség**. Ha emberek milliói ismernek egy programot, vállalatok építették rá működésüket, tanfolyamokon oktatják, könyvek és segédanyagok készültek hozzá, valamint hatalmas dokumentumállomány halmozódott fel a formátumában, akkor maga ez a környezet is értékké válik. Ehhez, bár ellentmondásos módon, de az illegális szoftvermásolás is hozzájárult.

Ez azonban csak addig jelent előnyt, amíg a fejlesztő képes továbblépni.

A Lotus 1-2-3 hatalmas felhasználói tábora nem mentette meg a programot az Exceltől. A WordPerfect korábbi sikere sem akadályozta meg a Word térnyerését.

A szoftver evolúciója

A programok története meglepően hasonlít a biológiai evolúcióra.

A környezet folyamatosan változik:

DOS → grafikus felület → Windows → internet → mobil eszközök → felhő → mesterséges intelligencia

Minden ilyen változás új szelekciós nyomást jelent.

A korábban sikeres tulajdonság már nem feltétlenül előny. Új versenytársak jelennek meg. Egyes programok alkalmazkodnak, mások eltűnnek, néhány pedig új formában él tovább.

Ezért a negyvenéves szoftver valójában nem negyvenéves programkód.

Negyven éve folyamatosan változó programcsalád.

Talán ez a szoftvertörténet egyik legfontosabb tanulsága:

nem feltétlenül a legjobb program marad életben, hanem az, amelyik képes újra és újra alkalmazkodni a megváltozott környezethez.

Rabló és pandúr – a szoftvermásolás története

A személyi számítógépek megjelenésével különös áru született. Egy autó, könyv vagy televízió lemásolásához nyersanyag, gép és munka kell. Egy számítógépes programból viszont néhány perc alatt készíthető az eredetivel gyakorlatilag azonos másolat.

A digitális információ sokszorosítása szinte semmibe sem kerül – az elkészítése viszont akár vagyonokba kerülhet.

Ez az ellentmondás a szoftveripar történetét kezdettől fogva végigkíséri.

Szoftverbörzék és lemezmásolás

A személyi számítógépek korai korszakában a programok gyakran kézről kézre jártak. Barátok, klubok és számítógépes találkozók résztvevői kazettákat és floppy lemezeket másoltak egymásnak. Magyarországon és más kelet-európai országokban különösen sajátos helyzet alakult ki: a nyugati programok hivatalos beszerzése sokáig nehéz volt, később pedig áruk a hazai jövedelmekhez képest rendkívül magas lehetett.

A **szoftverbörzék** ezért nem csupán kereskedelmi helyek voltak. Információs csomópontként is működtek: programokat, játékokat, segédprogramokat és tapasztalatokat cseréltek az emberek.

A jogtulajdonos számára azonban az engedély nélküli másolat jogosulatlan felhasználás volt.

Megkezdődött a több évtizedes **rabló–pandúr játék.**

Bill Gates már 1976-ban tiltakozott

A szoftvermásolás körüli vita szinte egyidős a személyi számítógéppel.

1976 februárjában a mindössze húszéves **Bill Gates** nyílt levelet írt a számítógépes hobbistákhoz *An Open Letter to Hobbyists* címmel. Gates ekkor Paul Allennel együtt az Altair BASIC fejlesztője volt, és felháborította, hogy programjukat a számítógépes klubok tagjai szabadon másolták egymás között.

Érvelése lényegében ma is ismerős volt: egy jó program elkészítéséhez rengeteg munka szükséges. Ha viszont mindenki lemásolja, és senki sem fizet érte, **ki fogja finanszírozni a következő program fejlesztését?**

A levél azért is érdekes, mert a korai számítógépes közösségben még korántsem volt magától értetődő, hogy a szoftvert ugyanolyan kereskedelmi terméknek kell tekinteni, mint a hardvert. A programokat korábban gyakran megosztották egymással, és a hobbisták közösségének kultúrájához természetesen hozzátartozott a programcsere.

Gates viszont már felismerte, hogy **önálló szoftveripar csak akkor jöhet létre, ha magáért a programért is hajlandók fizetni.**

A Microsoft-paradoxon

A történetnek azonban van egy különös folytatása.

A Microsoft később a világ egyik legnagyobb szoftvervállalatává vált, miközben DOS-, Windows- és Office-programjainak illegális másolatai világszerte óriási számban terjedtek.

A Microsoft természetesen védte szerzői jogait, használt termékkulcsokat, licencfeltételeket és később internetes aktiválást is. A korábbi rendszerek azonban a mai online ellenőrzéshez képest viszonylag könnyen másolhatók voltak.

Ennek különös mellékhatása lehetett.

Ha egy fiatal otthon egy lemásolt Wordöt vagy Excelt használt, **Wordöt és Excelt tanult meg használni**. Amikor később munkahelyre került, számára ezek a programok jelentették a megszokott környezetet. A vállalatok pedig annál inkább ezeket választották, minél több munkavállaló ismerte őket.

A folyamat önmagát erősíthette:

sok felhasználó → közös fájlformátumok → megszerzett tudás → vállalati igény → még több felhasználó

A jogosulatlan másolás tehát egyszerre jelenthetett veszteséget a szoftvergyártónak és – paradox módon – hozzájárulhatott programjának elterjedéséhez.

Nem tudjuk azt állítani, hogy a Microsoft tudatosan erre építette volna üzleti stratégiáját. Azt azonban igen, hogy a könnyen terjedő szoftverek világában a másolhatóság **hálózati hatást** is létrehozhatott.

Így Bill Gates 1976-os levelétől különös út vezetett a Windows és az Office világuralmáig:

az ember, aki a személyi számítógépes korszak hajnalán az elsők között emelte fel szavát a szoftverek szabad másolása ellen, később egy olyan szoftverbirodalmat épített fel, amelynek programjai a legális példányok mellett megszámlálhatatlan illegális másolatban is eljutottak a világ szinte minden részére.

Másolásvédelem – és annak feltörése

A gyártók először magukat az adathordozókat próbálták megvédeni.

Szokatlan lemezformátumokat, hibásnak jelölt szektorokat, rejtett adatokat és más technikai trükköket alkalmaztak. A program induláskor ellenőrizte, hogy valóban eredeti lemezről fut-e.

Hamarosan megjelentek azonban a védelem eltávolítására szakosodott programozók is.

Új másolásvédelem → feltörés → újabb védelem → újabb feltörés.

A játék évtizedeken át folytatódott.

Hardverkulcs

A másik megoldás az volt, hogy a program csak akkor működött, ha a számítógéphez egy különleges elektronikus eszközt – **hardverkulcsot, más néven dongle-t** – csatlakoztattak.

Ez főként drága professzionális CAD-, mérnöki és grafikai programoknál terjedt el.

A védelem erősebb lett, de megjelentek a hardverkulcs működését utánzó megoldások és a program ellenőrzését kiiktató módosítások is.

„Írd be a sorozatszámot!"

Az internet terjedésével a védekezés következő eszköze a **sorozatszám és a termékkulcs** lett.

A program telepítéséhez egy hosszú betű- és számsort kellett megadni. Csakhogy hamarosan internetes adatbázisokban kezdtek terjedni az érvényes vagy annak látszó kulcsok, illetve olyan programok, amelyek megfelelő formátumú kódokat generáltak.

A gyártók erre internetes aktiválással válaszoltak. A program már nemcsak azt ellenőrizte, hogy megfelelő kulcsot kapott-e, hanem kapcsolatba lépett a gyártó szerverével is.

A védelem így lassan **a lemezről az internetre költözött.**

Megjelenik a „szoftverrendőrség"

A jogosulatlan üzleti szoftverhasználat visszaszorításának egyik legismertebb szereplője az 1988-ban alapított **Business Software Alliance (BSA)** lett.

A BSA nem rendőrség és nem hatóság volt, hanem szoftvercégek érdekvédelmi szervezete. Kampányokat folytatott, bejelentéseket fogadott, jogi eljárásokat kezdeményezett vagy támogatott, és együttműködött a hatóságokkal.

A közvéleményben mégis kialakult róla a **„szoftverrendőrség"** képzete. Különösen a vállalkozások tartottak attól, hogy egy ellenőrzés során kiderül: a számítógépeiken több példányban használnak egy programot, mint ahány licenccel rendelkeznek.

A kilencvenes években és a 2000-es évek elején a szoftverlegalitás komoly vállalati kérdéssé vált.

Mennyit veszített valójában a szoftveripar?

Itt kezdődik a történet egyik legérdekesebb vitája.

A szoftveripar hosszú időn keresztül hatalmas összegekre becsülte a jogosulatlanul használt programok értékét. Az ilyen számokat azonban óvatosan kell értelmezni.

Egy lemásolt program ugyanis nem feltétlenül jelent egy elveszített eladást.

Ha egy diáknak vagy fiatal számítógép-rajongónak nem volt pénze egy több száz vagy több ezer dolláros programra, abból, hogy készített róla egy illegális másolatot, nem következik, hogy annak hiányában megvásárolta volna.

A tényleges gazdasági kár meghatározásához tehát azt kellene tudnunk, hogy a másolatok mekkora része helyettesített valódi vásárlást. Ezt sokkal nehezebb megállapítani, mint egyszerűen megszorozni a másolatok becsült számát a listaárral.

Másfelől a jogosulatlan másolás természetesen valódi bevételkiesést is okozhatott, különösen olyan vállalatoknál, amelyek képesek lettek volna megvásárolni a szükséges licenceket.

A kérdés tehát nem fekete-fehér.

Túl drága volt a szoftver?

A tömeges másolás egyik oka kétségtelenül az ár volt. Egy professzionális program ára akár egy teljes számítógép árával is összemérhető lehetett, Kelet-Európában pedig a nyugati ár és a helyi fizetések közötti különbség még nagyobb problémát jelentett.

Közben kialakult egy sajátos kulturális szemlélet is:

„Ha egy floppy lemásolásával az eredeti példány megmarad, akkor mit vettem el?”

A digitális termék éppen ezért kényszerítette ki a tulajdon fogalmának újragondolását. A felhasználó ugyanis általában nem magát a programot vásárolja meg, hanem **licencet, vagyis meghatározott feltételek szerinti használati jogot.**

A jogosulatlan másolás bizonyos esetekben **akaratlanul is segíthette egy-egy program vagy platform elterjedését.**

A rabló–pandúr játék vége?

Nem ért véget. Csak átalakult.

A fizikai adathordozó másolásvédelmét felváltotta a termékkulcs, azt az internetes aktiválás, majd az online felhasználói fiók és az előfizetés.

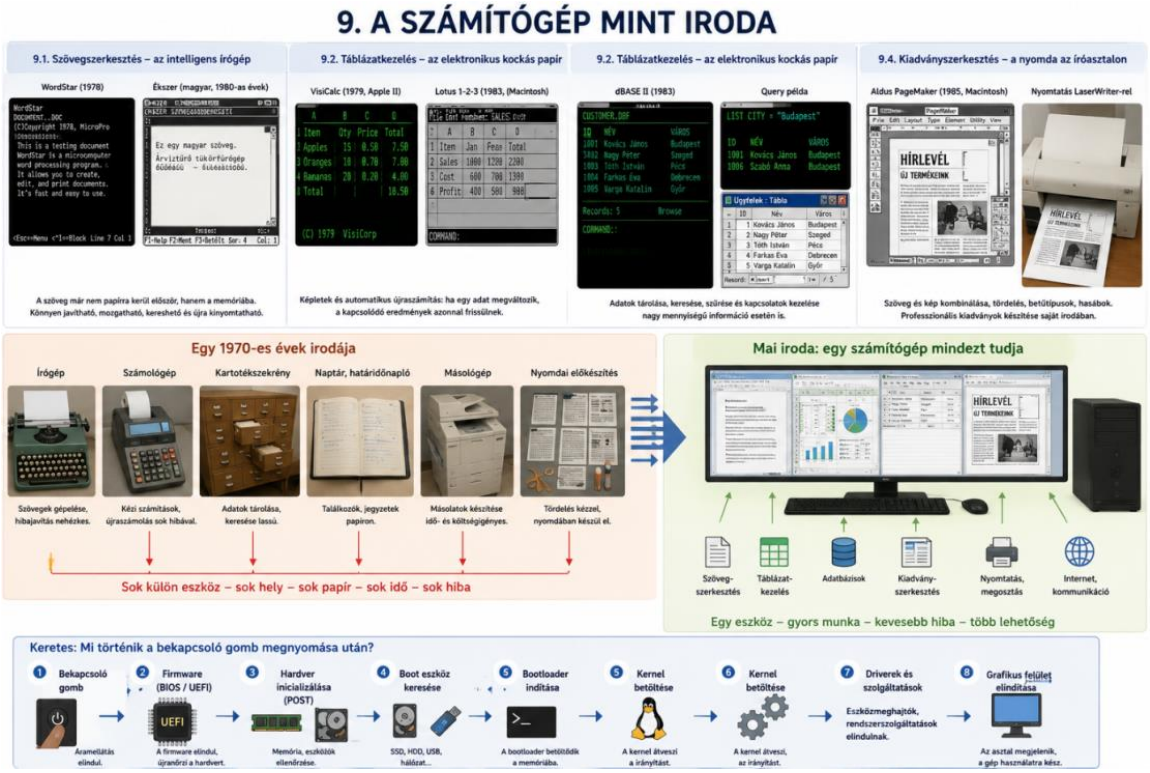
A felhőalapú szolgáltatást pedig már sokkal nehezebb egyszerűen „lemásolni”, hiszen a program lényeges része nem is a felhasználó számítógépén fut.

A szoftveripar részben gazdasági megoldást is talált a problémára. Megjelentek az olcsóbb otthoni és oktatási licencek, az ingyenes alapszolgáltatások, a reklámból finanszírozott programok, az előfizetéses konstrukciók és a nyílt forráskódú alternatívák.

A történet végül egy alapvető kérdéshez vezet:

Hogyan lehet árat szabni valaminek, amelynek az első példánya rendkívül drága lehet, a következő példány elkészítése viszont gyakorlatilag semmibe sem kerül?

A szoftveripar több mint fél évszázada keresi erre a választ.



10. Kép, hang és mozgókép

A számítógép eredetileg számokat dolgozott fel. Később betűket és szövegeket is megtanult kezelni.

De hogyan kerülhet egy fénykép, egy zenemű vagy egy film a számítógépbe?

A válasz meglepően egyszerű:**számokká kell alakítani.**



Egy digitális fénykép képpontok millióinak színét leíró számokból áll. A digitális hang egy hanghullám egymást követő mérési eredményeiből épül fel. A digitális film pedig képek és hangadatok hosszú sorozata.

A számítógép tehát valójában továbbra sem „lát” és nem „hall”.

Csak számokat kezel.

Mi látjuk ezeket a számokat fényképnek, halljuk zenének és érzékeljük mozgóképként.

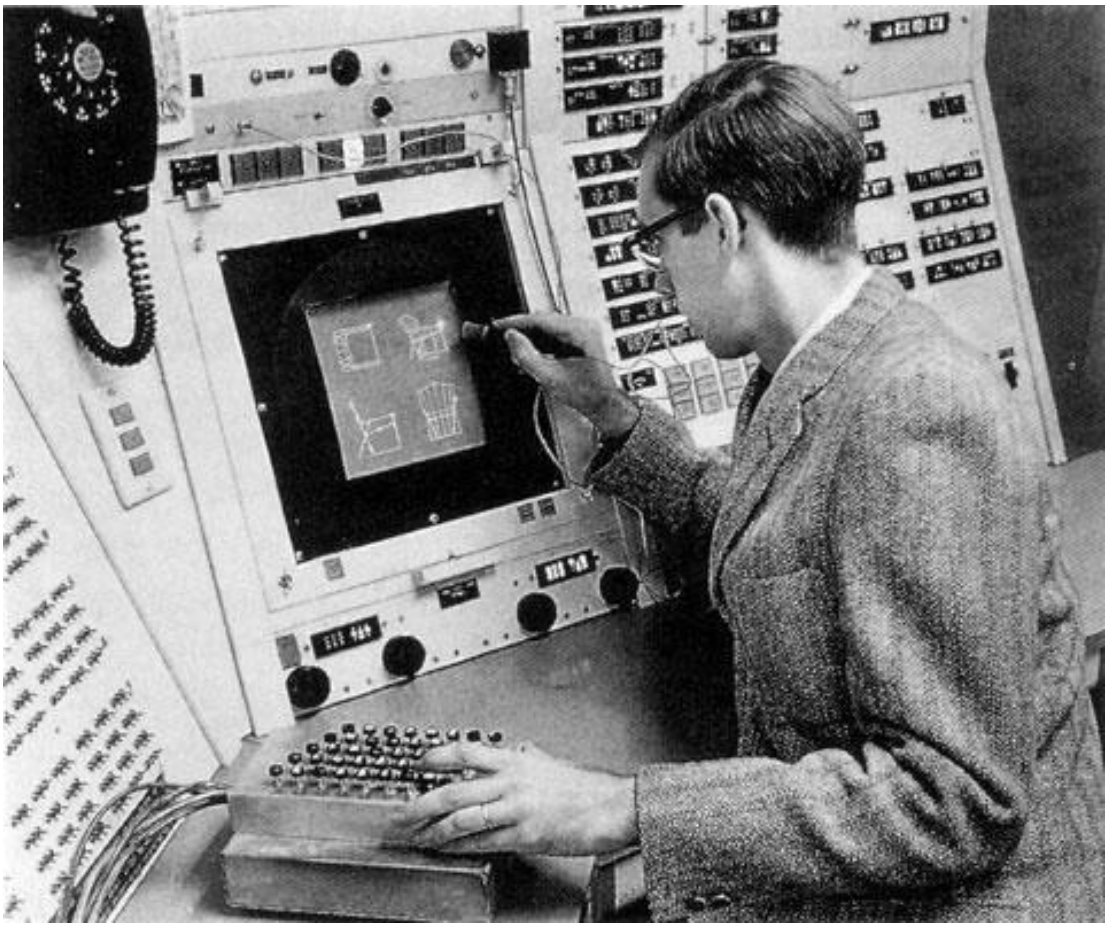
10.1. Számítógépes grafika

A legkorábbi számítógépeknek nem volt szükségük grafikus kijelzőre. Az eredményeket lámpák, lyukkártyák vagy nyomtatók közölték.

Ahogy azonban a számítógép és az ember kapcsolata közvetlenebbé vált, megjelentek a képernyők.

Kezdetben főként karaktereket jelenítettek meg. A következő lépés az volt, amikor a számítógép már a képernyő egyes pontjait is vezérelhette.

Megszületett a **számítógépes grafika**.



Ábra: Ivan Sutherland és a Sketchpad (1963). A fényceruzával vezérelhető rendszer a számítógépes grafika és a grafikus ember–gép kapcsolat egyik legfontosabb úttörője volt.

A grafika kezdetben főként tudományos, mérnöki és katonai alkalmazásokban jelent meg. Később a számítógépes tervezés, a játékok és végül a grafikus felhasználói felületek tették általánossá.

Ma pedig már háromdimenziós virtuális világokat és a valóságtól alig megkülönböztethető képeket is képesek vagyunk számítógéppel előállítani.

10.2. Vektor és bitmap

A számítógép alapvetően kétféleképpen írhat le egy képet.

Vektorgrafika

A vektorgrafika nem a kép minden egyes pontját tárolja, hanem azt mondja meg, **hogyan kell a képet megrajzolni**.

Például:

„húzz egy vonalat innen idáig”,

„rajzolj egy 30 egység sugarú kört”,

„töltsd ki ezt a területet”.

A kép geometriai objektumokból áll.

Nagy előnye, hogy szinte tetszőleges méretűre nagyítható anélkül, hogy „kockássá” válna.

Bitmap – rasztergrafika

A bitmap ezzel szemben apró képpontokra, **pixelekre** bontja a képet.



Ábra: Vektor és bitmap. A vektorgrafika matematikailag leírt vonalakból és alakzatokból, a bitmap apró képpontokból, pixelekből építi fel a képet.

A fényképek szinte mindig raszterképek, míg logókhoz, műszaki rajzokhoz és betűkhöz gyakran előnyösebb a vektorgrafika.

10.3. Digitális fénykép

A hagyományos fényképezőgépben a fény fényérzékeny filmre érkezik.

A digitális fényképezőgépben a film helyén elektronikus képérzékelő található.

A **CCD vagy CMOS érzékelő** apró fényérzékeny elemek sokaságából áll. Ezek a rájuk jutó fényt elektromos jellé alakítják.

A jeleket a fényképezőgép elektronikája számokká alakítja.

Megszületik a digitális kép.



Ábra: Steve Sasson és a Kodak digitális fényképezőgép-prototípusa, 1975. A mintegy 0,01 megapixeles fekete-fehér képet készített, amelyeket digitálisan rögzített. A digitális fényképezés néhány évtized alatt szinte teljesen átalakította a fotográfiát.

Nincs szükség filmre.

Nincs előhívás.

Nincs szükség minden egyes példány elkészítéséhez újabb fényképezési vagy laboratóriumi műveletre.

És a másolat – megfelelő digitális másolás esetén – pontosan ugyanazokat az adatokat tartalmazhatja, mint az eredeti.

10.4. Digitális hang

A hang fizikailag a levegő nyomásának folyamatos változása.

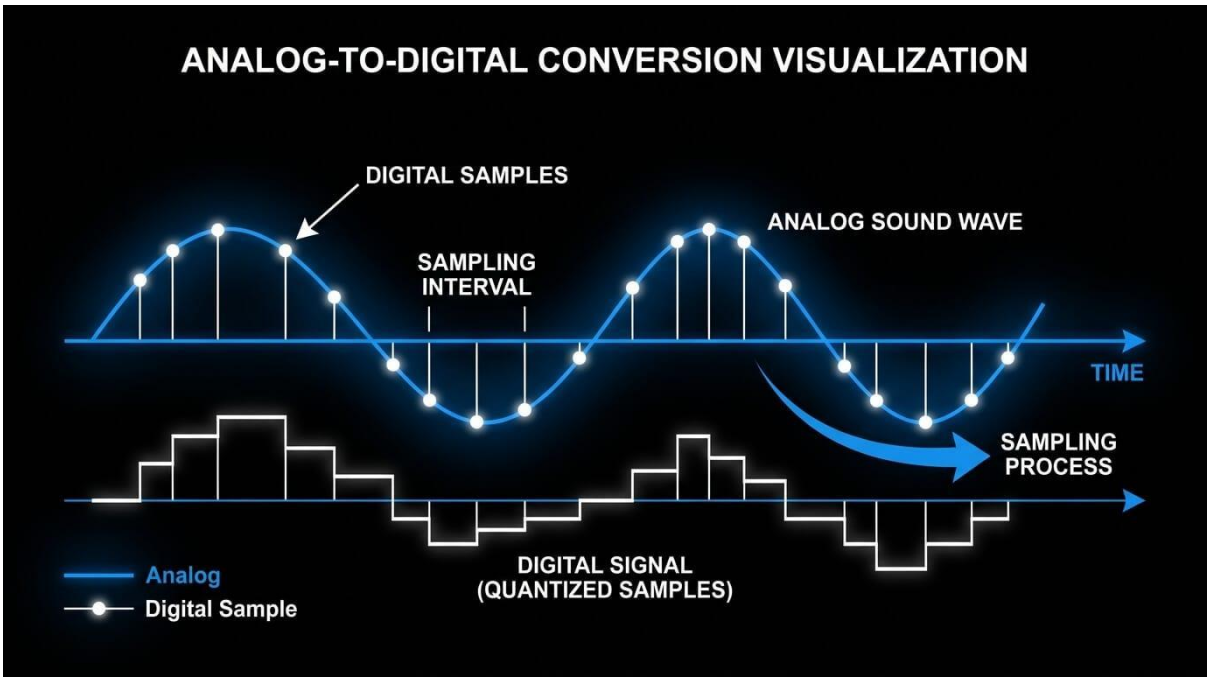
Ez **analóg jel**.

Ahhoz, hogy a számítógép kezelhesse, meg kell mérni a hanghullám pillanatnyi értékét.

Nem egyszer.

Hanem másodpercenként sok ezerszer.

Ezt nevezzük **mintavételezésnek**.



Ábra:
Az analóg hang digitalizálása. A folyamatos hanghullám pillanatnyi értékét meghatározott időközönként megmérjük. A mért értékeket számokként tároljuk – ezekből a számítógép később újra előállíthatja a hangot.

A CD-lemezek hangját például másodpercenként **44 100 alkalommal** mintavételezik, csatornánként 16 bites felbontással.

A digitalizálás után a hang ugyanúgy feldolgozható adattá válik, mint a szöveg vagy a kép.

Másolható.

Vágható.

Szűrhető.

Effektezhető.

Továbbítható.

10.5. MIDI

A MIDI egészen másképpen kezeli a zenét.

Nem magát a hangot tárolja.

Hanem azt írja le, **mit kell a hangszernek játszania**.

A MIDI – *Musical Instrument Digital Interface* – szabványt az 1980-as évek elején dolgozták ki elektronikus hangszerek összekapcsolására.



Ábra:
MIDI-rendszer az elektronikus zene világában. A MIDI nem a hangfelvételt továbbítja, hanem többek között azt közli a hangszerrel, melyik hangot, mikor, milyen erősen és meddig kell megszólaltatnia.

Ebben az értelemben a MIDI-fájl inkább hasonlít egy **digitális kottára**, mint hangfelvételre.

Ezért rendkívül kevés adatból hosszú zeneművek írhatók le.

10.6. MP3

A jó minőségű digitális hangnak van egy komoly hátránya: **sok adatot igényel.**

A CD-minőségű sztereó hang tömörítés nélkül percenként körülbelül 10 megabájtot foglal.

Az internet korai korszakának lassú hálózatain ez hatalmas mennyiség volt.

A megoldás a hangtömörítés lett.

Az egyik legsikeresebb eljárás az **MP3**, pontosabban MPEG-1 Audio Layer III.

Fejlesztésében különösen fontos szerepet játszott a német Fraunhofer Intézet kutatócsoportja Karlheinz Brandenburg közreműködésével.



Ábra: Karlheinz Brandenburg, az MP3 fejlesztésének egyik meghatározó kutatója. A hatékony hangtömörítés lehetővé tette, hogy jó minőségű zenék viszonylag kis fájlokban legyenek tárolhatók és továbbíthatók.

Az MP3 forradalma. A pszichoakusztikai tömörítés kihasználja az emberi hallás sajátosságait: olyan hanginformációkat is elhagyhat, amelyeket az ember alig vagy egyáltalán nem érzékel. Így a hangfájl mérete töredékére csökkenthető.

Az MP3 nem egyszerűen technikai újítás volt.

Megváltoztatta a zene terjesztését is.

A zeneszám többé nem feltétlenül egy fizikai tárgyhoz – hanglemezhez, kazettához vagy CD-hez – kötődött.

A zene fájllá vált.

10.7. Digitális videó

A film tulajdonképpen gyors egymásutánban megjelenített állóképek sorozata.

Ha egyetlen digitális fénykép sok adatot tartalmaz, könnyen belátható, hogy másodpercenként 25, 30 vagy akár 60 kép tárolása óriási adatmennyiséget eredményez.

Ehhez járul még a hang.

A digitális videó ezért csak akkor válhatott igazán általánossá, amikor elegendően gyors processzorok, nagy háttértárak és hatékony tömörítési eljárások jelentek meg.



Ábra:
Digitális videószerkesztés. A számítógépes, nemlineáris vágás alapvetően megváltoztatta a filmkészítést. A jeleneteket többé nem kellett fizikailag a kívánt sorrendbe vágni: a számítógép tetszőleges sorrendben elérhette és összeállíthatta őket.

A digitális videó később eljutott a személyi számítógépre, majd a telefonra is.

Ma egy zsebben hordott készülékkel olyan felbontású videót készíthetünk és szerkeszthetünk, amely néhány évtizede professzionális stúdiótechnikát igényelt.

10.8. Tömörítés

A digitális világ egyik alapvető problémája mindig ugyanaz:

hogyan tároljunk és továbbítsunk minél több információt minél kevesebb adattal?

Erre szolgál a tömörítés.

Két alapvető módszert különböztetünk meg.

Veszteségmentes tömörítés

Ilyenkor az eredeti adat pontosan visszaállítható.

Ilyen elvet használ például a ZIP, a PNG és számos más formátum.

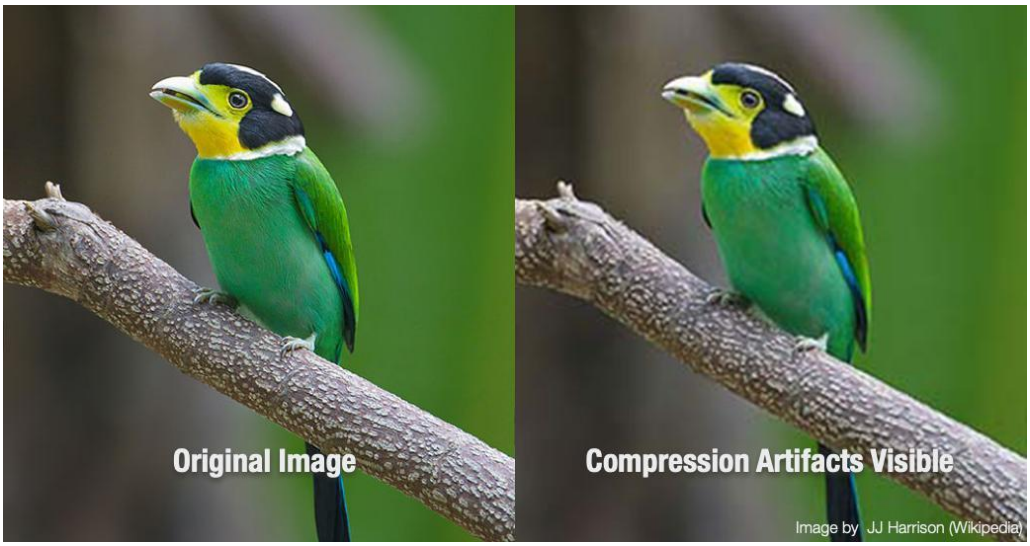
Szövegnél és programoknál ez elengedhetetlen. Egyetlen elveszett karakter vagy bit is megváltoztathatná a tartalmat.

Veszteséges tömörítés

Képnél, hangnál és videónál azonban gyakran elhagyhatunk olyan információkat, amelyeket az ember kevésbé érzékel.

Erre épül például:

JPEG – kép
MP3/AAC – hang
MPEG/H.264/H.265 és más kodekek – videó



Ábra:
A veszteséges tömörítés határa. Mérsékelt tömörítésnél a különbség alig észrevehető, erős tömörítésnél azonban megjelennek a jellegzetes képhibák, például a blokkosodás. A feladat mindig ugyanaz: a lehető legkevesebb adattal elfogadható minőséget megőrizni.

A modern videótömörítés még egy fontos tulajdonságot használ ki.

Egy film egymást követő képkockái gyakran nagyon hasonlóak.

Ha egy ember áll egy mozdulatlan fal előtt, felesleges másodpercenként harmincszor újra eltárolni ugyanazt a falat.

Elég azt is megadni: **mi változott az előző képhez képest.**

Ez a gondolat teszi lehetővé, hogy hatalmas felbontású filmeket tároljunk és az interneten továbbítsunk.

A világ számmá válik

A kép, a hang és a film digitalizálása ugyanannak az alapgondolatnak különböző változata.

Kép: a képet pontokra bontjuk, és számokkal leírjuk a pontok színét.

Hang: időnként megmérjük a hanghullám értékét, és számként eltároljuk.

Videó: digitális képek sorozatát és digitális hangot tárolunk.

Ha pedig már minden számokból áll, ugyanazzal az eszközzel kezelhető.

A számítógéppel.

Ez vezetett el egy különös technológiai összeolvadáshoz.

A fényképezőgép, magnetofon, lemezjátszó, videomagnó, filmvágó, rajzasztal és hanglemezgyűjtemény funkciói fokozatosan **egyetlen általános célú gépbe költöztek.**

Később pedig ez a gép is beköltözött a zsebünkbe.

11. Tervezés és tudomány

A számítógép történetének kezdetén maga a számolás volt a legfontosabb feladat. Az első elektronikus gépeket bonyolult matematikai problémák megoldására építették.

Később azonban kiderült, hogy a számítógép ennél sokkal többre képes.

Segítségével megtervezhetünk egy még nem létező gépet, kiszámíthatjuk egy repülőgép körül áramló levegő viselkedését, modellezhetjük egy híd terhelését, előre jelezhetjük az időjárást, vagy háromdimenziós képet készíthetünk az emberi test belsejéről.

A számítógép ezzel a mérnök és a kutató egyik legfontosabb munkaeszközévé vált.

11.1. CAD – amikor a rajztábla beköltözött a számítógépbe

Egy gépet, épületet vagy elektronikus berendezést először meg kell tervezni.

Évszázadokon keresztül ehhez papír, ceruza, vonalzó, körző és rajztábla kellett.

A műszaki rajzoló szakma külön mesterség volt.

A számítógép ezt a világot is alapvetően megváltoztatta.

A **CAD – Computer-Aided Design**, vagyis számítógéppel segített tervezés segítségével a rajz már nem papíron, hanem digitális modellként készül.

Graphisoft és az Archicad – magyar szoftver a világ építészeinek

A magyar számítástechnika egyik legnagyobb nemzetközi sikertörténete egy építészeti tervezőprogramhoz kapcsolódik.

A **Graphisoftot 1982-ben alapította Bojár Gábor és Tari István**. A vállalkozás Magyarországon, a személyi számítógépes forradalom kezdetén indult, amikor a korszerű nyugati számítástechnikai eszközökhöz való hozzáférés még korántsem volt egyszerű.

A cég kezdetben építészeti és mérnöki háromdimenziós modellezéssel foglalkozott. Fejlesztői hamar felismerték az akkor megjelenő személyi számítógépekben rejlő lehetőséget.

1984-ben jelent meg Apple Macintoshra az Archicad elődje, amelyből fokozatosan a Graphisoft legismertebb terméke fejlődött ki.

Az Archicad jelentősége azonban túlmutatott azon, hogy számítógéppel lehetett alaprajzokat készíteni.

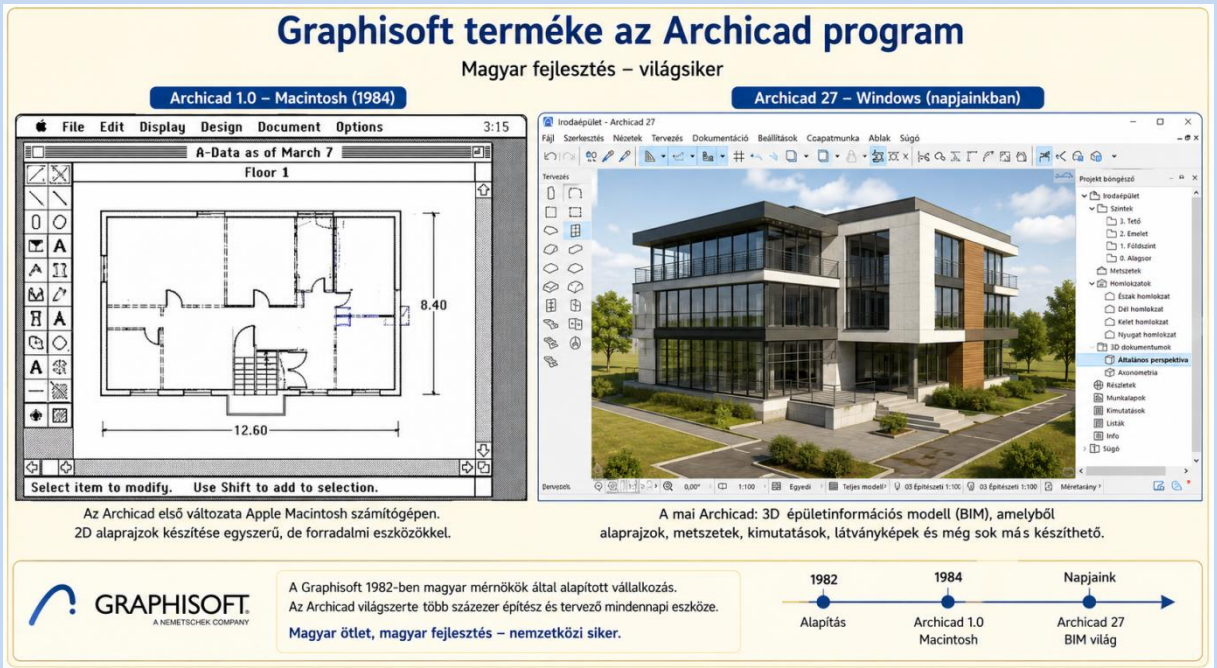
A hagyományos CAD-programban a tervező elsősorban **vonalakat rajzol**.

Az Archicad szemléletében viszont **épületet épít a számítógépben**.

Nem egyszerű vonalak szerepelnek a modellben, hanem falak, födémek, ajtók, ablakok, lépcsők és más építészeti elemek. Ha a tervező megváltoztatja az épület valamely részét, a modellből származó alaprajzok, metszetek és más dokumentumok is követhetik a változást.

Ez a gondolat vezetett a később széles körben elterjedt **BIM – Building Information Modeling**, vagyis épületinformációs modellezés felé.

A számítógép így már nem egyszerűen elektronikus rajztábla.



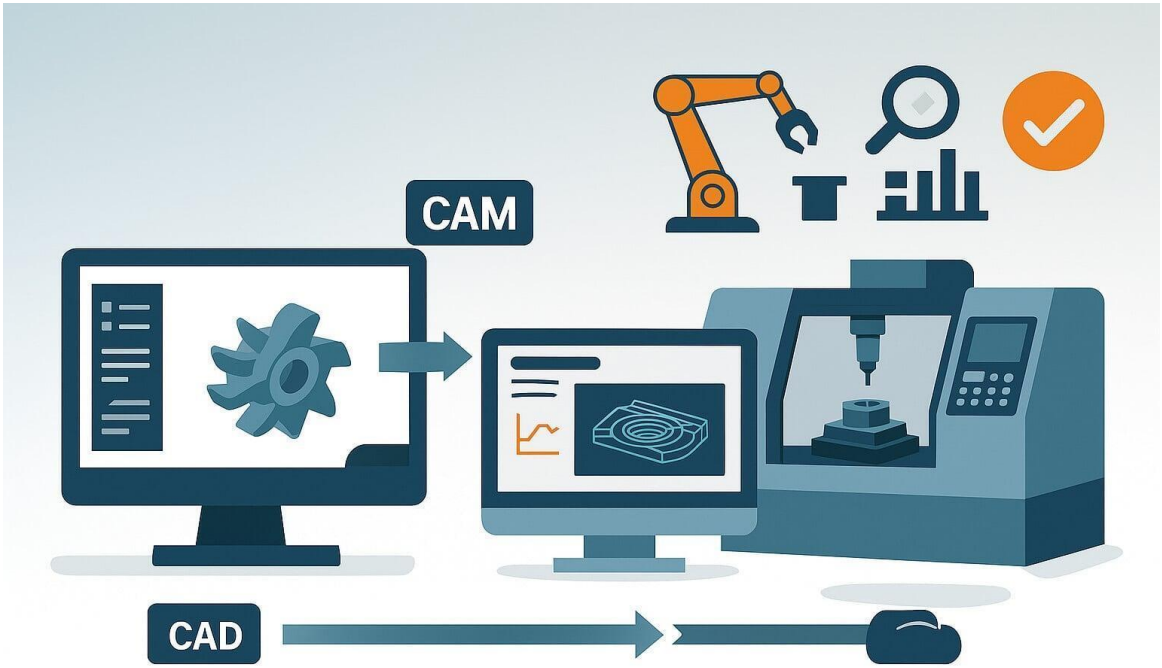
Az épület virtuális példánya jön létre benne még azelőtt, hogy az első téglát leraknák.

A Graphisoft különlegessége az is, hogy egy kis közép-európai országból indulva tudott nemzetközi szoftverpiaci szereplővé válni. Az Archicad világszerte ismert építészeti tervezőrendszer lett, a Graphisoft pedig a magyar szoftveripar egyik legismertebb vállalkozása.

Története jól példázza, hogy a számítástechnikában nem feltétlenül a nagy gyár, a hatalmas nyersanyagkészlet vagy a tömegtermelés jelenti a legfontosabb erőforrást.

Néha elegendő hozzá néhány számítógép, egy jó ötlet és megfelelő tudás.

A Graphisoft nem számítógépet gyártott Magyarországon – hanem olyan magyar szellemi terméket hozott létre, amelyet számítógépeken az egész világon használhattak.



Ábra: CAD–CAM – a tervezéstől a gyártásig. A számítógépen létrehozott háromdimenziós modellből a CAM-rendszer elkészítheti a CNC szerszámgép számára szükséges megmunkálási utasításokat.

A CAD egyik óriási előnye a módosíthatóság.

Ha egy alkatrész furatát két milliméterrel arrébb kell helyezni, nem szükséges újrarajzolni az egész tervet.

A mai CAD-rendszerek pedig már nem egyszerűen rajzokat készítenek.

Háromdimenziós virtuális tárgyakat építenek.

A mérnök forgathatja a modellt, belenézhet, szétszedheti, méreteket változtathat rajta, és még azelőtt ellenőrizheti működését, hogy az első valódi példány elkészülne.

11.2. CAM – a tervből tárgy lesz

Ha már elkészült a digitális terv, adódik a következő kérdés:

miért kellene azt ismét embernek lefordítania a gyártógép számára?

A **CAM – Computer-Aided Manufacturing**, vagyis számítógéppel segített gyártás összeköti a digitális tervezést a gyártással.

A CAD-rendszerben elkészített modell alapján program készülhet például egy CNC szerszámgép számára.

CAD-ből CAM. A számítógépen megtervezett háromdimenziós alkatrész adatai alapján a CAM-rendszer meghatározhatja a CNC szerszámgép mozgását. A virtuális modellből így közvetlenül valódi tárgy készülhet.

Ez alapvető változást hozott a gyártásban.

Tervezés → számítógépes modell → gyártási program → gép → kész alkatrész

A digitális információ közvetlenül fizikai tárggyá válik.

A 3D nyomtatás ezt a gondolatot még látványosabbá tette: a számítógépen megrajzolt modellből akár néhány órán belül kézbe vehető tárgy készülhet.

11.3. Szimuláció – kísérlet a virtuális világban

Vannak kísérletek, amelyeket nehéz, drága vagy veszélyes lenne a valóságban elvégezni.

Mi történik egy autóval ütközéskor?

Hogyan viselkedik egy repülőgép viharban?

Kibírja-e egy híd az erős szelet?

Hogyan terjed egy járvány?

Mi történik egy atomerőműben súlyos üzemzavar esetén?

A számítógépes **szimuláció** segítségével létrehozhatjuk a valóság matematikai modelljét, majd ezen végezhetünk kísérleteket.

A valóság kipróbálása a számítógépben. Az autó töréstesztje, a repülőgép körüli légáramlás vagy egy építmény terhelése már a tényleges prototípus elkészítése előtt vizsgálható számítógépes modelleken.

A szimuláció természetesen csak annyira jó, amennyire jó a modell.

A számítógép nem maga a valóság.

De ha a fizikai törvényeket és a kiindulási feltételeket megfelelően írjuk le, olyan folyamatokat is tanulmányozhatunk, amelyekhez a való világban nem férnénk hozzá.

11.4. Tudományos számítás

Tulajdonképpen itt térünk vissza a számítógép eredeti feladatához.

A számoláshoz.

Csakhogy egészen más nagyságrendben.

A tudomány számos területén olyan egyenleteket kell megoldani, amelyek kézzel gyakorlatilag kezelhetetlenek.

A fizika, kémia, csillagászat, geológia, biológia és mérnöki tudományok egyre nagyobb számítási teljesítményt igényeltek.

Megszülettek a **szuperszámítógépek.**

A tudományos számítás mára újfajta kutatási módszerré vált.

A hagyományos két út:

elmélet + kísérlet

mellé belépett egy harmadik:

számítógépes modell és szimuláció.

Bizonyos területeken pedig megjelent egy negyedik is:

hatalmas adatmennyiségek számítógépes elemzése.

11.5. Időjárás-előrejelzés

Kevés hétköznapi példa mutatja meg jobban a számítógépek tudományos jelentőségét, mint az időjárás-előrejelzés.

A légkör működését fizikai törvények határozzák meg.

Ha ismernénk a légkör minden pontjában a hőmérsékletet, nyomást, páratartalmat és a levegő mozgását, elvileg kiszámíthatnánk, hogyan változik az állapota.

A gyakorlatban természetesen nem ismerhetünk mindent.

Ezért a Föld légkörét képzeletbeli háromdimenziós rácsra bontjuk. A számítógép minden rácspont környezetére kiszámítja a légkör várható változását.

Majd újra.

És újra.

Órákkal, napokkal előre.



Ábra: Szuperszámítógépek az időjárás és az éghajlat modellezésében. A légkör matematikai modelljét hatalmas számítási teljesítményű rendszereken futtatják; a modell a megfigyelt kezdeti állapotból számítja ki a lehetséges jövőbeli állapotokat.

Numerikus időjárás-előrejelzés. A légkört háromdimenziós számítási rácsra bontják, majd szuperszámítógépek számítják ki, hogyan változhat a hőmérséklet, légnyomás, nedvesség és légmozgás az egyes területeken.

Az időjárás-előrejelzés azért különösen nehéz, mert a légkör **kaotikus rendszer.**

Nagyon kis eltérések a kiindulási adatokban idővel nagy eltéréseket okozhatnak.

Innen ered a híres „pillangóhatás” gondolata is.

Ezért nem lehet tetszőlegesen messzire pontos időjárás-előrejelzést készíteni – még akkor sem, ha a számítógépek teljesítménye tovább növekszik.

11.6. Orvosi informatika

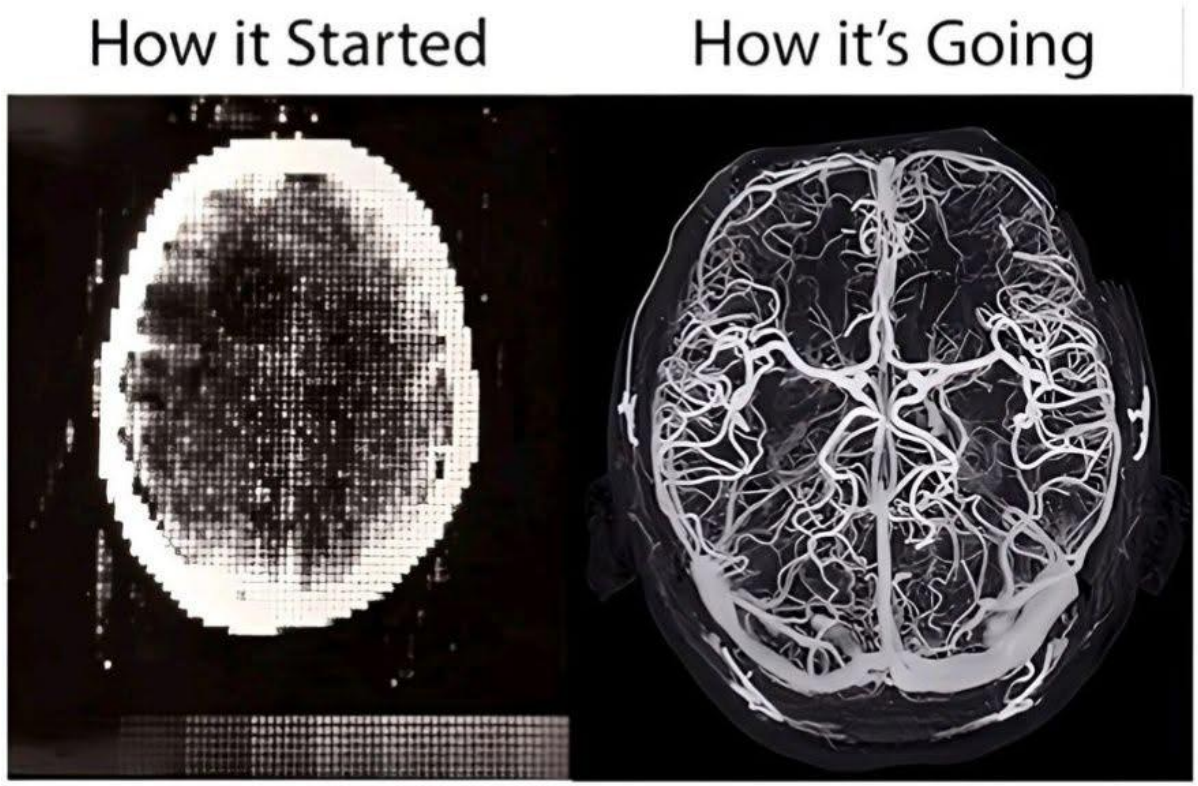
Kevés területen vált annyira látványossá a számítógép szerepe, mint az orvoslásban.

A számítógép kezdetben főként betegadatok tárolására és laboratóriumi eredmények feldolgozására szolgált.

Később azonban már közvetlenül részt vett a diagnosztikában.

Az egyik legfontosabb példa a **CT – számítógépes tomográfia**.

A röntgenberendezés sok különböző irányból végez méréseket. Ezekből a számítógép matematikai módszerekkel állítja elő a test keresztmetszeti képét.



Az első klinikai CT-felvételt **1971. október 1-jén** készítették el a londoni Atkinson Morley Kórházban. A vizsgálatot Godfrey Hounsfield brit mérnök prototípus készülékével végezték, és egy agydaganatos (cisztás elváltozásban szenvedő) beteg koponyáját vizsgálták meg vele

A számítógépes tomográfia kezdete. Godfrey Hounsfield és az EMI munkája nyomán a számítógép röntgenmérések sokaságából olyan keresztmetszeti képet tudott rekonstruálni, amelyet hagyományos röntgenfelvétellel nem lehetett elkészíteni.

A CT különösen érdekes informatikatörténeti szempontból.

A gép nem egyszerűen lefényképezi a test belsejét.

A kép számítás eredménye.

Ugyanez a gondolat más képalkotó eljárásokban is megjelenik.



Kép: **EMI Scanner Mark I – a CT úttörője.**

Godfrey Hounsfield rendszerében a számítógép sok különböző irányból végzett röntgenmérésből rekonstruálta a test keresztmetszeti képét. A kép tehát nem hagyományos fénykép: **matematikai számítás eredménye**.

Az orvosi képalkotás digitális világa. A CT, MRI és más berendezések hatalmas mennyiségű mérési adatából számítógépek állítanak elő képeket, metszeteket és háromdimenziós modelleket.

A számítógépek ma már a betegellátás szinte minden területén jelen vannak:

elektronikus betegnyilvántartásokban, laboratóriumi rendszerekben, betegmonitorokban, műtéti navigációban, robotsebészetben, gyógyszerkutatásban és egyre inkább a mesterséges intelligenciával támogatott diagnosztikában is.

Mesterséges intelligencia az orvoslásban – a doktor új segítőtársa

Ha valaki ma belép egy korszerű kórházba, jó eséllyel már olyan mesterséges intelligenciával találkozik, amelyet észre sem vesz. Az AI egyre több feladatban segíti az orvosokat – de nem helyettesíti őket.

A leglátványosabb eredményeket az orvosi képek elemzésében érte el. A neurális hálózatok képesek röntgen-, CT- és MRI-felvételeken olyan apró elváltozásokat is felismerni, amelyeket az emberi szem könnyen figyelmen kívül hagyhat. A bőrgyógyászatban fényképek alapján segíthetnek felismerni a melanómát, a szemészetben pedig a cukorbetegség okozta retinaelváltozásokat.

Az AI fontos szerepet játszik a daganatos betegségek kezelésében is. Segít meghatározni a daganat pontos kiterjedését, támogatja a sugárkezelés megtervezését, és egyre gyakrabban elemzi a daganatok genetikai jellemzőit is, hogy személyre szabott terápiát javasoljon.

A gyógyszerkutatásban a mesterséges intelligencia óriási mennyiségű molekula közül képes kiválasztani azokat, amelyekből új gyógyszerek születhetnek. Olyan feladatokat is elvégez néhány nap vagy hét alatt, amelyek korábban évekig tartottak. Az egyik legismertebb áttörés az AlphaFold volt, amely fehérjék millióinak térszerkezetét tudta nagy pontossággal előre jelezni, ezzel felgyorsítva a biológiai kutatásokat.

A kórházakban az AI nemcsak gyógyít, hanem szervez is. Segíthet a műtétek időzítésében, az intenzív osztályok kapacitásának tervezésében, a laboreredmények kiértékelésében vagy a betegutak optimalizálásában. A beszédfelismerő rendszerek már ma is képesek automatikusan elkészíteni az orvosi dokumentáció jelentős részét.

A robotsebészet szintén egyre fejlettebbé válik. A műtétet továbbra is a sebész végzi, de a robotrendszer nagyobb pontosságot, stabilitást és finomabb mozgást tesz lehetővé. A mesterséges intelligencia fokozatosan ezekbe a rendszerekbe is beépül, segítve a tervezést és a műtéti döntéseket.

Mindezek ellenére az AI nem helyettesíti az orvost. Nem érzi a beteg félelmét, nem tud vigaszt nyújtani, és nem vállalja a végső felelősséget a döntésekért. Az orvos feladata továbbra is az, hogy a mesterséges intelligencia javaslatait saját tudásával, tapasztalatával és a beteg személyes körülményeivel együtt mérlegelje.

Érdekesség: Egy korszerű egyetemi kórházban naponta akár több millió adat keletkezik – laboreredmények, képalkotó vizsgálatok, genetikai adatok, monitorok mérései és orvosi feljegyzések. Ekkora információtömeg áttekintésében az AI már ma is nélkülözhetetlen segítséget jelent.



A jövő valószínűleg nem az orvosok vagy a mesterséges intelligencia versenyéről szól, hanem arról, hogy az ember és a gép együtt pontosabb diagnózist, hatékonyabb kezelést és emberibb gyógyítást tud nyújtani.

A számítógép, mint virtuális laboratórium

A fejezetben szereplő alkalmazások első pillantásra nagyon különbözőnek tűnnek.

Egy autó tervezése.

Egy vihar előrejelzése.

Egy CT-felvétel elkészítése.

Egy molekula viselkedésének kiszámítása.

Mégis ugyanaz az alapelv köti össze őket:

a valóság egy részét számokkal írjuk le, majd a számítógép segítségével dolgozunk a modellen.

A CAD esetében egy még nem létező tárgyat modellezünk.

A CAM esetében a modellből valódi tárgyat készítünk.

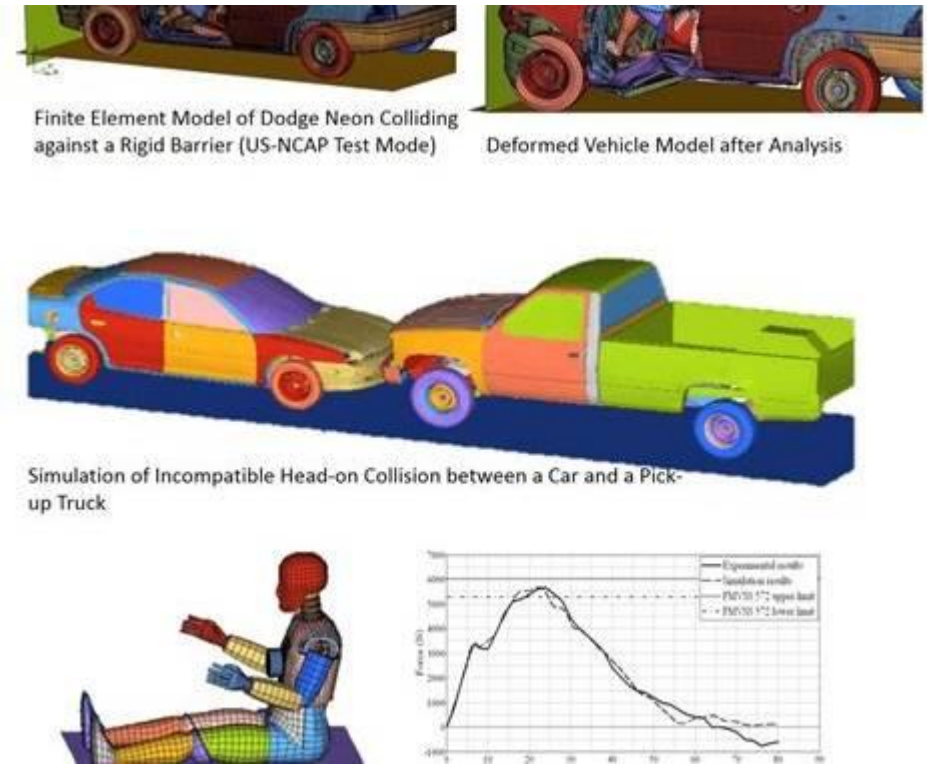
A szimulációban azt vizsgáljuk, hogyan viselkedne egy rendszer.

Az időjárás-előrejelzésben azt próbáljuk kiszámítani, hogyan változik majd.

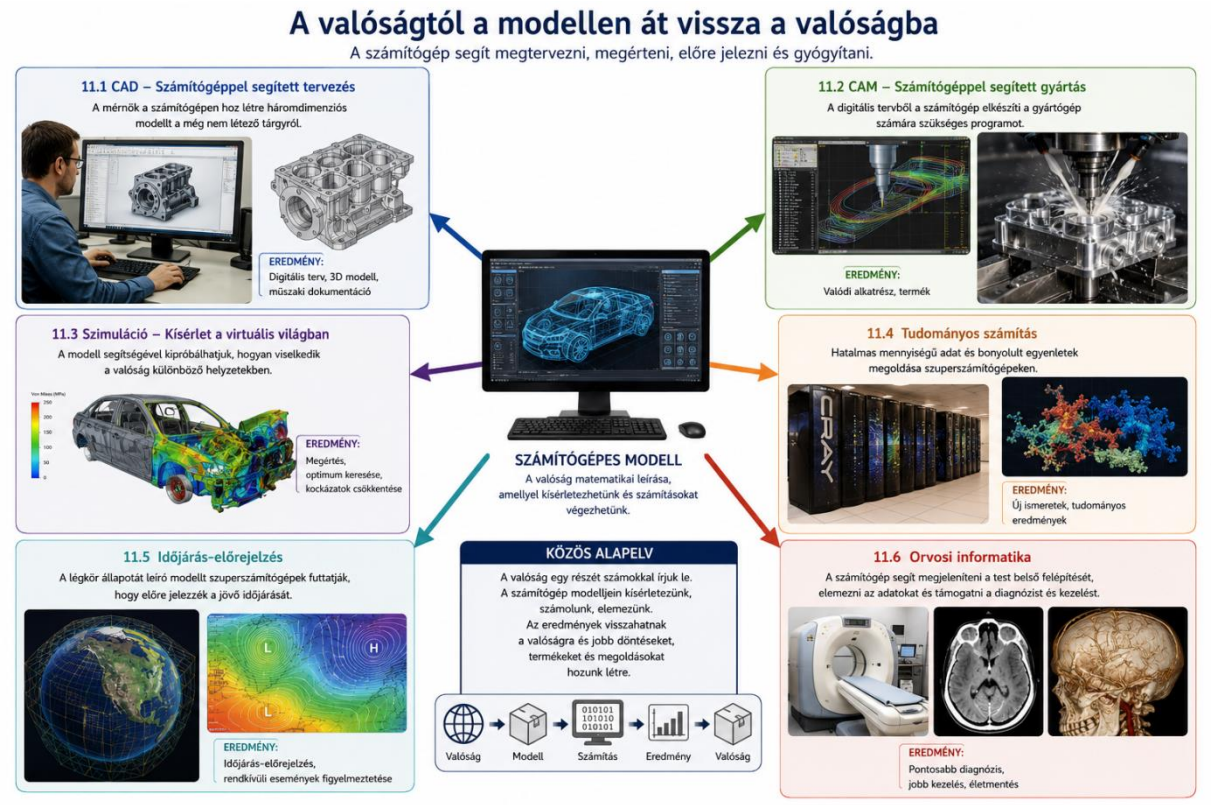
Az orvosi képalkotásban pedig mérésekből rekonstruálunk valamit, amit közvetlenül nem láthatunk.

A számítógép ezzel új képességet adott az embernek:

kipróbálhatjuk a valóságot anélkül, hogy előbb meg kellene építenünk, el kellene rontanunk vagy akár meg kellene várnunk.



Ábra: Számítógépes törésteszt. A végeselemes modell segítségével már a prototípus elkészítése előtt vizsgálható, hogyan deformálódik az autó karosszériája ütközéskor.



Ki írta a programot?

A közelmúltig emberek írták a számítógépek programjait. De most belépett az AI új szereplőként.

A Debian válasza az AI-korszakra

2026 nyarán a Debian közössége egy látszólag technikai, valójában azonban jóval messzebbre mutató kérdésben döntött: **használható-e generatív mesterséges intelligencia egy olyan rendszer fejlesztésében, amelyet emberek milliói használnak?**

A válasz igen lett – de egy fontos feltétellel.

A Debian nem tiltotta meg az AI által generált kód, dokumentáció vagy más tartalom használatát. Ehelyett kimondta, hogy aki ilyen tartalmat küld be a projektbe, annak ugyanúgy felelősséget kell vállalnia érte, mintha saját maga készítette volna. Az AI lehet segédeszköz, de nem válhat a felelősség címzettjévé.

Ez első pillantásra egyszerű szabálynak tűnik, valójában azonban az ember és a mesterséges intelligencia közötti munkamegosztás egyik lehetséges alapelvét fogalmazza meg:

AI javasol → ember megért → ember ellenőriz → ember dönt → ember felel.

Ez nem sokban különbözik attól, ahogyan más eszközöket használunk. Egy mérnök sem hivatkozhat arra, hogy a hibás számítást a számológép végezte, és egy tervező sem mentesül a felelősség alól azért, mert a hibás tervet egy számítógépes program készítette. Az AI azonban különleges helyzetet teremt, mert nem egyszerűen végrehajtja az ember utasításait, hanem maga is képes megoldásokat javasolni, programkódot írni, hibákat keresni vagy akár teljes munkafolyamatokat elvégezni.

Ezért a Debian döntésének legfontosabb szava talán nem is az **AI**, hanem a **felelősség**.

A probléma ráadásul egyre nehezebbé válhat. Ma még elképzelhető, hogy egy programozó elolvassa és ellenőrzi az AI által létrehozott néhány tucat vagy néhány száz programsort. De mi történik akkor, ha egy AI-agent már több másik AI-t irányít, több ezer fájlt módosít, teszteket futtat, hibákat javít, majd újabb AI-rendszerekkel ellenőrizteti saját munkáját?

A felelősségi lánc ekkor már így nézhet ki:

ember → AI-agent → másik AI → automatikus ellenőrzés → döntés → eredmény

Papíron továbbra is lehet egy ember a lánc elején. De vajon mondhatjuk-e még, hogy valóban megértette azt, amiért felelősséget vállalt?

Ez a kérdés messze túlmutat a programozáson. Ugyanez merül majd fel az orvosi diagnosztikában, a mérnöki tervezésben, a pénzügyekben, a közigazgatásban, az önvezető járműveknél vagy akár a tudományos kutatásban is.

A Debian döntése ezért egy korszakhatár apró, de érdekes dokumentuma. Nem próbálja megállítani a mesterséges intelligencia használatát, hanem megpróbálja megtalálni a helyét az emberi munkában.

Talán ez lesz az ember és az AI együttműködésének egyik legfontosabb szabálya:

Az AI írhat kódot. A nevét azonban egyelőre még az ember írja alá.

12. Játszani is engedd

A számítógépet komoly célokra fejlesztették: számításra, tudományos kutatásra, katonai feladatokra, adatfeldolgozásra. De amint a gépek képessé váltak az emberrel való közvetlen kapcsolatra, szinte törvényszerűen felmerült a kérdés:

Lehet vele játszani?

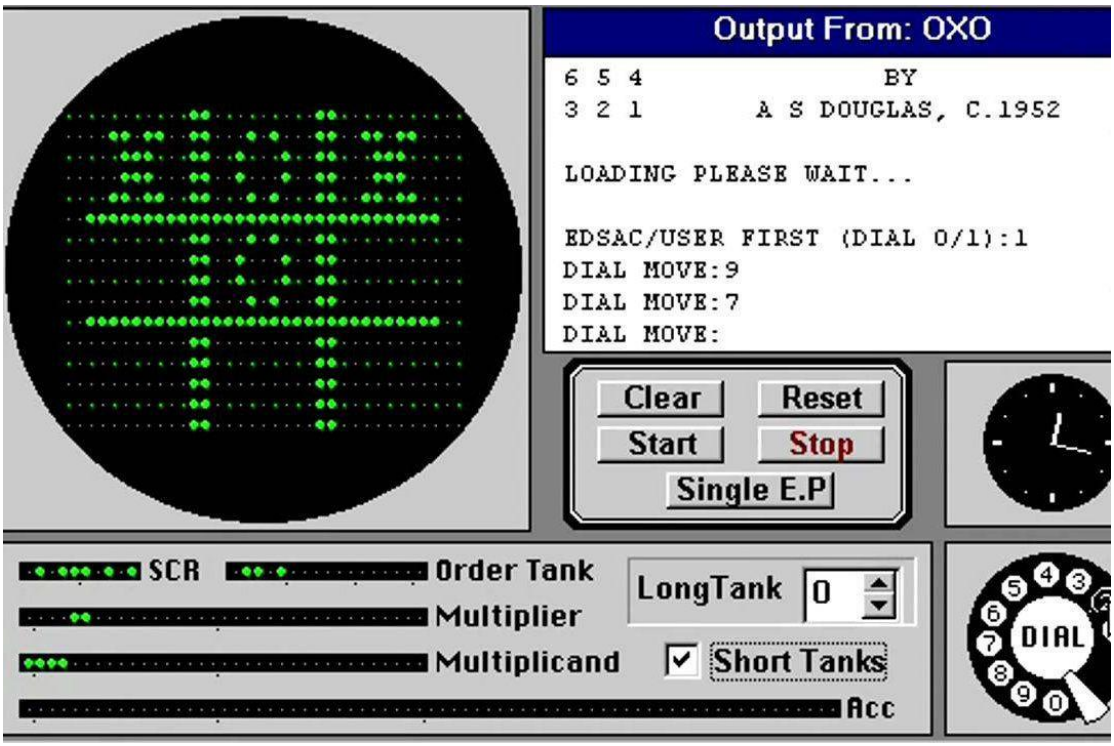
Lehetett. Sőt, a játék később a számítástechnika fejlődésének egyik fontos hajtóerejévé vált. A gyorsabb grafika, a jobb hang, a joystick és más vezérlők, a 3D gyorsítókártyák, majd az internetes hálózatok fejlődését jelentős részben a játékosok igényei ösztönözték.

12.1. Az első számítógépes játékok

Nehéz egyetlen programra rámutatni, és kijelenteni: **ez volt az első számítógépes játék**.

Az 1950-es években már készültek játékjellegű kísérleti programok. 1952-ben Alexander S. Douglas Cambridge-ben az EDSAC számítógépre elkészítette az **OXO** nevű amőbajátékot. 1958-ban William Higinbotham analóg számítógépet és oszcilloszkópot használt a **Tennis for Two** megalkotásához.

Az igazán nagy hatású korai játék azonban a Spacewar! lett.





Ábra: A számítógépes játék ősei. Az OXO, a Tennis for Two és az 1962-es Spacewar! még kutatóintézetek drága berendezésein futott. Már ezek megmutatták azonban a számítógép egyik új tulajdonságát: a gép nemcsak kiszámíthat valamit, hanem **valós időben reagálhat a játékos cselekedeteire**.

12.2. Pong – két ütő és egy pont

1972-ben Nolan Bushnell és Ted Dabney cége, az Atari piacra dobta a **Pong** játéktermi automatát.

A játék szinte nevetségesen egyszerű volt.

Két függőleges vonal jelentette az ütőket, egy világító pont a labdát. A két játékos forgatógombokkal mozgatta az ütőket.

Ábra: Atari Pong, 1972. Néhány vonal és egy mozgó pont elegendő volt ahhoz, hogy megszülessen a videojáték-ipar egyik első nagy kereskedelmi sikere.

A Pong jelentőségét nem grafikai bonyolultsága adta, hanem az, hogy bebizonyította:

az emberek hajlandók pénzt fizetni azért, hogy elektronikus játékkal játszhassanak.

Hamarosan otthoni változatok is megjelentek, amelyek a televíziót használták kijelzőként.

12.3. Játékgépek – az arcade aranykora

Az 1970-es évek végétől játéktermek ezrei teltek meg videojáték-automatákkal.

Space Invaders. Asteroids. Pac-Man. Donkey Kong.

A játék már önálló iparággá vált.

Ábra: Pac-Man játékautomata. Az 1980-ban megjelent játék a videojátékok egyik legismertebb kulturális ikonjává vált. A játéktermek gépei külön számítógépes hardvert, képernyőt és kezelőszerveket építettek egyetlen játék köré.

A játékgépek fontos technológiai laboratóriumként is működtek. Mivel a teljes elektronika egy meghatározott játékot szolgált, olyan grafikai és hanghatásokat lehetett létrehozni, amelyekre az akkori általános célú otthoni számítógépek sokszor még nem voltak képesek.

12.4. Otthoni számítógépek – a játék beköltözik a lakásba

Az 1980-as évek mikroszámítógépes forradalma alapvetően megváltoztatta a játék világát.

A ZX Spectrum, Commodore 64, Atari gépek, Amiga és más otthoni számítógépek tulajdonosainak jelentős része játékra is használta gépét.


JEEP-TOP
PUSH FIRE
30000
HELI-TOP
30000
TIME 0:00:26
PHILIPS


RED HEAT
WITH AUTO FIRE OPTION!
QuickShot II
FITS MOST POPULAR GAMES & COMPUTER SYSTEMS
HEMPSTON
ZX Spectrum
48K ZX Spectrum
THE HOBBIT



Ábra: Az otthoni számítógép mint játékgép. A Commodore 64 és kortársai generációk számára jelentették az első találkozást a számítógépes játékokkal – és gyakran magával a programozással is.

Ez utóbbi különösen fontos.

A gép bekapcsolásakor sok esetben BASIC várta a felhasználót. A játékos előbb játszott, aztán kíváncsi lett arra, hogyan működik a program.

Majd megpróbálta megírni a sajátját.

A játék így sok fiatal számára **bejárat lett a számítástechnikába**.

12.5. PC-játékok

Az IBM PC-t eredetileg nem játékgépnek tervezték. Grafikai és hangképességei kezdetben elmaradtak több otthoni számítógépétől.

A kompatibilis PC-k rohamos fejlődése azonban megváltoztatta a helyzetet.

Megjelent az EGA, majd a VGA grafika, az AdLib és Sound Blaster hangkártya, egyre gyorsabb processzorok és egyre nagyobb merevlemezek.

A PC lassan kiváló játékgéppé vált.

Az 1993-as **Doom** ennek egyik látványos jelképe.

Ábra: Doom, 1993. Az id Software játéka látványos, gyors, belső nézetű világával mérföldkő lett a PC-játékok történetében. Hálózati többjátékos módja a közös számítógépes játék új lehetőségeit is megmutatta.

A játékok egyre inkább kikényszerítették a gyorsabb processzorokat, nagyobb memóriát és jobb grafikus kártyákat.

Megjelent a PC-fejlesztés egyik legerősebb kérdése:

„Elfut rajta a játék?"

12.6. Konzolok – számítógép, amelynek a játék a dolga

Az általános célú számítógépekkel párhuzamosan egy másik fejlődési vonal is kialakult: a **játékkonzol.**

Az Atari 2600, majd a Nintendo, Sega, Sony és Microsoft gépei olyan számítógépek voltak, amelyeket elsősorban játékra terveztek.

Ábra: Nintendo Entertainment System – NES. Az 1980-as évek egyik meghatározó konzolja cserélhető játékkazettákkal működött. A konzol egyik előnye az egyszerűség volt: behelyeztük a játékot, bekapcsoltuk a gépet – és játszottunk.

A konzolok története egyben különleges verseny is.

Atari, Nintendo, Sega, Sony, Microsoft.

Minden generáció gyorsabb processzort, jobb grafikát és új kezelési lehetőségeket hozott.

A játékkonzol közben maga is egyre inkább általános multimédiás számítógéppé vált.

12.7. 3D grafika – belépünk a játék világába

A korai játékok sík képernyőjén kétdimenziós figurák mozogtak.

A következő nagy cél egy térbeli világ létrehozása volt.

Ehhez a számítógépnek folyamatosan ki kellett számítania, hogyan látszanak a háromdimenziós tárgyak a játékos pillanatnyi nézőpontjából.

A Wolfenstein 3D és a Doom fontos lépcsőfokok voltak, majd az 1996-ban megjelent **Quake** már valódi, poligonokból felépített háromdimenziós környezetet használt.


100
35


3dfx
N2544



Ábra: A háromdimenziós játékvilág megszületése. A Quake korszakára a számítógép már valós időben számította ki a térbeli világ perspektíváját, megvilágítását és mozgását. A játékok igénye nagyban hozzájárult a külön grafikus gyorsítók, majd a modern GPU-k elterjedéséhez.

A grafikus processzor később messze túlnőtt eredeti feladatán.

Ma GPU-kat használunk tudományos számításokra és **mesterséges intelligencia tanítására is.**

Különös történeti kör: a játékok kedvéért kifejlesztett hardver a modern AI egyik alapvető eszközévé vált.

12.8. Online játékok – már nem a gép ellen játszunk

Amikor a számítógépeket hálózatba kötötték, természetesen a játékok is megjelentek a hálózaton.

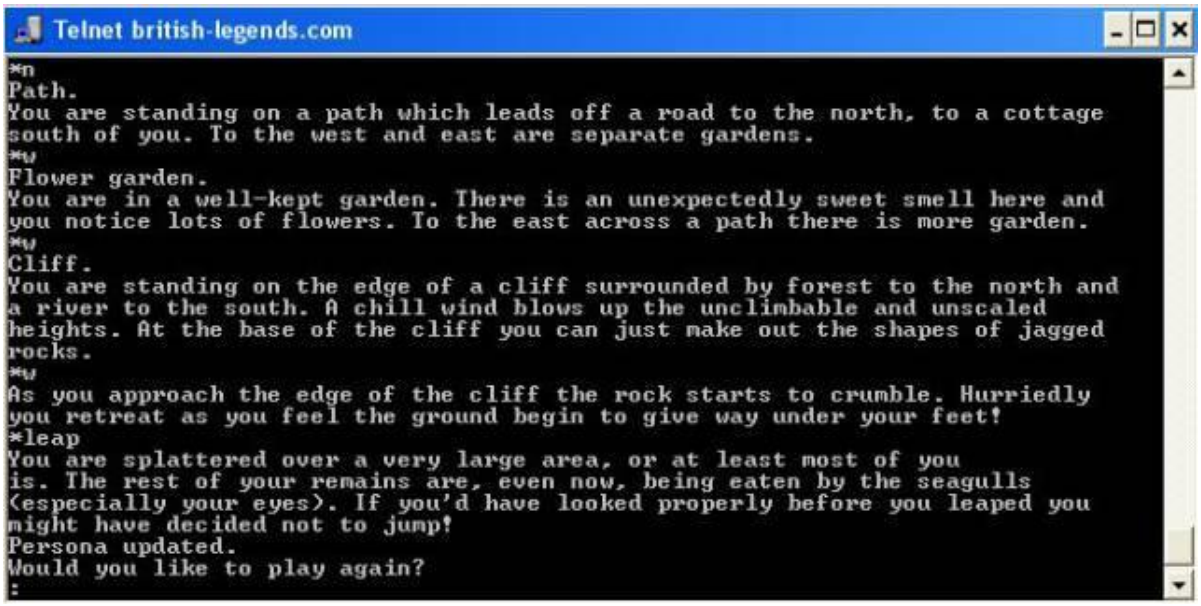
A korai egyetemi rendszereken már léteztek többfelhasználós szöveges kalandjátékok, az úgynevezett **MUD-ok.**

A helyi hálózatokon később egymás ellen lehetett játszani például Doomot és Quake-et.

Az internet pedig eltüntette a fizikai távolságot.


Annakie
Sen'jin Village
Clattering Scorpid misses ZincBar.
Clattering Scorpid hits ZincBar for 6.
Pygmy Surf Crawler hits ZincBar for 7.
Pygmy Surf Crawler attacks. ZincBar blocks.
bluegreenorc hits Pygmy Surf Crawler for 15.
bluegreenorc hits Pygmy Surf Crawler for 13.
ZincBar hits Pygmy Surf Crawler for
bluegreenorc hits Pygmy Surf Crawl
Sen'jin Watcher
Level 14
[1. General] HairyCrack: I mean where, sorry





Ábra: A játék hálózattá válik. A LAN-partiktól a tömeges online világokig a számítógépes játék társas tevékenységgé fejlődött. Egy virtuális világban akár több ezer, fizikailag egymástól távol lévő játékos találkozhat.

Ezzel valami érdekes történt.

A számítógépes játék kezdetben **ember és gép kapcsolata** volt.

Az internet korában a gép egyre inkább csak közvetítő:

ember → számítógép → hálózat → számítógép → ember.

Spacewar! – játék egy méregdrága számítógépen

1962-ben az MIT-n működő DEC PDP-1 számítógép különleges berendezésnek számított.

Nem játékra vásárolták.

A gép ára konfigurációtól függően százezres dolláros nagyságrendű volt – akkoriban.

Mégis ezen született meg a számítógépes játékok történetének egyik legendája.



Ábra: PDP-1 és Spacewar! A két játékos egy-egy űrhajót irányított, miközben nemcsak egymással, hanem a képernyő közepén lévő csillag gravitációjával is meg kellett küzdeniük.

Steve Russell és társai olyan programot akartak készíteni, amely valóban kihasználja a PDP-1 grafikus kijelzőjét és interaktív képességeit. Megszületett a **Spacewar!**.

Két űrhajó küzdött egymással.

Rakétákat lehetett kilőni, manőverezni kellett, miközben a központi csillag gravitációja befolyásolta az űrhajók pályáját. A program másodpercenként több mint százezer számítást végzett a mozgás, gravitáció és vezérlés kezelésére.

A játék gyorsan terjedt azokon az intézményeken belül, ahol megfelelő számítógép állt rendelkezésre, és a PDP-1 képességeinek bemutatására is használták.

A történetnek van egy szép iróniája.

A számítógépek fejlesztésének egyik fő célja az volt, hogy **komoly problémákat gyorsabban oldjunk meg velük**.

Amint azonban a gép elég interaktív lett, valakinek rögtön eszébe jutott: **játsszunk vele!**

Tetris – játék a vasfüggöny mögül

1984-ben Alekszej Pazsitnov a Szovjet Tudományos Akadémia számítógépes központjában dolgozott Moszkvában.

Az intézet Elektronika–60 számítógépére készített egy egyszerű játékot.

A gép grafikai lehetőségei korlátozottak voltak, de ez nem akadályozta meg az ötletet.

Különböző alakú, négy négyzetből álló elemek hullottak lefelé. A játékos forgathatta és mozgathatta őket. Ha egy teljes sor kialakult, az eltűnt.

Megszületett a **Tetris**.


ТЕТРИС
(TETRIS)
The Soviet Challenge
Spectrum Holobyte
TETRIS



Ábra: A Tetris születése a Szovjetunióban. Alekszej Pazsitnov 1984-ben egy Elektronika–60 számítógépre készítette a játék első változatát. A rendkívül egyszerű grafika mögött olyan játékmenet rejtőzött, amely később az egész világot meghódította.

A Tetris története azonban nemcsak programozástörténet.

A szovjet rendszerben a program szerzője nem rendelkezhetett egyszerűen úgy alkotásának kereskedelmi jogaival, mint egy nyugati magánvállalkozó. A játék nemzetközi licencelése ezért különösen bonyolult történetté vált.

Végül a Nintendo Game Boy egyik meghatározó játékaként jutott el emberek millióihoz.

A Tetris sikere különösen szép ellenpéldája annak az elképzelésnek, hogy egy jó játékhoz feltétlenül látványos grafika és óriási számítási teljesítmény szükséges.

A Spacewar! egy rendkívül drága számítógép képességeit demonstrálta.

A Tetris egy egyszerű ötlet erejét.

A jó játék legfontosabb alkatrésze nem mindig a processzor – hanem az ötlet.


60 ÉV SZÁMÍTÓGÉPES JÁTÉK – EGY TÖRTÉNET EGY KÉPEN
1962
Spacewar!
PDP-1 számítógép
Az egyik első számítógépes játék.
1972
Pong
Atari arcade gép
Az első nagy kereskedelmi siker.
1980
Pac-Man
Namco arcade gép
A játékgépek aranykora.
1982
Commodore 64
Otthoni számítógép
A játék beköltözik a lakásba.
1993
Doom
PC játék
3D világ, hálózati mód, új korszak kezdete.
1996
Nintendo 64
Konzolok 3D korszaka
Új élmények, új vezérlők.
1996
Quake
Valódi 3D grafika, új grafikus gyorsítók megjelenése.
2004–
Online világok
Többjátékos online világok, közösségek, e-sport.
12.1 Az első játékok
Kutatóintézetek drága számítógépein születtek az első kísérleti játékok.
12.2 Pong
Egyszerű grafika, egyszerű szabályok, határtalan szórakozás.
PONG
12.3 Játékgépek
A játéktermek tele voltak színes, izgalmas gépekkel.
12.4 Otthoni gépek
Kazetta, floppy, betöltés, de cserébe végtelen lehetőségek.
12.5 PC-játékok
Egyre jobb grafika, hang, történet és játékmenet.
12.6 Konzolok
Egyszerű használat, exkluzív játékok, nagy rajongótáborok.
12.7 3D grafika
A valósághű világok megjelenítéséhez hatalmas számítási teljesítmény kell.
3dfx
Voodoo3
12.8 Online játékok
A világ bármely pontjáról játszhatunk együtt milliókkal.
KERETES: Spacewar! – játék egy méregdrága számítógépen
1962-ben a PDP-1 számítógépen futott a Spacewar! két űrhajó küzdelméről szólt, miközben a csillag gravitációja is hatott rájuk. A gép ára akkoriban egy lakásével vetekedett, mégis sok kutató órákat töltött a játékkal.
KERETES: Tetris – játék a vasfüggöny mögül
Aleksej Pazsitnov 1984-ben készítette el a Tetris első változatát az Elektronika-60 számítógépen Moszkvában. Egyszerű ötlet, amely később az egész világot meghódította.
A JÁTÉK HAJTÓEREJE: A TECHNOLÓGIA FEJLŐDÉSE
8 bites grafika
3D világok
GPU-k megjelenése
Mesterséges intelligencia
A játékok igényei gyorsabbá, szebbé és okosabbá tették a számítógépeket – ma ugyanez a teljesítmény tudományos kutatást és AI-t is szolgál.

V. rész – A számítógépek összekapcsolódnak

13. Számítógép-hálózatok

A számítógép sokáig magányos gép volt. A felhasználó bevitte az adatokat, a gép kiszámította az eredményt, majd kiírta vagy kinyomtatta. A következő nagy változás akkor következett be, amikor rájöttek: **nem feltétlenül az embert kell a számítógéphez vinni – elég összekötni őket.**

Ebből az egyszerű gondolatból nőtt ki a számítógép-hálózat, majd végül az internet. A hálózatok története egyben annak története is, hogyan vált a számítógép egyetlen ember munkaeszközéből **egy világméretű információs rendszer csomópontjává.**

13.1. Terminál és nagyszámítógép

Az 1950-es és 1960-as évek számítógépei drágák voltak, ezért egy intézményben rendszerint egyetlen nagy központi gépet használtak. Kezdetben a programokat lyukkártyán vagy lyukszalagon vitték a géphez, az eredményt pedig később, többnyire papíron kapták vissza.

Az időosztásos rendszerek megjelenése alapvetően megváltoztatta ezt. Egyetlen nagyszámítógéphez egyszerre több felhasználó kapcsolódhatott saját **termináljáról.**

Ábra: *IBM 3278 terminál. A billentyűzetből és képernyőből álló munkaállomás önmagában alig végzett számításokat: a tényleges munkát a hozzá kapcsolódó nagyszámítógép végezte.*

A terminál lényegében billentyűzetből és képernyőből állt. Az úgynevezett „buta terminál” nem volt önálló számítógép: a beírt adatokat továbbította a központi gépnek, majd megjelenítette annak válaszát. Az IBM 3270 család például 1971-től vált a nagyszámítógépes világ egyik jellegzetes terminálrendszerévé.

Ez érdekes módon előrevetítette a mai felhőalapú számítástechnikát:

terminál → hálózat → központi számítógép

ma pedig sokszor:

böngésző → internet → felhőben működő szerver

A technika teljesen megváltozott, az alapgondolat meglepően hasonló maradt.

13.2. Modem

Hogyan lehet azonban egy távoli terminált összekötni a számítógéppel, ha nincs közöttük külön adatvezeték?

Kézenfekvő megoldást kínált a már létező telefonhálózat.

A számítógép digitális jeleit azonban az eredetileg emberi beszéd továbbítására épített telefonvonal nem tudta közvetlenül kezelni. Szükség volt egy átalakító berendezésre: a **modemre**.

A név két szóból született:

MOdulátor + DEModulátor = MODEM



Ábra: *Korai akusztikus modem. A telefon kézibeszélőjét a két gumiharangba kellett helyezni. A számítógép adatai hallható hangokká alakítva utaztak a telefonvonalon.*

A korai modemek sebessége mai szemmel szinte elképzelhetetlenül kicsi volt: néhány száz bit másodpercenként. Később az otthoni számítógépek korszakában elterjedtek az 1200, 2400, 9600, 14 400, 28 800, 33 600, végül az 56 000 bit/s körüli modemek.

Aki használta őket, aligha felejti el a kapcsolat felépítésekor hallható különös sípoló, fütyülő és recsegő hangsorokat. Ezek nem véletlen zajok voltak: **két modem egyeztette egymással a kapcsolat paramétereit.**

A modem tette lehetővé, hogy az otthoni számítógép előbb BBS-ekhez és különféle online szolgáltatásokhoz, később pedig az internethez kapcsolódjon.

1978. február 16-án Chicago városában elindult a világ első **számítógépes bulletin board rendszere (BBS – Computer Bulletin Board System)**, amelyet két amerikai programozó, **Ward Christensen és Randy Suess** hozott létre. A rendszert **CBBS-nek (Computerized Bulletin Board System)** nevezték el, és célja az volt, hogy a felhasználók üzeneteket hagyhassanak egymásnak, hasonlóan egy fizikai hirdetőtáblához.

A CBBS egy **telefonvonalon keresztül működött**, és a felhasználók egy modem segítségével csatlakozhattak hozzá. Bár mai szemmel nézve ez kezdetleges technológiának tűnik, akkoriban forradalmi újításnak számított, hiszen lehetőséget adott az embereknek arra, hogy **távolról is kommunikáljanak egymással egy számítógépes hálózaton keresztül**.



Miért hozták létre a BBS rendszert?

Ward Christensen és Randy Suess ötlete egy különösen hideg és havas tél során született meg, amikor a helyi számítógépes klub tagjai nem tudtak személyesen találkozni. Arra gondoltak, hogy egy központi számítógépen keresztül is megoszthatnák egymással az információkat és az üzeneteket. Így született meg a CBBS, amelyet az **S-100 alapú mikroprocesszoros számítógépükön futtattak egy 110 baud sebességű modemen keresztül**.

A CBBS és a későbbi BBS rendszerek működése meglehetősen egyszerű volt, de a kor technológiai szintjéhez képest lenyűgöző:

A felhasználók **modemmel csatlakoztak a rendszerhez** telefonvonalon keresztül.

Miután bejelentkeztek, **üzeneteket olvashattak vagy hagyhattak másoknak**, akárcsak egy digitális üzenőfalon.

Bizonyos rendszerek lehetővé tették **fájlok megosztását**, valamint programok és játékok letöltését is.

Mivel egy BBS rendszer általában **csak egyetlen telefonvonalon keresztül volt elérhető**, egyszerre csak egy felhasználó csatlakozhatott, így várakozni kellett, míg mások befejezték a használatot.

Bár az internet végül felváltotta a BBS rendszereket, ezek mégis **megalapozták az online kommunikáció számos formáját**. A BBS rendszerek voltak az első olyan platformok, ahol az emberek:

Elektronikus leveleket (e-mail) küldhettek egymásnak, évekkel azelőtt, hogy az interneten elterjedt volna az e-mail használata.

Digitális közösségeket hozhattak létre, amelyek később az online fórumok és a közösségi média elődjeivé váltak.

Fájlokat és programokat oszthattak meg, elősegítve a szoftverfejlesztők és a technológiai közösségek együttműködését.

Kapcsolódó érdekességek

A 80-as és 90-es években a BBS rendszerek virágkorukat élték, és **több ezer független BBS működött világszerte**.

Magyarországon is működtek BBS rendszerek a 90-es években, és sok számítógépes klub használta őket információmegosztásra.

A BBS-ek **alapvetően különböztek az internettől**, mert közvetlen telefonvonalas kapcsolaton keresztül működtek, míg az internet egy összekapcsolt hálózat volt.

A 90-es évek végére az internet térhódítása miatt a BBS-ek lassan eltűntek, de **egyes retró közösségek a mai napig fenntartanak BBS szervereket nosztalgikus céllal**.

Bár ma már a legtöbben a webes fórumokat és a közösségi médiát használják, a **CBBS és a BBS rendszerek úttörő szerepet játszottak** az online kommunikáció fejlődésében. Ezek voltak azok a platformok, ahol az első „geekek” először megtapasztalhatták az internet előtti digitális közösségek erejét.

Forrás: https://linuxmint.hu/hir/2025/02/az-elso-bbs-rendszer-indulasa

13.3. LAN és WAN

Ha egy intézményben már sok számítógép működött, ésszerű volt ezeket egymással is összekötni.

Megszületett a **LAN – Local Area Network**, vagyis helyi hálózat.

Egy LAN rendszerint egy épület, iroda, iskola, laboratórium vagy üzem számítógépeit kapcsolja össze. A gépek így közösen használhatnak például nyomtatót, háttértárat vagy szervert, és adatokat cserélhetnek egymással.

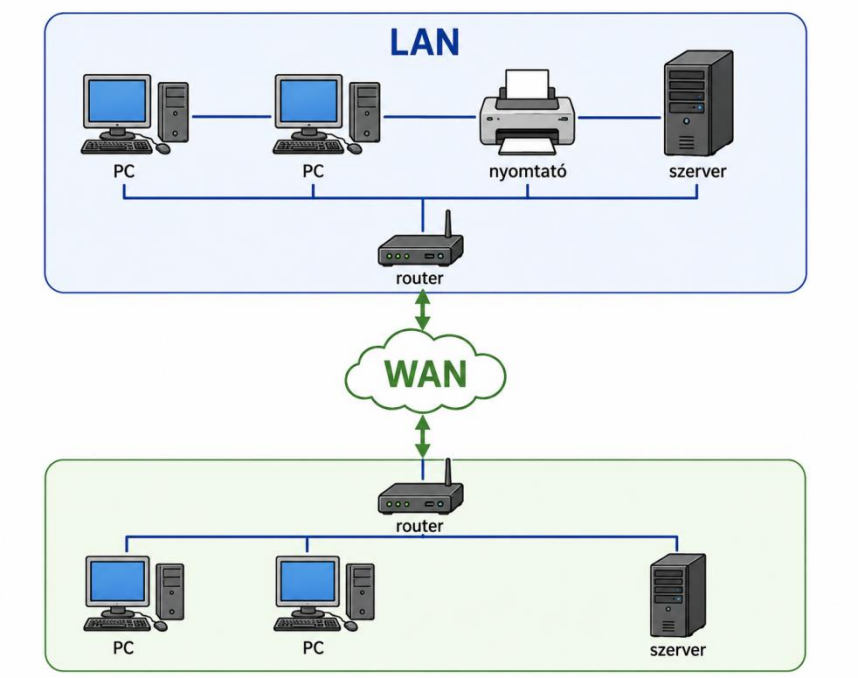
Ha a hálózat nagyobb földrajzi területre terjed ki, **WAN-ról – Wide Area Networkről**, nagy kiterjedésű hálózatról beszélünk.

Egyszerűen:

LAN → egy helyen lévő gépek

WAN → távoli hálózatok összekapcsolása

Az internet lényegében hálózatok hálózata – a világ LAN-jait és más hálózatait összekapcsoló globális rendszer.



Ábra: *A helyi hálózat (LAN) kis területen kapcsolja össze az eszközöket. A távoli helyi hálózatokat nagy kiterjedésű hálózatok (WAN) kötik össze.*

13.4. Ethernet

A helyi hálózatok történetének egyik legfontosabb találmánya az **Ethernet.**

Az alapötletet az 1970-es évek elején Robert Metcalfe és munkatársai dolgozták ki a Xerox PARC kutatóközpontban. Az Ethernet korai változatában a számítógépek közös koaxiális kábelre csatlakoztak.

A rendszer egyik fontos problémája az volt: mi történjen, ha két számítógép egyszerre akar adni?

A gépek figyelték a vezetéket. Ha szabad volt, megkezdték az adást. Ha két állomás mégis egyszerre kezdett továbbítani, ütközés történt. Ezt érzékelték, rövid, véletlenszerű ideig vártak, majd újra próbálkoztak.

Ez volt a **CSMA/CD** eljárás.

A korai Ethernet 2,94 Mbit/s körüli sebességéről hamarosan 10 Mbit/s-ra fejlődött, majd következett a Fast Ethernet 100 Mbit/s, a Gigabit Ethernet és az ennél is gyorsabb változatok.

Közben a koaxiális kábeleket nagyrészt felváltotta a sodrott érpár, illetve nagy sebességnél az optikai kábel.

Az Ethernet az egyik legjobb példája annak, amikor egy több mint fél évszázados alapötlet folyamatos fejlesztéssel ma is a számítástechnika alapvető technológiája.

Képjavaslat

Korai Ethernet-hálózat Xerox Alto számítógépekkel és koaxiális kábellel.

Ábra: *Az Ethernet a Xerox PARC-ban született meg az 1970-es években. A korai hálózatok közös koaxiális kábelen továbbították az adatokat.*

13.5. Csomagkapcsolás

A hagyományos telefonhálózatban egy beszélgetéshez kapcsolatot – áramköri utat – hoztak létre a két fél között.

A számítógépek kommunikációjához azonban ez pazarló megoldás.

Miért kellene egy teljes vonalat lefoglalni, miközben a számítógép az idő nagy részében nem küld semmit?

A megoldás a **csomagkapcsolás** lett.

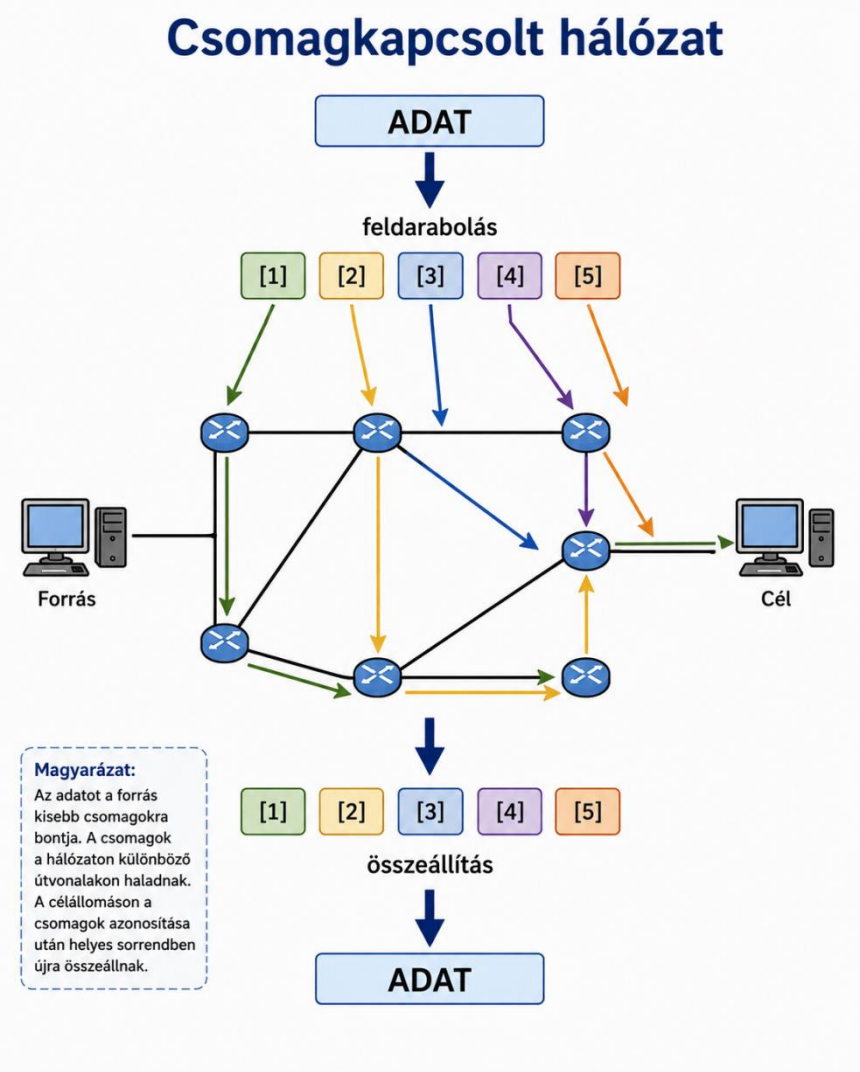
Az elküldendő adatot kisebb részekre, **csomagokra** bontják. Minden csomag tartalmazza a célba juttatásához szükséges információkat. A hálózat továbbítja őket, akár eltérő útvonalakon is, a fogadó számítógép pedig ismét összeállítja belőlük az eredeti adatot.

Ezt jól szemlélteti a postai hasonlat:

Nem egy egész tehervonatot bérelünk ki egyetlen levél számára. A levelet betesszük a többi közé, ráírjuk a címet, a postai hálózat pedig eljuttatja a címzetthez.

A csomagkapcsolás elméleti alapjain több kutató dolgozott, köztük **Paul Baran** az Egyesült Államokban és **Donald Davies** Nagy-Britanniában. Davies használta a „packet” elnevezést is.

Ez lett a modern számítógép-hálózatok egyik legfontosabb alapelve.



Ábra: *Csomagkapcsolás: az információ kisebb adatcsomagokra bomlik, amelyek a hálózaton akár különböző útvonalakon juthatnak el ugyanahhoz a címzetthez.*

13.6. TCP/IP

Ahogy egyre több különböző számítógép és hálózat jött létre, új probléma jelentkezett:

hogyan beszéljenek egymással olyan hálózatok, amelyeket eredetileg nem egymáshoz terveztek?

Ehhez közös szabályokra – protokollokra – volt szükség.

Az internet alapvető protokollcsaládjává a **TCP/IP** vált, amelynek kidolgozásában Vint Cerf és Bob Kahn játszott meghatározó szerepet.

Az IP – **Internet Protocol** – alapvetően a csomagok címzéséről és továbbításáról gondoskodik.

A TCP – **Transmission Control Protocol** – többek között azt biztosítja, hogy az adatfolyam megfelelő sorrendben és megbízhatóan érkezzen meg.

Nagyon leegyszerűsítve:

IP: Hová menjen?

TCP: Minden rendben megérkezett?

1983. január 1-jén az ARPANET átállt a TCP/IP használatára. Ezt a dátumot ezért gyakran az internet történetének egyik fontos fordulópontjaként említik.

A TCP/IP egyik nagy ereje éppen az volt, hogy **nem követelte meg az összekapcsolt számítógépektől és hálózatoktól, hogy egyformák legyenek.**

Ez tette lehetővé a „network of networks” – **hálózatok hálózata** – létrejöttét.

Internet és web – nem ugyanaz!

A hétköznapi beszédben gyakran ugyanazt értjük az „internet” és a „web” szavakon. Technikailag azonban két különböző dologról van szó.

Az **internet** a világ számítógépes hálózatainak összekapcsolt rendszere. Kábelekből, optikai vezetékekből, rádiós kapcsolatokból, routerekből, szerverekből és kommunikációs szabályokból – protokollokból – áll.

Az internet tehát elsősorban **infrastruktúra.**

Ezen az infrastruktúrán sokféle szolgáltatás működhet:

- elektronikus levelezés,
- fájlátvitel,
- távoli bejelentkezés,
- internetes telefonálás,
- online játék,
- üzenetküldés,

- videóközvetítés,
- valamint a World Wide Web.

A **World Wide Web**, röviden **WWW vagy web**, az interneten működő információs rendszer. Egymással hiperhivatkozásokkal összekapcsolt dokumentumokból, képekből, videókból és más tartalmakból áll, amelyeket rendszerint böngészőprogrammal érünk el.

Egyszerű hasonlattal: **Az internet az úthálózat.**

A web az egyik rajta működő szállítási rendszer.

Vagy másként: **Internet = hálózatok és kapcsolatok**

Web = az interneten elérhető, egymással összekapcsolt oldalak

Az internet már a web megszületése előtt is létezett. E-mailt, fájlátvitelt és más hálózati szolgáltatásokat már akkor is használtak, amikor még nem léteztek weboldalak és böngészők.

Tim Berners-Lee tehát nem az internetet találta fel. A világhálót dolgozta ki: azt a rendszert, amely az internetet könnyen kezelhető, összekapcsolt információs térré alakította.

A web az internet legismertebb szolgáltatása – de nem maga az internet.

13.7. Wi-Fi

A hálózati kábel jó – egészen addig, amíg mozgatni nem akarjuk a számítógépet.

A hordozható számítógépek terjedésével egyre nagyobb igény mutatkozott arra, hogy a helyi hálózat rádióhullámokon keresztül is elérhető legyen.

Az IEEE 802.11 szabvány első változatát 1997-ben fogadták el. A szélesebb körű elterjedést azonban különösen az 1999-ben megjelent 802.11b segítette.

Ábra: *Korai 802.11b Wi-Fi hálózati kártya. A szabvány 11 Mbit/s névleges adatsebességet tett lehetővé – akkoriban ez már valóban használható vezeték nélküli hálózatot jelentett.*

A **Wi-Fi** nem az internet másik neve. Ez fontos különbség.

A Wi-Fi csupán egy módszer arra, hogy egy eszköz vezeték nélkül csatlakozzon egy helyi hálózathoz. A hálózat ezután kapcsolódhat az internethez – de nem feltétlenül kell kapcsolódnia.

A Wi-Fi sebessége azóta nagyságrendekkel nőtt, és ma már számítógépek, telefonok, televíziók, kamerák, háztartási eszközök és számtalan más készülék használja.

A hálózati csatlakozó ezzel gyakorlatilag **láthatatlanná vált.**

13.8. Bluetooth

Nem minden eszköznek van szüksége teljes értékű helyi hálózatra.

Egy egér, billentyűzet, fejhallgató vagy érzékelő esetében fontosabb lehet a kis fogyasztás, az egyszerű kapcsolat és a rövid hatótávolság.

Erre született meg a **Bluetooth.**

A technológia fejlesztése az 1990-es években indult az Ericssonnál, majd több nagyvállalat együttműködésével vált ipari szabvánnyá.

A Bluetooth egyik legérdekesebb része maga a név.

Még a Bluetooth embléma is történelmi játék: Harald nevének két rúnaírásos kezdőbetűjéből alakították ki.

Képjavaslat



A Bluetooth különösen fontos lett a mobiltelefonok elterjedésével. Ma fejhallgatók, hangszórók, egerek, billentyűzetek, órák, autók, egészségügyi eszközök és érzékelők milliárdjai kommunikálnak vele.

A hálózat fejlődése egyetlen sorban



Miért hívják Bluetoothnak?

Kevés számítástechnikai szabvány visel egy több mint ezer éve élt királyról elnevezett nevet. A Bluetooth ilyen.

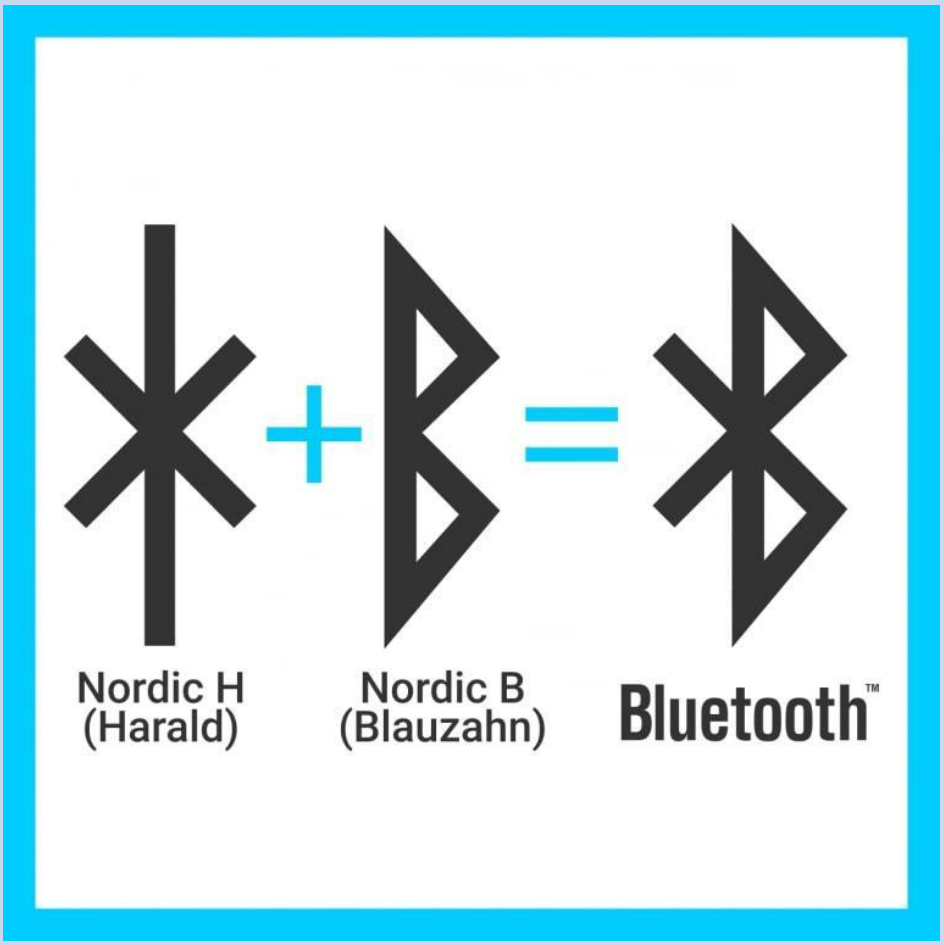
A technológia fejlesztése az 1990-es években indult az Ericssonnál. A cél az volt, hogy mobiltelefonok, számítógépek és különféle kiegészítő eszközök rövid távolságon, vezeték

nélkül kommunikálhassanak egymással. Amikor az Ericsson, az IBM, az Intel, a Nokia és a Toshiba együttműködésével megkezdődött a közös szabvány kialakítása, szükség volt egy névre is.

Az Intel mérnöke, Jim Kardach a **Bluetooth** nevet javasolta. Harald „Bluetooth" Gormsson 10. századi dán királyról nevezte el, akit Dánia és Norvégia egyesítésével hoztak kapcsolatba. A hasonlat kézenfekvő volt: ahogy Harald különböző területeket egyesített, az új technológiának is különböző elektronikus eszközöket kellett összekapcsolnia.

A név eredetileg csak ideiglenes munkanévnek készült, végül azonban rajta maradt a technológián.

Még érdekesebb a jól ismert Bluetooth-jel eredete. Az embléma két germán rúna összevonása: a **H** és a **B**, vagyis Harald Bluetooth kezdőbetűi.



Ábra: *A Bluetooth jel két rúna összevonásából született: Harald Bluetooth dán király nevének kezdőbetűiből. A technológia célja különböző elektronikus eszközök egyszerű, rövid hatótávolságú összekapcsolása.*

Így amikor vezeték nélküli fejhallgatónkat vagy egerünket Bluetooth segítségével csatlakoztatjuk, egy 10. századi viking király emlékét idéző jelet használunk.

Ezer év választja el Harald királyt az okostelefontól – a Bluetooth mégis összeköti őket.

Mi történik, amikor beírunk egy webcímet?

Beírjuk a böngészőbe egy weboldal címét, megnyomjuk az Entert, és egy pillanat múlva megjelenik az oldal.

Egyszerűnek látszik. A háttérben azonban néhány tizedmásodperc alatt számítógépek és hálózati eszközök egész sora kezd dolgozni.

Tegyük fel, hogy beírjuk:

www.example.com

A számítógépek valójában nem ilyen nevekkel azonosítják egymást. Először tehát meg kell tudni, milyen **IP-cím** tartozik a névhez. Ebben a **DNS – Domain Name System**, az internet „telefonkönyve” segít.

Ha megvan az IP-cím, számítógépünk már tudja, melyik szervert keresi.

De hogyan jut el hozzá?

Az elküldendő információ kisebb **adatcsomagokra** bomlik. Ezek először helyi hálózatunkon haladnak. Notebookunk például Wi-Fi-kapcsolaton keresztül elküldi őket a routernek.

A router továbbadja őket az internetszolgáltató hálózatának, onnan pedig újabb és újabb routereken keresztül haladnak a cél felé.

Az interneten nincs feltétlenül egyetlen előre kijelölt út. Az egyes csomagok akár eltérő útvonalakon is eljuthatnak ugyanahhoz a szerverhez.

Ebben segít az **IP – Internet Protocol**: a csomagokon szereplő címek alapján a hálózat megpróbálja őket eljuttatni a megfelelő célhoz.

A **TCP – Transmission Control Protocol** használatakor a rendszer azt is ellenőrzi, hogy az adatfolyam megfelelően megérkezett-e. Ha valamelyik szükséges csomag elveszett, újra elkérhető; az adatokat pedig a fogadó oldalon megfelelő sorrendbe lehet rendezni.

A kapcsolat ma rendszerint titkosított. A böngésző és a webszerver a **HTTPS** használatával biztonságos kapcsolatot hoz létre, hogy mások ne tudják egyszerűen elolvasni vagy észrevétlenül módosítani a közöttük haladó adatokat.

Ezután böngészőnk elküldi kérését a webszervernek:

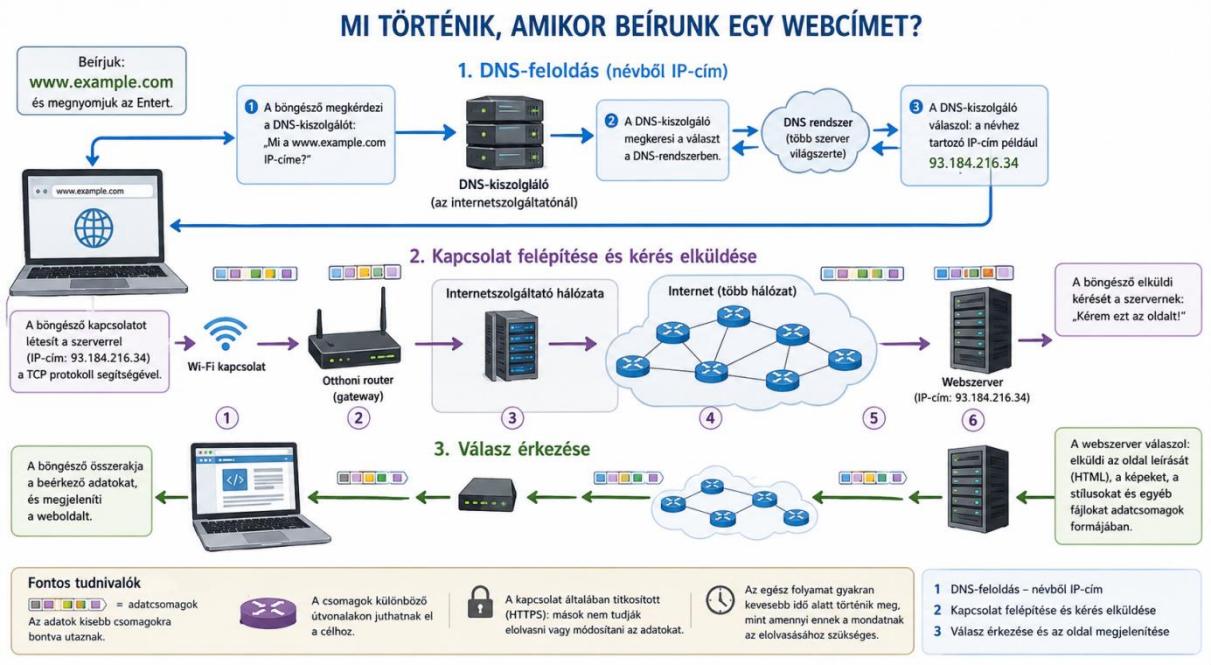
Kérem ezt az oldalt!

A szerver válaszol. Elküldi az oldal leírását, majd a böngésző további fájlokat – például képeket, betűkészleteket és programokat – kérhet le.

A válasz ismét adatcsomagok formájában utazik át a hálózaton. A számítógép összerakja a beérkező adatokat, a böngésző pedig ezekből felépíti a képernyőn látható oldalt.

Mindez gyakran rövidebb idő alatt történik meg, mint amennyi ennek a mondatnak az elolvasásához szükséges.

Webcím → DNS → IP-cím → adatcsomagok → routerek → szerver → válasz → adatcsomagok → böngésző → weboldal



A világháló használatakor tehát nem egyetlen vezetéken „megnézünk" egy távoli oldalt. Számítógépünk és egy akár több ezer kilométerre működő szerver **apró adatcsomagok sokaságával beszélget egymással**.

14. Internet

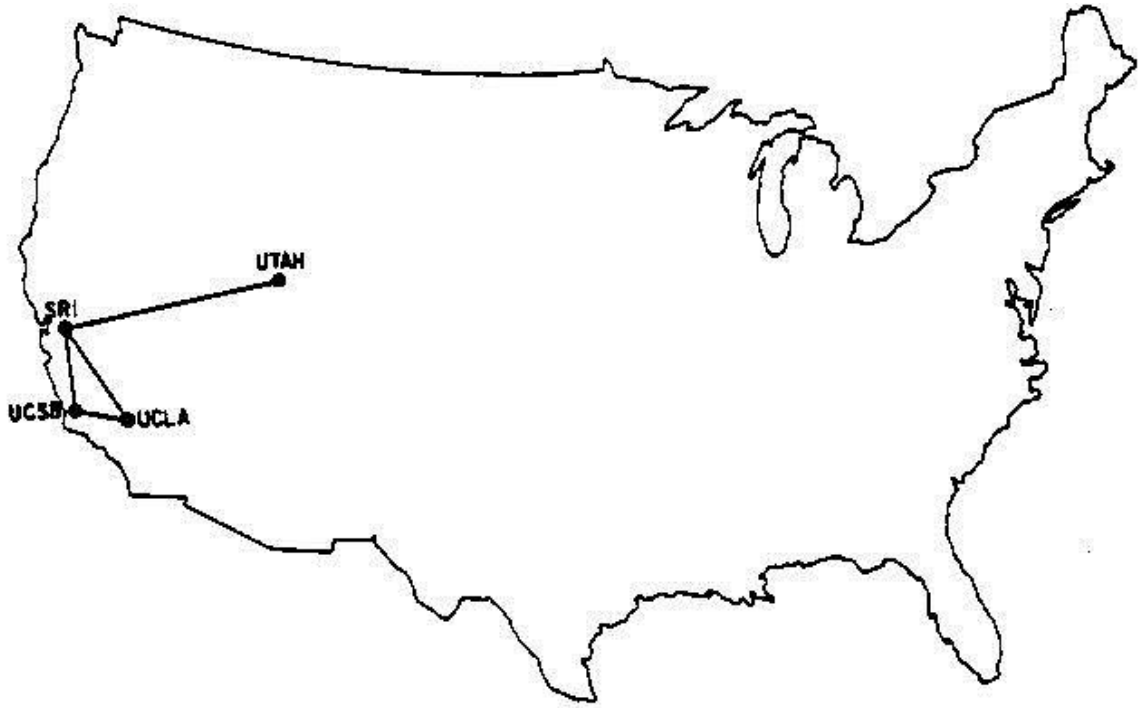
A személyi számítógép önmagában is forradalmi eszköz volt, de az igazi változás akkor következett be, amikor a gépeket hálózatba kapcsolták. Ettől kezdve már nemcsak a saját számítógépünkön tárolt programokat és adatokat használhattuk: üzeneteket küldhettünk, távoli számítógépeket érhettünk el, információkat kereshettünk, később pedig a világ szinte teljes információs rendszeréhez kapcsolódhattunk.

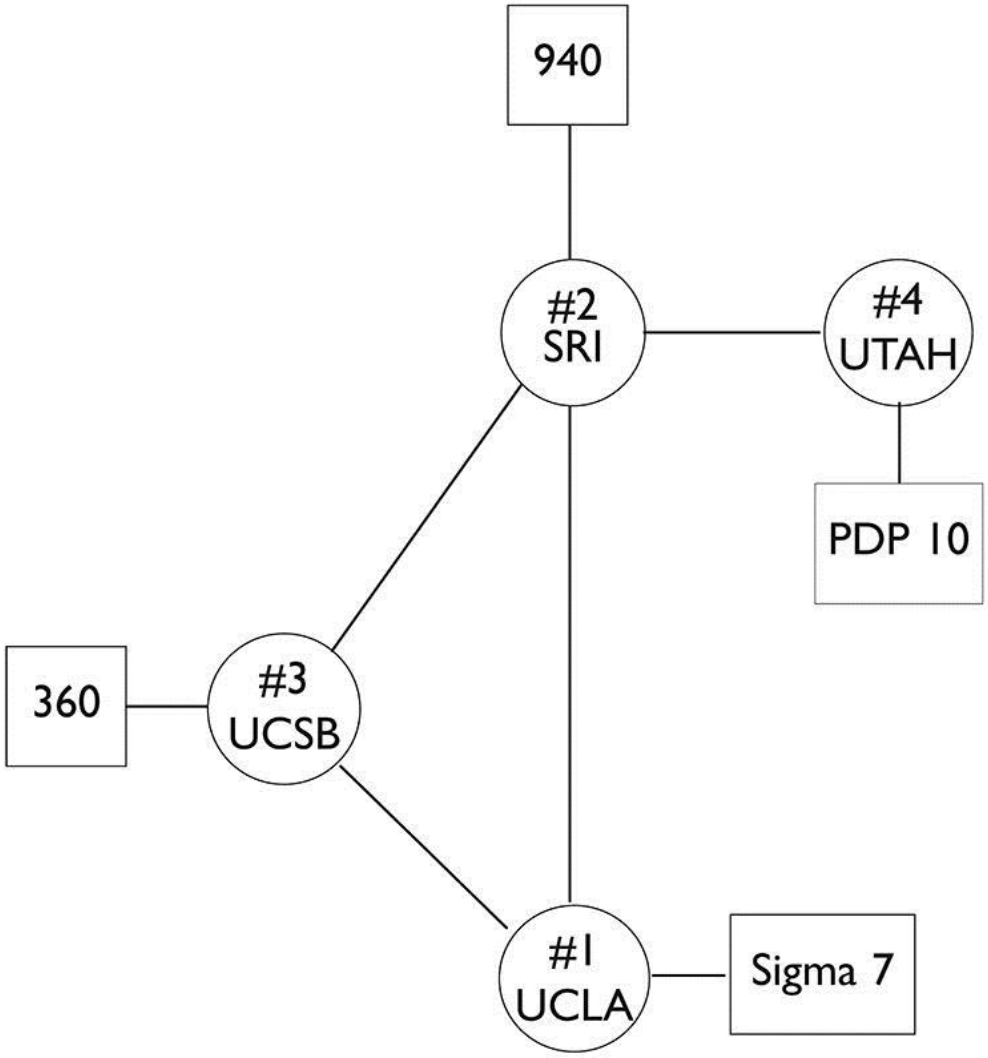
Az internet nem egyetlen találmány eredménye. Sok évtized fejlesztése kellett hozzá: csomagkapcsolás, hálózati protokollok, modemek, szerverek, személyi számítógépek és végül a World Wide Web.

14.1. ARPANET

Az internet egyik legfontosabb elődje az amerikai **ARPANET** volt. Az ARPA – az Egyesült Államok Védelmi Minisztériumának kutatási ügynöksége – támogatásával létrehozott hálózat célja különböző kutatóintézetek számítógépeinek összekapcsolása volt.

1969-ben négy amerikai intézmény számítógépei alkották az első hálózatot.





Ábra: *Az ARPANET első négy csomópontja 1969-ben. A hálózat még néhány számítógépből állt, de működésében már megjelentek a későbbi internet alapelvei.*

Az első kísérlet során a UCLA kutatói a Stanford Research Institute számítógépére próbálták elküldeni a **LOGIN** szót. A rendszer azonban két betű után összeomlott.

Így az internet egyik őse által továbbított első üzenet mindössze ennyi lett:

LO

Nem túl ünnepélyes kezdet egy olyan technológiának, amely később az egész világot behálózta.

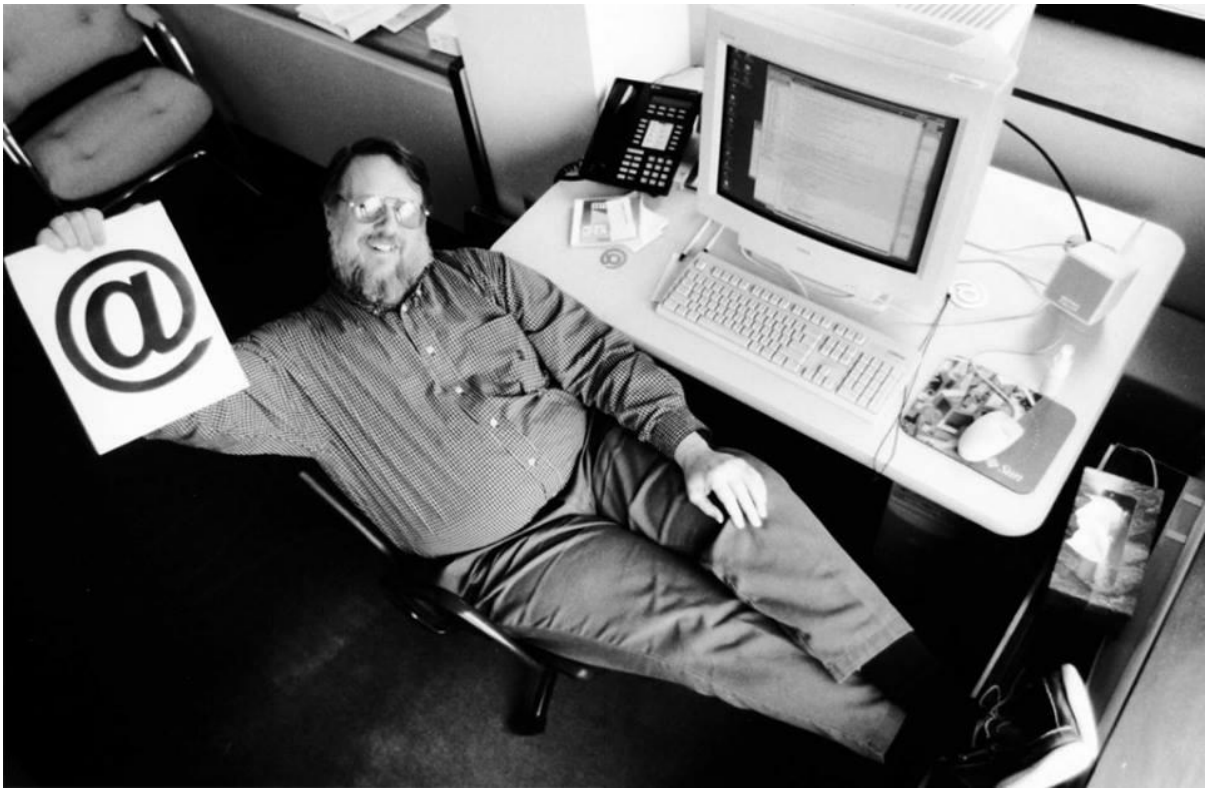
14.2. E-mail

Az elektronikus levelezés meglepően régi: már jóval a világháló megszületése előtt használták.

1971-ben **Ray Tomlinson** olyan rendszert készített, amely lehetővé tette üzenetek küldését hálózatba kapcsolt számítógépek között. Ő vezette be a ma már magától értetődőnek tűnő **@** jel használatát is.

Az elv: **felhasználó @ számítógép**

A @ jel azért volt praktikus, mert az angolban „at”-nek olvassák: **felhasználó – ezen a számítógépen.**



Ábra: *Ray Tomlinson az 1970-es évek elején dolgozta ki a hálózaton keresztüli elektronikus levelezés egyik meghatározó rendszerét. Neki köszönhetjük az e-mail-címek @ jelét is.*

Az e-mail hamarosan a hálózat egyik legfontosabb szolgáltatásává vált – még azelőtt, hogy a legtöbb ember egyáltalán hallott volna az internetről.

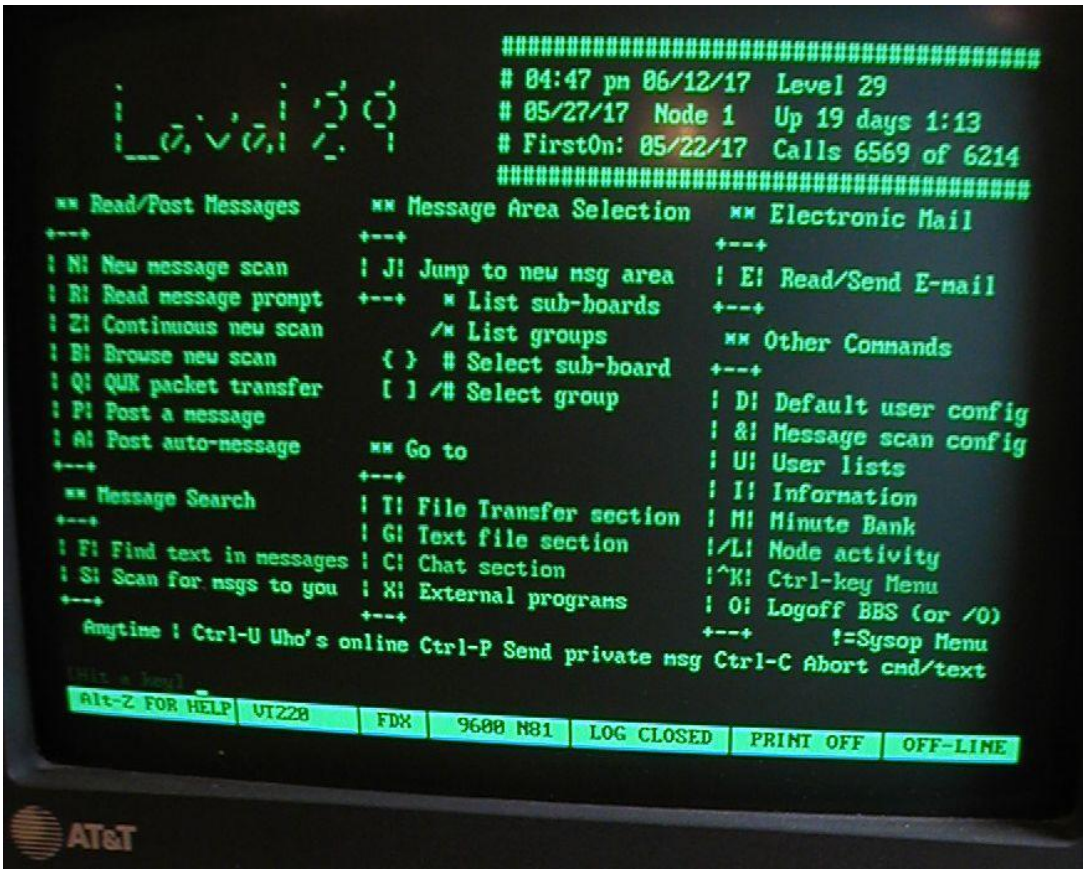
14.3. BBS

A személyi számítógépek és a modemek elterjedésével az otthoni felhasználók is elkezdtek hálózatokat építeni.

A **BBS – Bulletin Board System** olyan számítógép volt, amelyet modemmel, telefonvonalon lehetett felhívni.

A felhasználók üzeneteket hagyhattak egymásnak, fájlokat tölthettek fel és le, beszélgethettek, programokat cserélhettek.







Ábra: *BBS használata az 1980-as években. A felhasználó modemmel felhívta a szolgáltatást működtető számítógépet, majd szöveges menük segítségével üzeneteket és fájlokat érhetett el.*

A BBS-ek tulajdonképpen a későbbi internetes fórumok és közösségek elődei voltak.

14.4. Internet

Az egymástól független számítógép-hálózatok összekapcsolásához közös kommunikációs szabályokra volt szükség.

Ezt a szerepet töltötte be a **TCP/IP protokollcsalád**. Az ARPANET 1983. január 1-jén állt át TCP/IP-re, és ezt a dátumot gyakran az internet történetének egyik fontos határköveként említik.

Az „internet” szó lényegében azt fejezi ki, ami a rendszer legfontosabb tulajdonsága:

interconnected networks – összekapcsolt hálózatok.

Az internet tehát nem egyetlen hálózat, hanem **hálózatok hálózata.**

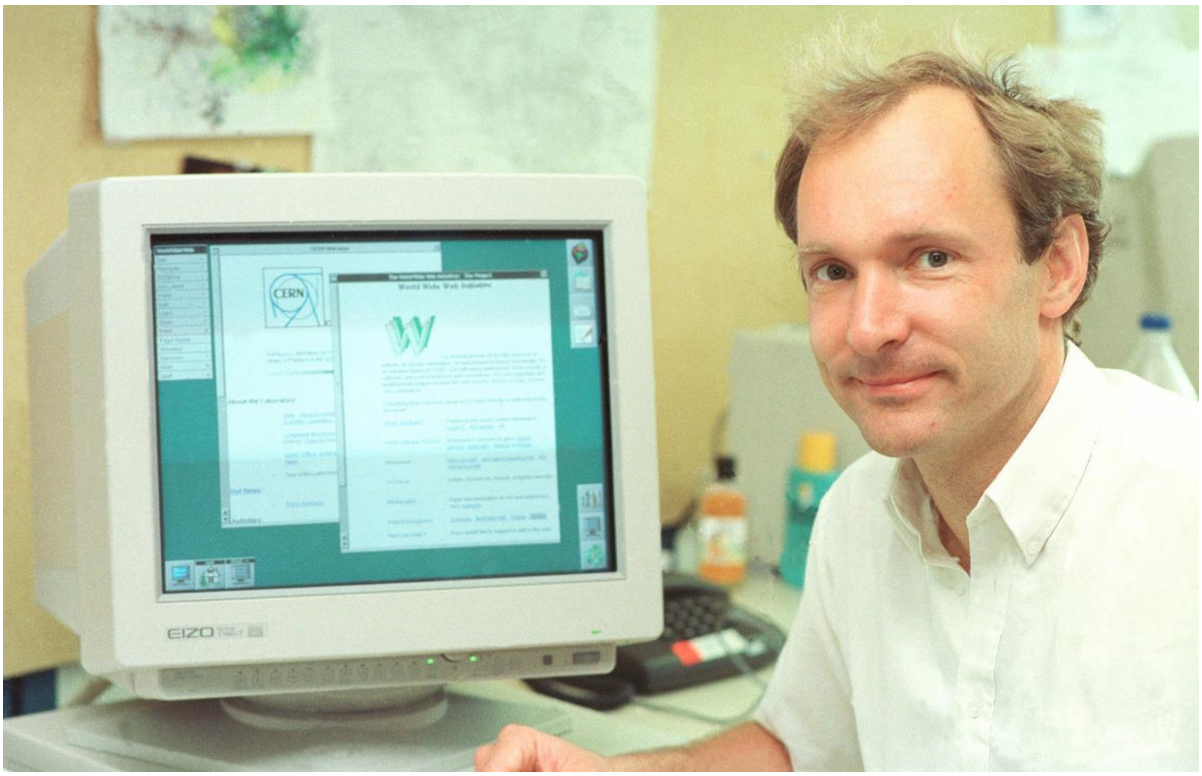
És fontos különbség:

az internet nem azonos a World Wide Webbel.

Az internet az infrastruktúra. A web ennek egyik szolgáltatása – ugyanúgy, ahogy az e-mail is.

14.5. Tim Berners-Lee és a WWW

Az internet már létezett, amikor **Tim Berners-Lee** brit informatikus 1989-ben a CERN-ben olyan rendszert javasolt, amelyben a dokumentumokat hiperhivatkozások kapcsolják össze.



Tim Berners-Lee

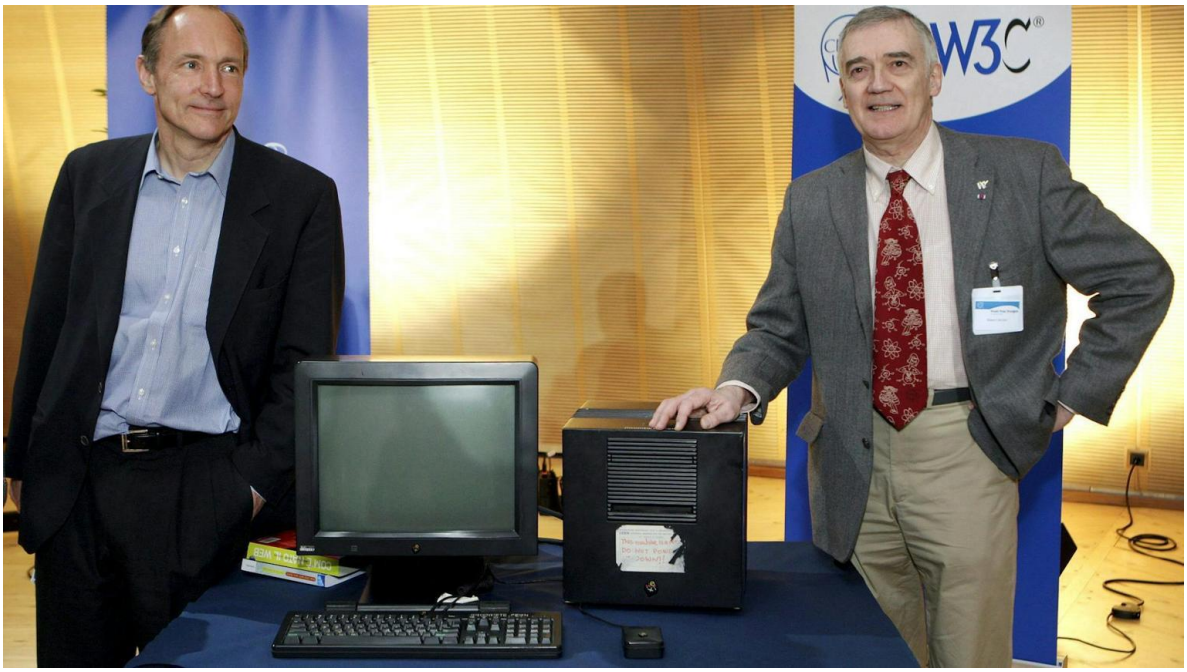
1990-re kialakult a rendszer három alapvető eleme:

HTML – a dokumentumok leírására,

HTTP – a dokumentumok továbbítására,

URL – a hálózati erőforrások címzésére.

Megszületett a **World Wide Web – WWW**, a világháló.



Ábra: *Tim Berners-Lee NeXT számítógépe a CERN-ben. Ezen működött a világ első webszervere. A gépre még figyelmeztetést is ragasztottak: „This machine is a server. DO NOT POWER IT DOWN!!”*

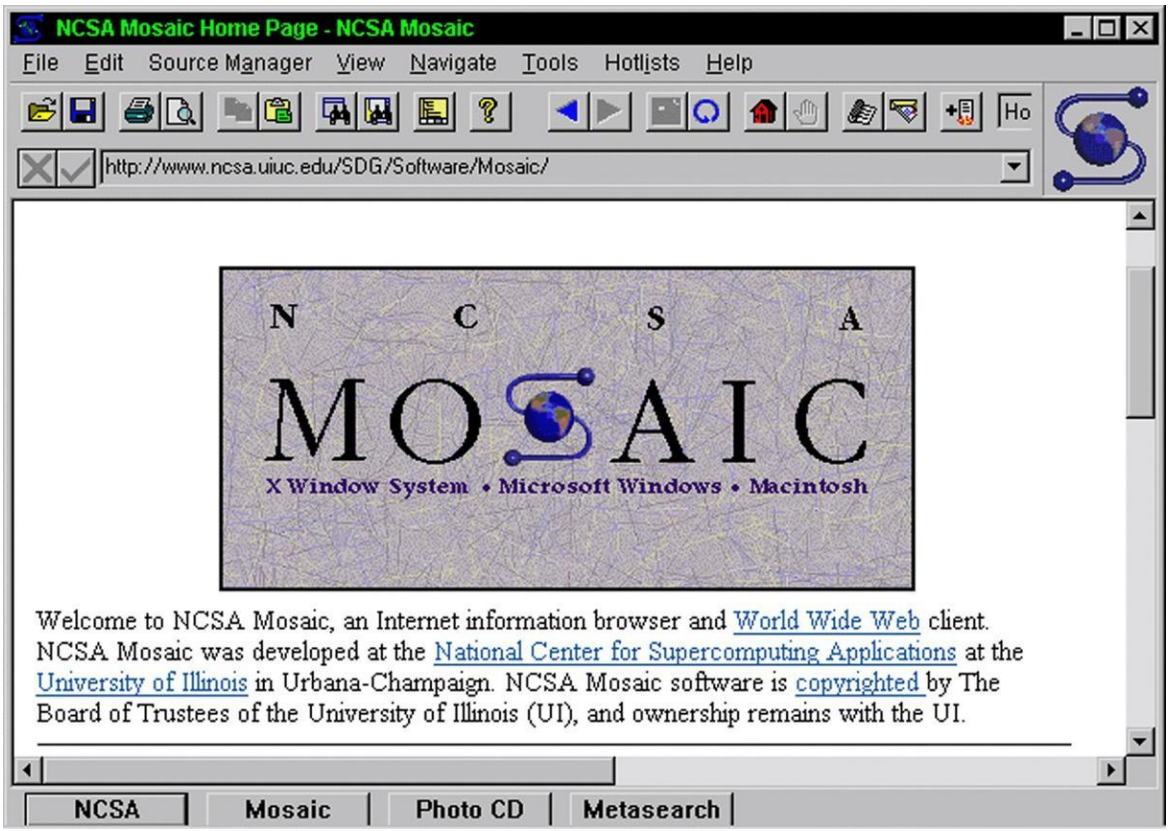
A web tette az internetet igazán könnyen használható, egymással összekapcsolt információs térré.

14.6. Böngészők

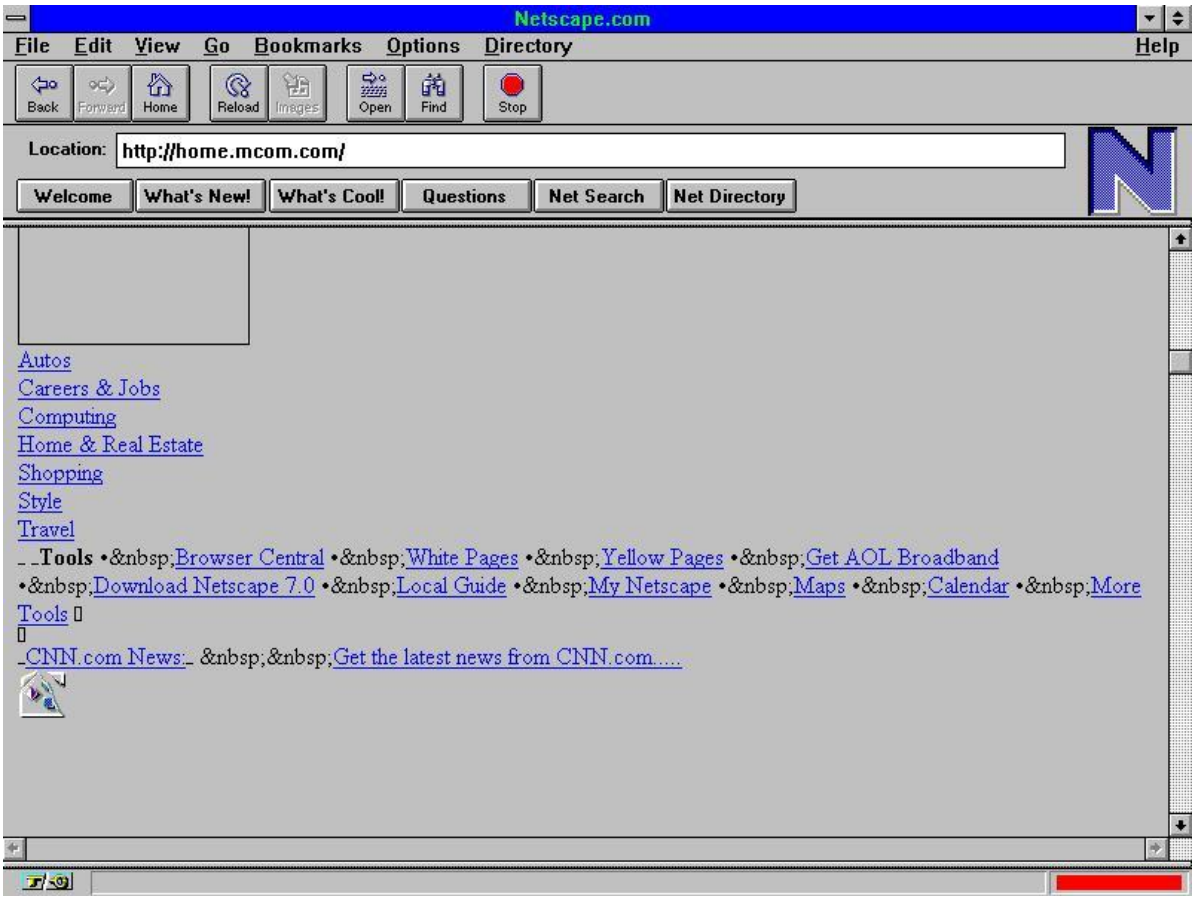
A weboldalak használatához böngészőprogram kellett.

A korai böngészők után az áttörést az 1993-ban megjelent **Mosaic** hozta, amely már képeket is kényelmesen megjelenített a szöveg mellett.

Ezt követte többek között a Netscape Navigator, majd az Internet Explorer, később a Firefox, a Chrome, a Safari és más böngészők.


NCSA Mosaic Home Page - NCSA Mosaic
File Edit Source Manager View Navigate Tools Hotlists Help
http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/
N C S A
MOSAIC
X Window System • Microsoft Windows • Macintosh
Welcome to NCSA Mosaic, an Internet information browser and World Wide Web client.
NCSA Mosaic was developed at the National Center for Supercomputing Applications at the
University of Illinois in Urbana-Champaign. NCSA Mosaic software is copyrighted by The
Board of Trustees of the University of Illinois (UI), and ownership remains with the UI.
NCSA
Mosaic
Photo CD
Metasearch


Netscape - [Welcome to Netscape]
File Edit View Go Bookmarks Options Directory Help
Netsite: http://www.netscape.com/
What's New! What's Cool! Upgrades Net Search Net Directory Newsgroups
WELCOME TO NETSCAPE
NETSCAPE STORE
EXPLORING THE NET
COMPANY & PRODUCTS
NETSCAPE STORE
NEWS & REFERENCE
ASSISTANCE
COMMUNITY
NETSCAPE SERVER GALLERIA
NETSCAPE SECURITY UPGRADE AVAILABLE
Find out how Netscape is providing upgrades to customers and then
download the final software. Users of the expired Navigator 1.2 beta can
also download this upgrade for evaluation. A patch for Netscape Commerce Server and a new
version of Netscape Proxy Server are now available.
NETSCAPE Now!



Ábra: *Az NCSA Mosaic a web első igazán sikeres grafikus böngészőinek egyike volt. A világháló ezzel kezdett kilépni a kutatóintézetek világából a nagyközönség elé.*

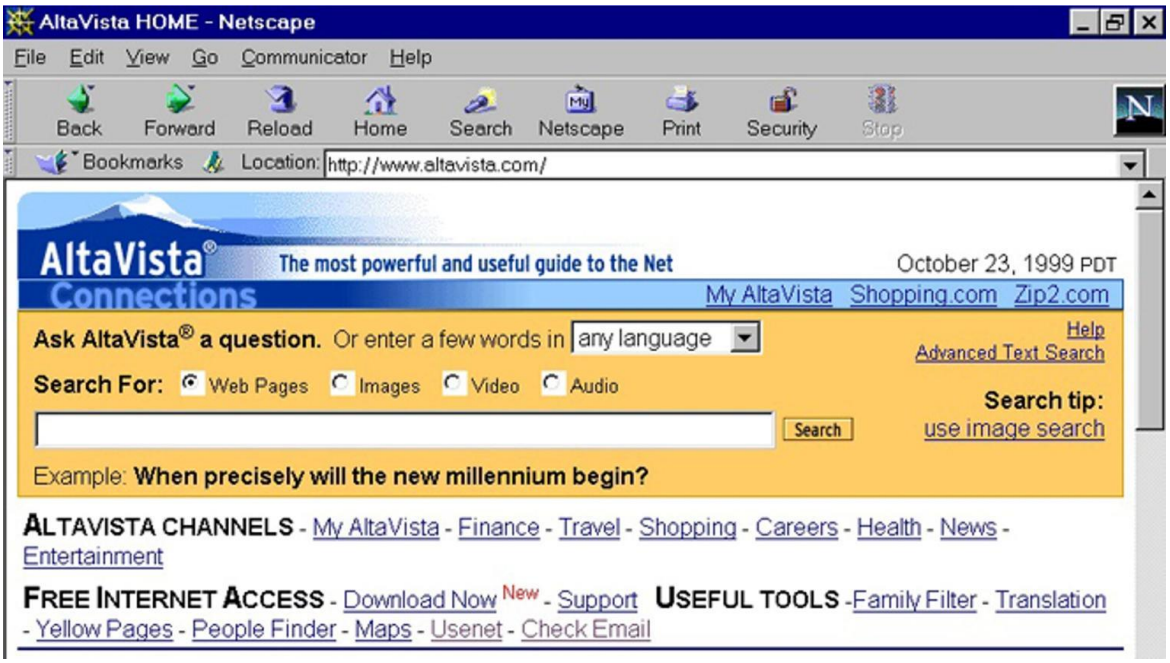
A böngésző idővel szinte **operációs rendszerré az operációs rendszeren belül** vált: ma már levelezünk, dolgozunk, filmet nézünk és programokat futtatunk benne.

14.7. Keresők

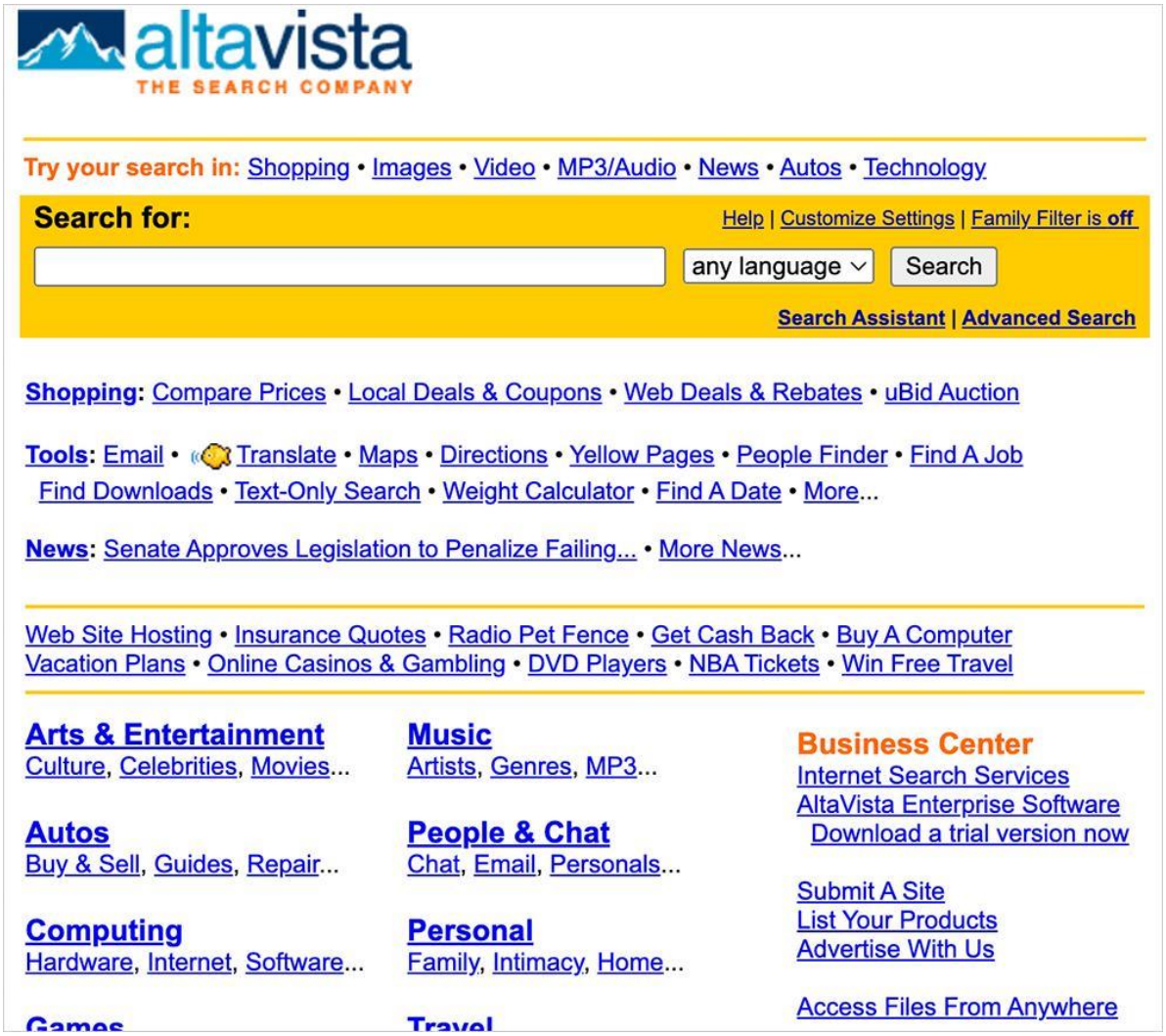
Amíg néhány száz vagy néhány ezer weboldal létezett, különféle katalógusokban még össze lehetett gyűjteni őket.

A web rohamos növekedésével ez lehetetlenné vált.

Megjelentek a keresőprogramok: WebCrawler, Lycos, AltaVista, Yahoo!, majd 1998-ban a Google.


AltaVista HOME - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://www.altavista.com/
AltaVista
Connections
The most powerful and useful guide to the Net
October 23, 1999 PDT
My AltaVista Shopping.com Zip2.com
Ask AltaVista® a question. Or enter a few words in any language
Help
Advanced Text Search
Search For: Web Pages Images Video Audio
Search tip:
use image search
Search
Example: When precisely will the new millennium begin?
ALTAVISTA CHANNELS - My AltaVista - Finance - Travel - Shopping - Careers - Health - News - Entertainment
FREE INTERNET ACCESS - Download Now New - Support USEFUL TOOLS -Family Filter - Translation - Yellow Pages - People Finder - Maps - Usenet - Check Email


Google!
Search the web using Google!
10 results
Google Search
I'm feeling lucky
Index contains ~25 million pages (soon to be much bigger)
About Google!
Stanford Search Linux Search
Get Google! updates monthly!
your e-mail
Subscribe
Archive
Copyright ©1997-8 Stanford University



A keresők automatikusan feltérképezik a webet, adatbázist készítenek a megtalált oldalakról, majd megpróbálják a kérdésünkhöz leginkább kapcsolódó találatokat megmutatni.

Ez legalább akkora változás volt, mint maga a web:

Már nem kellett tudnunk, hol található az információ. Elég volt tudnunk, mit keresünk.

Altavizsla és társai – a magyar keresők hőskora

Ma már szinte magától értetődő, hogy ha valamit meg szeretnénk találni az interneten, egyszerűen megnyitjuk a Google-t. A kilencvenes évek végén azonban korántsem volt ilyen egyértelmű a helyzet.

A világháló rohamosan növekedett, de még nem létezett egyetlen, mindenki által használt kereső. A nemzetközi szolgáltatások – például az AltaVista, a Yahoo! vagy később a Google – mellett számos ország saját keresőket is fejlesztett. Magyarország sem maradt ki ebből a folyamatból.

A magyar internet első éveiben több hazai kereső is működött:

Kereső	Jellemző
Altavizsla	1998-ban indult. Neve játékosan az AltaVista keresőre utalt, a „Vista” helyére a magyar „vizsla” került.
Vizsla	Az Origo keresőszolgáltatása, hosszú ideig az egyik legismertebb magyar kereső.
Heuréka	Önálló magyar fejlesztésű keresőrendszer.
Kurzor	A hazai internet egyik korai keresőszolgáltatása.
Góliát	Magyar webkereső a kétezer novemberében indult.
Ariadnet	Metakereső, amely egyszerre több kereső eredményeit gyűjtötte össze.

Ezek a szolgáltatások elsősorban a magyar weboldalak között segítettek eligazodni. Akkoriban a hazai internet még viszonylag kis méretű volt, ezért egy magyar nyelvre és magyar tartalmakra specializálódott kereső valódi előnyt jelentett.

Az **Altavizsla** különösen emlékezetes maradt játékos neve miatt. Az AltaVista akkoriban a világ egyik legnépszerűbb keresője volt, a magyar változat pedig szellemes szóviccel utalt rá: a „Vista” helyét a magyar vadászkutya, a **vizsla** vette át.

A keresők világa azonban gyorsan változott. A Google új keresési módszere – amely nemcsak a weboldalak tartalmát, hanem az egymásra mutató hivatkozásokat is figyelembe vette – lényegesen pontosabb találatokat adott. Néhány év alatt világszerte átvette a vezető szerepet, és a kisebb nemzeti keresők fokozatosan eltűntek.

A magyar keresők története mégis fontos fejezete a hazai internet fejlődésének. Egy olyan korszakot idéz fel, amikor a magyar web még saját szolgáltatásokat, saját ötleteket és saját megoldásokat keresett.

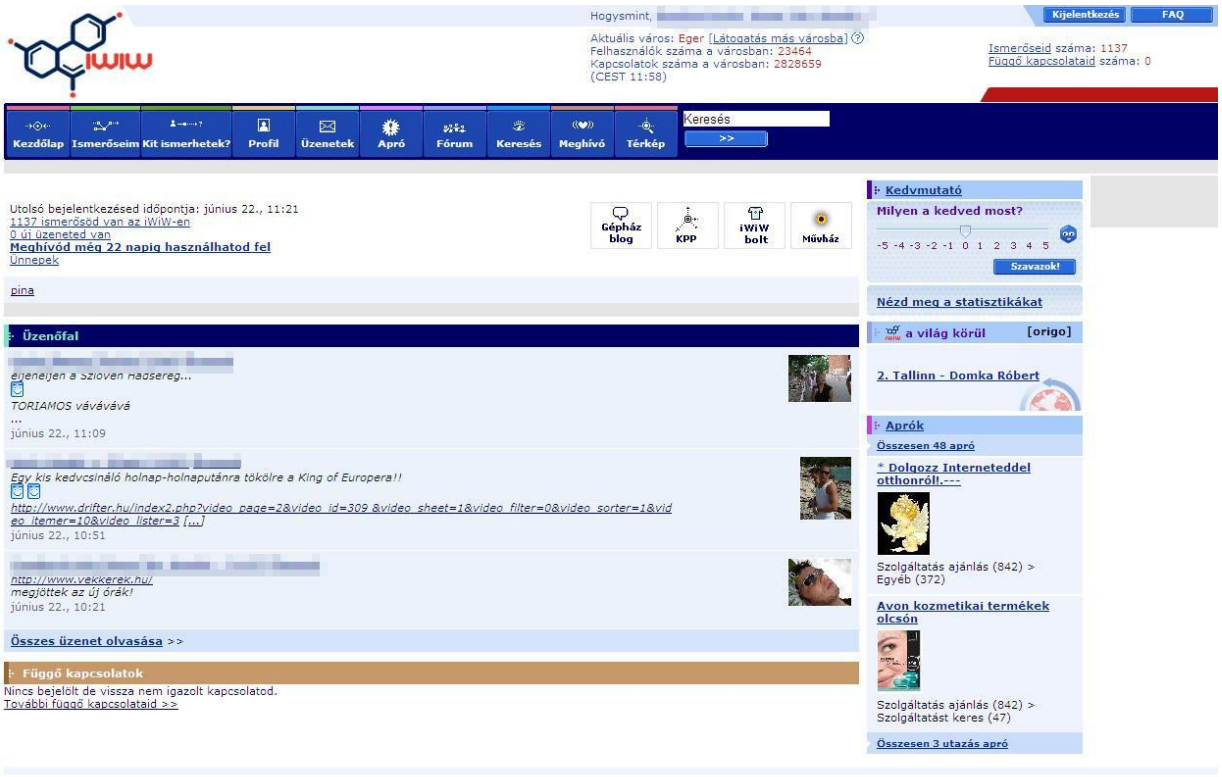
A világháló kezdetén nem egyetlen kapun keresztül léptünk be az internetre. Több magyar „kapuőr” is segített eligazodni a folyamatosan növekvő információs világban.

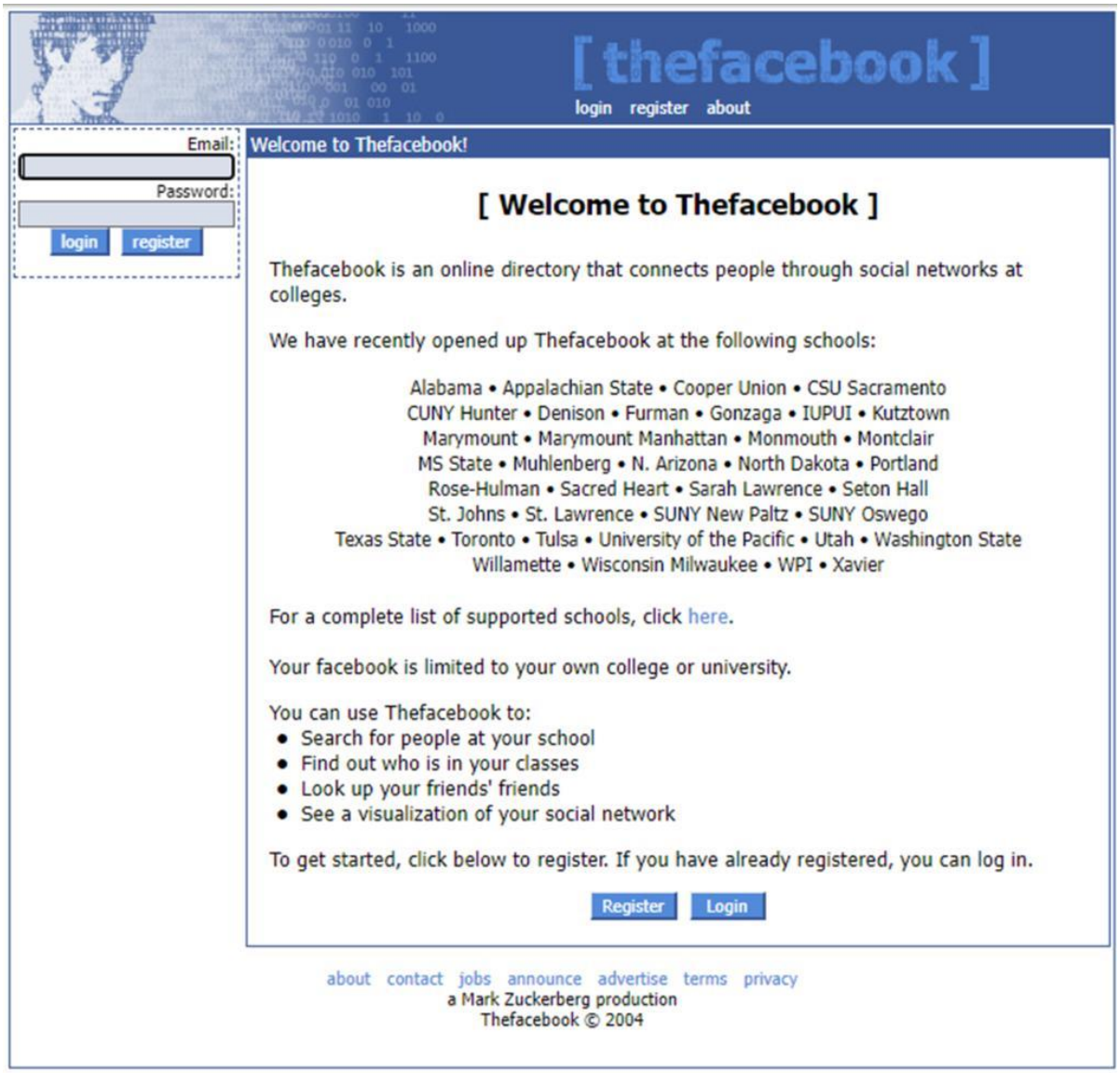
14.8. Közösségi hálózatok

Az internet kezdetben elsősorban számítógépeket kötött össze. A web információkat kapcsolt össze. A közösségi hálózatok pedig **embereket.**

A korai közösségi szolgáltatásokat később olyan rendszerek követték, mint a MySpace, a Facebook, a LinkedIn, a YouTube, a Twitter/X, az Instagram és a TikTok.

És itt van egy fontos magyar történet is: az **iWiW**, amely 2002-ben indult WiW néven, tehát a Facebook előtt.


Hogysmint,
Kijelentkezés
FAQ
Aktuális város: Eger [Látogatás más városba]
Felhasználók száma a városban: 23464
Kapcsolatok száma a városban: 2828659
(CEST 11:58)
Ismerőseid száma: 1137
Függő kapcsolataid száma: 0
Kezdőlap
Ismerőseim
Kit ismerhetek?
Profil
Üzenetek
Apró
Fórum
Keresés
Meghívó
Térkép
Keresés
>>
Utolsó bejelentkezésed időpontja: június 22., 11:21
1137 ismerősöd van az iWiW-en
0 új üzeneted van
Meghívód még 22 napig használhatod fel
Ünnepek
Gépház blog
KPP
iWiW bolt
Művház
pina
Üzenőfal
TORIAMOS vávávává
...
június 22., 11:09
Egy kis kedvcsináló holnap-holnaputánra tökölre a King of Europera!!
http://www.drifter.hu/index2.php?video_page=2&video_id=309 &video_sheet=1&video_filter=0&video_sorter=1&video_itemer=10&video_lister=3 [...]
június 22., 10:51
http://www.vekkerek.hu/
megjöttek az új órák!
június 22., 10:21
Összes üzenet olvasása >>
Függő kapcsolatok
Nincs bejelölt de vissza nem igazolt kapcsolatod.
További függő kapcsolataid >>
Kedvmutató
Milyen a kedved most?
-5 -4 -3 -2 -1 0 1 2 3 4 5
Szavazok!
Nézd meg a statisztikákat
a világ körül
[origo]
2. Tallinn - Domka Róbert
Aprók
Összesen 48 apró
* Dolgozz Interneteddel otthonról!.---
Szolgáltatás ajánlás (842) > Egyéb (372)
Avon kozmetikai termékek olcsón
Szolgáltatás ajánlás (842) > Szolgáltatást keres (47)
Összesen 3 utazás apró


[thefacebook]
login register about
Email:
Password:
login register
Welcome to Thefacebook!
[Welcome to Thefacebook]
Thefacebook is an online directory that connects people through social networks at colleges.
We have recently opened up Thefacebook at the following schools:
Alabama • Appalachian State • Cooper Union • CSU Sacramento
CUNY Hunter • Denison • Furman • Gonzaga • IUPUI • Kutztown
Marymount • Marymount Manhattan • Monmouth • Montclair
MS State • Muhlenberg • N. Arizona • North Dakota • Portland
Rose-Hulman • Sacred Heart • Sarah Lawrence • Seton Hall
St. Johns • St. Lawrence • SUNY New Paltz • SUNY Oswego
Texas State • Toronto • Tulsa • University of the Pacific • Utah • Washington State
Willamette • Wisconsin Milwaukee • WPI • Xavier
For a complete list of supported schools, click here.
Your facebook is limited to your own college or university.
You can use Thefacebook to:
• Search for people at your school
• Find out who is in your classes
• Look up your friends' friends
• See a visualization of your social network
To get started, click below to register. If you have already registered, you can log in.
Register Login
about contact jobs announce advertise terms privacy
a Mark Zuckerberg production
Thefacebook © 2004



Ábra: *A közösségi hálózatok a webet dokumentumok hálózatából emberek hálózatává változtatták. Magyarországon az iWiW már a Facebook világsikere előtt rendkívül népszerűvé vált.*

A közösségi média azonban nemcsak kommunikációs eszköz lett. Átalakította a hírfogyasztást, a reklámot, a politikai kommunikációt és részben társas kapcsolatainkat is.

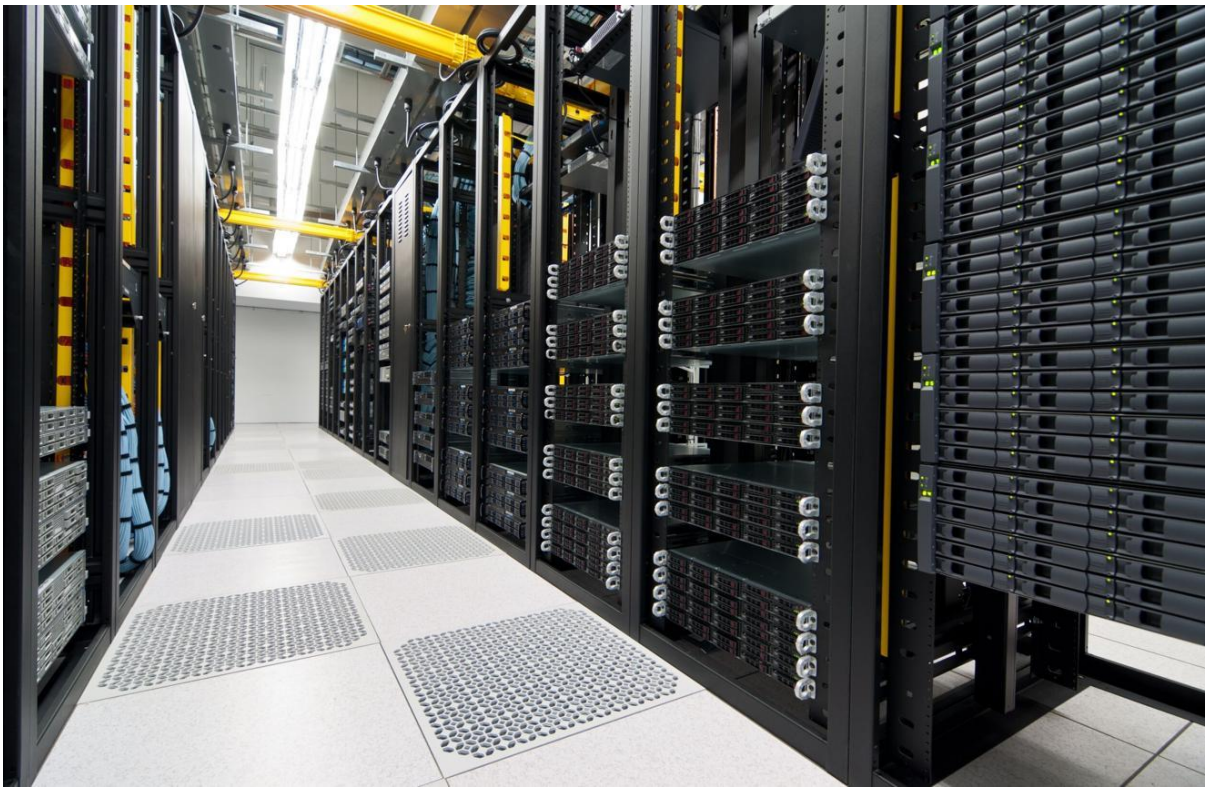
14.9. Felhő

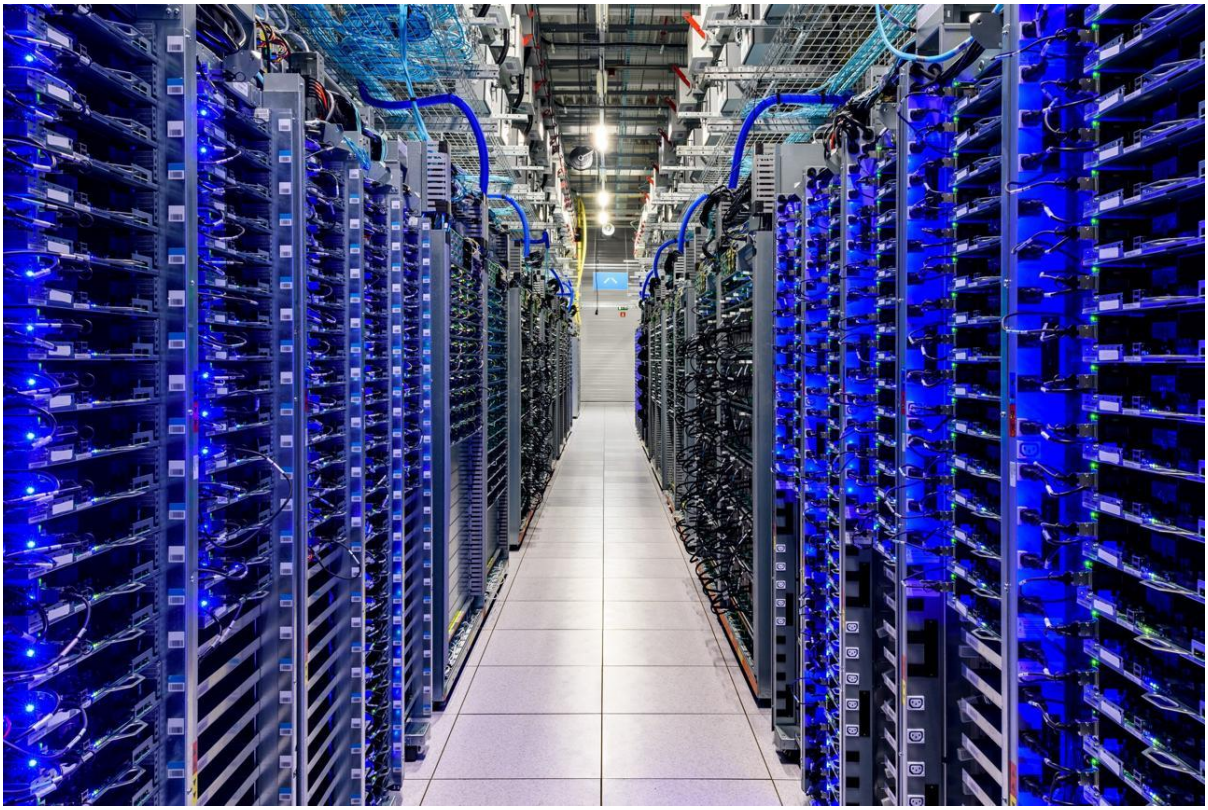
Régen a program a saját számítógépünkön futott, az adat pedig a saját merevlemezünkön volt.

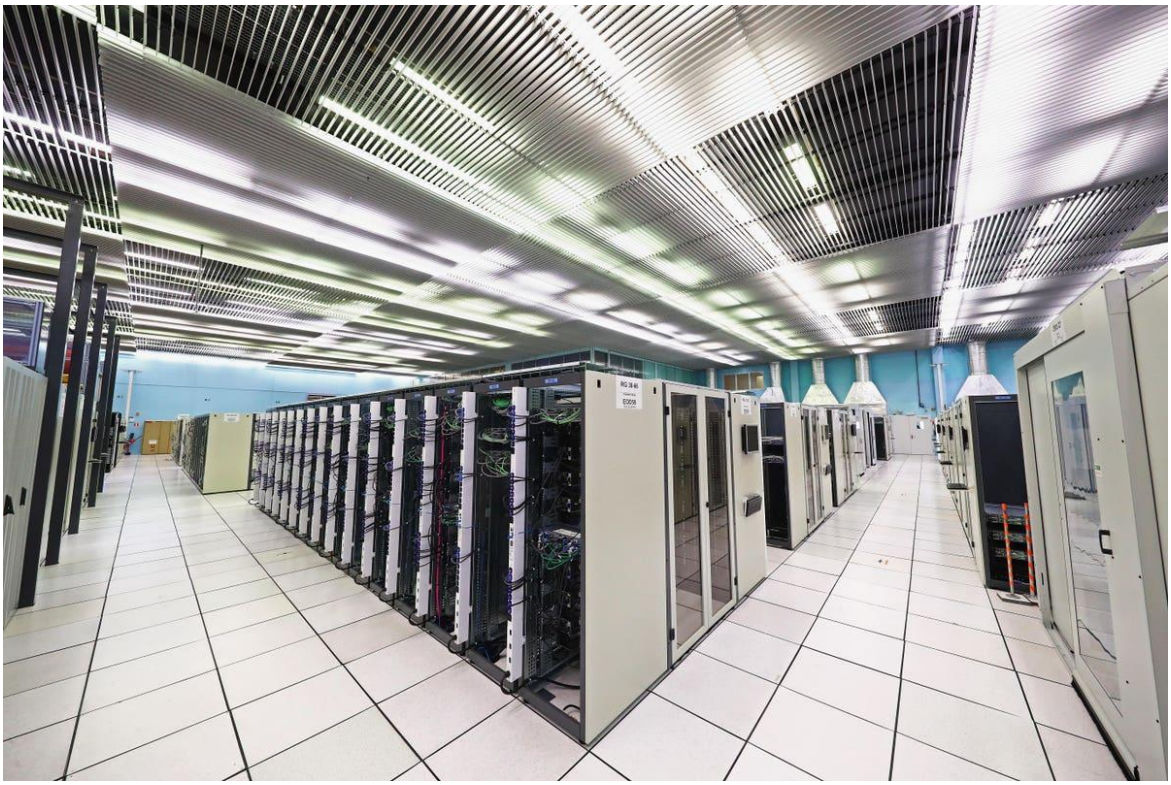
Az internet fejlődésével mindkettő távoli számítógépekre költözhetett.

Ez a **cloud computing – felhőalapú számítástechnika**.

A „felhő” valójában nagyon is kézzelfogható: hatalmas adatközpontokban működő szerverek, háttértárak és hálózati berendezések sokasága.







Ábra: *A „felhő” valójában nagyon is földi: hatalmas adatközpontokban működő számítógépek ezrei szolgálják ki a felhasználókat.*

A felhasználó számára azonban mindegy, hogy pontosan melyik gépen található a fájl vagy fut a program. Elég, ha van hálózati kapcsolat.

Ezzel bizonyos értelemben visszatértünk a nagyszámítógép és terminál világához – csak most **az egész internet lett a vezeték a terminál és a számítógép között.**

14.10. Internet of Things

A hálózatba először nagyszámítógépeket kötöttünk.

Aztán személyi számítógépeket.

Majd telefonokat.

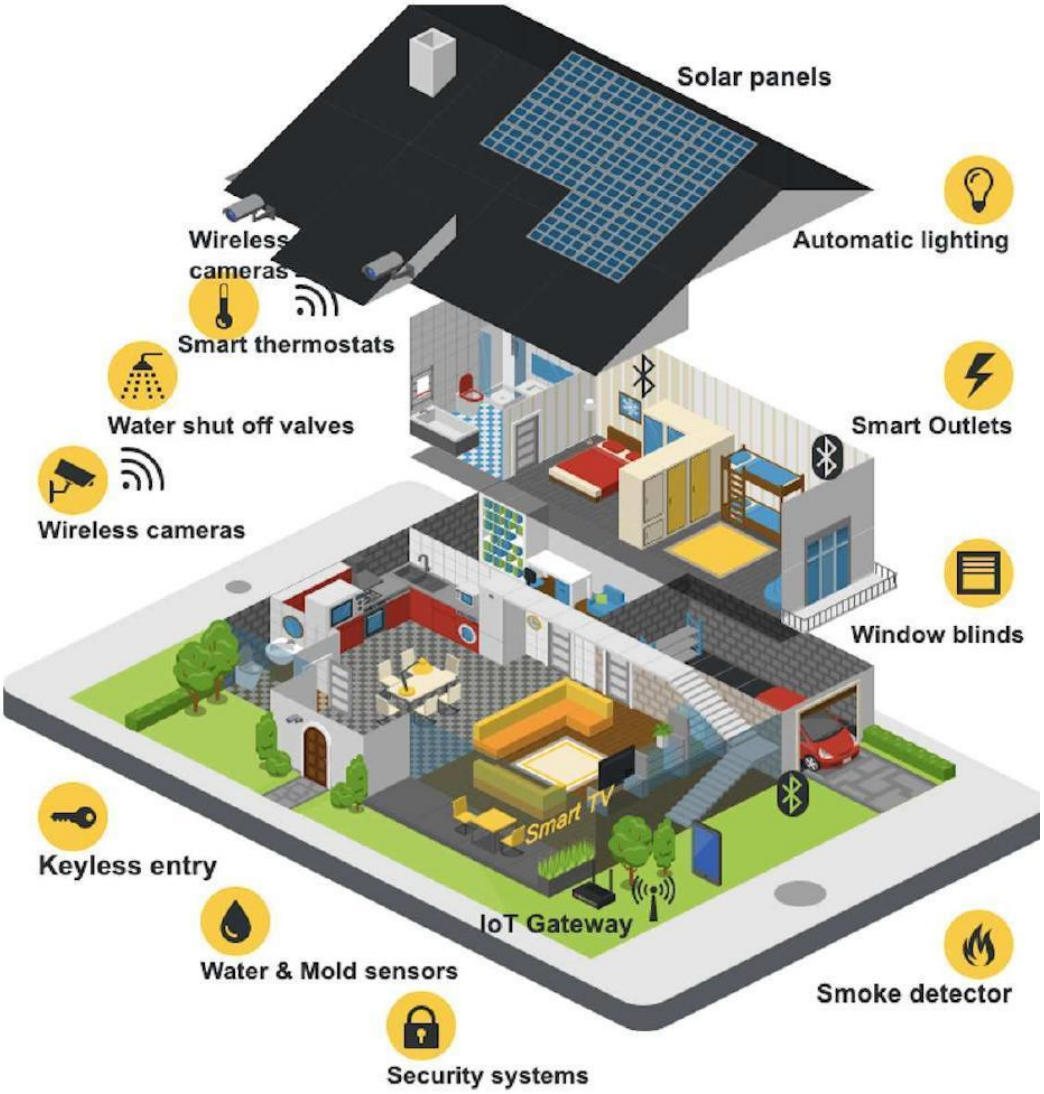
Végül szinte mindent.

Az **Internet of Things – IoT**, magyarul a dolgok internete azt jelenti, hogy hétköznapi eszközök, gépek és érzékelők is hálózati kapcsolatot kapnak.



Lehet ilyen egy termosztát, villanyóra, biztonsági kamera, autó, gyártógép, mezőgazdasági érzékelő vagy akár egy háztartási készülék.



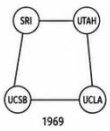


Ábra: *A dolgok internetében már nemcsak számítógépek és telefonok, hanem érzékelők, járművek, háztartási és ipari berendezések is a hálózat részei.*

Ezzel a számítógép-hálózat történetében különös fordulat következett be. Maga a számítógép egyre kevésbé látható: **beépül a tárgyainkba és a környezetünkbe.**

14. INTERNET – AZ INFORMÁCIÓ HÁLÓZATA

14.1. ARPANET



1969-ben indult az **ARPANET**, az első csomagkapcsolt számítógép-hálózat.



14.2. E-mail



1971-ben Ray Tomlinson megalkotta az e-mail rendszert, és bevezette az @ jelet.



14.3. BBS



A BBS-ek a modemes hálózatok világa volt a személyi számítógépek korai közössége.



14.4. Internet



Az 1980-as évekre az egymásba kapcsolt hálózatok hálózata lett: megszületett az Internet.



14.5. WWW



Tim Berners-Lee 1989–90-ben a CERN-ben kifejlesztette a World Wide Web alapjait: HTML, HTTP, URL.



14.6. Böngészők



A Mosaic (1993) és a későbbi böngészők (Netscape, IE, Firefox, Chrome…) elhozták a Webet a nagyközönségnek.



14.7. Keresők



A keresők (AltaVista, Yahoo!, Google…) segítenek eligazodni a hatalmas információ-tömegben.



14.8. Közösségi hálózatok



Az internet a közösségi kapcsolatok új formáit is megteremtette (iWiW, Facebook, YouTube, Twitter, Instagram…).



14.9. Felhő



A felhőalapú szolgáltatások lehetővé tették, hogy adataink és programjaink távoli szervereken legyenek elérhetők.



14.10. Internet of Things



Az IoT-ben már nemcsak számítógépek, hanem eszközök, érzékelők, járművek, háztartási készülékek is kapcsolódnak az internetre.



KERETES: ALTAVIZSLA – A MAGYAR KERESŐ



Az Altavizsla Magyarország első webes keresője volt. 1996-ban indult, magyar nyelvű oldalakat kutatott fel és rendezett kategóriákba. Nevében az AltaVista keresőre utalt, valamint a magyar vizslára. Az Altavizsla fontos szerepet játszott abban, hogy a magyar internethasználók könnyebben megtalálhassák a hazai tartalmakat.



KERETES: AZ INTERNET ÉS A WEB NEM UGYANAZ!

INTERNET

Az internet a világ számítógép-hálózatainak hálózata. Alapja a TCP/IP protokoll. Szolgáltatásai közé tartozik az e-mail, a fájlátvitel, a távoli bejelentkezés, a VoIP stb.

Az internet = infrastruktúra (hálózatok, protokollok)



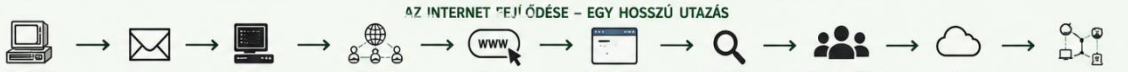
≠

WEB (WWW)

A World Wide Web az interneten működő egyik szolgáltatás, amely hiperhivatkozásokkal összekapcsolt dokumentumokat (oldalakat) érhető el böngészők segítségével.

A web = információs rendszer az interneten

AZ INTERNET FEJLŐDÉSE – EGY HOSSZÚ UTAZÁS



VI. rész – Rabló és pandúr

15. Információbiztonság

Az információ érték. Amióta az ember üzeneteket küld, létezik olyan információ is, amelyet csak a címzettnek szán. Hadvezérek, uralkodók, diplomaták és kémek évezredek óta próbálják úgy továbbítani üzeneteiket, hogy az ellenség ne értse meg őket.

A számítógépek és az internet megjelenésével ez mindennapi problémává vált. Banki műveleteink, levelezésünk, jelszavaink, dokumentumaink és személyes adataink folyamatosan számítógépeken és hálózatokon haladnak keresztül.

A titkosítás története ezért különös folytonosságot mutat:

ugyanazt a problémát próbáljuk megoldani több ezer éve – csak egyre bonyolultabb eszközökkel.

15.1. Az ókori titkosírás

A titkosítás egyik legegyszerűbb módszere, ha az üzenet betűit valamilyen szabály szerint más betűkre cseréljük.

A legismertebb ókori példa a **Caesar-rejtjel**. Julius Caesar állítólag olyan helyettesítő írást használt katonai üzeneteiben, amelyben az ábécé minden betűjét néhány hellyel eltolták.

Például három betűs eltolással:

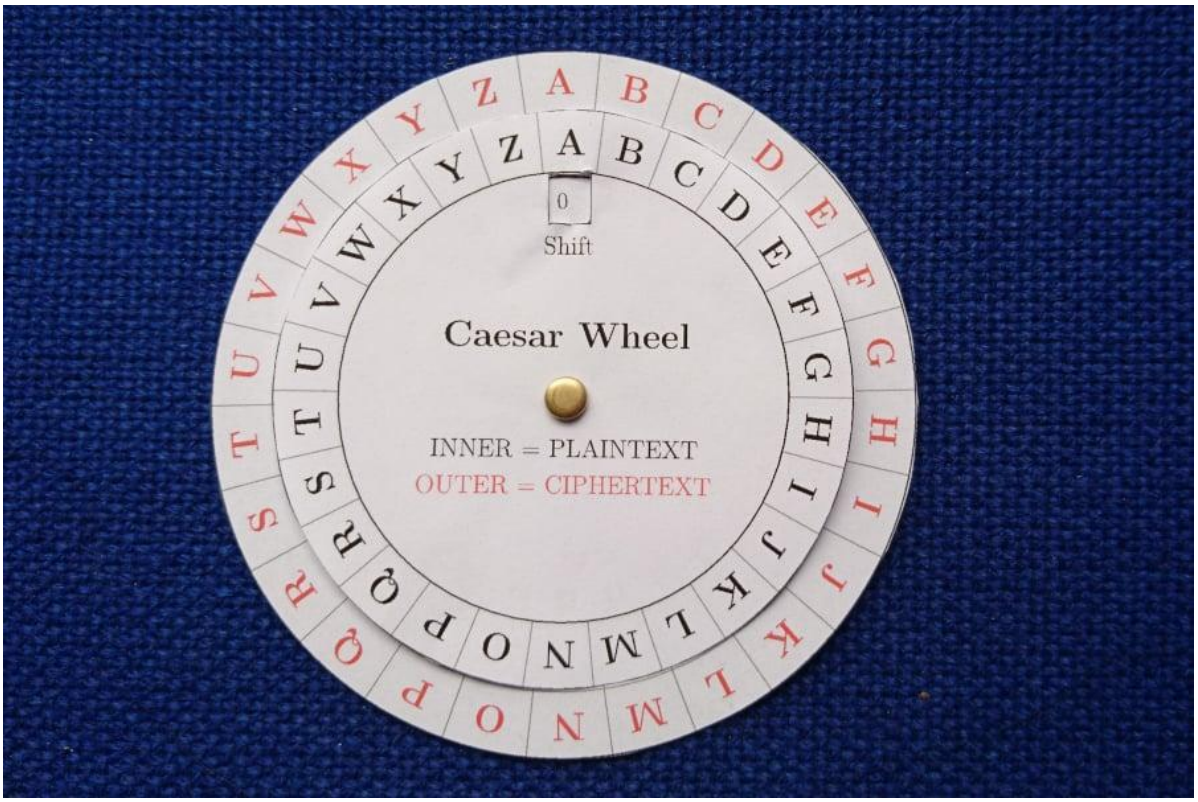
A → D
B → E
C → F

Így például:

ROMA → URPD


A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z





Ábra: *A Caesar-rejtjel modern szemléltetése titkosító tárcsával. A betűk eltolásával az eredeti szöveg olvashatatlanná válik annak, aki nem ismeri a szabályt.*

A görögök ennél érdekesebb mechanikus megoldást is ismertek, a **szkütalét**. Egy bőrszalagot megfelelő átmérőjű botra tekertek, és arra írták az üzenetet. Levéve a szalag betűi értelmetlen sorrendbe kerültek; az üzenetet egy ugyanolyan átmérőjű botra tekerve lehetett ismét elolvasni.

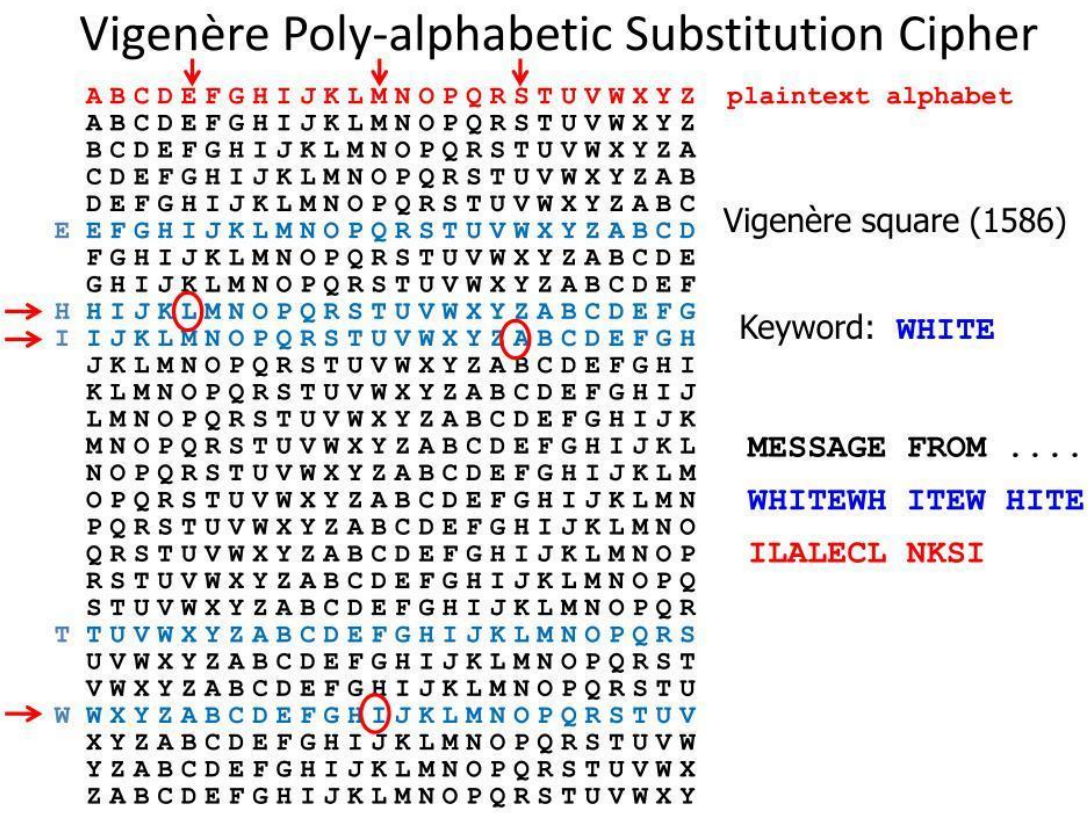
15.2. Vigenère

Az egyszerű betűhelyettesítés nagy gyengesége, hogy a nyelvekben bizonyos betűk gyakrabban fordulnak elő másoknál. Ha valaki elegendően hosszú titkosított szöveget szerez meg, a betűgyakoriság alapján következtethet az eredeti szövegre.

A polialfabetikus titkosítás ezt próbálta megnehezíteni.

A **Vigenère-rejtjelben** nem mindig ugyanazzal az eltolással helyettesítjük a betűket. Az eltolást egy ismétlődő kulcsszó határozza meg.

A	a	b	c	d	e	f	g	h	i	l
B	m	n	o	p	q	r	s	t	u	x
C	a	b	c	d	e	f	g	h	i	l
D	x	m	n	o	p	q	r	s	t	u
E	a	b	c	d	e	f	g	h	i	l
F	u	x	m	n	o	p	q	r	s	t
G	a	b	c	d	e	f	g	h	i	l
H	t	u	x	m	n	o	p	q	r	s
I	a	b	c	d	e	f	g	h	i	l
L	s	t	u	x	m	n	o	p	q	r
M	a	b	c	d	e	f	g	h	i	l
N	r	s	t	u	x	m	n	o	p	q
O	a	b	c	d	e	f	g	h	i	l
P	q	r	s	t	u	x	m	n	o	p
Q	a	b	c	d	e	f	g	h	i	l
R	p	q	r	s	t.	u	x	m	n	o
S	a	b	c	d	e	f	g	h	i	l
T	o	p	q	r	s	t	u	x	m	n
V	a	b	c	d	e	f	g	h	i	l
X	n	o	p	q	r	s	t	u	x	m



Ábra: *A Vigenère-tábla. A titkosításhoz az eredeti szöveg betűjét és a kulcsszó megfelelő betűjét kellett megkeresni a táblázatban.*

A módszert hosszú ideig rendkívül nehezen feltörhetőnek tartották, ezért később a „le chiffre indéchiffrable” – **a megfejthetetlen rejtjel** – elnevezést is kapta.

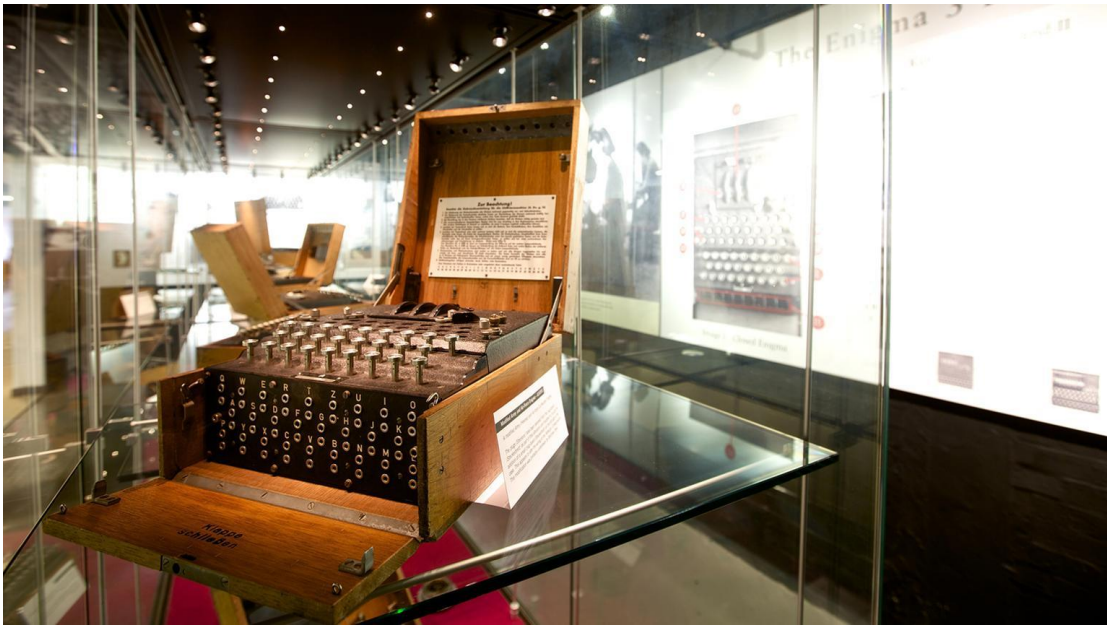
Megfejthetetlen azonban ez sem volt.

15.3. Enigma és a navahó kódbeszélők

A 20. század háborúiban a titkosítás már ipari méretűvé vált.

A legismertebb titkosítógépek egyike a német **Enigma** volt. Billentyűzet, forgó rotorok, kapcsolótábla és lámpák segítségével rendkívül sokféle helyettesítést tudott létrehozni.





Ábra: *Német Enigma titkosítógép. A rotorok minden leütés után megváltoztatták a helyettesítés szabályát, ezért ugyanaz a betű a szöveg különböző helyein más-más betűvé alakulhatott.*



Az Enigma feltörésében lengyel matematikusok – különösen Marian Rejewski, Jerzy Różycki és Henryk Zygalski – alapozták meg a módszereket, majd a brit Bletchley Parkban Alan Turing, Gordon Welchman és sok más szakember folytatta a munkát.

Ugyanebben a háborúban az Egyesült Államok egészen más módszert alkalmazott.

A csendes-óceáni hadszíntéren **navahó indián rádiósok** továbbítottak katonai üzeneteket saját nyelvükre épülő speciális kódrendszer segítségével.



Ábra: *Navahó kódbeszélők a második világháborúban. A ritka nyelvre épített különleges katonai kódrendszer gyors és hatékony kommunikációt tett lehetővé.*

Az Enigma és a kódbeszélők jól mutatják: **titkosítani nemcsak matematikával lehet – akár egy ritka emberi nyelv is a rendszer része lehet.**

15.4. DES

A számítógépek elterjedésével már nem kézzel vagy mechanikus gépekkel kellett titkosítani: maga a számítógép végezhette el a műveleteket.

Az 1970-es években az IBM fejlesztése alapján létrejött a **DES – Data Encryption Standard**, amelyet 1977-ben amerikai szövetségi szabványként fogadtak el.

A DES 56 bites kulcsot használt.

Ez akkor hatalmasnak tűnő számú lehetőséget jelentett:

$2^{56} \approx$ 72 000 000 000 000 000 kulcs.

A számítógépek gyorsulásával azonban ez már kevésnek bizonyult. 1998-ban az Electronic Frontier Foundation speciális DES-törő gépe kevesebb mint három nap alatt megtalált egy DES-kulcsot.

A történet fontos tanulsága:

A titkosítás biztonsága nem örök. Ami ma feltörhetetlennek látszik, a holnap számítógépével már könnyen feltörhető lehet.

15.5. Diffie–Hellman

A hagyományos titkosításnak volt egy alapvető problémája.

Ha Alice titkosított üzenetet akar küldeni Bobnak, mindketten ugyanazt a titkos kulcsot használhatják.

De **hogyan juttassák el egymásnak biztonságosan a kulcsot?**

Ha már lenne biztonságos csatornájuk a kulcs átadására, azon akár magát az üzenetet is elküldhetnék.

Whitfield Diffie és Martin Hellman 1976-ban olyan módszert publikált, amely forradalmi megoldást kínált a problémára.

A két fél **nyilvános csatornán keresztül is létrehozhat közös titkot anélkül, hogy magát a titkot elküldenék egymásnak.**

Ez a Diffie–Hellman-kulcscsere.

A mögötte álló matematika egyik fontos tulajdonsága, hogy bizonyos műveleteket könnyű az egyik irányban elvégezni, de rendkívül nehéz visszafelé kiszámítani.

Ez nyitotta meg az utat a modern nyilvános kulcsú kriptográfia előtt.

15.6. RSA

1977-ben **Ron Rivest, Adi Shamir és Leonard Adleman** kidolgozta az RSA algoritmust.

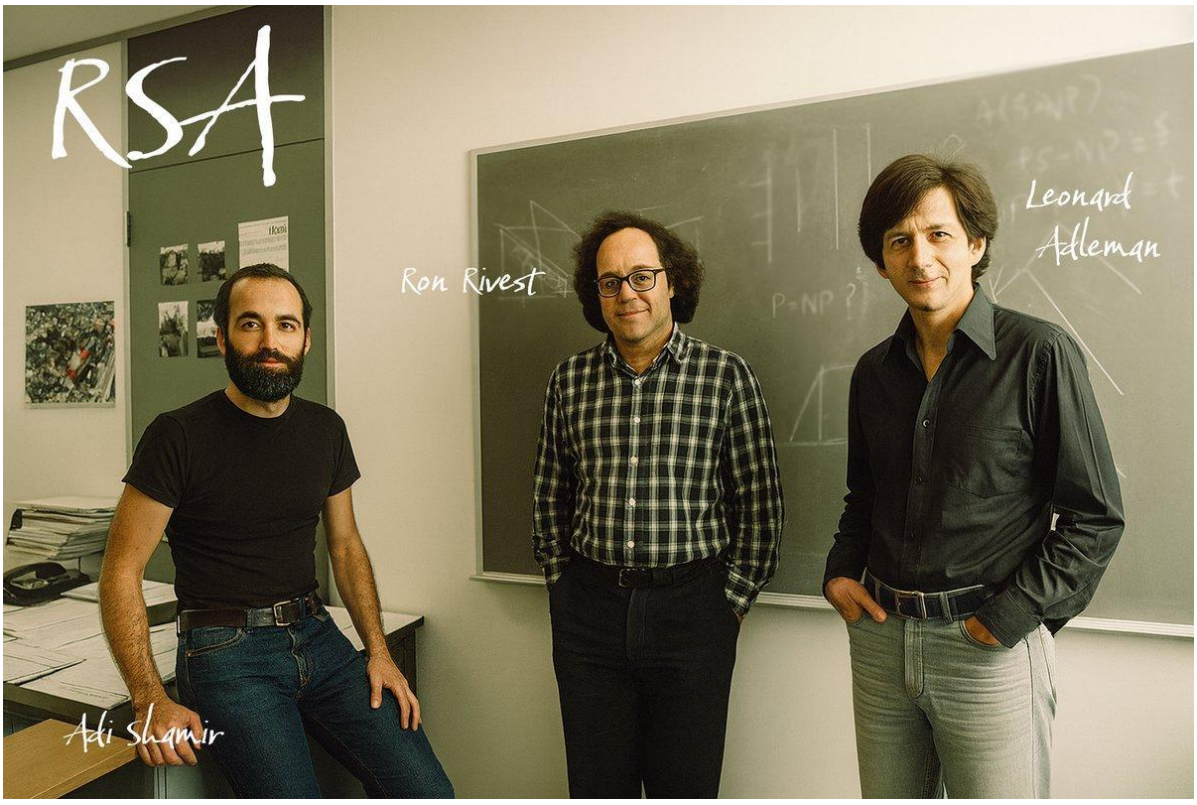
A név vezetéknevük kezdőbetűiből származik:

Rivest – Shamir – Adleman → RSA

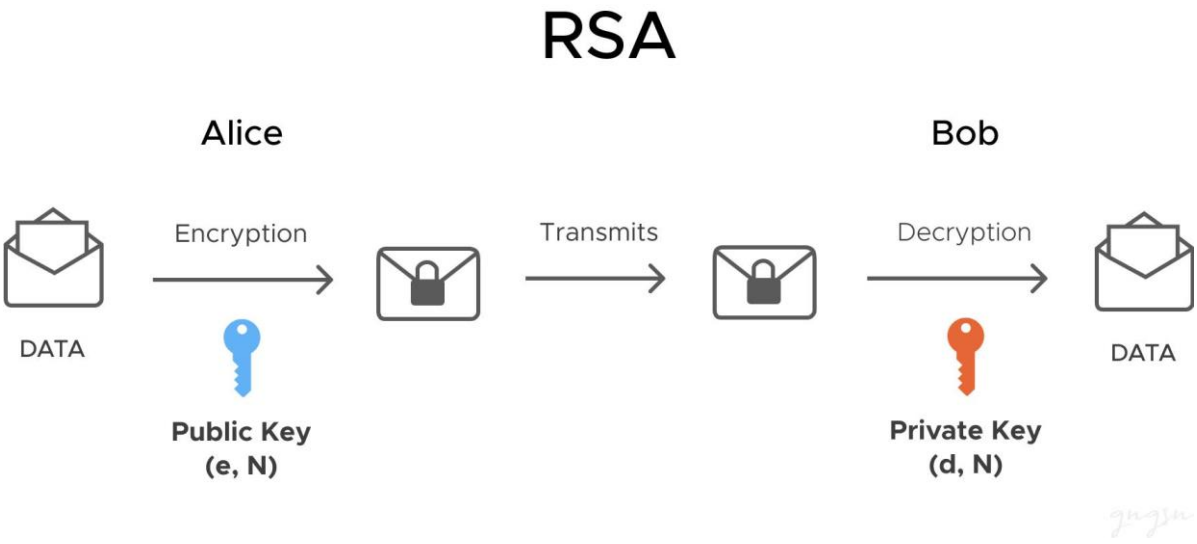
Az RSA különleges tulajdonsága, hogy két összetartozó kulcsot használ.

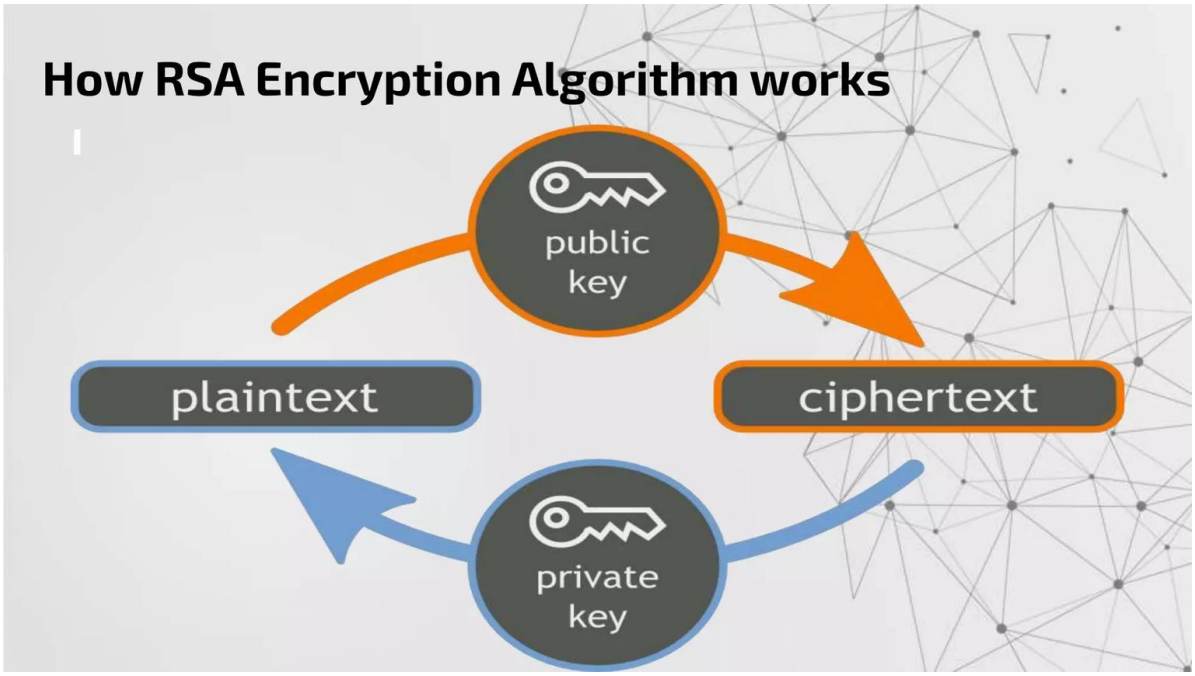
Nyilvános kulcs – bárki ismerheti.

Titkos kulcs – csak a tulajdonos ismeri.



Ábra: *Ron Rivest, Adi Shamir és Leonard Adleman nevének kezdőbetűiből született az RSA elnevezés. Algoritmusuk a nyilvános kulcsú kriptográfia egyik legismertebb megoldása lett.*





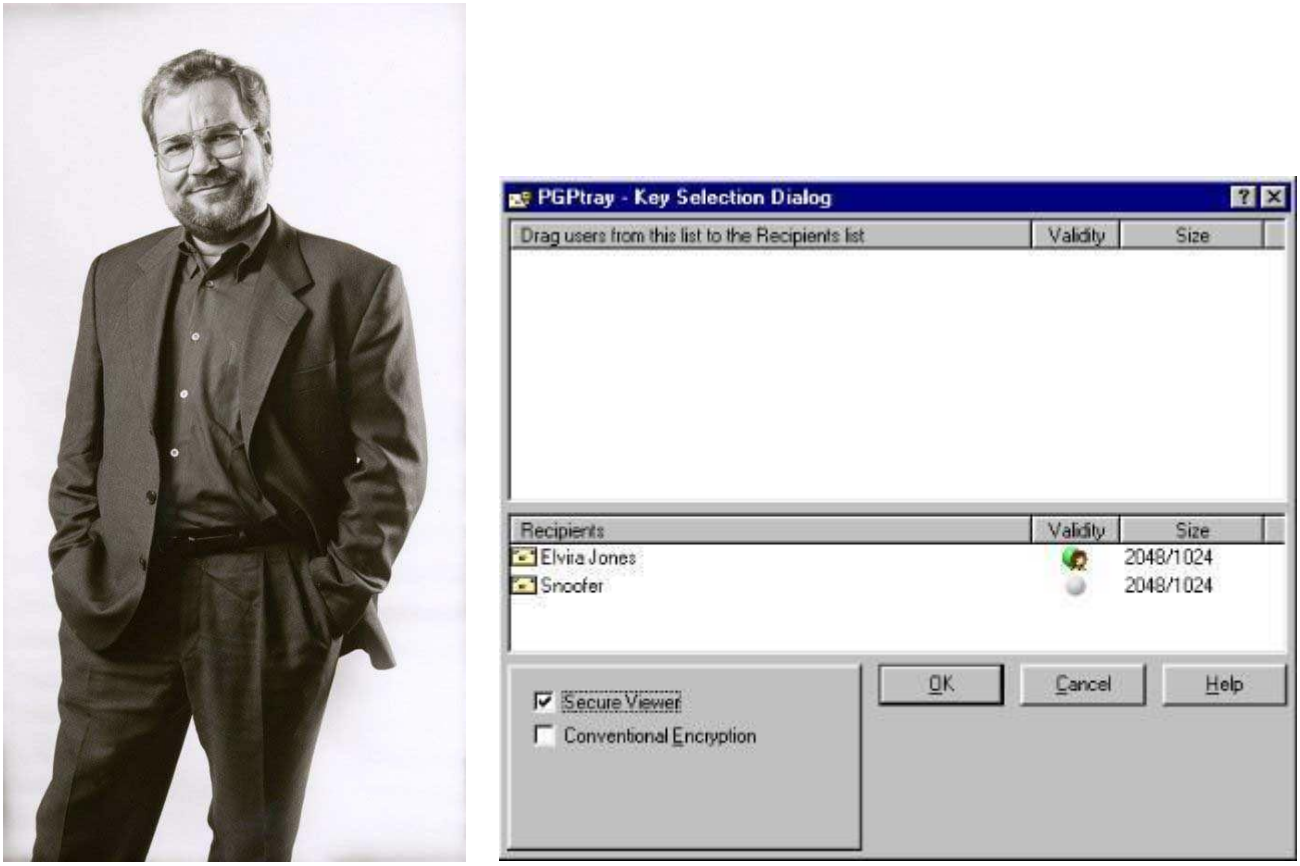
A módszer biztonsága klasszikusan a nagy egész számok prímtényezőkre bontásának számítási nehézségére épül.

15.7. PGP – Phil Zimmermann

A korszerű titkosítás sokáig főként kormányok, katonai szervezetek, bankok és nagyvállalatok eszköze volt.

Phil Zimmermann ezen akart változtatni.

1991-ben elkészítette a **PGP – Pretty Good Privacy** programot, amellyel hétköznapi számítógép-felhasználók is erős titkosítást alkalmazhattak például elektronikus leveleikre és fájljaikra.



Ábra: *Phil Zimmermann és a PGP a korszerű titkosítást a személyi számítógépek felhasználói számára is elérhetővé tette.*

A PGP körül komoly jogi és politikai vita bontakozott ki. Az erős titkosítás akkoriban az Egyesült Államok exportszabályozása alá esett, és Zimmermann ellen vizsgálat is indult.

A PGP története ezért már nem pusztán számítástechnika. Egy ma is aktuális kérdést vetett fel:

Kinek van joga olyan titkosítást használni, amelyet még az állam sem tud egyszerűen feltörni?

15.8. Nyilvános és titkos kulcs

Ez első hallásra nehezen érthető, ezért érdemes egy hétköznapi hasonlattal elképzelni.

Alice készít egy **nyitott lakatot**, és bárkinek adhat belőle.

Bob egy dobozba teszi az üzenetet, majd Alice nyilvánosan hozzáférhető lakatjával lezárja.

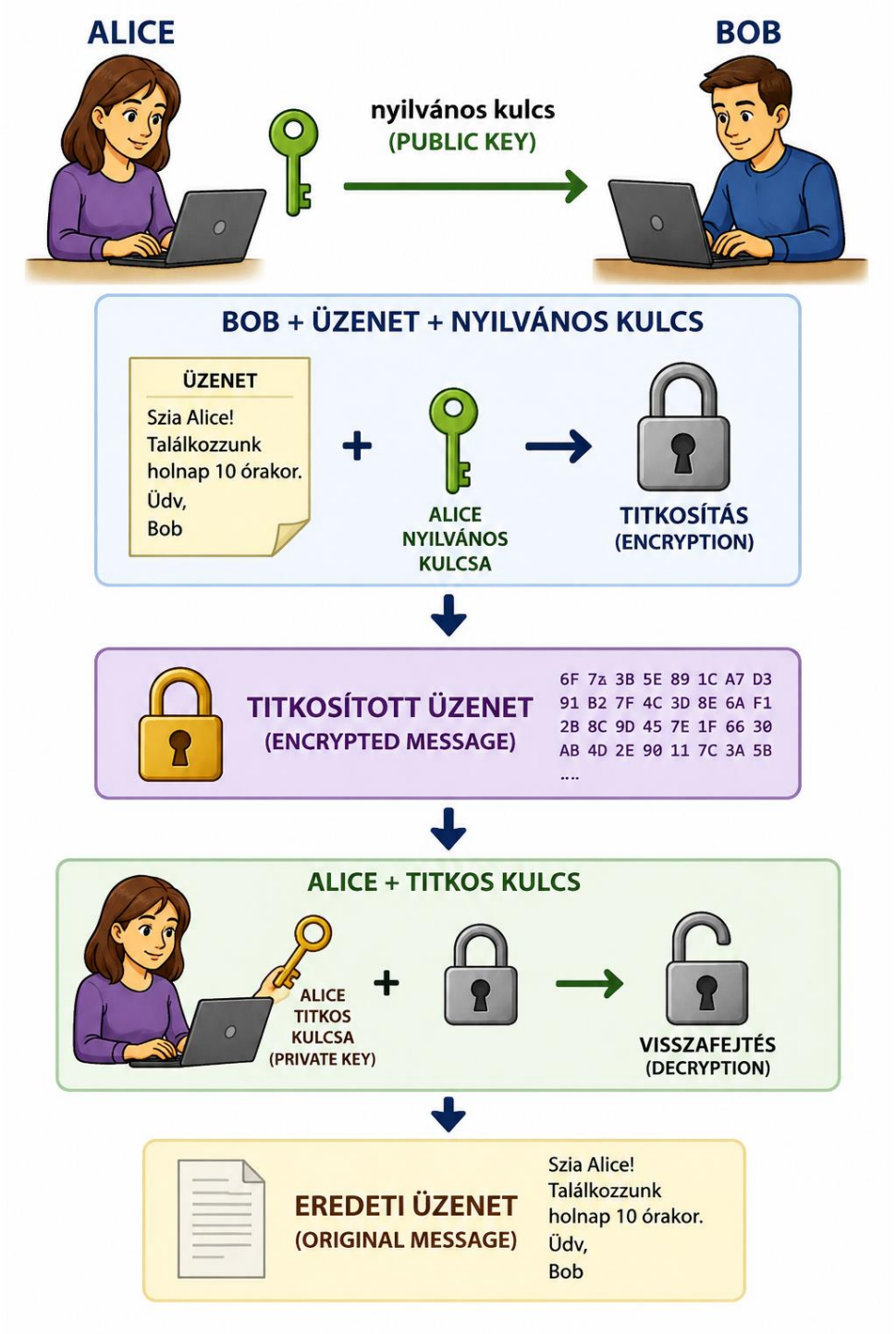
A lezárt dobozt azonban már csak Alice saját kulcsa nyitja.

Nyilvános kulcs → bezárhatjuk vele a titkot.

Titkos kulcs → csak ezzel lehet kinyitni.

Ez az aszimmetrikus kriptográfia alapgondolata.

A valóságban természetesen nem lakatokat, hanem matematikai műveleteket használunk.



15.9. Digitális aláírás

A titkosítás megakadályozhatja, hogy illetéktelenek elolvassák az üzenetet.

De honnan tudjuk, hogy valóban attól érkezett, aki feladóként szerepel?

És honnan tudjuk, hogy útközben nem változtatták meg?

Erre szolgál a **digitális aláírás**.

A dokumentumból először egy rövid matematikai lenyomat – **hash** – készül. Ezt a feladó a saját titkos kulcsához kötött kriptográfiai művelettel aláírja.

A címzett a nyilvános kulcs segítségével ellenőrizheti az aláírást.

Ezzel két fontos dolgot vizsgálhat:

Ki írta alá?

Megváltozott-e azóta?

A digitális aláírás tehát nem egy kézzel írt aláírás beszkennelt képe, hanem **matematikai bizonyíték**.

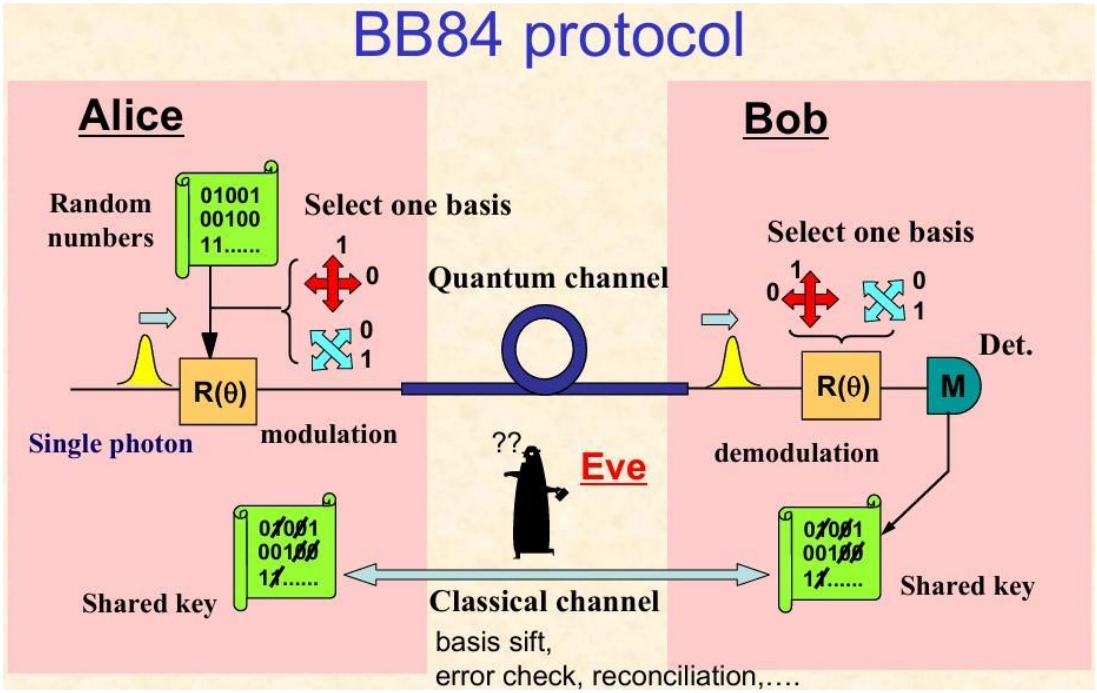
15.10. Kvantumkriptográfia

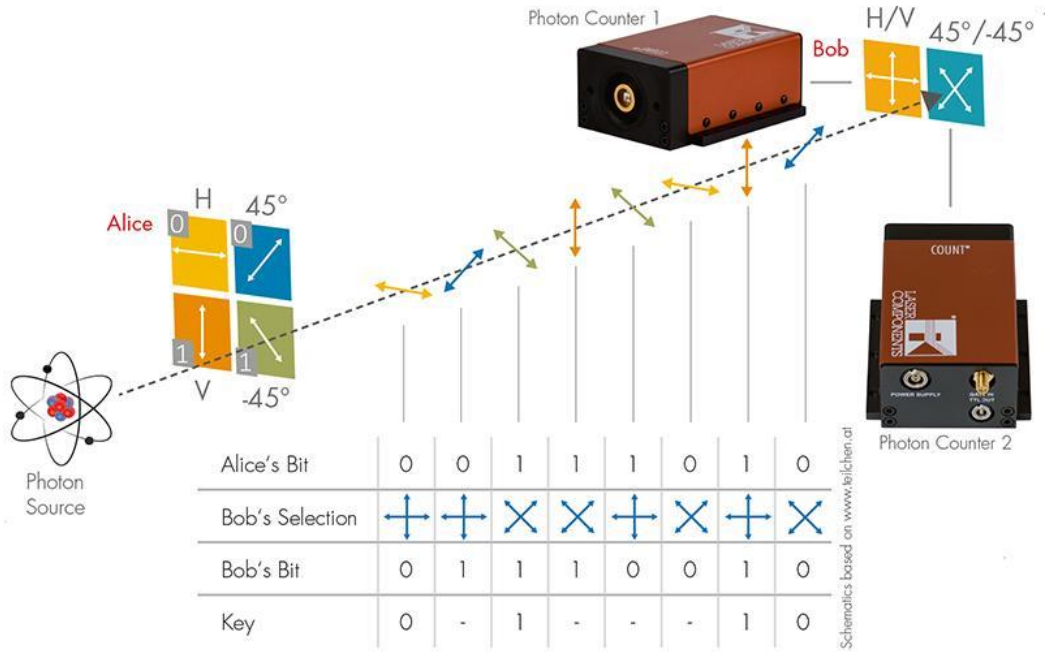
A kvantumszámítógépek megjelenése új kihívást jelent a kriptográfia számára.

Egy kellően nagy és hibajavított kvantumszámítógép Shor algoritmusával olyan matematikai problémákat tudna hatékonyan megoldani, amelyekre például az RSA biztonsága épül.

De a kvantumfizika nemcsak veszélyt, hanem új lehetőséget is kínál.

A **kvantumkulcs-elosztás**, például a BB84 protokoll, kvantumállapotok segítségével teszi lehetővé titkos kulcs létrehozását.





Ábra: *A kvantumkulcs-elosztásban az információ kvantumállapotokban továbbítható. A mérés megváltoztathatja ezeket az állapotokat, ezért a lehallgatás nyomot hagyhat.*

A kvantummechanika különös törvényei miatt egy lehallgató mérése megzavarhatja az átvitt kvantumállapotokat. Alice és Bob így megfelelő eljárással észlelheti, ha a kulcslétrehozás során valaki beleavatkozott a kommunikációba.

Fontos azonban: a **kvantumkriptográfia nem azt jelenti, hogy minden internetes adatot kvantumállapotban titkosítunk**. Elsősorban a biztonságos kulcselosztás egyik különleges módszeréről beszélünk.

Hogyan törték fel az Enigmát?

Az Enigma a második világháború egyik legismertebb titkosítógépe volt. Első pillantásra írógéphez hasonlított, belsejében azonban forgó tárcsák – rotorok – gondoskodtak arról, hogy minden leütött betű más betűvé alakuljon.

A nehézséget az jelentette, hogy minden billentyűleütés után legalább az egyik rotor továbbfordult. Emiatt ugyanaz a betű a szöveg egyik helyén például K-vá, néhány karakterrel később pedig egészen más betűvé változhatott. A rotorok sorrendje, kezdőállása és a kapcsolótábla beállítása együtt óriási számú lehetőséget eredményezett.

A németek ezért az Enigmát rendkívül biztonságosnak tartották.

A megfejtés története azonban nem Alan Turinggal kezdődött.

Már az 1930-as években három lengyel matematikus, **Marian Rejewski, Jerzy Różycki és Henryk Zygalski** jelentős eredményeket ért el az Enigma működésének feltárásában. Rejewski matematikai módszerekkel rekonstruálta a katonai Enigma belső huzalozását, és a lengyelek speciális gépeket és eljárásokat is készítettek a napi kulcsok megtalálására.

1939-ben, közvetlenül a háború kitörése előtt eredményeiket átadták brit és francia szövetségeseiknek.

A brit kódfejtés központja a **Bletchley Park** lett. Matematikusok, nyelvészek, sakkozók, mérnökök és más szakemberek ezrei dolgoztak itt a német üzenetek megfejtésén.

Közöttük volt **Alan Turing** és **Gordon Welchman** is.

A britek továbbfejlesztették a lengyel eredményeket, és létrehozták a **Bombe** nevű elektromechanikus berendezéseket. Ezek nem egyszerűen „végigpróbálták az összes kulcsot”, hanem a feltételezett szövegrészletek és az Enigma ismert matematikai tulajdonságai alapján zárták ki nagy sebességgel a lehetetlen beállításokat.

A kódfejtőknek az emberi hibák is sokat segítettek. A katonai üzenetek gyakran tartalmaztak kiszámítható fordulatokat, időjárás-jelentéseket vagy szabványos kifejezéseket. Ha a megfejtők helyesen sejtették egy üzenetrész eredeti szövegét, az fontos kapaszkodót adott a kulcs megtalálásához.

Az Enigma története ezért nagyszerű példája annak, hogy a biztonság nemcsak az alkalmazott technikától függ.

Egy titkosító rendszer lehet matematikailag rendkívül bonyolult – ha az emberek kiszámíthatóan használják, gyenge pontokat teremthetnek benne.

És még egy fontos tanulság: az Enigmát nem egyetlen zseni „törte fel”. Lengyel, brit és más szövetséges szakemberek hosszú éveken át végzett közös munkájának eredménye volt.

A kriptográfia fegyvernek számít?

1991-ben Phil Zimmermann amerikai programozó közzétette a **PGP – Pretty Good Privacy** nevű programját.

A PGP különlegessége nem pusztán az volt, hogy erős titkosítást alkalmazott. Ilyen technológia korábban is létezett.

A valódi újdonság az volt, hogy **bárki használhatta személyi számítógépén.**

Zimmermann úgy gondolta, hogy az elektronikus levelezés korában az embereknek ugyanúgy joguk van magánbeszélgetéshez, mint amikor becsukják maguk mögött az ajtót vagy borítékba tesznek egy levelet.

Csakhogy az erős kriptográfiát az Egyesült Államok akkoriban szigorú exportszabályokkal korlátozta. A titkosító szoftverek bizonyos esetekben olyan exportellenőrzés alá estek, amelyet katonai jelentőségű technológiákra is alkalmaztak.

Amikor a PGP az interneten keresztül külföldre is eljutott, Zimmermann ellen szövetségi vizsgálat indult.

Különös helyzet alakult ki.

Lehet-e egy matematikai algoritmus exportját megtiltani?

A számítógépes program ugyanis információ. Leírható papíron, kinyomtatható könyvben, megtanulható és újra begépelhető.

A PGP forráskódját ezért könyv formájában is kiadták. A nyomtatott könyveket az amerikai alkotmány szólásszabadságot védő rendelkezései miatt egészen más jogi helyzetben kezelték, mint az exportkorlátozás alá eső szoftvert.

A Zimmermann elleni vizsgálat 1996-ban vádemelés nélkül lezárult, az amerikai kriptográfiai exportszabályokat pedig később jelentősen enyhítették.

A vita azonban nem tűnt el.

Ha csak a feladó és a címzett képes elolvasni egy üzenetet, az megvédi az állampolgárt a bűnözőktől, az ipari kémkedéstől és az illetéktelen megfigyeléstől.

Ugyanez a titkosítás azonban a bűnözők kommunikációját is megvédheti a hatóságoktól.

Ezért a kérdés ma is ugyanaz:

Jogunk van-e olyan titkosításhoz, amelyhez rajtunk és a címzetten kívül senki másnak nincs kulcsa?

A kriptográfia története ezen a ponton már nem pusztán matematika és számítástechnika.

A magánélet, a szabadság és a biztonság közötti egyensúly kérdésévé válik.

Feltöri-e a kvantumszámítógép az internet titkosítását?

Gyakran hallani, hogy ha elkészül a kellően nagy kvantumszámítógép, „minden titkosítást fel fog törni”.

Ez így nem igaz – de van mögötte valódi probléma.

A mai internet biztonságának egy része olyan matematikai feladatokra épül, amelyeket a hagyományos számítógépek nagyon nehezen tudnak megoldani.

Ilyen például nagy számok prímtényezőkre bontása, amely az RSA biztonságának alapvető része.

Egy hagyományos számítógép számára megfelelően nagy számok esetében ez rendkívül hosszadalmas feladat.

1994-ben azonban **Peter Shor** amerikai matematikus megmutatta, hogy egy megfelelő kvantumszámítógép egészen más módon képes megközelíteni bizonyos ilyen problémákat. A róla elnevezett **Shor-algoritmus** elvileg lehetővé teszi a nagy számok prímtényezőkre bontását olyan hatékonysággal, amely veszélyezteti az RSA-t és több más nyilvános kulcsú rendszert.

Ez azonban nem jelenti azt, hogy egy mai kvantumszámítógéppel egyszerűen feltörhetjük az internetet.

Ehhez a jelenlegieknél lényegesen nagyobb, megbízhatóbb, hibajavított kvantumszámítógépekre lenne szükség.

A veszélyre azonban már most készülni kell.

Miért?

Mert egy támadó ma is elmentheti a titkosított adatforgalmat, és megpróbálhatja évekkel később megfejteni, amikor már rendelkezésére áll megfelelő technológia.

Ezt nevezik: **„harvest now, decrypt later” – gyűjtsd be most, fejtsd meg később.**

Ezért születnek olyan új titkosítási eljárások, amelyek a jelenlegi ismereteink szerint a kvantumszámítógépek támadásainak is ellenállnak. Ezeket összefoglaló néven **posztkvantum-kriptográfiának** nevezzük.

Fontos különbség: **A posztkvantum-kriptográfia nem kvantumkriptográfia.**

A posztkvantum-kriptográfia hagyományos számítógépeken futó új matematikai algoritmusokat használ.

A kvantumkriptográfia ezzel szemben magukat a kvantumfizikai jelenségeket használja fel, például titkos kulcs biztonságos létrehozására.

Különös verseny kezdődött tehát.

A kvantumszámítógép veszélyezteti a régi titkosítás egy részét – és ezzel egy időben új, erősebb titkosítási módszerek kifejlesztésére kényszerít bennünket.

A kriptográfia több ezer éves története ezzel ismétli önmagát:

amikor erősebb lesz a kódfejtő, erősebb zárat kell készíteni.



16. Számítógépes kártevők – Rabló-pandúr

A számítógépes program alapvetően utasítások sorozata. A gép végrehajtja őket – függetlenül attól, hogy az utasítások a felhasználót segítik vagy éppen ellene dolgoznak.

Amikor a számítógépek elkezdtek programokat és adatokat cserélni, szinte törvényszerűen megjelentek azok a programok is, amelyek **kéretlenül terjedtek egyik gépről a másikra**.

Eleinte sok ilyen program inkább kísérlet vagy tréfa volt. Később azonban megjelent az adatrombolás, a kémkedés, az adatlopás, majd a szervezett kiberbűnözés.

A biológiai hasonlat meglepően találó:

fertőzés → szaporodás → terjedés → járvány → védekezés → alkalmazkodás.

16.1. Az első vírusok

A történet egyik korai szereplője az 1971 körül megjelent **Creeper** volt. Bob Thomas kísérleti programja az ARPANET-en keresztül DEC PDP-10 számítógépek között tudott mozogni. Nem rombolt, hanem üzenetet jelenített meg. Ray Tomlinson később módosította úgy, hogy másolatokat is készítsen magáról. A Creeper eltávolítására megszületett a **Reaper** – a kártevő és az ellene harcoló program párharca tehát szinte rögtön elkezdődött.

Az első, személyi számítógépeken „vadon” is elterjedt vírusnak általában az **Elk Clonert** tekintik. 1982-ben egy 15 éves amerikai diák, Rich Skrenta készítette Apple II számítógépre, eredetileg tréfának. Floppylemezekkel terjedt.

1986-ban pedig megjelent a **Brain**, az IBM PC-kompatibilis gépek első széles körben ismert vírusainak egyike, amely már a lemez boot szektorát fertőzte meg.

```
BRAIN.ASM  - A Brain vírus részlete                (C) 1986  Basit & Amjad Farooq Alvi
-------------------------------------------------------------------------------------

0100    FA 33 C0        CLI             ; megszakítások tiltása
0103    8E D0           MOV  DX,AX      ; DS = 0
0106    BC 00 7C        MOV  SP,7C00h   ; verem a kód fölé
0109    8A 34           MOV  AL,[SI]    ; olvasás a következő szektorból
010C    3C 00           CMP  AL,00h     ; üres szektor?
010F    74 06           JE   SKIP
0112    B4 09           MOV  AH,09h     ; DOS: írj karaktert
0115    CD 21           INT  21h
0118    B8 01 00        MOV  AX,0001h   ; videó mód beállítása
011B    8E D8           MOV  DS,AX
011E    8E C0           MOV  ES,AX
0121    FB              STI             ; megszakítások engedélyezése
....
0167    B9 00 00        MOV  CX,0000h   ; másolás kezdete
016A    F3 A4           REP  MOVSB      ; másolja a vírust a memóriában
016D    EA 00 01        JMP  0100h      ; visszaugrás a kezdetre
0170    00 00           DB   00h ; víruskód és adatok
0173    00 00           DB   00h ; víruskód és adatok
0176    00 00           DB   00h ; víruskód és adatok
0179    00 00           DB   00h ; víruskód és adatok
017C    00 00           DB   00h ; víruskód és adatok
017F    00 00           DB   00h ; víruskód és adatok

A Brain vírus (1986)

Az első ismert PC vírus.
A floppy lemez boot
szektorát fertőzi meg.
Induláskor a memóriába
másolja önmagát,
majd visszaadja az
irányítást a rendszernek,
így a felhasználó nem
vesz észre semmit.

Írói: Basit és Amjad
Farooq Alvi (Pakistan)

-------------------------
CLI     = Clear Interrupt Flag
STI     = Set Interrupt Flag
INT 21h = DOS megszakítás
MOVSB   = byte másolás
-------------------------

Megjegyzés:
Oktatási célú bemutatás -
Ne használd rosszindulatú
célra!
```

Ábra: *A Brain vírus kódjának részlete. Az 1986-ban megjelent kártevő a PC-k floppylemezeinek boot szektorát fertőzte.*

16.2. Bootvírus

A PC-k korai korszakában a programokat és adatokat gyakran **floppylemezen** vitték egyik számítógépről a másikra.

Ez tökéletes terjedési lehetőséget biztosított a vírusok számára.

A bootvírus a lemeznek azt a részét fertőzte meg, amelyről a számítógép induláskor programot töltött be. Ha a felhasználó fertőzött lemezről indította a gépet, a vírus bekerült a memóriába. Ezután az újonnan behelyezett floppylemezeket is megfertőzhette.

Így a vírus útja lehetett:

fertőzött floppy → számítógép → másik floppy → másik számítógép

A Brain ennek klasszikus példája volt.

Érdekes, hogy az internet előtti világban a vírusok **fizikai adathordozókon utaztak.** A fertőzés sebességét tehát nagyrészt az határozta meg, milyen gyorsan vitték az emberek a lemezeket egyik géptől a másikig.

16.3. Fájlvírus

A fájlvírus nem a lemez indítóterületét, hanem egy végrehajtható programot fertőz meg.

A vírus saját kódját hozzákapcsolja vagy beépíti a programba. Amikor a felhasználó elindítja a fertőzött programot, a vírus is végrehajtódik.

Ezután további fájlokat kereshet és fertőzhet meg.

program → fertőzés → programindítás → újabb programok fertőzése

A vírus tehát egy másik programot használ „gazdaszervezetként” – innen származik a biológiai vírusokkal való hasonlóság.

A védekezés és támadás versenye itt gyorsult fel igazán. Az antivírus-programok felismerhető kódrészleteket, **vírusszignatúrákat** kerestek. Erre válaszul megjelentek olyan vírusok, amelyek módosították vagy titkosították saját kódjukat, hogy nehezebb legyen felismerni őket.

Megkezdődött a valódi **rabló-pandúr játék.**

16.4. Makrovírus

Az 1990-es években új meglepetés érte a felhasználókat.

Nemcsak programfájl lehet veszélyes.

Egy dokumentum is tartalmazhat programot.

A Microsoft Word és más irodai programok makrónyelvei lehetővé tették, hogy dokumentumokhoz automatikusan végrehajtható programrészek tartozzanak.

1995-ben a **Concept** nevű Word-makrovírus megmutatta, milyen hatékonyan használható ez terjedésre.

A fertőzött Word-dokumentum megnyitása után a vírus más dokumentumokat is megfertőzhetett.

A probléma azért vált különösen súlyossá, mert az emberek természetes módon megbíztak a dokumentumokban:

„Ez csak egy Word-fájl. Mi baj lehet?”

Mint kiderült: sok minden.

16.5. Féreg

A számítógépes féreg egyik legfontosabb tulajdonsága, hogy **önállóan képes hálózaton terjedni**.

Nem feltétlenül kell hozzá fertőzött programot másolgatni.

Ennek egyik leghíresebb példája az **1988-as Morris-féreg** volt. Robert Tappan Morris programja Unix rendszerek hibáit és gyenge pontjait használta ki, és az akkori internet jelentős részét érintette.

A féreg egyik programozási hibája miatt ugyanazt a számítógépet többször is megfertőzhette. A gépek lelassultak, egy részük gyakorlatilag használhatatlanná vált.


NEVER FORGET





Ábra: *Az 1988-as Morris-féreg megmutatta, milyen gyorsan terjedhet egy önmagát másoló program az összekapcsolt számítógépek között.*

A hálózat ettől kezdve nemcsak kommunikációs lehetőség volt.

A fertőzés terjedési útvonala is lett.

16.6. Trójai program

A trójai program nevét természetesen a görög mondából ismert **trójai falóról** kapta.

Kívülről hasznos vagy ártalmatlan programnak látszik, belül azonban valami egészen mást rejt.

A felhasználó maga telepíti vagy indítja el, mert azt hiszi, hogy játékot, segédprogramot vagy dokumentumot kapott.

A trójai ezután például:

- adatokat lophat,
- jelszavakat figyelhet meg,
- további kártevőket tölthet le,
- vagy távoli hozzáférést biztosíthat a támadónak.

A trójai tehát nem feltétlenül önmagát szaporítja.

Legfontosabb fegyvere a megtévesztés.

16.7. Rootkit

A rootkit célja elsősorban nem a terjedés, hanem a **rejtőzködés és a tartós hozzáférés fenntartása.**

Ha a támadó már bejutott a számítógépbe, olyan eszközöket telepíthet, amelyek megpróbálják elrejteni jelenlétét a felhasználó és a biztonsági programok elől.

Elrejthetnek fájlokat, folyamatokat vagy hálózati kapcsolatokat, és akár az operációs rendszer alacsony szintű működését is manipulálhatják.

Itt a rabló-pandúr játék új szintre jutott:

már nemcsak be kell jutni – láthatatlannak is kell maradni.

16.8. Zsarolóprogram

A számítógépes kártevők fejlődésének egyik legsötétebb iránya a **ransomware**, vagyis zsarolóprogram.

A módszer egyszerű:

bejutás → fájlok titkosítása → váltságdíj követelése

A felhasználó dokumentumai, fényképei vagy egy vállalat teljes informatikai rendszere elérhetetlenné válhat.

A zsarolóprogramok különösen veszélyessé váltak, amikor a kriptovaluták megkönnyítették a nehezen követhető nemzetközi pénzfizetést.

A 2017-es **WannaCry** támadás például világszerte számítógépek százezreit érintette, köztük kórházak, vállalatok és közintézmények rendszereit.

Ettől kezdve világossá vált, hogy egy számítógépes kártevő már **fizikai infrastruktúrák működését is veszélyeztetheti.**

16.9. Spam és adathalászat

Nem mindig a számítógépet kell becsapni.

Sokszor sokkal egyszerűbb **az embert becsapni.**

Az adathalász – phishing – levél banktól, futárszolgálattól, hatóságtól vagy akár egy ismerőstől érkező üzenetnek látszhat.

A felhasználót ráveszi, hogy kattintson egy linkre, adja meg jelszavát vagy nyisson meg egy mellékletet.

A 2000-ben pusztító **ILOVEYOU** féreg például egy egyszerű emberi tulajdonságra épített: a kíváncsiságra. A „LOVE-LETTER-FOR-YOU” mellékletet emberek milliói nyitották meg.



Ábra: *Az ILOVEYOU 2000-ben e-mailben terjedt. A támadás egyik legfontosabb eszköze nem technikai trükk volt, hanem az emberi kíváncsiság.*

A biztonsági rendszerek lehetnek rendkívül fejlettek – az ember gyakran továbbra is a rendszer egyik legsebezhetőbb pontja.

16.10. Antivírusok – és egy magyar történet

A vírusok megjelenése természetesen létrehozta ellenfeleiket is: az **antivírus-programokat.**

Kezdetben ezek elsősorban ismert vírusokra jellemző kódrészleteket, úgynevezett szignatúrákat kerestek.

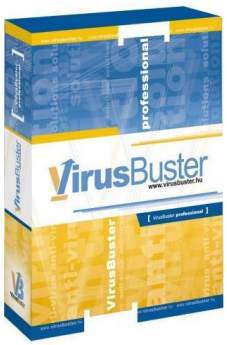
A vírusírók erre olyan kártevőket készítettek, amelyek módosították saját kódjukat.

Az antivírusok válasza a heurisztikus és viselkedéselemzés lett.

A támadók újabb rejtőzködési módszereket dolgoztak ki.

És ez a verseny mindmáig tart.

Magyar szál – VirusBuster



Ebben a történetben Magyarország is helyet kap.

A **VirusBuster** elődjénél már az 1980-as évek végén megkezdődött antivírus-termék fejlesztése. A korai program **Virsec** néven jelent meg, majd 1992-ben VirusBusterre nevezték át. A később megalakult magyar VirusBuster Kft. hazai és nemzetközi piacra is fejlesztett biztonsági szoftvereket. A cég 2012-ben jelentette be saját antivírus-termékvonalának megszüntetését.

Ábra: *A magyar fejlesztésű VirusBuster egyik változata. Magyarország az antivírus-szoftverek fejlesztésében is megjelent a nemzetközi informatikai térképen.*

16.11. Kiberbűnözés

A korai vírusok szerzői gyakran kíváncsiságból, kísérletezésből vagy egyszerű csínytevésként írták programjaikat.

A helyzet mára gyökeresen megváltozott.

A kiberbűnözés **üzletággá** vált.

Léteznek adatlopással, banki csalással, zsarolóprogramokkal, vállalati rendszerek feltörésével és ellopott hozzáférési adatok értékesítésével foglalkozó bűnözői csoportok.

Sőt a kártevők és támadási infrastruktúrák egy része már szolgáltatásként is megjelenik: a támadónak nem feltétlenül kell saját maga minden programot elkészítenie.

A számítógépes bűnözés tehát végigjárta ugyanazt az utat, mint sok más technológia:

amatőr kísérlet → specializáció → szervezett iparág.

A számítógépes vírus, mint digitális járvány

Nem véletlenül nevezzük **vírusnak** az önmagukat terjesztő számítógépes kártevők egy részét. Fred Cohen az 1980-as években használta és definiálta tudományos értelemben a „computer virus” fogalmát, és a biológiai hasonlat meglepően találónak bizonyult.

A biológiai vírus önmagában nem képes szaporodni. Bejut egy élő sejtbe, majd annak működését használja fel saját másolatainak elkészítésére.

A klasszikus számítógépes vírus hasonlóan viselkedik. Szüksége van valamilyen „gazdára”: egy programra, rendszerterületre vagy dokumentumra. Ha a fertőzött objektum működésbe lép, a vírus aktiválódhat, majd újabb fájlokat vagy adathordozókat fertőzhet meg.

A hasonlóság azonban nem ér véget a szaporodásnál.

Biológiai világ → Számítógépes világ

fertőzött élőlény → fertőzött számítógép
vírushordozó → fertőzött fájl vagy adathordozó
érintkezés → adatcsere
fertőzési út → hálózati kapcsolat
lappangás → rejtett működés
járvány → tömeges hálózati fertőzés
immunrendszer → számítógépes védelem
vírusfelismerés → antivírus
védőoltás → biztonsági frissítés
karantén → fertőzött fájl elkülönítése

A számítógépes járványok terjedési sebességét alapvetően a **kapcsolatok száma és gyorsasága** változtatta meg.

A személyi számítógépek korai korszakában egy vírus napok, hetek vagy akár hónapok alatt járhatta be a világot. Floppylemezen utazott egyik számítógéptől a másikig – vagyis végső soron az ember vitte magával.

A hálózatok mindent megváltoztattak.

Egy számítógépes féreg már önállóan kereshet újabb megtámadható gépeket. Az interneten pedig néhány óra, szélsőséges esetben akár percek alatt hatalmas számú számítógépet érhet el.

A biológiai és a digitális járványok között azonban van egy alapvető különbség.

A biológiai vírus az evolúció eredménye.

A számítógépes kártevőt **ember írja.**

Ezért képes tudatosan kihasználni egy operációs rendszer hibáját, jelszavakat keresni, biztonsági programokat kikapcsolni vagy éppen elrejteni saját jelenlétét.

És itt kezdődik a **rabló-pandúr játék.**

Megjelenik egy vírus.

Elkészül az antivírus, amely felismeri.

A vírus készítője módosítja a kódot, hogy ne lehessen felismerni.

A víruskereső már nemcsak ismert kódrészleteket keres, hanem a program viselkedését is figyeli.

A támadó új módszert talál a rejtőzködésre.

A védelem ismét alkalmazkodik.

Vírus → antivírus → új vírus → új felismerési módszer → új támadás → új védelem → …

A folyamat feltűnően emlékeztet a természetben zajló evolúciós fegyverkezési versenyre: minden új védekezési módszer előnyt jelent azoknak a támadóknak, amelyek képesek megkerülni.

Van azonban még egy fontos tanulság.

A járványokhoz hasonlóan a számítógépes fertőzések ellen sem elegendő kizárólag akkor védekezni, amikor már megtörtént a baj. A rendszeres frissítés, a biztonsági mentés, a hozzáférések megfelelő védelme és a felhasználók óvatossága ugyanazt a szerepet tölti be a digitális világban, mint a **megelőzés a járványvédelemben.**

A hálózat pedig ugyanazt a kettősséget hordozza, mint az emberi társadalom:

ami lehetővé teszi az információ gyors terjedését, az a fertőzés gyors terjedését is lehetővé teszi.

Ez a számítógépes korszak egyik alapvető paradoxona:

Minél jobban összekötjük a világot, annál több előnyünk származik a kapcsolatokból – és annál gyorsabban terjedhet mindaz, amit nem szeretnénk továbbadni.

Ezért a számítógépes vírus története egyben a hálózatok történetének árnyoldala is.

Hacker vagy cracker? – Nem minden hacker bűnöző

A filmekben és a hírekben a **hacker** többnyire számítógépes betörőt jelent. A szó eredeti jelentése azonban egészen más.

A hacker-kultúra az 1950-es és 1960-as évek amerikai műszaki egyetemeinek, különösen az MIT-nek a világában alakult ki. Hackernek azt a rendkívül kíváncsi és kreatív műszaki embert nevezték, aki meg akarta érteni egy rendszer működését, kísérletezett vele, átalakította, és gyakran olyan dolgokra is képessé tette, amelyekre eredetileg nem tervezték.

A **hack** tehát eredetileg inkább ötletes vagy szokatlan műszaki megoldást jelentett.

A számítógépes hálózatok terjedésével azonban megjelentek azok is, akik tudásukat rendszerek engedély nélküli feltörésére használták. A hacker-közösség egy része ezért a rosszindulatú rendszerfeltörőkre a **cracker** elnevezést kezdte használni.

A köznyelvben ez a különbségtétel nem igazán honosodott meg. Ma ezért gyakran „kalapokkal” jelölik a különböző szerepeket.

White hat – fehér kalapos: engedéllyel keres biztonsági hibákat, hogy azokat ki lehessen javítani.

Black hat – fekete kalapos: engedély nélkül tör be rendszerekbe, adatot lop, károkozásra vagy haszonszerzésre törekszik.

Grey hat – szürke kalapos: a kettő közötti bizonytalan terület; tevékenységének nem feltétlenül célja a károkozás, de engedély nélkül vizsgálhat vagy támadhat rendszereket.

Kevin Mitnick – amikor a rablóból pandúr lesz



A hackertörténet egyik legismertebb alakja **Kevin Mitnick (1963–2023)** volt.

Az 1980-as és 1990-es években számítógépes rendszerekbe és hálózatokba tört be, több vállalat rendszeréhez szerzett jogosulatlan hozzáférést. Az FBI egyik legkeresettebb számítógépes szökevényévé vált, míg 1995-ben elfogták.

Mitnick azonban nem pusztán technikai módszereiről lett híres.

Felismerte, hogy egy informatikai rendszer leggyengébb pontja gyakran **nem a számítógép, hanem az ember**.

Telefonon felhívott alkalmazottakat, rendszergazdának vagy munkatársnak adta ki magát, és megfelelően felépített történettel rávette őket, hogy jelszavakat, telefonszámokat vagy más fontos információkat áruljanak el.

Ezt nevezzük **social engineeringnek – társadalmi manipulációnak**.

Mitnick börtönbüntetése után különös fordulatot vett az élete. Biztonsági tanácsadó lett, vállalatok rendszereinek gyenge pontjait kereste, előadásokat tartott és könyveket írt az informatikai biztonságról.

A története így szinte szó szerint megtestesíti a **rabló-pandúr** hasonlatot:

az egykori üldözött hacker később ugyanazt a tudását már a rendszerek védelmére használta.

És története egy máig érvényes biztonsági szabályra is figyelmeztet:

Lehet bármilyen erős a titkosítás és bármilyen jó a tűzfal, ha valakit sikerül rávenni arra, hogy önként elárulja a jelszavát.

A rablónak és a pandúrnak ugyanis sokszor **ugyanazt kell tudnia – csak nem ugyanarra használják a tudásukat.**

VII. rész – Amikor egy számítógép már nem elég

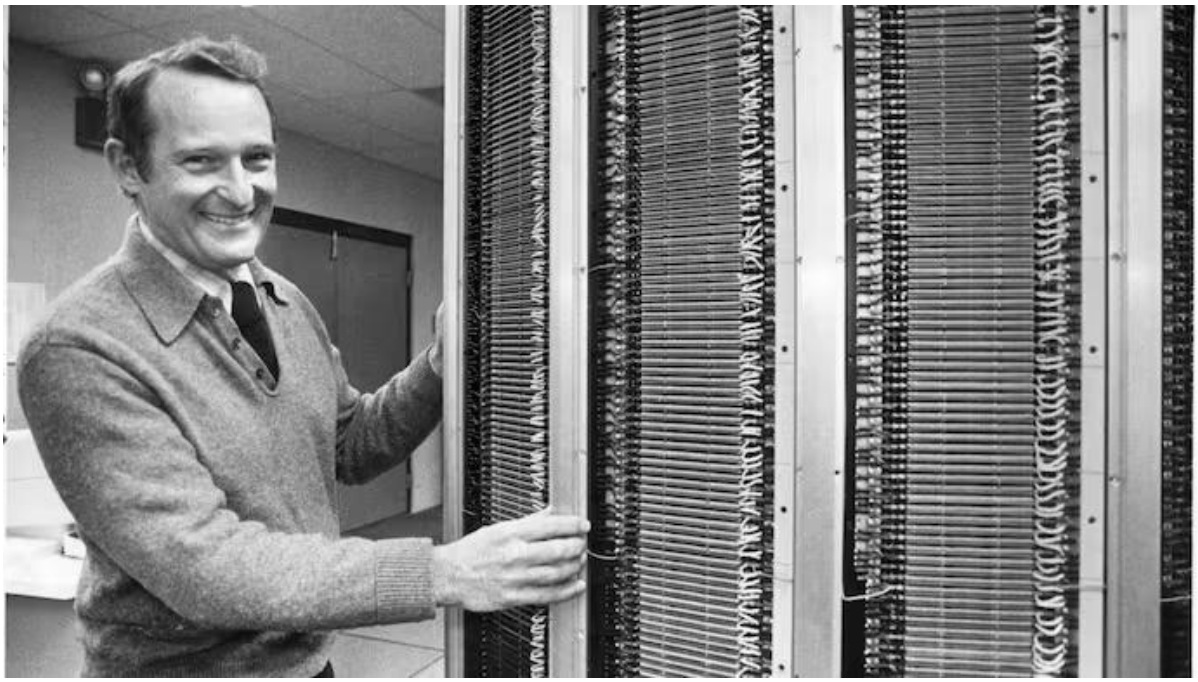
17. Szuperszámítógépek

A számítógépek történetében mindig volt egy különleges kategória, amelynek tervezőit kevésbé érdekelte az ár, a méret vagy az általános használhatóság. Egyetlen cél lebegett előttük: **hogyan lehet az adott kor technológiájából a lehető legnagyobb számítási teljesítményt kihozni?**

Ezek a szuperszámítógépek.

Érdekes módon a „szuperszámítógép” nem állandó műszaki kategória. Ami tegnap szuperszámítógép volt, az ma akár egy hétköznapi eszköznél is lassabb lehet. A szuperszámítógép tehát mindenekelőtt **saját korának legnagyobb teljesítményű számítógépe**.

17.1. Seymour Cray



A szuperszámítógépek történetéből egyetlen nevet kellene kiemelnünk, az minden bizonnyal **Seymour Cray** lenne. Az amerikai mérnök olyan meghatározó alakja lett a nagy teljesítményű számítástechnikának, hogy gyakran egyszerűen a „szuperszámítógépek atyjaként” emlegetik. A Control Data Corporationnél, majd saját vállalatánál, a Cray Researchnél dolgozott.

Cray felismerte, hogy egy számítógépet nem lehet pusztán azzal gyorsabbá tenni, hogy gyorsabb processzort építünk. Az adatokat el is kell juttatni hozzá, az eredményeket el kell vezetni, a memóriának megfelelő sebességgel kell kiszolgálnia a processzort, a gépet pedig hűteni kell.

A nagy teljesítmény tehát **az egész rendszer összehangolásának eredménye**.

Cray különös figyelmet fordított még a vezetékek hosszára is. Nagy sebességnél ugyanis már az elektromos jel terjedési ideje sem hanyagolható el. Ez később a Cray-1 különleges alakját is meghatározta.



Ábra: *Seymour Cray (1925–1996), a szuperszámítógépek történetének legendás tervezője. Számára a sebesség nem pusztán gyors processzort, hanem a teljes számítógép gondos megtervezését jelentette.*

17.2. CDC 6600





Ábra: *A CDC 6600 1964-ben. Seymour Cray gépe a központi processzor mellett tíz kisebb perifériaprocesszort használt, hogy a fő processzor elsősorban a számításokkal foglalkozhasson.*

1964-ben jelent meg a **Control Data Corporation CDC 6600** számítógépe. A gépet általában az első igazán sikeres szuperszámítógépnek tekintik.

A CDC 6600 egyik nagy újítása az volt, hogy a központi processzort igyekeztek megszabadítani a mellékes feladatoktól. Tíz perifériaprocesszor foglalkozott többek között az adatforgalommal, miközben a központi processzor a számításokra összpontosíthatott.

Ez a munkamegosztás fontos elvet mutatott meg:

nem feltétlenül egyetlen mindent végző processzor a leggyorsabb megoldás.

A CDC 6600 mintegy tízszer gyorsabb volt számos korábbi rendszerénél, és többek között atomfizikai számításokra, időjárás-előrejelzésre és tudományos szimulációkra használták.

17.3. Cray-1

Seymour Cray 1972-ben megalapította a Cray Research vállalatot. Első gépe, az **1976-ban bemutatott Cray-1** a számítástechnika egyik legismertebb formatervezési ikonjává vált.

A gép C alakú, majdnem kör alakú szekrényekből állt, alját pedig párnázott ülőfelület vette körül. A forma azonban nem pusztán látványosság volt.

A processzor, különböző részeit összekötő vezetékeket minél rövidebbre kellett készíteni. A kör alakú elrendezés ebben segített. A nagy teljesítmény miatt a hőelvezetés is komoly mérnöki feladatot jelentett. A Computer History Museum szerint a Cray-1 1976 és 1982 között a világ leggyorsabb számítógépének számított.

A gép körülbelül **80 MHz-es órajellel** működött, és csúcsteljesítménye nagyságrendileg százmillió lebegőpontos művelet volt másodpercenként.

Ma ez nevetségesen kevésnek tűnhet – akkor azonban elképesztő teljesítmény volt.



Ábra: *A Cray-1 jellegzetes kör alakú felépítése nem csupán dizájnelem volt: a rövidebb összeköttetések növelték a gép sebességét. Az alsó „ülőpad” mögött a tápegységek egy része rejtőzött.*

17.4. Vektorprocesszorok

A hagyományos processzor egy utasítással általában egy vagy néhány adattal végez műveletet. Tudományos számításokban azonban gyakran ugyanazt a műveletet kell hatalmas adatsorokon ismételni.

Például két számsor megfelelő elemeit kell összeadni:

$$\mathbf{A_1 + B_1, A_2 + B_2, A_3 + B_3 \ldots}$$

A **vektorprocesszor** ezeket nem egymástól teljesen független műveletekként kezeli, hanem egyetlen adatfolyamként. A Cray-1 egyik legfontosabb tulajdonsága éppen a rendkívül hatékony vektorfeldolgozás volt.

Ez különösen alkalmas:

- fizikai szimulációkra,
- meteorológiai számításokra,
- áramlástani modellekre,
- mátrixműveletekre,
- képfeldolgozásra.

A mai processzorok SIMD utasításai és a grafikus processzorok működésében ugyanez az alapgondolat él tovább: **ugyanazt a műveletet egyszerre sok adaton végezzük el.**

17.5. Párhuzamos számítás

Egy idő után világossá vált, hogy egyetlen processzort nem lehet korlátlanul gyorsítani.

Miért ne dolgozzon akkor egyszerre több processzor?

A párhuzamos számítás során egy nagy problémát kisebb részekre bontunk, és ezeket egyszerre több számítóegység végzi el.

Képzeljük el például az időjárás modellezését. A Föld légkörét képzeletbeli térbeli cellák millióira oszthatjuk. Mindegyik cellában ki kell számítani a hőmérséklet, nyomás, páratartalom és légmozgás változását.

Ahelyett, hogy egyetlen processzor sorban végigszámolná őket, processzorok ezrei vagy milliói dolgozhatnak rajtuk egyszerre.

Van azonban egy nehézség: a részeredményeket folyamatosan meg kell osztani egymással.

A modern szuperszámítógép ezért nem egyszerűen **sok processzor**, hanem **sok, rendkívül gyorsan kommunikáló processzor**.

17.6. Klaszterek

A szuperszámítógép sokáig különleges, egyedi konstrukciót jelentett. Az 1990-es évektől azonban egy másik megoldás egyre vonzóbbá vált.

Mi történik, ha sok, viszonylag hétköznapi számítógépet gyors hálózattal összekötünk?

Létrejön a **számítógép-klaszter**.

A klaszter csomópontjai különálló számítógépek, de megfelelő programok segítségével közösen dolgozhatnak ugyanazon problémán.

Ennek híres korai változata volt a **Beowulf-klaszter**, amely szabványos PC-alkatrészekből és Linux operációs rendszerből épített olcsó párhuzamos számítógépet.

Ez alapvetően megváltoztatta a szuperszámítógépek világát. A különleges processzorok mellett egyre nagyobb szerepet kaptak a nagy sorozatban gyártott CPU-k, majd később a **GPU-k**, vagyis grafikus processzorok.

A szuperszámítógép lassan egyetlen különleges gépből **hatalmas számítógép-hálózattá** alakult.

17.7. TOP500

De hogyan dönthetjük el, melyik a világ leggyorsabb számítógépe?

1993 óta évente kétszer közzéteszik a **TOP500 listát**, amely a világ legerősebb nagy teljesítményű számítógépes rendszereit rangsorolja.

A mérés hagyományosan a **LINPACK** tesztre épül, amely nagy lineáris egyenletrendszerek megoldásával méri a lebegőpontos számítási teljesítményt.

A teljesítmény mértékegysége a **FLOPS**:

Floating Point Operations Per Second,
vagyis másodpercenként végrehajtott lebegőpontos műveletek száma.

A fejlődés nagyságrendje önmagában is lenyűgöző:

megaflop → gigaflop → teraflop → petaflop → exaflop

Minden lépcső ezerszeres növekedést jelent.

Így: **1 exaflop = 1 000 000 000 000 000 000 művelet másodpercenként.**

Ez egymilliárdszor egymilliárd lebegőpontos művelet másodpercenként.

17.8. Az exaflop korszak

2022-ben az amerikai Oak Ridge National Laboratory **Frontier** nevű szuperszámítógépe elsőként lépte át hivatalosan az egy exaflopos határt a TOP500 LINPACK mérésében.

A gép már egészen másképpen fest, mint a Cray-1. Nem egyetlen különleges processzor köré épül, hanem rengeteg számítási egységből álló hatalmas párhuzamos rendszer.

A modern szuperszámítógépek teljesítményének jelentős részét GPU-k, illetve más gyorsítók adják. Ezek ugyanazt az alapelvet viszik tovább, amelyet a vektorprocesszoroknál már megismertünk: **sok hasonló számítást kell egyszerre elvégezni.**



Ábra: *A Frontier szuperszámítógép az Oak Ridge National Laboratoryban. 2022-ben elsőként ért el hivatalosan exaflop feletti teljesítményt a TOP500 LINPACK tesztjén.*

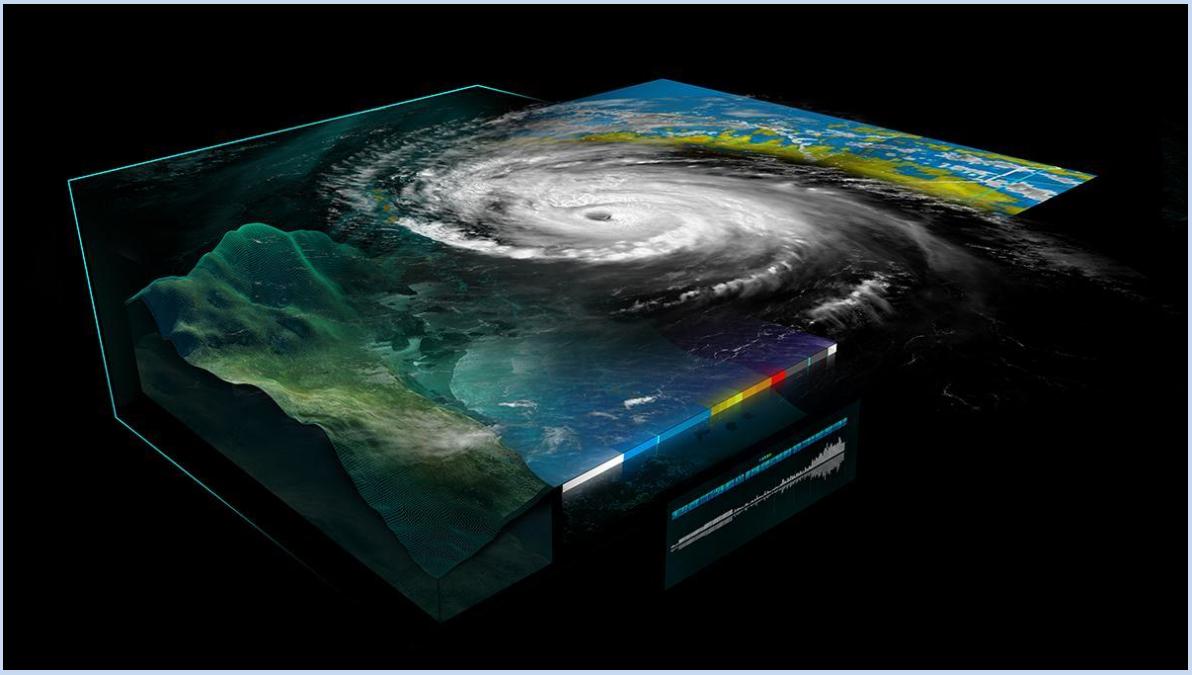
Az exaflop azonban nem a történet vége. Már a következő nagyságrend, a **zettaflop** felé vezető út technikai problémáit kutatják.

És itt különösen fontos kérdéssé vált az energiafogyasztás. Nem elég egy gépet megépíteni, amely elképesztően gyors – azt gazdaságosan működtetni és hűteni is kell.

A szuperszámítógépek története így furcsa módon visszavezet Seymour Cray egyik alapgondolatához:

a sebesség mindig az egész rendszer problémája.

Mit számol egy szuperszámítógép?





Amikor azt halljuk, hogy egy számítógép másodpercenként trilliónyi vagy trilliószor milliónyi műveletet végez, joggal merül fel a kérdés: **mire kell ennyi számítás?**

A szuperszámítógép általában nem egyetlen rendkívül nehéz összeadást végez. Ehelyett egyszerű matematikai műveletek elképesztő tömegét hajtja végre.

Időjárás és éghajlat. A légkört háromdimenziós cellákra osztják, majd minden cellára újra és újra kiszámítják, hogyan változik a hőmérséklet, a légnyomás, a páratartalom és a levegő mozgása.

Csillagászat. Galaxisok kialakulását, csillagok fejlődését, szupernóvákat vagy akár fekete lyukak összeolvadását modellezhetjük. Egy több milliárd éves kozmikus folyamatot természetesen nem várhatunk végig – matematikai modelljét számoljuk ki.

Atom- és molekulamodellezés. Gyógyszermolekulák kölcsönhatását, fehérjék szerkezetét és új anyagok tulajdonságait lehet számítógépen vizsgálni még azelőtt, hogy a valódi anyagot előállítanánk.

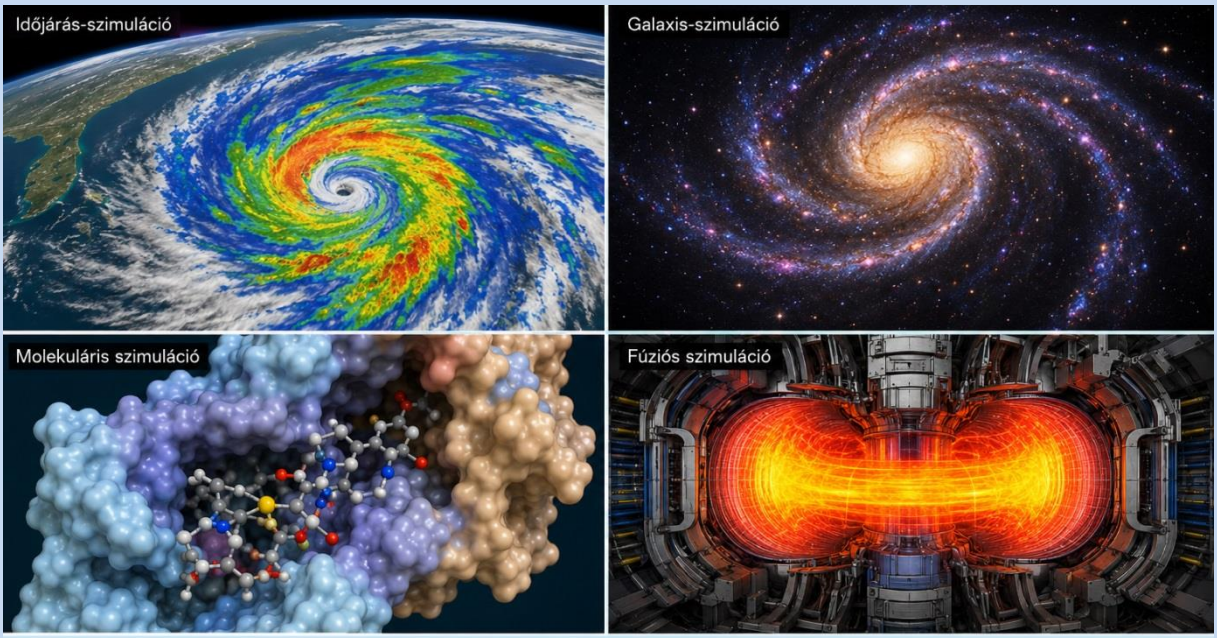
Mérnöki tervezés. Repülőgépek körüli légáramlás, autók ütközése, turbinák működése vagy egy épület földrengés alatti viselkedése virtuálisan kipróbálható.

Atomfizika és fúzió. Olyan szélsőséges fizikai folyamatokat lehet modellezni, amelyeket valódi kísérletben rendkívül nehéz, veszélyes vagy költséges lenne létrehozni.

És egyre fontosabb feladat a **mesterséges intelligencia** is. A nagy neurális hálózatok tanítása ugyancsak óriási mennyiségű párhuzamos mátrixműveletet kíván.

A szuperszámítógép tehát tulajdonképpen **virtuális laboratórium.**

A tudós felírja a természet ismert törvényeit, megadja a kiindulási állapotot, majd azt mondja a gépnek:



A szuperszámítógép egyik legfontosabb szerepe a szimuláció: olyan folyamatokat számolhatunk ki vele, amelyek túl nagyok, túl kicsik, túl gyorsak, túl lassúak vagy túl veszélyesek ahhoz, hogy közvetlenül kísérletezzünk velük.

A tudós felírja a természet ismert törvényeit, megadja a kiindulási állapotot, majd azt mondja a gépnek: „Számold ki, mi történik ezután!"

Ábra: *A szuperszámítógép egyik legfontosabb szerepe a szimuláció: olyan folyamatokat számolhatunk ki vele, amelyek túl nagyok, túl kicsik, túl gyorsak, túl lassúak vagy túl veszélyesek ahhoz, hogy közvetlenül kísérletezzünk velük.*

18. A GPU különös története

Kevés számítógépes alkatrész futott be olyan meglepő pályát, mint a **GPU – Graphics Processing Unit**, vagyis grafikus feldolgozóegység. Azért született, hogy gyorsabban rajzolja ki a számítógépes játékokat. Később kiderült, hogy felépítése miatt olyan számításokra is kiváló, amelyeknek semmi közük a grafikához. Ma GPU-k dolgoznak szuperszámítógépekben, tudományos laboratóriumokban és a mesterséges intelligencia mögött.

A történet egyik különös tanulsága:

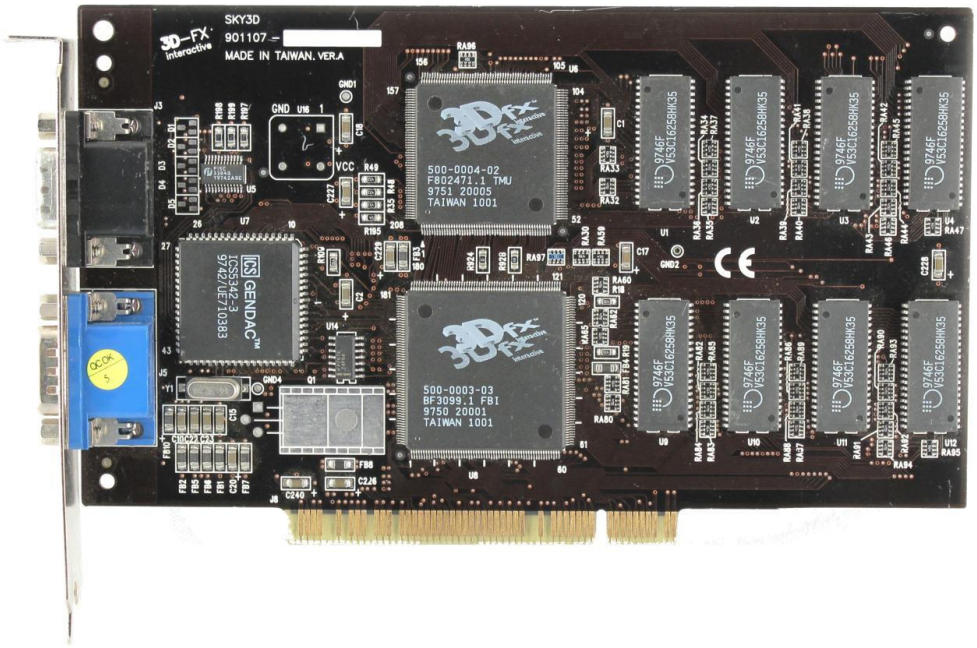
A játékosok egyre szebb virtuális világokat akartak – és közben olyan processzor fejlődött ki, amely végül az AI forradalmának egyik legfontosabb eszköze lett.

18.1. Grafikus gyorsítók

A korai személyi számítógépekben a képernyő tartalmának előállításában még jelentős szerepe volt a központi processzornak. Ahogy azonban nőtt a felbontás, a színmélység és különösen a grafikus felületek bonyolultsága, egyre több munkát kellett elvégezni.

Megjelentek a **grafikus gyorsítók**. Ezek speciális áramkörök voltak, amelyek átvették a CPU-tól például a vonalak és téglalapok rajzolását, a képrészletek mozgatását, majd később egyre több 3D műveletet is.

Az 1990-es évek közepén a PC-s játékok fejlődése óriási piacot teremtett számukra. Különösen híressé vált az **1996-ban megjelent 3dfx Voodoo Graphics.** A Voodoo kifejezetten 3D gyorsításra szolgált, ezért eleinte egy hagyományos 2D videokártya mellett működött. A 3dfx sikere jól mutatta, mekkora igény alakult ki a valósidejű háromdimenziós grafikára.



Ábra: *3dfx Voodoo Graphics. Az 1990-es évek közepének 3D-gyorsítói látványosan javították a PC-játékok grafikáját, és felgyorsították a grafikus processzorok fejlődését.*

18.2. 3D grafika

A háromdimenziós kép előállítása meglepően sok számítást igényel.



Egy virtuális tárgyat háromszögek sokaságából építenek fel. Minden egyes háromszögnél meg kell határozni többek között annak helyét és perspektivikus vetületét, majd a felület színét, textúráját, megvilágítását és láthatóságát.

És mindezt nem egyszer.

Egy játékban **másodpercenként több tucat vagy akár több száz új képet** kell előállítani.

A grafikai számításnak azonban van egy rendkívül fontos tulajdonsága. A kép rengeteg pontján hasonló matematikai műveleteket kell végrehajtani.

A CPU hagyományosan viszonylag kevés, de nagyon sokoldalú processzormagot használ. A grafikus processzor ezzel szemben sokkal több, egyszerűbb számítási egységet állíthat munkába párhuzamosan.

Ez lett később a GPU legnagyobb előnye.

18.3. Programozható GPU

1999-ben az NVIDIA bemutatta a **GeForce 256**-ot, amelyet a vállalat „a világ első GPU-jaként” hirdetett. A chip egyetlen processzorban egyesítette többek között a geometriai transzformációt, megvilágítást és raszterizálást.

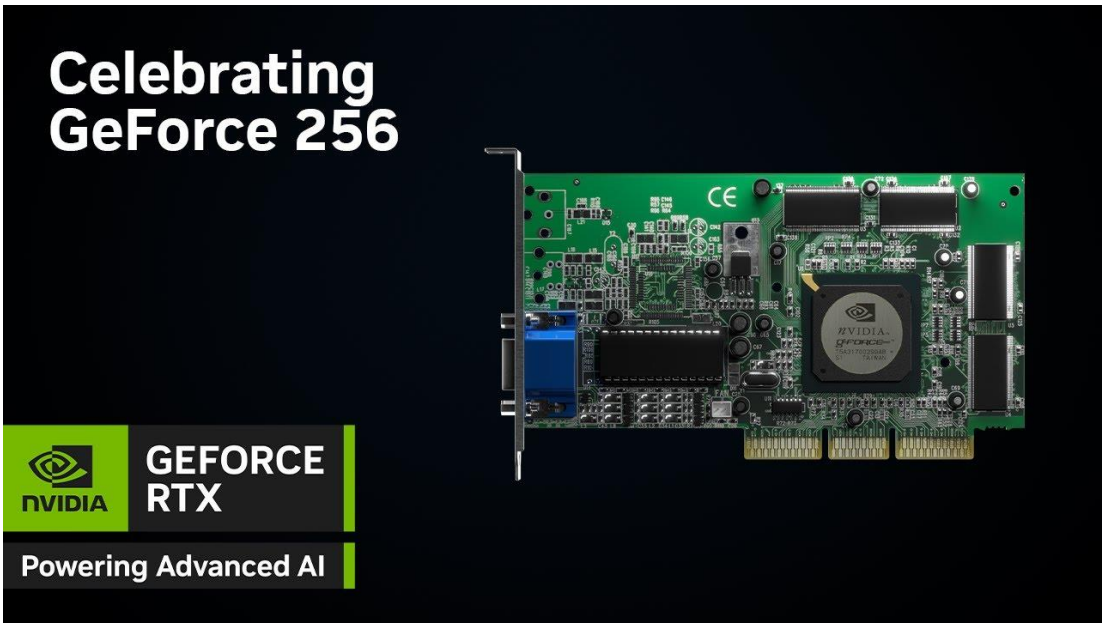
A következő nagy lépést azonban a **programozhatóság** jelentette.

A korábbi grafikus chipek többnyire előre meghatározott műveletsorokat végeztek. A 2000-es évek elején megjelent programozható shader-egységek esetében a programozó már kis programokat futtathatott a GPU-n.

Eredetileg ezek továbbra is grafikai feladatokat szolgáltak: például azt számolták ki, hogyan csillan meg a fény egy fémfelületen, hogyan hullámzik a víz vagy milyen színű legyen egy képpont.

De ezzel valami nagyon fontos történt:

a grafikus chipből programozható párhuzamos processzor lett.



Ábra: *NVIDIA GeForce 256, 1999. A „GPU” elnevezést az NVIDIA ennél a terméknél tette széles körben ismertté. A következő generációk programozható shader-egységei nyitották meg az utat az általános célú GPU-s számítások felé.*

18.4. GPGPU

A kutatók hamar észrevették, hogy ha a GPU programozható, akkor talán nemcsak képpontokat lehet vele számoltatni.

Megpróbáltak teljesen más problémákat **grafikai feladatnak álcázni.**

Az adatokat textúraként töltötték a grafikus memóriába, a matematikai számítást shaderprogrammal végeztették el, majd az eredményt képként olvasták vissza.

Működött – de meglehetősen körülményesen.

Innen származik a **GPGPU – General-Purpose computing on Graphics Processing Units,** vagyis általános célú számítás grafikus processzoron.

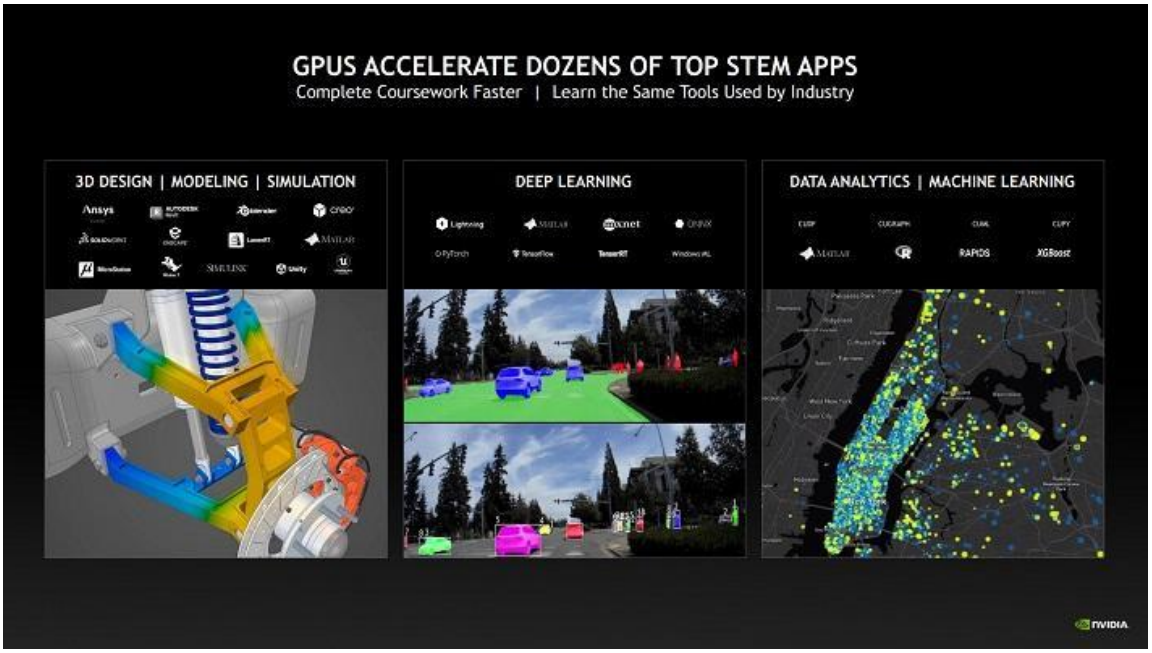
Az igazi áttörést 2006 hozta, amikor az NVIDIA bemutatta a **CUDA** párhuzamos programozási platformot. A programozóknak ettől kezdve nem kellett grafikai problémának álcázniuk a számításaikat: közvetlenül használhatták a GPU számítási egységeit.

A CUDA mellett később más megoldások – például az OpenCL – is lehetővé tették az általános GPU-programozást.

Itt vált igazán külön a GPU története a számítógépes grafikáétól.

18.5. GPU a tudományban





A tudományban számtalan olyan feladat létezik, ahol ugyanazt a műveletet kell hatalmas adatmennyiségen végrehajtani. Pontosan ez az, amiben a GPU kiváló.

GPU-kat kezdtek használni például:

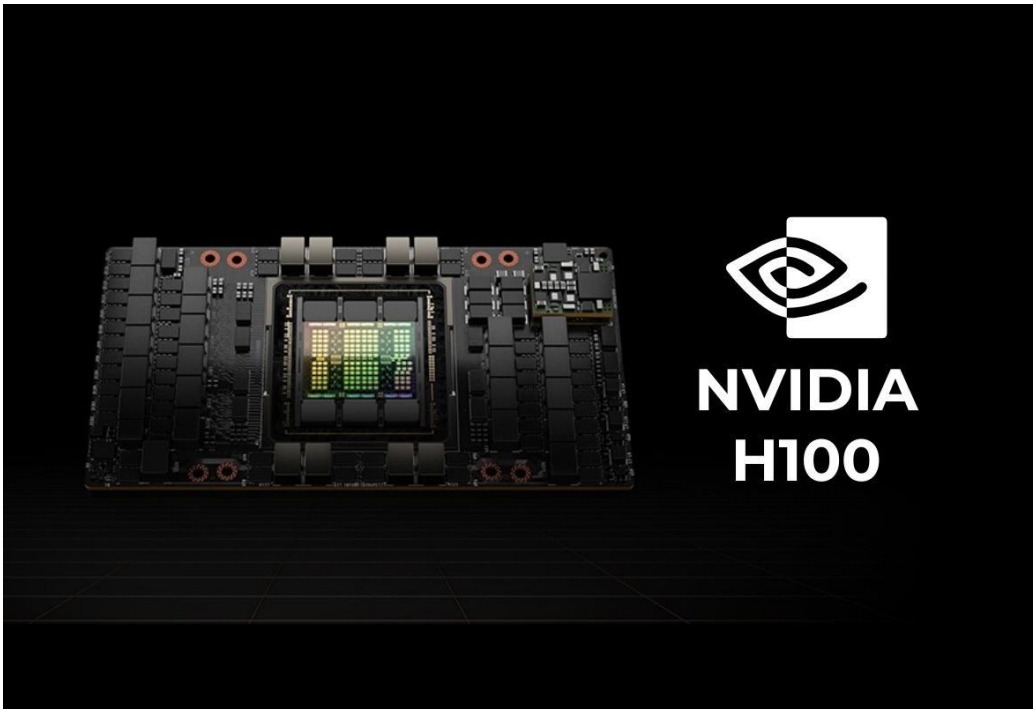
- molekuladinamikai számításokra,
- időjárási és éghajlati modellezésre,
- áramlástani szimulációkra,
- csillagászati számításokra,
- orvosi képalkotásra,
- szeizmikus adatok feldolgozására,
- genomikai számításokra.

A GPU így lassan beköltözött a **szuperszámítógépekbe** is. A mai nagy teljesítményű rendszerek jelentős része CPU-kat és GPU-kat vagy más gyorsítókat együtt használ.

A CPU irányítja a bonyolult programfolyamatot, a GPU pedig átveszi azokat a részeket, amelyek hatalmas mértékben párhuzamosíthatók.

A grafikus kártyából **tudományos gyorsító** lett. A CUDA ma többek között csillagászati, biológiai, kémiai és fizikai alkalmazások gyorsítására használható.

18.6. GPU és mesterséges intelligencia

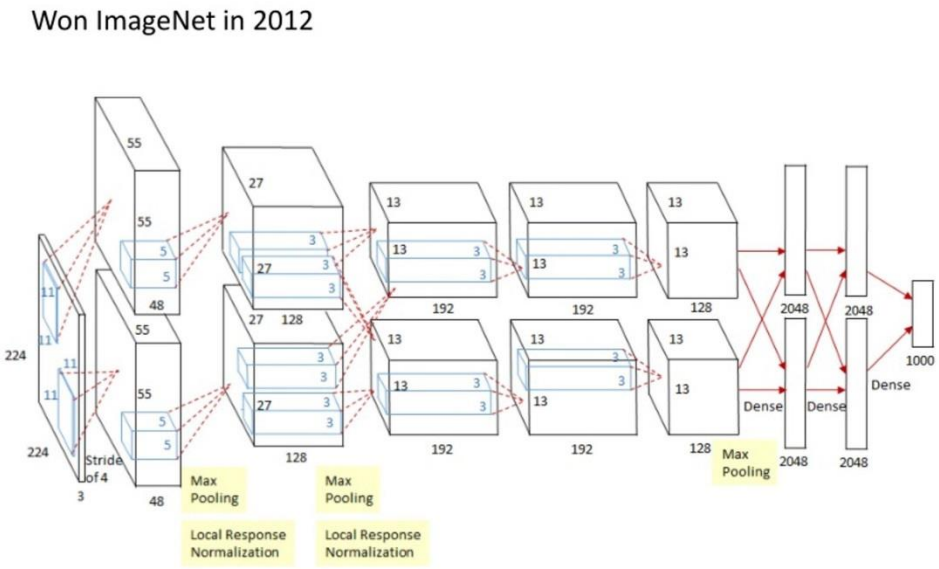


És ekkor következett a GPU történetének talán legmeglepőbb fordulata.

A neurális hálózatok tanítása rengeteg **mátrixműveletből** áll. Ezek nagy része egymással párhuzamosan végezhető.

Vagyis pontosan olyan feladat, amelyre a GPU kiválóan alkalmas.

AlexNet



2012-ben az **AlexNet** képfelismerő neurális hálózat GPU-k segítségével látványos eredményt ért el az ImageNet versenyen. Ez a siker a mélytanulás történetének egyik fordulópontjává vált, és megmutatta, milyen jól használhatók a GPU-k neurális hálózatok tanítására. Az NVIDIA saját történeti áttekintése is 2012-höz köti ezt a fordulatot.

A GPU-k ezután egyre inkább az AI igényeihez is alkalmazkodtak. Megjelentek például a **Tensor Core** egységek, amelyeket kifejezetten a neurális hálózatokban gyakori mátrixműveletek gyorsítására terveztek.

Így alakult ki a különös fejlődési lánc:

képernyővezérlő
↓
grafikus gyorsító
↓
3D gyorsító
↓
programozható GPU
↓
általános célú párhuzamos processzor
↓
tudományos gyorsító
↓
AI-gyorsító

A játékosok építették meg az AI motorját?

Első pillantásra meglepőnek tűnhet, de a mesterséges intelligencia fejlődésének egyik legfontosabb mozgatórugója hosszú időn át a **számítógépes játékok világa** volt.

Ennek egyszerű oka van: a játékok jól meghatározott szabályok szerint működnek, a számítógép azonnal visszajelzést kap arról, hogy egy lépés jó vagy rossz volt, és ugyanazt a helyzetet akár milliószor is újrajátszhatja. Ez ideális környezet a tanuláshoz.

Az első nagy mérföldkő a sakk volt. 1997-ben az IBM **Deep Blue** számítógépe legyőzte Garri Kaszparov világbajnokot. Bár ez a rendszer még főként hatalmas számítási teljesítményre épült, bebizonyította, hogy a gépek képesek felülmúlni az embert összetett stratégiai feladatokban.

A következő áttörést a **Go** játék hozta. A lehetséges lépések száma olyan óriási, hogy puszta számolással nem kezelhető. A DeepMind által fejlesztett **AlphaGo** ezért már neurális hálózatokat és önálló tanulást alkalmazott. 2016-ban legyőzte a világ egyik legerősebb játékosát, Li Szedolt, ami sok kutató szerint a mesterséges intelligencia történetének egyik legfontosabb pillanata volt.

Ezután következett az **AlphaZero**, amelynek már nem volt szüksége emberi játszmákra. Mindössze a szabályokat ismerte, majd önmaga ellen játszva néhány óra alatt emberfeletti szintre jutott sakkban, góban és sógiban is.

A videojátékok újabb kihívásokat jelentettek. Az Atari játékai megtanították a gépeket képekből döntéseket hozni, a **StarCraft II** pedig már valós idejű stratégiai gondolkodást, hosszú távú tervezést és hiányos információk kezelését követelte meg. Ezek a feladatok sokkal közelebb állnak a való élet problémáihoz.

Az itt kifejlesztett algoritmusok ma már messze túlmutatnak a játékokon. Ugyanazok az eljárások segítenek robotok irányításában, önvezető járművek fejlesztésében, gyógyszerkutatásban, energiafelhasználás optimalizálásában vagy éppen nagy nyelvi modellek tanításában.

Érdekesség: amikor ma egy számítógépes játékban intelligens ellenféllel találkozunk, érdemes arra gondolni, hogy a játékok nemcsak szórakoztatnak. Évtizedeken át olyan „edzőtermek” voltak, ahol a mesterséges intelligencia megtanulta, hogyan hozzon egyre jobb döntéseket.

Tanulság: Sokszor a játék nem a valóság ellentéte, hanem a jövő technológiáinak kísérleti laboratóriuma.

A PC-s játékok hatalmas piaca évtizedeken át finanszírozta az egyre gyorsabb grafikus processzorok fejlesztését. A játékosok jobb felbontást, részletesebb világokat, valóság hűbb fényeket és nagyobb képfrissítést akartak.

Ehhez egyre nagyobb párhuzamos számítási teljesítmény kellett.

Amikor pedig a kutatók rájöttek, hogy ugyanez a hardver a neurális hálózatok matematikájához is kitűnően használható, már rendelkezésre állt egy évtizedek alatt kifejlesztett, tömeggyártásban készülő processzortípus és a hozzá tartozó programozási ökoszisztéma. A GPU fejlődése ezért jó példája annak, hogy egy technológia **egészen más területen válhat igazán fontossá, mint amire eredetileg megalkották**.



Bitcoin-bányászat – amikor a videokártya pénzt kezdett „termelni"

A grafikus processzorok egyik legkülönösebb felhasználási területe a kriptovaluták bányászata lett. A történet 2009-ben, a **Bitcoin** megjelenésével kezdődött.

A Bitcoin mögött nincs bank, amely egy központi nyilvántartásban vezeti, kinek mennyi pénze van. A tranzakciókat egy sok számítógépen tárolt, közösen ellenőrzött adatbázis, a **blokklánc** tartalmazza.

De ki dönti el, mely új tranzakciók kerülhetnek bele?

A Bitcoin rendszerében ezért verseny zajlik a hálózat résztvevői között. A „bányászok" számítógépei egy kriptográfiai hash-függvényt, az **SHA-256-ot** számolják újra és újra, mindig kissé módosított bemenettel. Olyan eredményt keresnek, amely megfelel a hálózat által előírt feltételnek.

Itt nincs ügyes matematikai rövidítés: **rengeteg próbálkozást kell rendkívül gyorsan végrehajtani.**

Éppen ez az a feladat, amelyre a párhuzamos feldolgozás kiválóan alkalmas.

A Bitcoin első időszakában még hagyományos CPU-val is lehetett bányászni. 2010 körül azonban megjelent a **GPU-s bányászat**. Egy grafikus processzor sok számítási egysége egyszerre hatalmas mennyiségű hash-műveletet tudott elvégezni, ezért nagyságrendekkel hatékonyabbnak bizonyult a CPU-nál.

A fejlődés azonban nem állt meg:

CPU → GPU → FPGA → ASIC

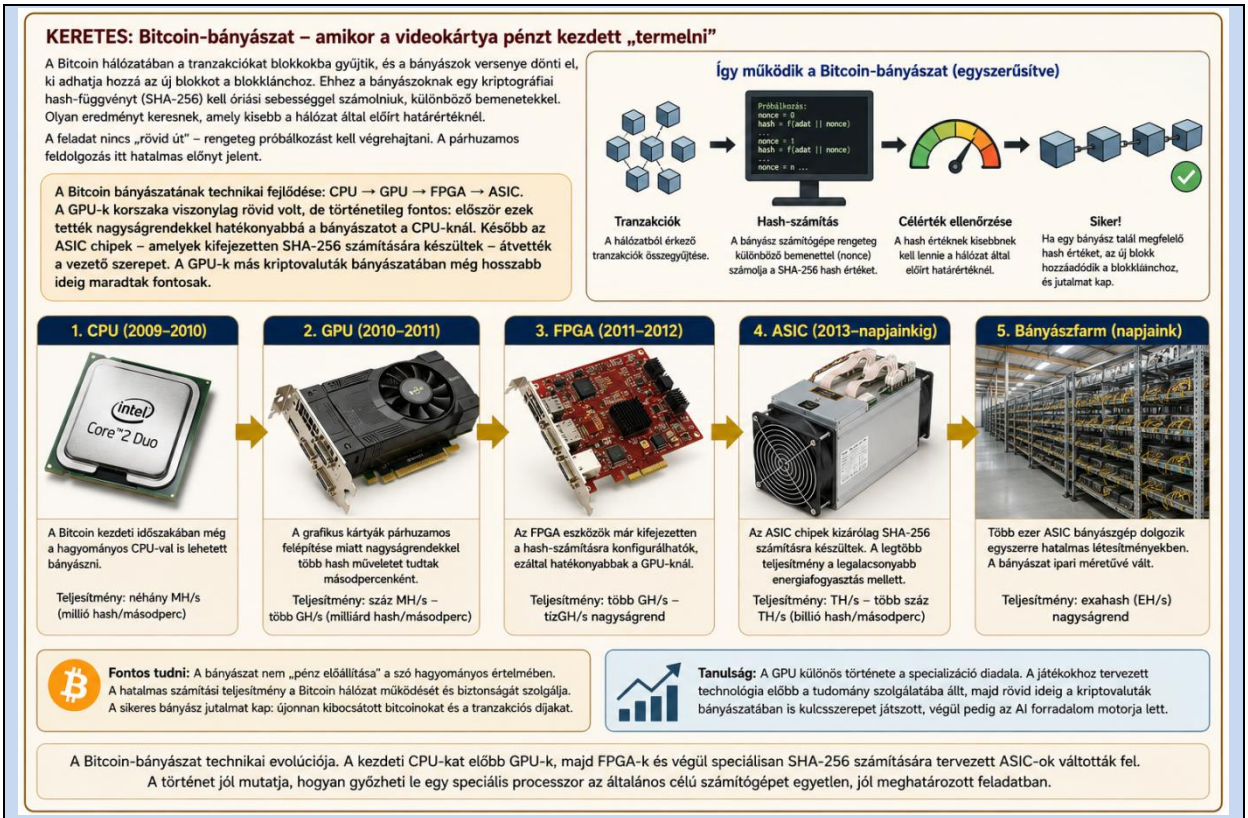
Az ASIC – *Application-Specific Integrated Circuit* – már kifejezetten egy adott feladatra tervezett integrált áramkör. A Bitcoin esetében ezek a chipek az SHA-256 számítására vannak optimalizálva. Sokkal hatékonyabbak a GPU-knál, ezért a Bitcoin ipari méretű bányászatából a grafikus kártyákat gyakorlatilag kiszorították.

A GPU-k történetében azonban a kriptobányászat fontos epizód maradt. Más kriptovaluták bányászatában még hosszabb ideig jelentős szerepet játszottak, és a kriptovaluta-láz időszakaiban olyan nagy lett irántuk a kereslet, hogy **világszerte megdrágultak és időnként hiánycikké váltak a videokártyák.**

A történet egyben a specializáció szép példája is:

CPU: szinte bármit meg tud csinálni.
GPU: sok hasonló számítást tud egyszerre elvégezni.
ASIC: kevés dolgot tud – de azt rendkívül gyorsan és hatékonyan.

A Bitcoin-bányászat tehát valójában nem „pénz előállítása” a szó hagyományos értelmében. A hatalmas számítási teljesítmény a Bitcoin hálózatának működését és biztonságát szolgálja, miközben a sikeres bányász jutalmat kaphat érte.



Ábra: *A Bitcoin-bányászat technikai evolúciója. A kezdeti CPU-kat előbb GPU-k, majd FPGA-k és végül speciálisan SHA-256 számítására tervezett ASIC-ok váltották fel. A történet jól mutatja, hogyan győzheti le egy speciális processzor az általános célú számítógépet egyetlen, jól meghatározott feladatban.*

VIII. rész – A programozott géptől a tanuló gépig

19. A mesterséges intelligencia története

Itt valóban érdemes **szigorúan technikatörténeti pályán maradni**. Nem azt vizsgáljuk, hogy „gondolkodik-e” vagy „ért-e” a gép, hanem azt, hogyan jutottunk el a kézzel megadott szabályoktól a tanuló neurális hálózatokon át a mai generatív rendszerekig.

A történet ráadásul egyáltalán nem egyenes vonalú. A mesterséges intelligenciát többször beharangozták mint közeli forradalmat, majd csalódás következett. A valódi áttöréshez végül **algoritmusok, hatalmas adatmennyiség és nagy számítási teljesítmény** egyszerre kellett.

19.1. Turing kérdése

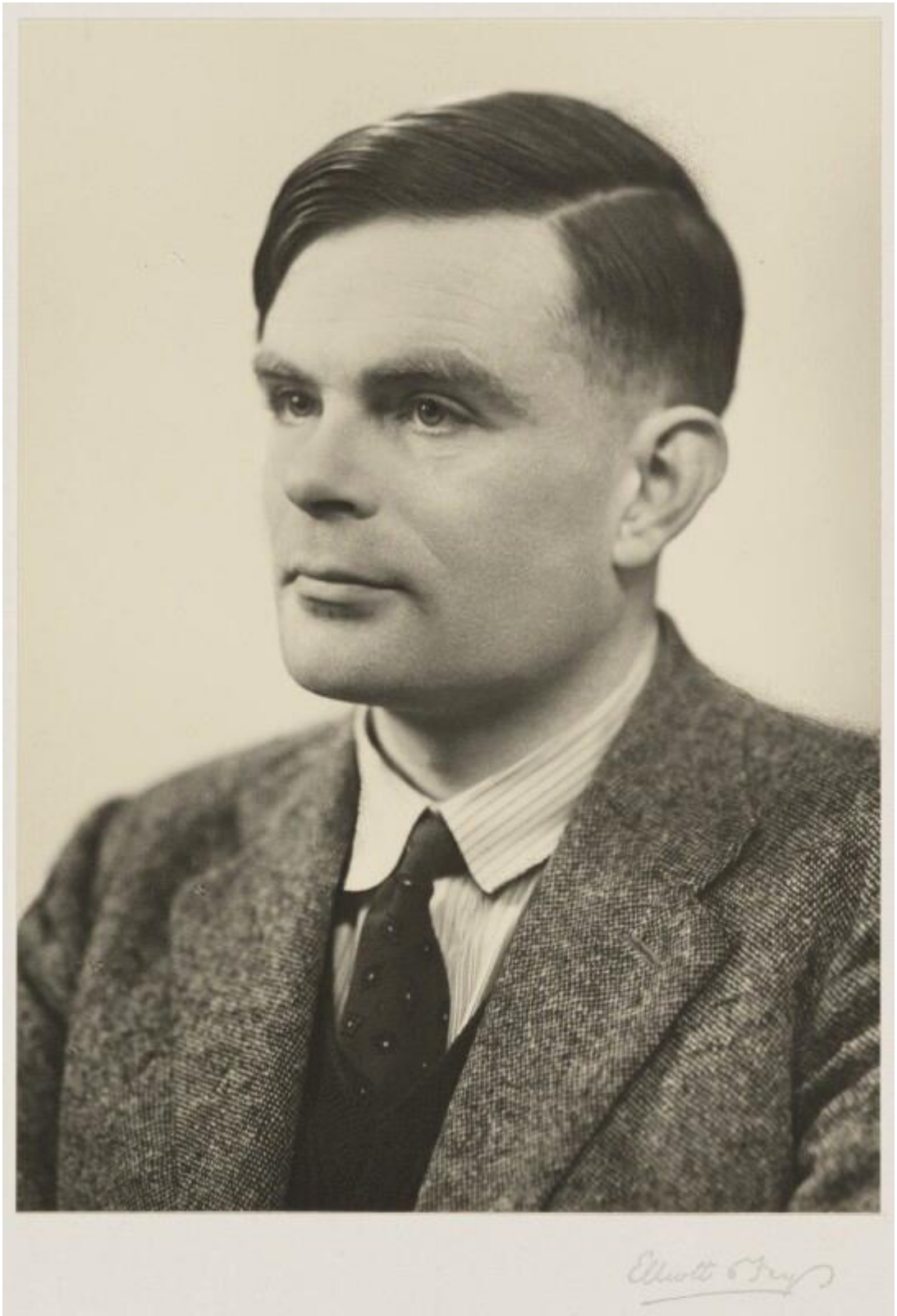
1950-ben **Alan Turing** „Computing Machinery and Intelligence” című tanulmányát egy provokatív kérdéssel kezdte:

„Can machines think?” – Tudnak-e gondolkodni a gépek?

Turing azonban gyorsan felismerte, hogy a „gondolkodás” fogalmán vég nélkül lehet vitatkozni. Ezért gyakorlati próbát javasolt, az **utánzási játékot**, amelyből később a Turing-teszt néven ismert gondolatkísérlet született.

Egy ember írásban beszélget egy másik emberrel és egy géppel. Ha a válaszok alapján nem tudja megbízhatóan eldönteni, melyik a gép, akkor a gép sikeresen teljesítette a próbát.

Technikatörténeti szempontból azonban még fontosabb, hogy Turing már 1950-ben felvetette: talán nem minden intelligens viselkedést kell kézzel beprogramoznunk. Lehetne olyan gépet készíteni, amely **tanul.**



Ábra: *Alan Turing 1951-ben. Egy évvel korábban megjelent tanulmányában feltette a mesterséges intelligencia történetének egyik leghíresebb kérdését: „Tudnak-e gondolkodni a gépek?”*

19.2. Dartmouth – megszületik az AI neve

1956 nyarán a Dartmouth College-ban kutatók kis csoportja gyűlt össze egy műhelykonferenciára. A kezdeményezők között volt **John McCarthy, Marvin Minsky, Claude Shannon és Nathaniel Rochester**.

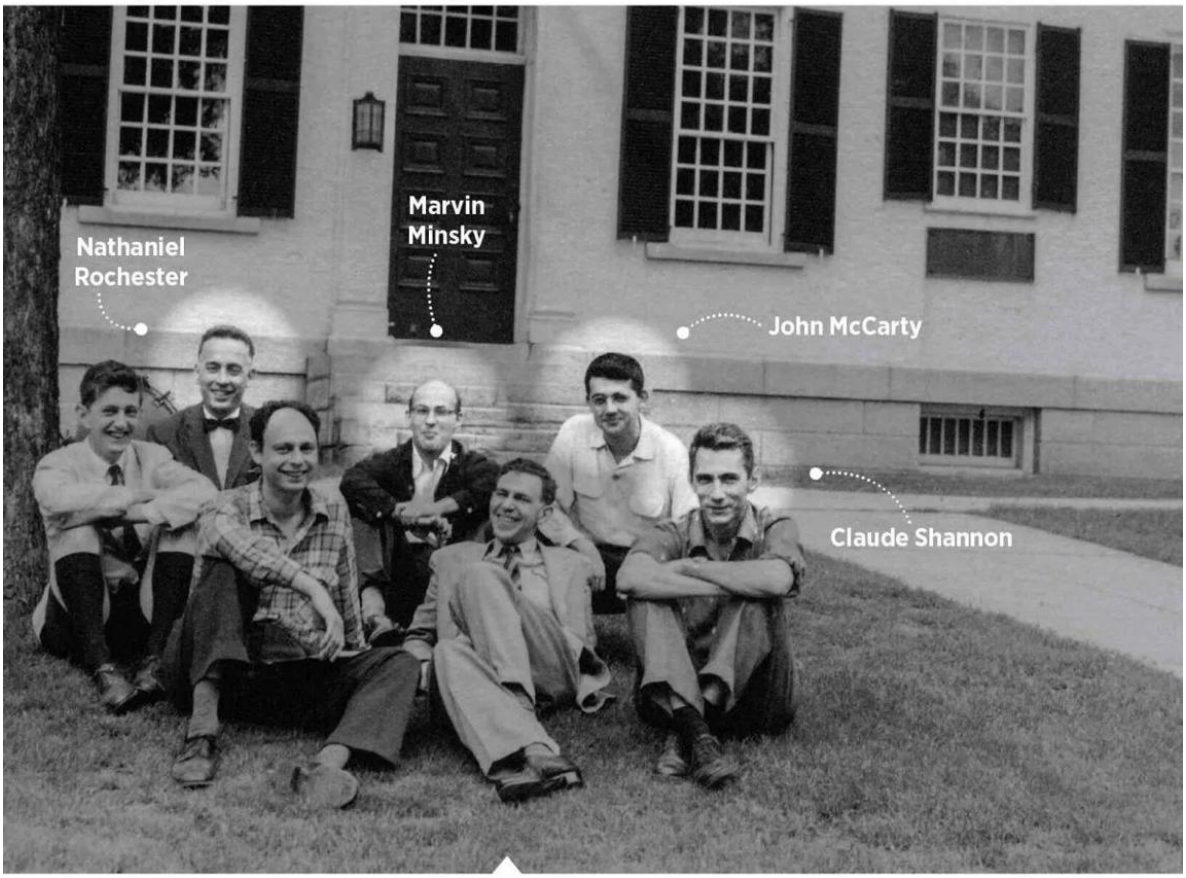
A konferencia javaslatában szerepelt az új elnevezés:

Artificial Intelligence – mesterséges intelligencia.

A résztvevők meglehetősen optimisták voltak. Úgy gondolták, hogy a tanulás és az intelligencia más jelenségei elvileg olyan pontosan leírhatók, hogy géppel szimulálhatók.

Az AI ezzel önálló kutatási területté kezdett válni.

A korai programok főként szimbólumokkal és logikai szabályokkal dolgoztak. Sakk, matematikai bizonyítás és különféle rejtvények megfelelő terepnek látszottak, mert ezek szabályai egyértelműen megfogalmazhatók.



Ábra: *Az 1956-os Dartmouth Summer Research Project on Artificial Intelligence az AI történetének jelképes kezdőpontja. Itt vált szélesebb körben használttá az „artificial intelligence” elnevezés.*

19.3. Perceptron

Az AI történetének másik iránya nem szabályokat próbált írni, hanem az idegrendszer működéséből merített ötletet.

Frank Rosenblatt az 1950-es évek végén fejlesztette ki perceptronját. A rendszer egyszerű mesterséges neuronokból állt, és példák alapján képes volt módosítani a kapcsolatok súlyait.

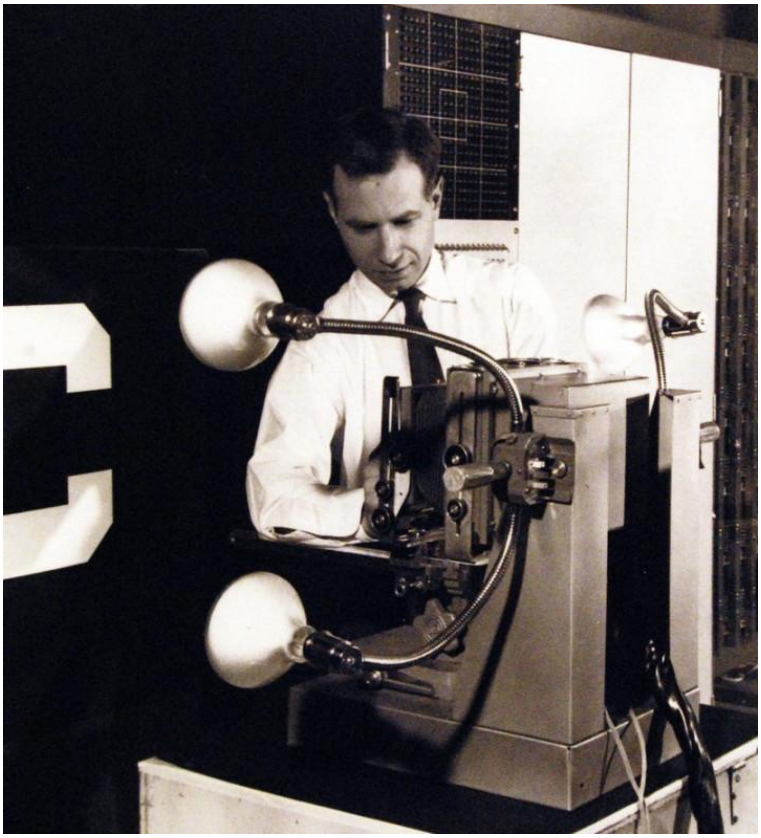
Ez alapvetően más gondolat volt, mint a hagyományos programozás.

Nem azt mondjuk: **„Ha ezt látod, tedd ezt."**

Hanem példákat mutatunk: **„Ezek ilyenek, azok olyanok – találd meg a különbséget."**

Rosenblatt Mark I Perceptronja hardveresen is megépült. Fényérzékelők, elektronikus áramkörök és állítható kapcsolatok alkották.

A korai eredmények hatalmas várakozásokat keltettek, de a perceptron képességei erősen korlátozottak voltak.



Ábra: *Frank Rosenblatt és a perceptronkutatások egyik korai berendezése. A perceptron már nem csupán végrehajtotta a programozó szabályait: példák alapján módosította saját paramétereit.*

19.4. Neurális hálózatok

Egyetlen mesterséges neuron természetesen nagyon keveset tud. Ha azonban sokat összekapcsolunk, **neurális hálózatot** kapunk.

A hálózat rétegekből épülhet fel:

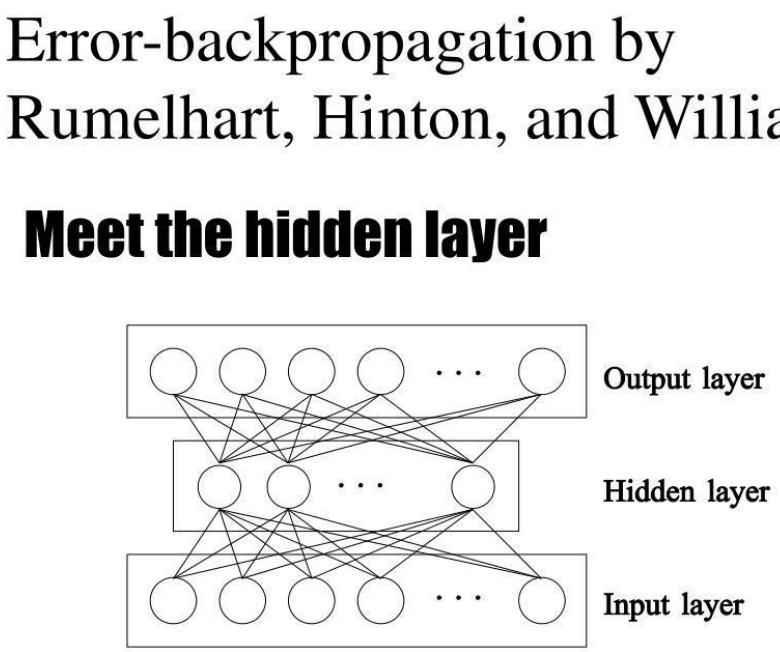
bemeneti réteg → rejtett rétegek → kimeneti réteg

A kapcsolatokhoz számértékek, úgynevezett **súlyok** tartoznak. A tanítás során lényegében ezeket változtatjuk úgy, hogy a hálózat egyre jobb eredményt adjon.

Az egyik kulcsfontosságú módszer a **backpropagation**, vagyis hibavisszaterjesztés lett. Az algoritmus kiszámítja, hogy a hálózat egyes súlyai mennyiben járultak hozzá a hibához, majd ennek megfelelően módosítja őket.

Az alapötletek már évtizedekkel korábban léteztek, de a neurális hálózatok igazán nagy sikereihez még hiányzott két dolog:

elegendő adat és elegendő számítási teljesítmény.



Ábra: *A többrétegű neurális hálózatban a bemeneti és kimeneti réteg között rejtett rétegek dolgozzák fel az információt. A tanítás során a hálózat kapcsolatainak súlyai változnak.*

19.5. Szakértői rendszerek

Az 1970-es és 1980-as években az AI egyik legsikeresebb iránya egészen más megközelítést alkalmazott.

A **szakértői rendszer** megpróbálta egy emberi szakértő tudását szabályok formájában számítógépbe vinni.

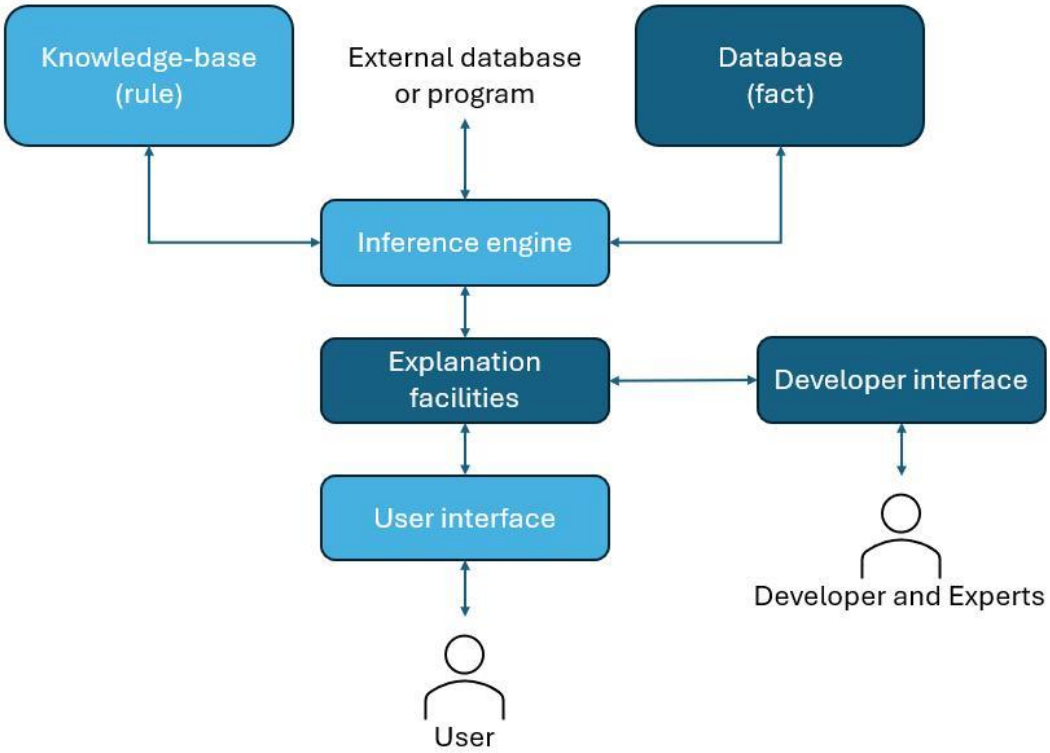
Például:

HA a betegnek magas láza van
ÉS bizonyos laboreredmények pozitívak,
AKKOR valószínűsíthető egy adott fertőzés.

Az egyik híres rendszer az 1970-es évek **MYCIN** programja volt, amely bakteriális fertőzésekkel és antibiotikum-választással kapcsolatos döntéseket vizsgált.

A szakértői rendszerekhez két fontos elem kellett:

tudásbázis + következtető rendszer.



Ábra: *A MYCIN az 1970-es évek híres szakértői rendszere volt. Tudásbázisban tárolt szabályok alapján próbált bakteriális fertőzésekkel kapcsolatos orvosi döntéseket támogatni.*

Az elképzelés ipari alkalmazásokat is eredményezett, de volt egy súlyos problémája. Az emberi szakértők tudását rendkívül nehéz volt szabályok ezreire lefordítani, a szabályrendszerek karbantartása pedig egyre bonyolultabbá vált.

19.6. AI-telek

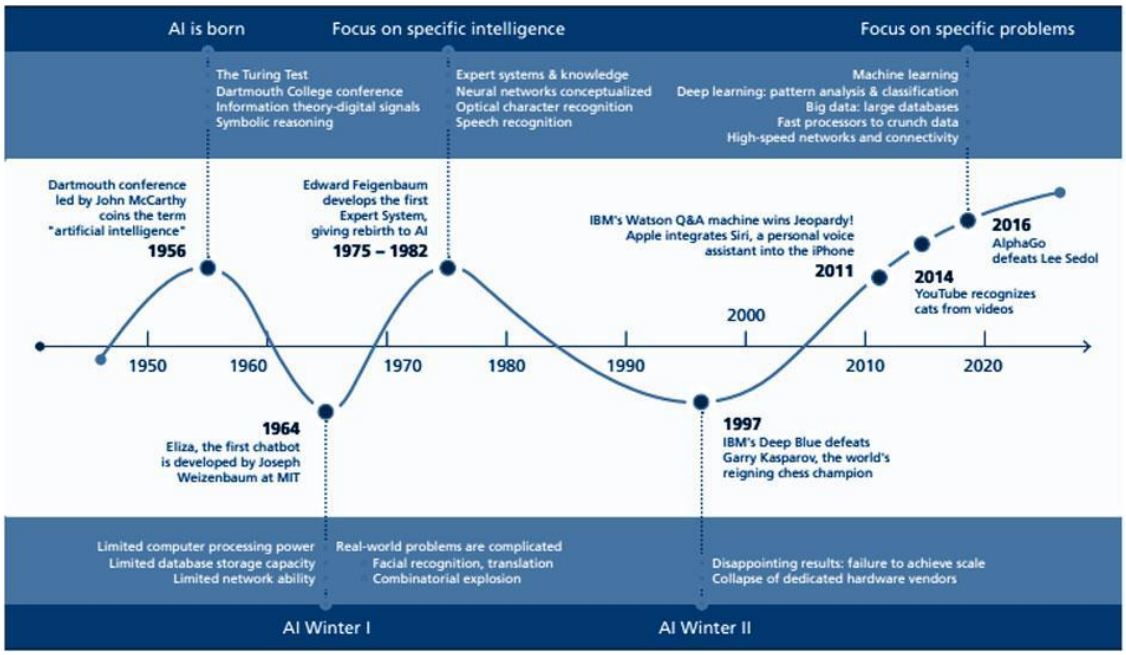
A mesterséges intelligencia története többször ugyanazt a mintát követte:

nagy ígéretek → hatalmas várakozások → csalódás → támogatások csökkenése.

Ezeket az időszakokat nevezzük **AI-teleknek**.

Az első jelentős visszaesés az 1970-es években következett. A korabeli gépek lassúak voltak, kevés memóriával rendelkeztek, és az AI-programok a gondosan kiválasztott bemutatófeladatokon kívül sokszor gyengén teljesítettek.

Újabb visszaesés következett az 1980-as évek végén és az 1990-es évek elején, amikor a szakértői rendszerekhez és a speciális AI-hardverekhez fűzött üzleti remények egy része sem teljesült.



Az AI azonban nem halt meg.

A kutatás folytatódott – csak kevesebb reflektorfényben.

19.7. Gépi tanulás

A következő fordulatot egy szemléletváltás hozta.

Ahelyett, hogy megpróbálnánk **minden szabályt előre megadni**, adjunk a számítógépnek adatokat, és engedjük, hogy statisztikai módszerekkel maga találjon mintázatokat.

Ez a **machine learning – gépi tanulás**.

A módszerek között megjelentek és elterjedtek többek között a döntési fák, Bayes-módszerek, klaszterező algoritmusok, support vector machine-ek és különböző neurális hálózatok.

A számítógép immár nem feltétlenül ember által megfogalmazott szabályokból dolgozott.

A szabály egy részét az adatokból tanulta meg.

A korszak szimbolikus eseménye volt 1997, amikor az IBM **Deep Blue** sakk-számítógépe mérkőzésen legyőzte Garri Kaszparov világbajnokot. Fontos azonban, hogy a Deep Blue nem a mai értelemben vett mélytanuló rendszer volt: speciális sakkhardverrel, hatalmas keresési teljesítménnyel, értékelő függvényekkel és sakk-szakértők által kialakított tudással dolgozott.



Ábra: *Garri Kaszparov az IBM Deep Blue elleni 1997-es visszavágón. A gép győzelme látványosan demonstrálta, hogy speciális algoritmusokkal és óriási számítási teljesítménnyel egy számítógép már a sakkvilágbajnokot is felülmúlhatja.*

19.8. Deep learning

A neurális hálózatok régi ötlete a 2000-es és 2010-es években új életre kelt.

Három dolog találkozott:

nagy adathalmazok + jobb algoritmusok + GPU-k.

És itt kapcsolódik közvetlenül ez a fejezet az előzőhöz.

A játékok számára kifejlesztett GPU-k hatalmas párhuzamos számítási kapacitása kiválóan megfelelt a neurális hálózatok tanításához szükséges mátrixműveletekhez.

A mesterséges intelligencia fejlődésének motorja egyre kevésbé az ember által írt szabályrendszer, és egyre inkább **az adatokból megtanult matematikai modell** lett.

19.9. Transformer

2017-ben jelent meg az **„Attention Is All You Need”** című tanulmány, amely bemutatta a transformer architektúrát.

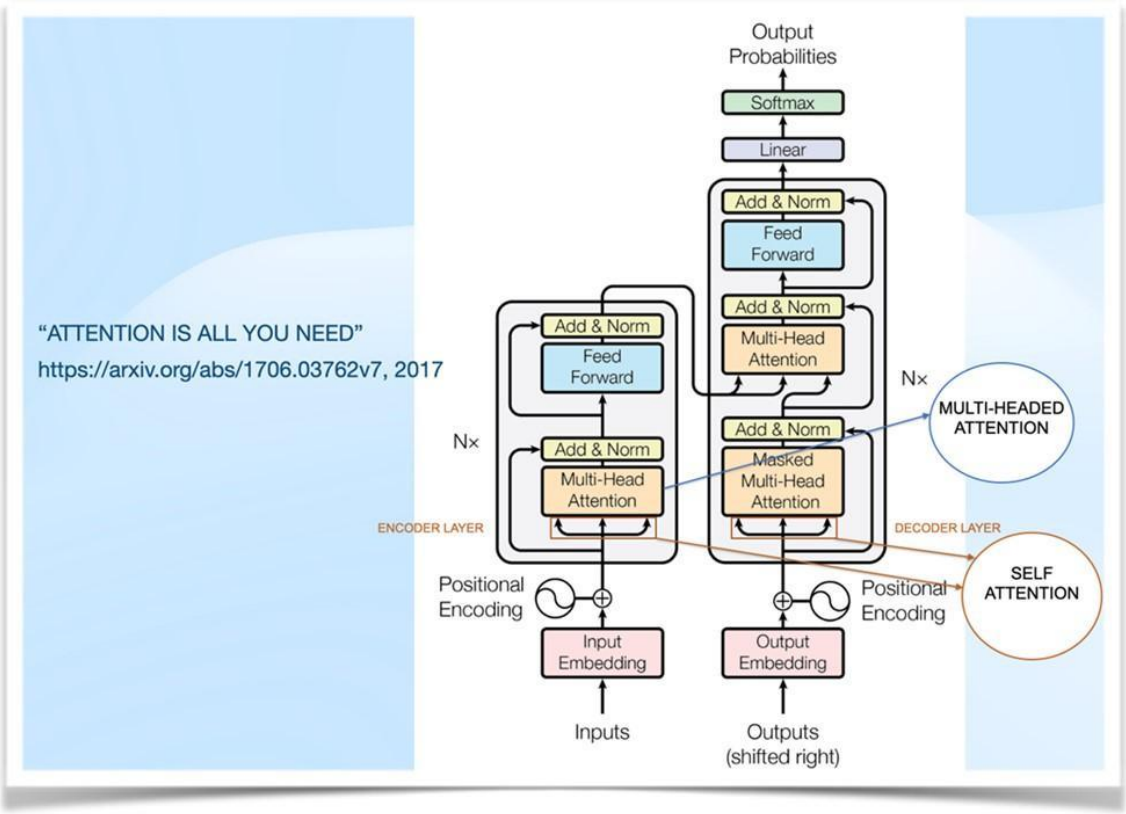
A transformer egyik kulcseleme az **attention**, vagyis figyelmi mechanizmus. Ennek segítségével a modell egy szöveg feldolgozásakor képes meghatározni, hogy az egyes elemek között milyen kapcsolatok fontosak.

A korábbi szekvenciális modellekhez képest a transformer számításai sokkal jobban párhuzamosíthatók voltak.

Ez döntő előnyt jelentett.

A GPU-k és később az AI-ra specializált gyorsítók segítségével egyre nagyobb transformer modelleket lehetett tanítani egyre nagyobb szövegállományokon.

A transformer lett a modern nagy nyelvi modellek egyik alapvető technológiai építőköve.



Ábra: *A 2017-ben bemutatott transformer eredeti architektúrájának sematikus felépítése. A figyelmi mechanizmus és a jól párhuzamosítható számítás alapvetően megváltoztatta a természetes nyelv gépi feldolgozását.*

19.10. Nagy nyelvi modellek

A következő lépés első pillantásra meglepően egyszerű volt:

építsünk egyre nagyobb modelleket, és tanítsuk őket egyre több szövegen.

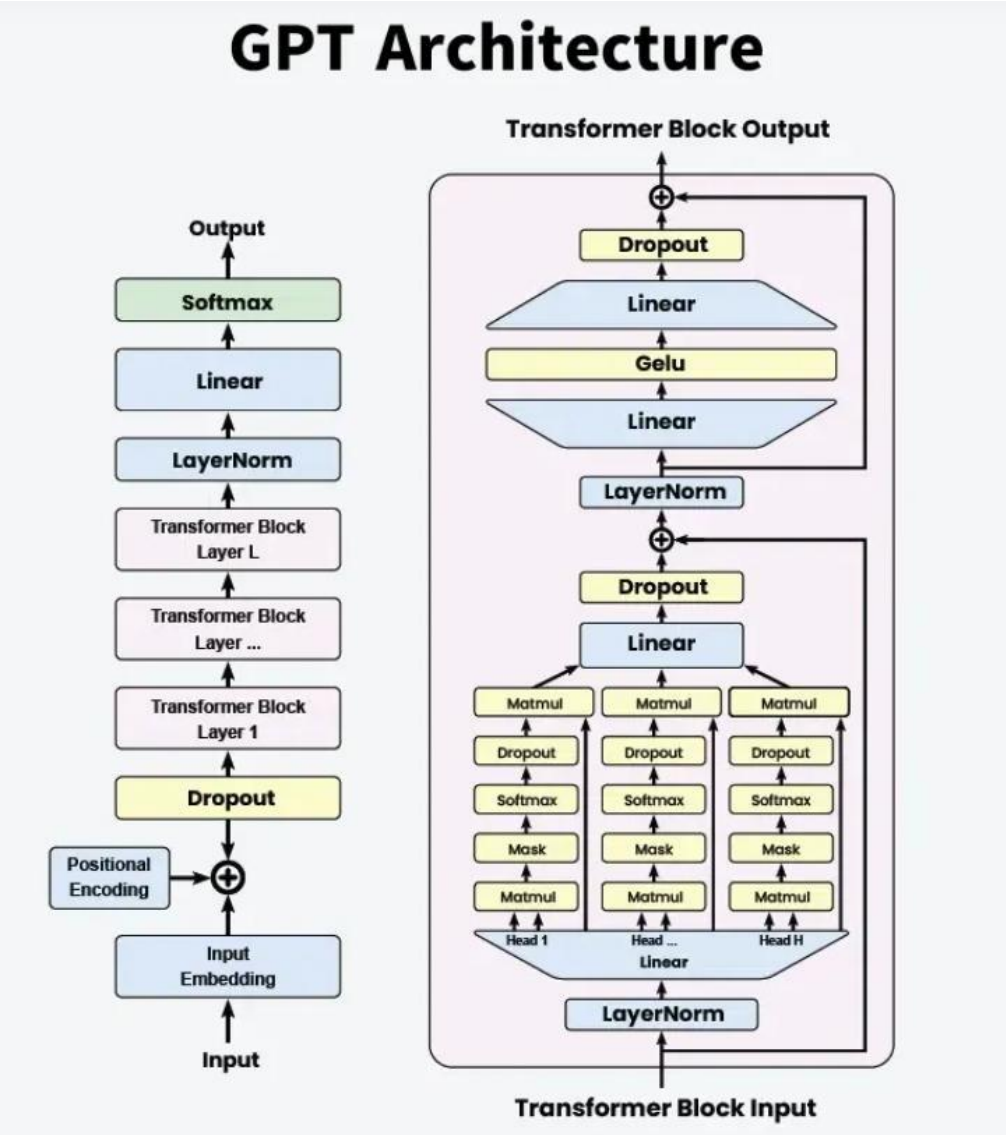
A nagy nyelvi modell – **Large Language Model, LLM** – hatalmas mennyiségű szövegből tanulja meg a nyelvi mintázatokat.

Technikai szempontból fontos változás történt. Régebben külön programot készítettünk fordításra, szövegosztályozásra, kérdések megválaszolására vagy összefoglalásra.

A nagy nyelvi modell ezzel szemben **általános alapmodell** lehet, amely sok különböző feladatot képes elvégezni.

A GPT elnevezés is erre utal:

Generative Pre-trained Transformer.



Generatív, mert új tartalmat állít elő; előtanított, mert hatalmas adathalmazon előzetesen tanul; transformer, mert erre az architektúrára épül.

A modell méretével együtt a tanítás számítási igénye is óriásira nőtt. A modern AI ezért már elválaszthatatlan az adatközpontoktól és a GPU-k, illetve más AI-gyorsítók ezreit összekapcsoló számítógépes rendszerektől.

19.11. Generatív mesterséges intelligencia

A mesterséges intelligencia sokáig elsősorban **felismert, osztályozott, keresett és előrejelzett**.

A generatív AI viszont létrehoz.

Szöveget, programkódot, képet, zenét, beszédet és videót képes generálni a tanult mintázatok alapján.

A 2020-as évek elején ezek a technológiák kiléptek a kutatólaboratóriumokból. A képgenerátorok, majd 2022 végétől a **ChatGPT** tömegek számára tették közvetlenül megtapasztalhatóvá a generatív mesterséges intelligenciát.

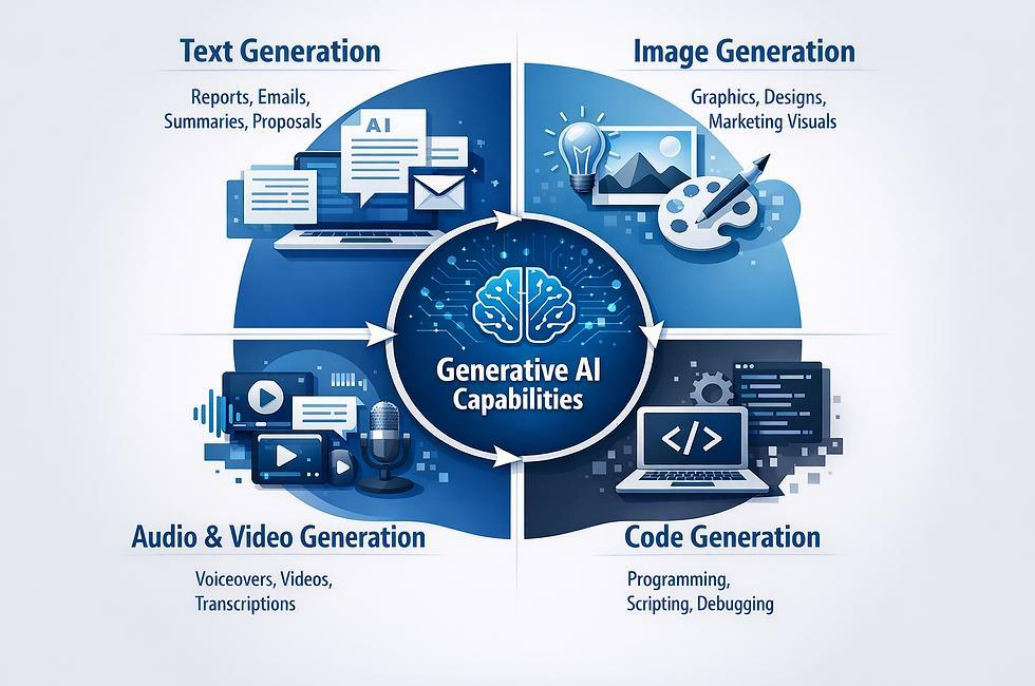
A technológiai fejlődés közben több korábban különálló terület kezdett összekapcsolódni. A **multimodális modellek** már nem kizárólag szöveggel dolgoznak: képet, hangot és más információtípusokat is képesek feldolgozni, illetve létrehozni.

Ezzel érdekes módon visszaérkezünk a számítógép történetének egyik alapgondolatához.

A gép kezdetben **számokat** dolgozott fel.

Később betűket, képeket és hangokat.

Most pedig mindezeket ugyanannak a digitális információs világnak különböző megjelenési formáiként kezeli.



Ábra: *A generatív mesterséges intelligencia a 2020-as években tömegtechnológiává vált. A számítógép már nemcsak adatokat elemez, hanem szöveget, képet, hangot, programkódot és más új tartalmat is képes létrehozni.*

Mikor programozunk és mikor tanítunk egy számítógépet?

Hagyományos programozás

A programozó ismeri az eljárást, és utasítások formájában megadja a számítógépnek.

ADAT + PROGRAM → EREDMÉNY

Ha például egy fizetésből ki kell számítani az adót, pontosan ismerjük a szabályokat. Ezeket algoritmussá alakítjuk, a számítógép pedig végrehajtja őket.

A programozó lényegében azt mondja: **„Megmondom, hogyan kell megoldani."**

Gépi tanulás

Mi történik azonban, ha azt szeretnénk eldönteni egy fényképről, hogy macska vagy kutya látható rajta?

Megpróbálhatnánk szabályokat írni:

„Ha hegyes a füle…"
„Ha ilyen alakú az orra…"
„Ha ekkora a szeme…"

Hamar kiderülne, hogy reménytelenül sok kivétellel találkozunk.

Ehelyett mutatunk a rendszernek sok megfelelően megjelölt példát:

ADAT + HELYES EREDMÉNYEK → TANÍTÁS → MODELL

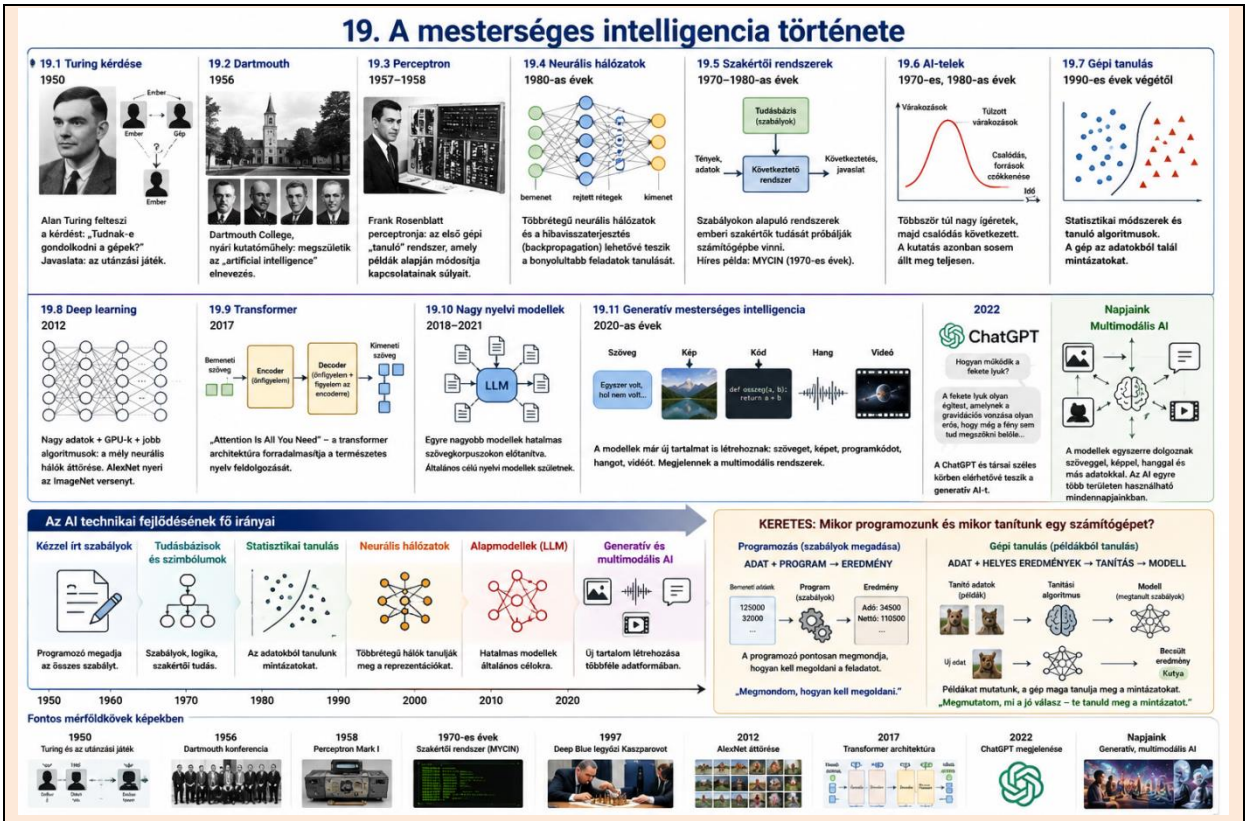
Ezután: **ÚJ ADAT + MODELL → BECSÜLT EREDMÉNY**

A programozó itt már nem minden felismerési szabályt ír le. A tanító algoritmust, a modell felépítését és a tanítási eljárást készíti el, a konkrét paraméterek jelentős részét pedig **a gép az adatokból állítja be.**

A különbséget egyetlen mondatban így lehetne összefoglalni:

A hagyományos programozásnál megadjuk a gépnek a szabályokat. A gépi tanulásnál példákat adunk neki, amelyekből modellt tanul.

És fontos hozzátenni: a kettő nem váltja ki egymást. A mai AI-rendszerek körül rengeteg hagyományosan programozott szoftver működik. Maga a tanító algoritmus is program.



A technológia sohasem csak egy új eszköz

„Walter, hogyan fogod rábírni ezeket a robotokat, hogy fizessenek szakszervezeti díjat?" – kérdezte Ford.

Reuther azonnal válaszolt: „Henry, és Te hogyan fogod rávenni őket, hogy vásárolják meg az autóidat?"

Hajlamosak vagyunk úgy tekinteni egy új technológiára, mintha egyszerűen hozzáadna valamit a már létező világhoz. A telefon gyorsabbá tette a kommunikációt, az autó a közlekedést, a számítógép a számolást, az internet pedig az információcserét. Csakhogy a történelmi tapasztalat szerint ennél sokkal több történik.

Neil Postman amerikai médiateoretikus ezt nevezte a technológia **„ökológiai" hatásának.** Ha egy ökoszisztémába új élőlény kerül, nem egyszerűen eggyel több faj lesz benne: megváltozhatnak a táplálkozási láncok, az egyensúlyok, végső soron az egész rendszer. Postman szerint a technológia hasonlóan működik: **„Egy új technológia sem egyszerűen hozzáad, vagy elvesz valamit. Mindent átalakít."** Molnár Szilárd ezt a gondolatot használja a negyedik ipari forradalom társadalmi hatásainak magyarázatára.

Az emberiség története tele van ilyen átalakulásokkal. Az **írás** nem csupán lehetővé tette a gondolatok rögzítését: megváltoztatta az emlékezés szerepét. A **könyvnyomtatás** nem egyszerűen olcsóbbá tette a könyvet: átalakította a tudás terjedését. A **telefon** nem csupán gyorsabb levelezés lett: új értelmet adott a távolságnak. A **fényképezés** nemcsak képeket készített: megváltoztatta azt is, hogyan őrizzük meg személyes és közös múltunkat.

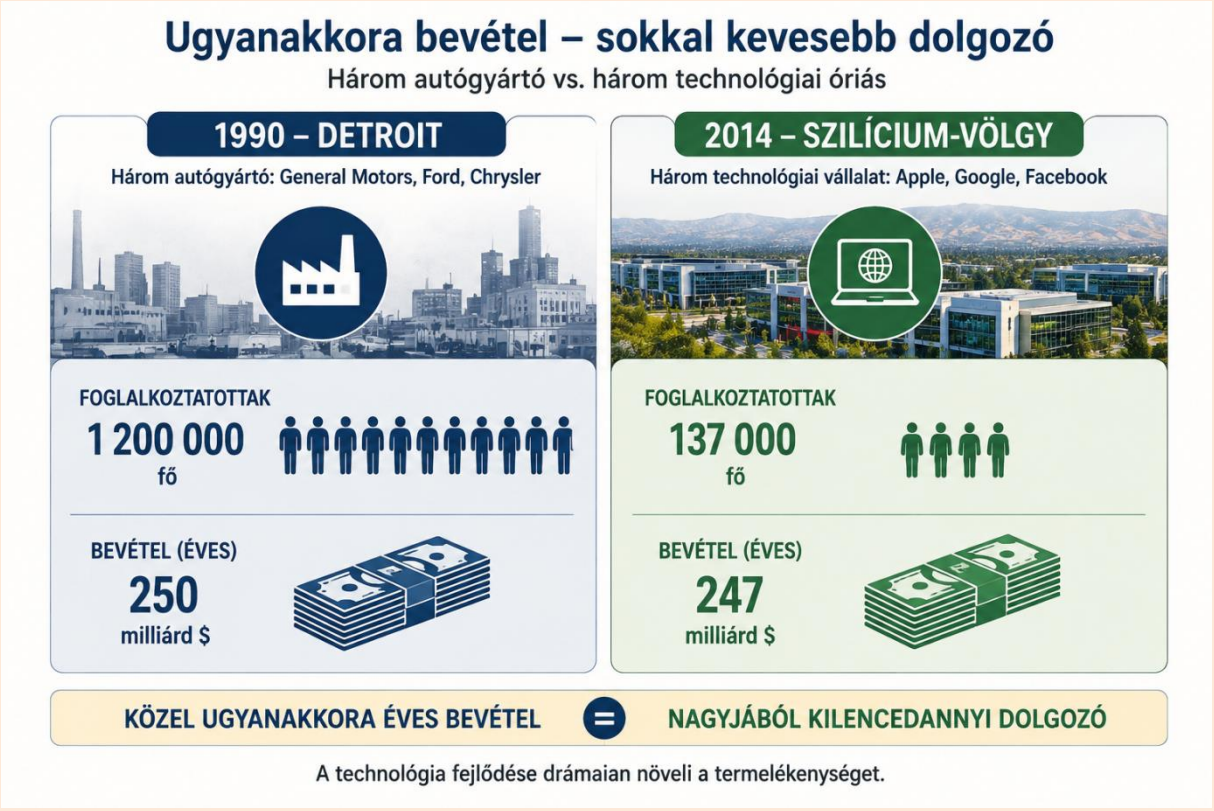
A számítógéppel ugyanez történt. Kezdetben elsősorban rendkívül gyors számológépnek tekintették. Ma már azonban nemcsak számolunk vele: írunk, tervezünk, kommunikálunk, fényképezünk, zenét hallgatunk, filmet nézünk, vásárolunk, dolgozunk és tanulunk rajta. A számítógép nem egyszerűen bekerült a korábbi világba – **átalakította azt.**

Az internet hatása még látványosabb. Nem csupán új kommunikációs eszköz lett, hanem átalakította a sajtót, a kereskedelmet, a bankrendszert, a zene- és filmipart, a reklámot és az emberi kapcsolatokat is. Molnár Szilárd tanulmánya éppen ezért a digitalizációt már nem pusztán technológiai változásként, hanem új **paradigmaként** értelmezi.

És most itt a mesterséges intelligencia

Az MI esetében talán ugyanez történik a szemünk előtt. Első pillantásra egy újabb eszköznek látszik: program, amely szöveget ír, képet készít, adatokat elemez vagy kérdésekre válaszol.

De mi történik, ha nem egyszerűen gyorsabban végzi el az ember korábbi feladatait, hanem **megváltoztatja magát a munkafolyamatot**?



Az ipari forradalom gépei elsősorban az ember izomerejét helyettesítették. Az automatizálás később a rutinszerű munkát vette át. A tanulmány egyik fontos felismerése, hogy az új hullám már a középosztály és a szellemi foglalkozások egy részét is érintheti: jogászokat, könyvelőket, tolmácsokat, pénzügyi szakembereket és más, korábban nehezen automatizálhatónak gondolt foglalkozásokat is említ.

A kérdés ezért talán nem az, hogy:

„Milyen munkát tud elvégezni helyettünk a mesterséges intelligencia?”

Hanem inkább:

„Milyen lesz az ember munkája és gondolkodása egy olyan világban, amelyben a mesterséges intelligencia állandó társa lesz?”

A választ még nem ismerjük. A technológia története azonban egy dologra megtanított bennünket:

az igazán nagy találmányok nem egyszerűen megváltoztatják azt, amit csinálunk – végül bennünket és a társadalmat is megváltoztatják.

*Forrás: Molnár Szilárd: **A negyedik ipari forradalom nem várt hatása***


ÍRÁS → megváltoztatta az emlékezést
NYOMTATÁS → megváltoztatta a tudás terjedését
FÉNYKÉPEZÉS → megváltoztatta az emlékeink megőrzését
TELEFON → megváltoztatta a távolság jelentését
SZÁMÍTÓGÉP → megváltoztatta a munkát
INTERNET → megváltoztatta az információhoz jutást
OKOSTELEFON → megváltoztatta a mindennapi életet
MESTERSÉGES INTELLIGENCIA → megváltoztatja a gondolkodást?
AI

A világ, amely figyel bennünket – a térfigyelő kamerától az arcfelismerésig

Alig néhány évtized alatt a kamera a modern városok egyik leggyakoribb érzékelőjévé vált. Kamerák figyelik az utcákat, közlekedési csomópontokat, üzleteket, bankokat, repülőtereket, gyárakat és lakóházakat. Autók fedélzeti kamerái rögzítik az utakat, kapucsengők az ajtónk előtt állókat, mobiltelefonok pedig milliárdszámra készítik a felvételeket.

A kamerás megfigyelés eredetileg egyszerű elven működött: **egy ember nézte a monitorokat**. A felvételek elsősorban arra szolgáltak, hogy egy eseményt utólag vissza lehessen keresni.

A digitális technika ezt alapvetően megváltoztatta. A kamerák hálózatba kapcsolódtak, a felvételek kereshető adatokká váltak, majd megjelent a számítógépes képfeldolgozás és a mesterséges intelligencia.

A gép ma már nemcsak rögzítheti a képet, hanem megpróbálhatja **értelmezni is, mi látható rajta**: felismerhet rendszámokat, járműtípusokat, tárgyakat, mozgásokat, és megfelelő rendszerben arcokat is összevethet adatbázisokkal.

Ezzel azonban minőségi változás történt.

A kamera megfigyelőeszközből információgyűjtő rendszerré vált.

Miért jó, ha kamerák vesznek körül bennünket?

A kamerák számos esetben növelhetik a biztonságot. Segíthetnek bűncselekmények felderítésében, eltűnt személyek keresésében, közlekedési balesetek rekonstruálásában, forgalmi dugók felismerésében vagy veszélyes ipari folyamatok ellenőrzésében.

Egy intelligens közlekedési rendszer például észlelheti a balesetet, a torlódást vagy az úttestre került akadályt. Egy ipari kamera olyan helyen is dolgozhat, ahol az ember jelenléte veszélyes lenne. Egy üzlet kamerafelvétele pedig fontos bizonyíték lehet egy rablás után.

A technológia tehát önmagában nem feltétlenül ellensége a szabadságnak. **Ugyanaz a kamera, amely megfigyel bennünket, meg is védhet bennünket.**

És hol kezdődik a veszély?



Régen a megfigyelés drága volt. Ha valakit folyamatosan követni akartak, emberekre volt szükség. A digitális kamerák, az óriási adattárolók és az MI ezt a gazdasági korlátot nagyrészt megszüntették.

Egy számítógépes rendszer elvileg egyszerre hatalmas mennyiségű kameraképet elemezhet. Ha ehhez arcfelismerés, rendszámfelismerés, mobiltelefonos helyadatok, vásárlási vagy más adatbázisok társulnak, már nem egyszerű térfigyelésről beszélünk.

Létrejöhet az ember **mozgásának digitális története**:

Ki volt? → Hol járt? → Mikor? → Kivel találkozott? → Merre ment tovább?

És itt válik a technikai kérdés társadalmi kérdéssé.

Kína – a lehetőségek és a veszélyek laboratóriuma

Kína különösen fontos példa, mert nagy léptékben alkalmaz hálózatba kötött kamerarendszereket, arcfelismerést és más digitális megfigyelési technológiákat. Ezek használhatók közbiztonsági, közlekedési és bűnüldözési célokra, ugyanakkor az állami megfigyelés olyan mértékét is lehetővé tehetik, amely komoly adatvédelmi és emberi jogi kérdéseket vet fel.

Itt azonban érdemes elkerülni azt az egyszerű képletet, hogy **„Kínában megfigyelés van, nálunk nincs”**. A kamerák, rendszámfelismerő rendszerek, arcfelismerési technológiák és kereskedelmi célú adatgyűjtés a világ számos országában jelen vannak. A lényeges különbség sokszor abban rejlik, **ki gyűjtheti az adatokat, milyen célból, mennyi ideig őrizheti meg**

őket, összekapcsolhatja-e más adatbázisokkal, és milyen jogorvoslata van a megfigyelt embernek.

A megfigyelés paradoxona

A probléma ezért nehezen oldható fel.

Több kamera → nagyobb biztonság lehetősége
de egyben
Több kamera → kisebb anonimitás lehetősége

A demokratikus társadalmak egyik nagy kérdése éppen az, hol húzzák meg a határt.

Egy balesetet észlelő kamera aligha fenyegeti a szabadságunkat. Egy gyilkosság elkövetőjének azonosítása társadalmilag kívánatos lehet. De mi történik, ha ugyanazzal a rendszerrel már azt is meg lehet állapítani, ki vett részt egy tüntetésen, ki ment be egy politikai rendezvényre, templomba, orvosi rendelőbe vagy kivel találkozott?

A technológia ugyanaz. A felhasználás célja változott meg.

A következő lépés: a kamera, amely már „ért”

Talán ez a történet legfontosabb fordulata. Az emberiség több mint száz éve készít fényképeket, de ezek sokáig passzív információk voltak. Valakinek meg kellett néznie őket.

A mesterséges intelligencia megjelenésével a kép **géppel feldolgozható információvá** válik.

Így a fejlődés láncolata:

Kamera → digitális kamera → hálózatba kapcsolt kamera → képfelismerés → arcfelismerés → adatbázisok összekapcsolása → automatikus követés

A végén pedig ott áll egy kellemetlenül egyszerű kérdés:

Nagyobb biztonságot akarunk, vagy nagyobb magánszférát?

Valószínűleg mindkettőt.

A 21. század egyik fontos feladata éppen annak eldöntése lesz, **hogyan kaphatjuk meg az elsőt anélkül, hogy elveszítenénk a másodikat.**

Megjósolható-e, ki lesz bűnöző? – amikor az algoritmus előítéletet tanul

A bűnüldözés egyik régi álma, hogy ne csak felderítse a már elkövetett bűncselekményeket, hanem **előre jelezze, hol vagy kinél nagyobb egy újabb bűncselekmény kockázata.** Ami egykor tudományos fantasztikumnak tűnt, az a nagy adatbázisok és a mesterséges intelligencia megjelenésével részben valósággá vált.

Az Egyesült Államokban többféle úgynevezett **predictive policing**, vagyis előrejelző rendészeti rendszert próbáltak ki. Ezek egy része korábbi bűnügyi adatokból azt becsülte meg, hogy **hol és mikor** valószínűbb egy újabb bűncselekmény. Az eredmény alapján a rendőrség több járőrt küldhetett a veszélyesebbnek ítélt területekre.

Más rendszerek már **emberek kockázatát** próbálták számszerűsíteni. Az egyik legismertebb ilyen rendszer a COMPAS, amelyet az amerikai büntető igazságszolgáltatásban a visszaesés kockázatának becslésére is alkalmaztak.

Első pillantásra mindez kifejezetten ésszerűnek tűnik. Egy számítógépnek nincsenek érzelmei, személyes ellenszenvei vagy előítéletei. Miért ne lehetne tehát objektívebb egy embernél?

A probléma az, hogy **az algoritmus az emberek által létrehozott adatokból tanul.**

Amikor a múlt előítélete bekerül az adatokba

„Ha egy algoritmus a múlt adataiból tanul, a múlt előítéleteit is megtanulhatja."

2016-ban a ProPublica amerikai oknyomozó szervezet több mint tízezer floridai vádlott adatain vizsgálta meg a COMPAS kockázatbecsléseit. Elemzésük szerint jelentős különbség mutatkozott abban, hogyan tévedett a rendszer fekete és fehér vádlottak esetében.

Azok között, akik a következő két évben nem követtek el újabb bűncselekményt, a fekete vádlottakat lényegesen gyakrabban sorolta tévesen magasabb kockázatú csoportba. Fordítva pedig a később ténylegesen visszaeső fehér vádlottakat gyakrabban értékelte alacsony kockázatúnak. A vizsgálat és annak értelmezése komoly szakmai vitát váltott ki. A rendszer készítője vitatta a ProPublica következtetéseit, és azóta is vita folyik arról, hogyan kell matematikailag meghatározni egy kockázatbecslő algoritmus „igazságosságát".

A történet legfontosabb tanulsága azonban ennél általánosabb.

Egy algoritmusnak nem kell ismernie valakinek a bőrszínét ahhoz, hogy társadalmilag torzított eredményt adjon.

Képzeljünk el egy városrészt, ahol történelmi vagy társadalmi okokból eleve több rendőr járőrözik. A több rendőr több szabálysértést és bűncselekményt észlel. A rendőrségi adatbázisban ezért erről a területről több eset szerepel.

Az algoritmus megvizsgálja az adatokat, és logikusan arra jut:

itt történik több bűncselekmény, tehát ez veszélyesebb terület.

Ezért a rendszer azt javasolja, hogy küldjenek oda még több rendőrt. A több rendőr ismét több szabálysértést fedez fel, amit az adatbázis újabb bizonyítékként rögzít arra, hogy a terület veszélyes.

Ezzel bezárult a kör.

Az algoritmikus visszacsatolás

Több rendőr egy területen
↓
Több észlelt és regisztrált eset
↓
Az adatbázis szerint magasabb bűnözés
↓
Az algoritmus magasabb kockázatot jelez
↓
Még több rendőrt küldenek oda
↓
Még több észlelt eset
↺

A számítógép közben látszólag teljesen objektíven dolgozott. A probléma a kiindulási adatokban és az ezekből létrejövő visszacsatolásban volt. Ez rávilágít a mesterséges intelligencia egyik alapvető problémájára:

A gép nem feltétlenül a világot tanulja meg, hanem azt, amit a világról adatként megmutattunk neki.

Ha a múlt adatai magukban hordozzák egy társadalom korábbi egyenlőtlenségeit vagy előítéleteit, az algoritmus ezeket nem szükségképpen felismeri és kijavítja. Akár **matematikai szabályszerűségként meg is tanulhatja őket**.

A térfigyelő kamerák, az arcfelismerés és az automatizált kockázatelemzés összekapcsolása ezért különösen érzékeny terület:

Kamera → arcfelismerés → azonosítás → adatbázis → kockázati pontszám → döntés → rendőri intézkedés

Minden egyes lépés önmagában hasznos technológia lehet. Összekapcsolva azonban olyan rendszer jöhet létre, amely már nemcsak azt rögzíti, **mit tettünk**, hanem megpróbálja megjósolni azt is, **mit fogunk tenni**.

És ezzel egy régi jogelv egészen új technológiai kihívással találkozik:

Büntethető-e valaki nem azért, amit elkövetett, hanem azért, amit egy algoritmus szerint talán el fog követni?

Ez már nem pusztán informatikai kérdés. **A jogállam és az emberi szabadság egyik 21. századi alapkérdése.**

Amikor az AI már AI-t fejleszt – kitől kell félnünk?

A mesterséges intelligencia egyik különös tulajdonsága, hogy még a fejlesztői sem mindig tudják pontosan megmondani, miért jutott egy bonyolult neurális hálózat éppen arra a válaszra, amelyre jutott. Ez azonban nem jelenti azt, hogy a gép titokzatos módon átvette volna az irányítást. A rendszert emberek építették, ők választották ki a tanítás módját, az adatokat, a célokat és azt is, milyen lehetőségeket kapjon a külvilágban.

De meddig marad ez így?

A hagyományos programozás során az ember lépésről lépésre megmondta a számítógépnek, mit tegyen. A gépi tanulásnál már inkább a tanulás feltételeit határozzuk meg, a megoldást pedig maga a rendszer alakítja ki. A következő lépcsőfok az, amikor az AI már a saját utódainak létrehozásában is részt vesz: programot ír, algoritmusokat javasol, kísérleteket tervez, modelleket értékel és segít jobb AI-rendszereket létrehozni.

Elképzelhető ennél egy további lépés is:

Ember → AI_1 → jobb AI_2 → még jobb AI_3 → ?

Ezt nevezik **rekurzív önfejlesztésnek**. Ha minden új rendszer az elődjénél ügyesebb lenne a következő megtervezésében, elméletileg rendkívül gyors fejlődés indulhatna el. Nem tudjuk, hogy ez valóban bekövetkezhet-e. Az intelligencia fejlődéséhez ugyanis nem elegendő a programkód: számítási teljesítmény, energia, chipek, adatközpontok és végső soron az egész emberi technikai infrastruktúra szükséges hozzá.

A veszélyhez azonban nincs is szükség arra, hogy a gép „öntudatra ébredjen”, vagy elhatározza, hogy átveszi a hatalmat.

Lehet, hogy továbbra is az ember a nagyobb veszély.

Az emberiség történetében szinte minden nagy technikai találmány egyszerre növelte alkotó és pusztító képességeinket. A tűzzel főzni és várost felgyújtani is lehetett. A kémiából gyógyszer és méreg egyaránt született. Az atommag energiája erőművet és atombombát adott. Az internet összekötötte az emberiséget, de lehetőséget teremtett soha nem látott méretű megfigyelésre, csalásra és dezinformációra is.

Az AI ugyanezt a kettősséget hordozza, csak különösen nagy erővel, mert más technológiák képességeit is megsokszorozhatja.

Egy mesterséges intelligencia, amely tanácsot ad egy orvosnak, egészen más kockázatot jelent, mint az, amely önállóan rendelkezhet pénzzel, informatikai rendszerekkel, kritikus infrastruktúrával vagy fegyverekkel.

Ezért talán rosszul tesszük fel a kérdést, amikor azon vitatkozunk, hogy **az embertől vagy az AI-tól kell-e jobban félnünk.**

A valódi veszély a kettő találkozása lehet.

Rossz emberi cél + rendkívüli gépi képesség = korábban elképzelhetetlen hatalom.

Ehhez nem kell a gépek lázadása. Nem kell öntudatra ébredő számítógép, és nem kell egy sci-fikből ismert gonosz szuperintelligencia sem. Elég egy olyan ember, vállalat, politikai rendszer vagy állam, amely felismeri, hogy az AI segítségével hatékonyabban figyelhet meg, manipulálhat, támadhat vagy gyakorolhat hatalmat.

És van egy ennél is nehezebb kérdés. Mi történik akkor, ha egy általunk létrehozott rendszer már olyan bonyolulttá válik, hogy ugyan mi határoztuk meg a célját, de többé nem tudjuk előre látni, milyen úton és milyen következmények árán éri el azt?

Talán nem attól kell leginkább tartanunk, hogy a mesterséges intelligencia egyszer megtanul **emberként gondolkodni**.

Hanem attól, hogy az ember megtanulja **embertelen célokra használni a mesterséges intelligenciát**.

IX. rész – Másképpen számoló számítógépek

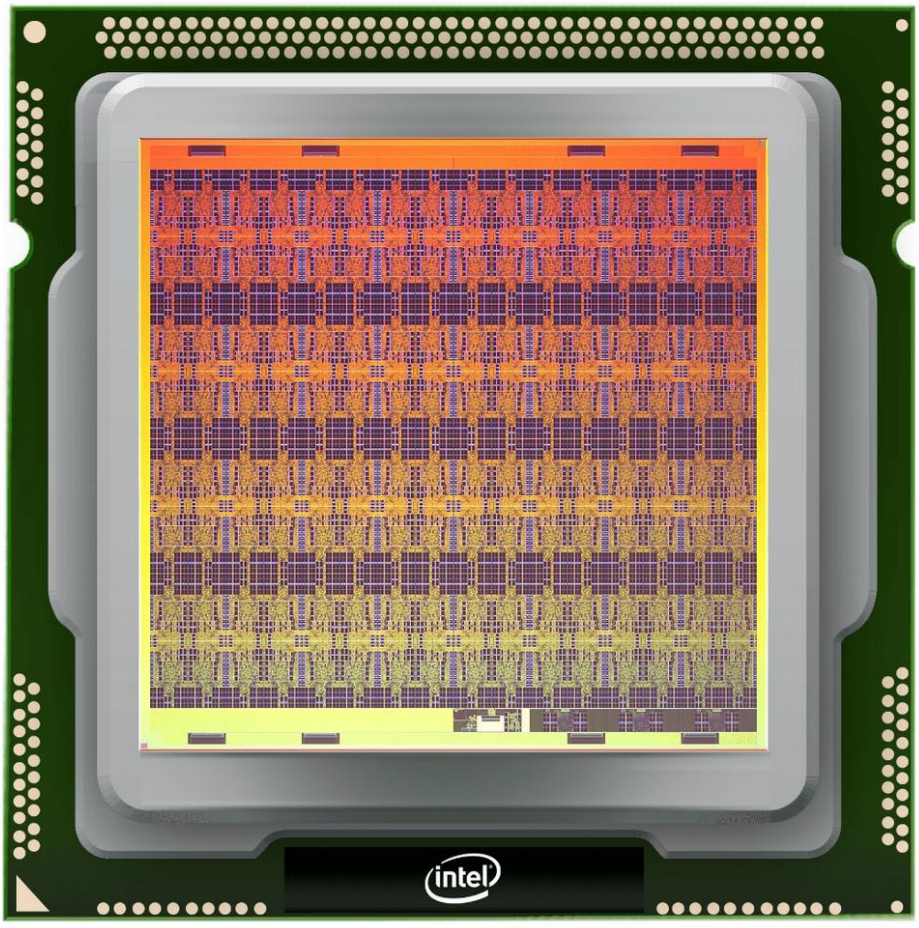
20. A klasszikus számítógépen túl

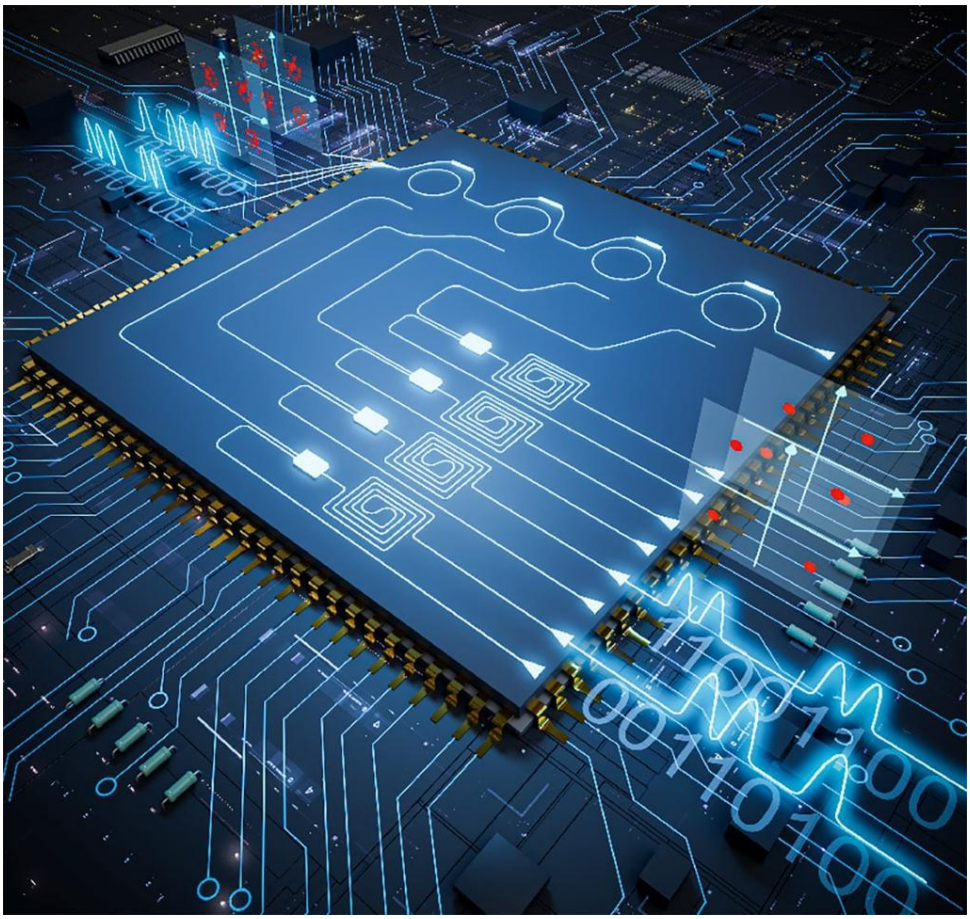
A számítógép történetének nagy részében ugyanazt az alapelvet tökéletesítettük: egyre kisebb és gyorsabb elektronikus kapcsolókból építettünk egyre nagyobb teljesítményű gépeket. A tranzisztorok mérete azonban nem csökkenthető korlátlanul, az órajel növelésének útjába az energiafogyasztás és a hőtermelés állt, az adatok processzor és memória közötti mozgatása pedig egyre jelentősebb korláttá vált.

Ezért felmerült egy merészebb kérdés:

Muszáj-e minden számítógépnek úgy működnie, ahogyan az elmúlt évtizedek számítógépei működtek?

A válasz számos különös számítási módszerhez vezetett. Egyesek az emberi agyat utánozzák, mások elektronok helyett fényt használnak, sőt olyan kísérletek is születtek, amelyekben **DNS-molekulák végzik a számítás egy részét**.





Ábra: *A klasszikus számítógépen túl több út vezethet: hatalmas párhuzamos rendszerek, az agy működését utánzó neuromorf chipek, fotonikus processzorok és akár DNS-molekulák is felhasználhatók számításra.*

20.1. Miért keresünk új számítási módszereket?

Évtizedeken keresztül szinte természetesnek vettük, hogy az új számítógép gyorsabb lesz a réginél. A tranzisztorok egyre kisebbek lettek, egy lapkára egyre több fért belőlük, és az órajel is folyamatosan emelkedett.

Ez a fejlődés azonban fizikai korlátokhoz közeledett.

A processzor teljesítményének növelése egyre nagyobb **energiafogyasztással és hőtermeléssel** jár. Eközben maga a számítás sok esetben már gyorsabb, mint az adatok mozgatása a memória és a processzor között.

Ez az úgynevezett **von Neumann-szűk keresztmetszet** egyik megjelenési formája.

A modern AI különösen jól megmutatta a problémát. Egy neurális hálózat működtetése során óriási mennyiségű adatot kell újra és újra mozgatni a memória és a számítóegységek között. Az energia jelentős része nem feltétlenül magára a matematikai műveletre, hanem **az adatok szállítására** fordítódik.

A kérdés ezért már nem pusztán az:

Hogyan készítsünk gyorsabb processzort?

Hanem: **Lehet-e egészen másképpen számolni?**

Visszatér az analóg számítás? – A mikrohullámú agy

A számítástechnika történetének egyik nagy győzelme az analóg világról a digitálisra való áttérés volt. A folytonosan változó feszültségeket és fizikai mennyiségeket nullákra és egyesekre cseréltük. A digitális számítógép pontos, programozható, másolható és szinte tetszőleges feladat elvégzésére alkalmassá vált.

Az AI rohamosan növekvő számításigénye azonban érdekes fordulatot hozhat: **bizonyos feladatoknál ismét érdemes lehet magát a fizikát számolásra használni.**

A Cornell Egyetem kutatói ennek különös példáját készítették el: egy szilíciumchipbe integrált **mikrohullámú neurális hálózatot**, amelyet tréfásan „mikrohullámú agynak" is neveznek. A rendszer nem a hagyományos digitális processzorok módjára, órajelről órajelre hajt végre utasításokat. A chipben terjedő mikrohullámok egymással kölcsönhatásba lépnek, és a rendszer ezeknek a hullámoknak az analóg, nemlineáris viselkedését használja fel a számításhoz.

Másképpen fogalmazva: **a fizika maga végzi el a művelet egy részét.**

A 2025-ben bemutatott kísérleti chip több tíz gigahertzes tartományban képes jeleket feldolgozni, miközben fogyasztása 200 milliwatt alatt marad. Rádiójelek felismerését, radaradatok feldolgozását és nagy sebességű adatfolyamok elemzését is demonstrálták vele.

A kutatók 2026-ban továbbfejlesztették az elképzelést. A mikrohullámú hálózat már sajátos „tokeneket" is képes létrehozni: a beérkező információ jellemzőit mikrohullámú mintázatokba kódolja. Egy műholdas viharkép kísérletében például a kép fontos tulajdonságainak jelentős részét úgy sikerült megőrizni, hogy az átvitt adatmennyiség körülbelül a nyolcadára csökkent.

Ez nem jelenti azt, hogy visszatérnek a régi analóg számítógépek. Sokkal inkább egy új korszak körvonalazódik, amelyben a digitális számítógép mellett különféle **analóg, neuromorf, fotonikus és más specializált processzorok** végezhetik azokat a feladatokat, amelyekhez saját fizikai működésük különösen jól illeszkedik.

Ennek ára is van. Az analóg rendszerek természetüknél fogva érzékenyebbek a zajra, a gyártási eltérésekre és a környezet változásaira, pontosságuk pedig nem feltétlenül éri el a digitális számítását. Sok AI-feladatnál azonban nem tizenöt tizedesjegy pontosságra van szükség, hanem gyors mintafelismerésre és jó közelítésre. Itt az analóg megoldások előnye jelentős lehet.

A számítástechnika története így különös kört írhat le. Évtizedeken keresztül azon dolgoztunk, hogy a folytonos, zajos fizikai világot nullákra és egyesekre alakítsuk, hogy a számítógép fel tudja dolgozni.

Most pedig ismét feltesszük a kérdést: **Biztos, hogy mindent nullákra és egyesekre kell alakítanunk?**

Talán a jövő számítógépe nem tisztán digitális vagy analóg lesz. A digitális processzor maradhat a karmester, miközben egyre több speciális „hangszeren” maga a fizika végzi el a számítás egy részét.

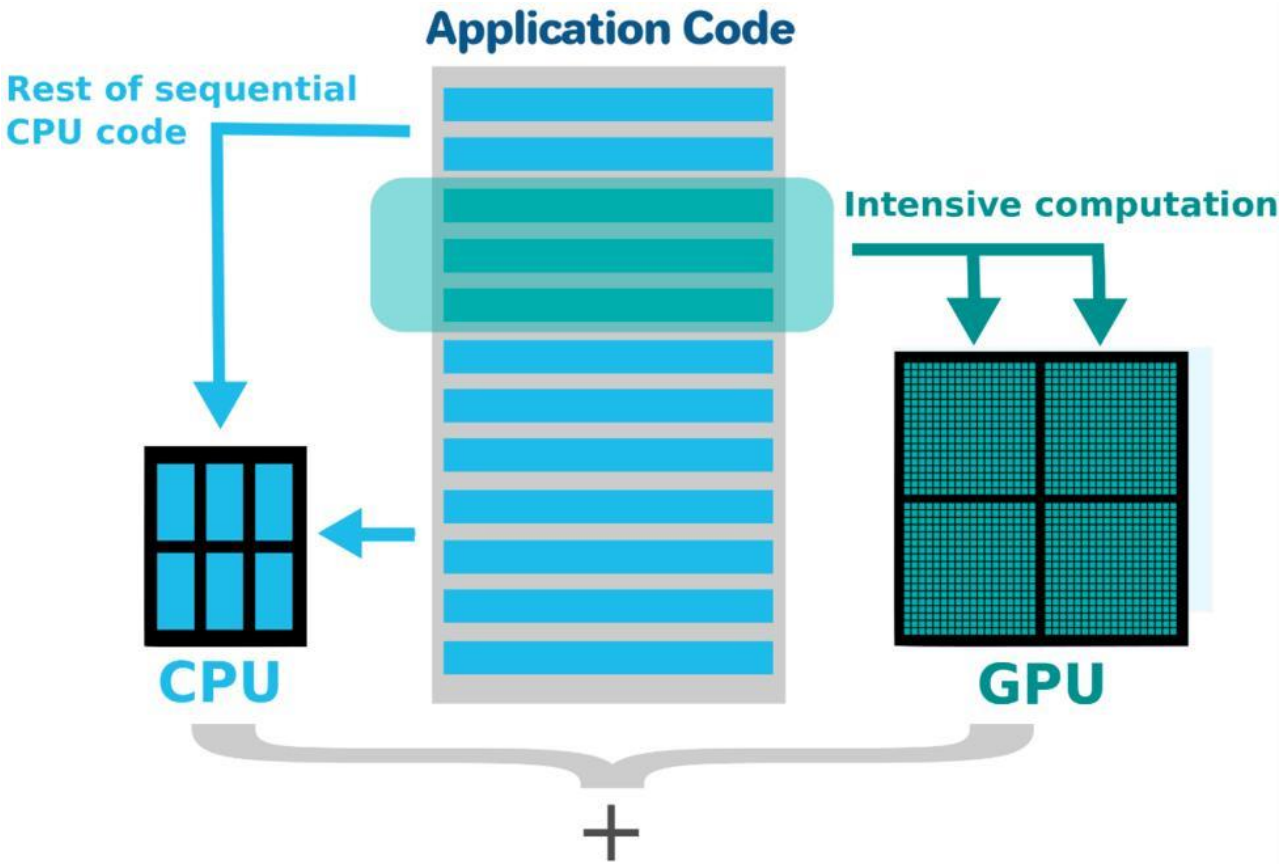
A digitális számítógép száz év után újra segítségül hívja a fizikát.

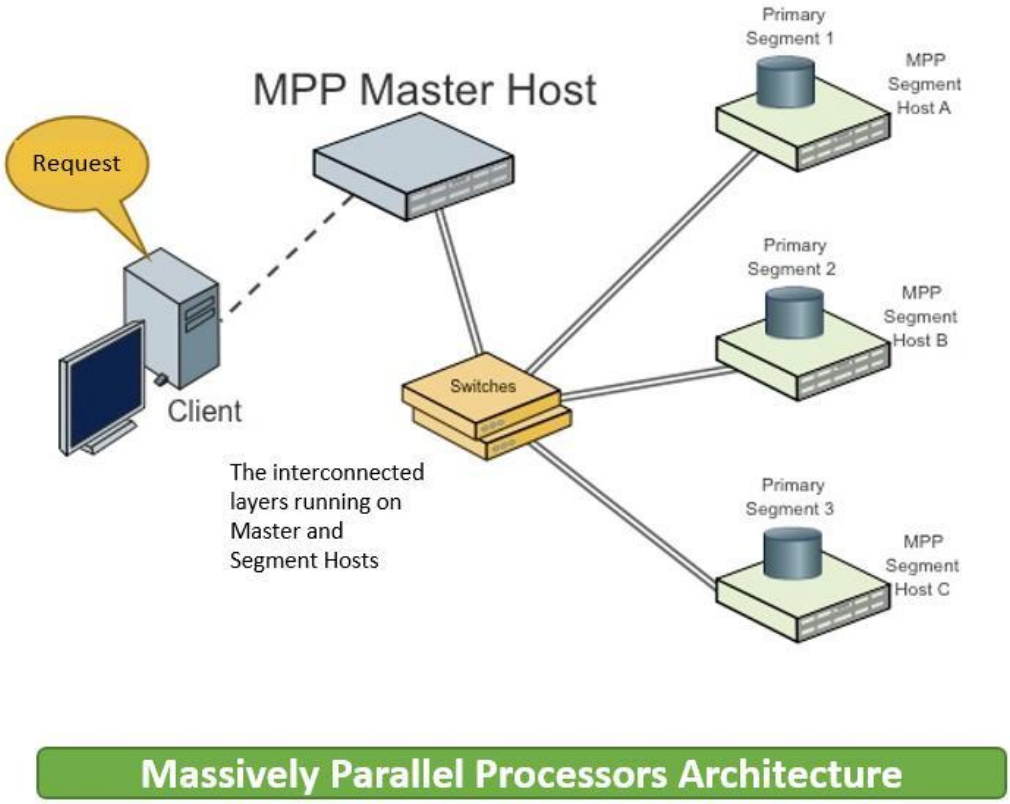
20.2. Párhuzamos architektúrák

Az első válasszal már találkoztunk a szuperszámítógépeknél és a GPU-knál.

Ha egyetlen processzort nem tudunk korlátlanul gyorsítani, **használjunk sokat egyszerre**.

A mai processzorok több magot tartalmaznak. A GPU-k több ezer számítási egységgel dolgozhatnak. A szuperszámítógépek pedig processzorok és gyorsítók óriási hálózatát kapcsolják össze.





Csakhogy nem minden feladat párhuzamosítható egyformán.

Ha egy számítás következő lépéséhez feltétlenül szükségünk van az előző eredményére, a két műveletet nem végezhetjük egyszerre.

Ezért a párhuzamosításnak is van elméleti korlátja. Hiába növeljük a processzorok számát, ha a program egy része szükségképpen sorosan fut.

A modern számítógép ezért egyre kevésbé egyetlen univerzális processzor.

Inkább különböző számítási egységek együttese:

CPU + GPU + AI-gyorsító + egyéb speciális processzorok.

Ábra: *A számítás gyorsításának egyik útja az egyre több párhuzamos számítási egység alkalmazása. A CPU néhány nagy teljesítményű magja mellett a GPU sok egyszerűbb egységgel végez egyszerre hasonló műveleteket.*

20.3. Neuromorf számítógépek

Az emberi agy különös összehasonlítási alap.

Körülbelül 86 milliárd neuronnal működik, rendkívül összetett feladatokat végez, mégsem igényel erőművet.

Energiafelhasználása nagyságrendileg **néhány tíz watt.**

Mi lenne, ha a számítógépet nem a hagyományos processzor, hanem **az idegrendszer működése alapján** építenénk fel?

Ezt az elképzelést követi a **neuromorf számítástechnika**.





A hagyományos számítógépben a processzor és a memória elkülönül. A neuromorf rendszerek ezzel szemben sok apró, neuronszerű egységet és közöttük szinapszisokhoz hasonló kapcsolatokat alkalmaznak.

Gyakori megközelítés a **spiking neural network**, amelyben az információt időben megjelenő impulzusok – „tüskék” – továbbítják.

Ilyen kutatási rendszerek például az IBM **TrueNorth**, az Intel **Loihi**, valamint a Manchesteri Egyetem **SpiNNaker** rendszere.

A neuromorf processzor különösen olyan feladatokban lehet érdekes, ahol folyamatosan érkező érzékelőadatokra kell kis energiafelhasználással reagálni: robotikában, gépi látásban vagy intelligens érzékelőkben.

Nem egyszerűen **gyorsabb CPU-t** próbál készíteni.

Más számítógép-architektúrát keres.



Ábra: *Neuromorf processzorok az idegrendszer működéséből merítenek ötleteket. Sok, egymással kapcsolatban álló számítási egységgel és eseményvezérelt működéssel próbálnak rendkívül energiahatékony információfeldolgozást megvalósítani.*

20.4. Optikai számítástechnika

Az elektronikus számítógépben elektronok mozgásával továbbítjuk és dolgozzuk fel az információt.

De miért ne használhatnánk **fényt?**

A fotonok rendkívül gyorsan terjednek, és különböző hullámhosszú fénynyalábok ugyanabban a közegben egymással párhuzamosan továbbíthatnak információt.

Az optikai számítástechnika gondolata ezért régi.

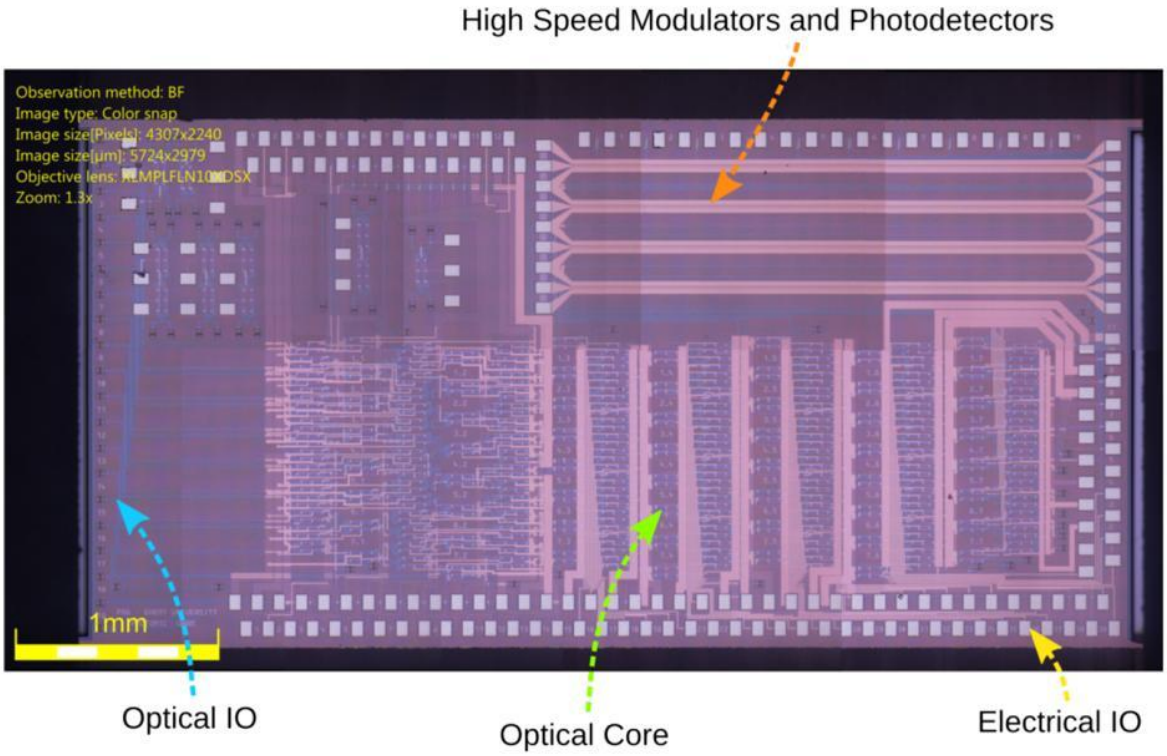
A modern **fotonikus integrált áramkörökben** apró hullámvezetők irányítják a fényt a chipen belül. Interferenciával, modulátorokkal és más optikai elemekkel matematikai műveletek is végrehajthatók.

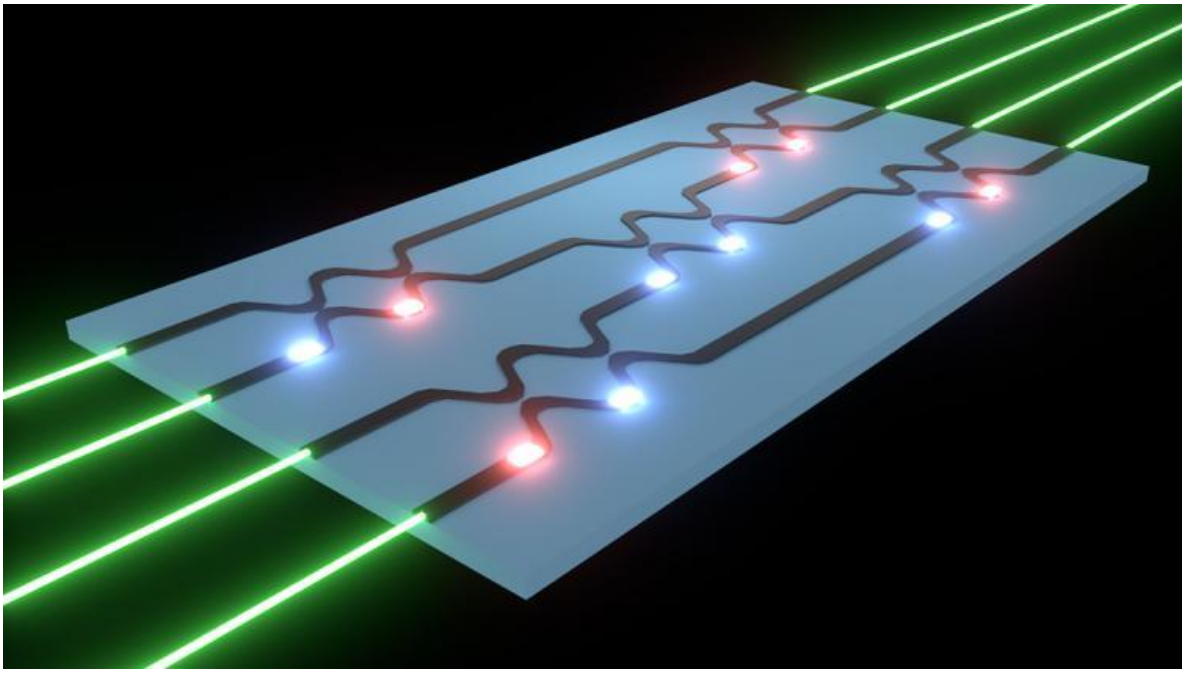
Különösen érdekes lehet ez a neurális hálózatoknál gyakori **mátrixműveletek** gyorsítására.

Az optikai számítógépnek azonban komoly problémái vannak. Az adatokat be kell juttatni az optikai rendszerbe, az eredményeket érzékelni kell, memóriára továbbra is szükség van, és a vezérlés jelentős része elektronikus maradhat.

Ezért a közeljövő reális megoldása valószínűleg nem a teljesen optikai PC.

Sokkal inkább: **elektronika + fotonika.**





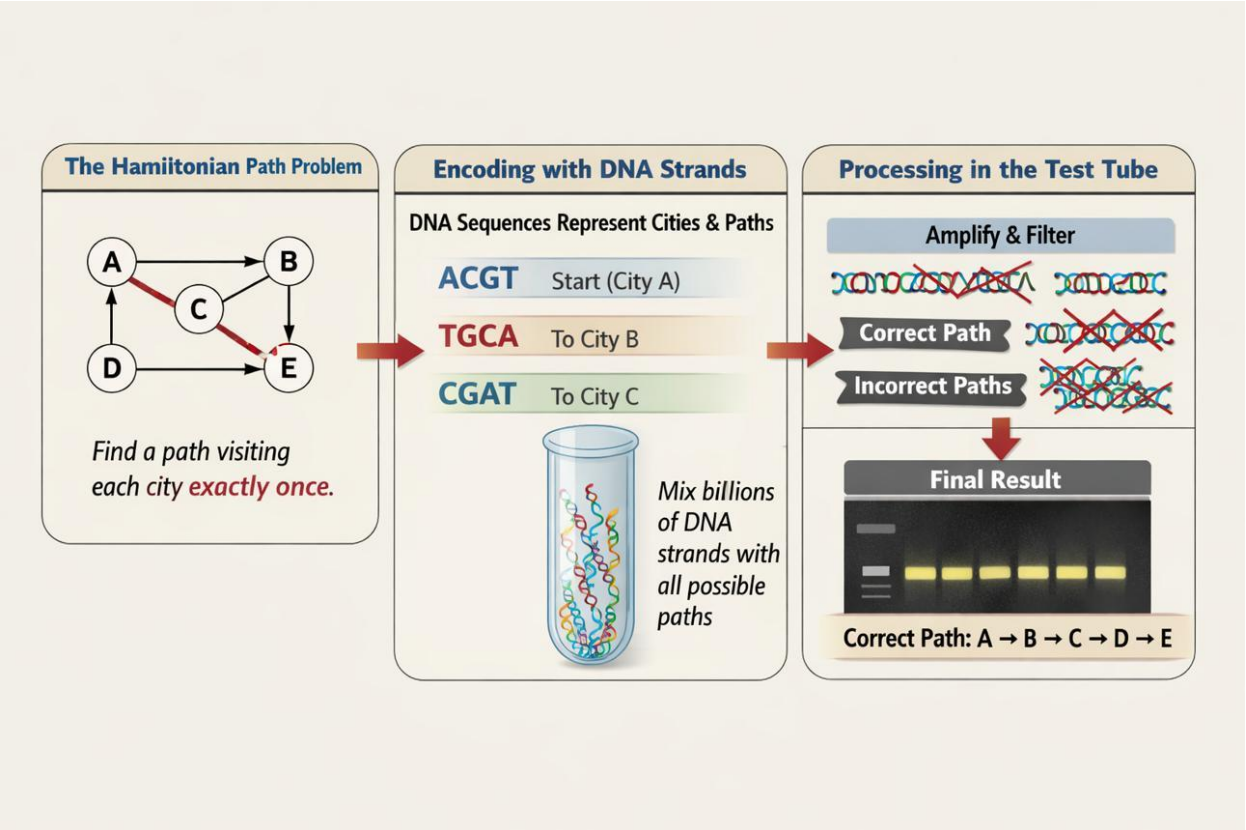
Ábra: *Fotonikus integrált áramkör. Az elektromos vezetékekhez hasonlóan apró optikai hullámvezetők irányítják a fényt a chipen belül. A fotonika különösen gyors adatátvitelben és bizonyos párhuzamos számításokban ígéretes.*

20.5. DNS-számítás

Talán ez a fejezet legmeglepőbb része.

Lehet egy molekulával számolni?

1994-ben Leonard Adleman amerikai informatikus megmutatta, hogy igen.



Adleman egy gráfelméleti problémát oldott meg DNS-molekulák segítségével.

A lehetséges megoldások egyes részeit különböző DNS-szekvenciák képviselték. A molekulák természetes kémiai tulajdonságaik alapján hatalmas mennyiségű kombinációt hoztak létre párhuzamosan. Ezután laboratóriumi módszerekkel kiválogatták azokat a molekulákat, amelyek megfeleltek a probléma feltételeinek.

A „processzor” tehát egy kémcsőben volt.

A DNS különleges tulajdonsága az elképesztő **információsűrűség** és a molekulák óriási számából adódó párhuzamosság.

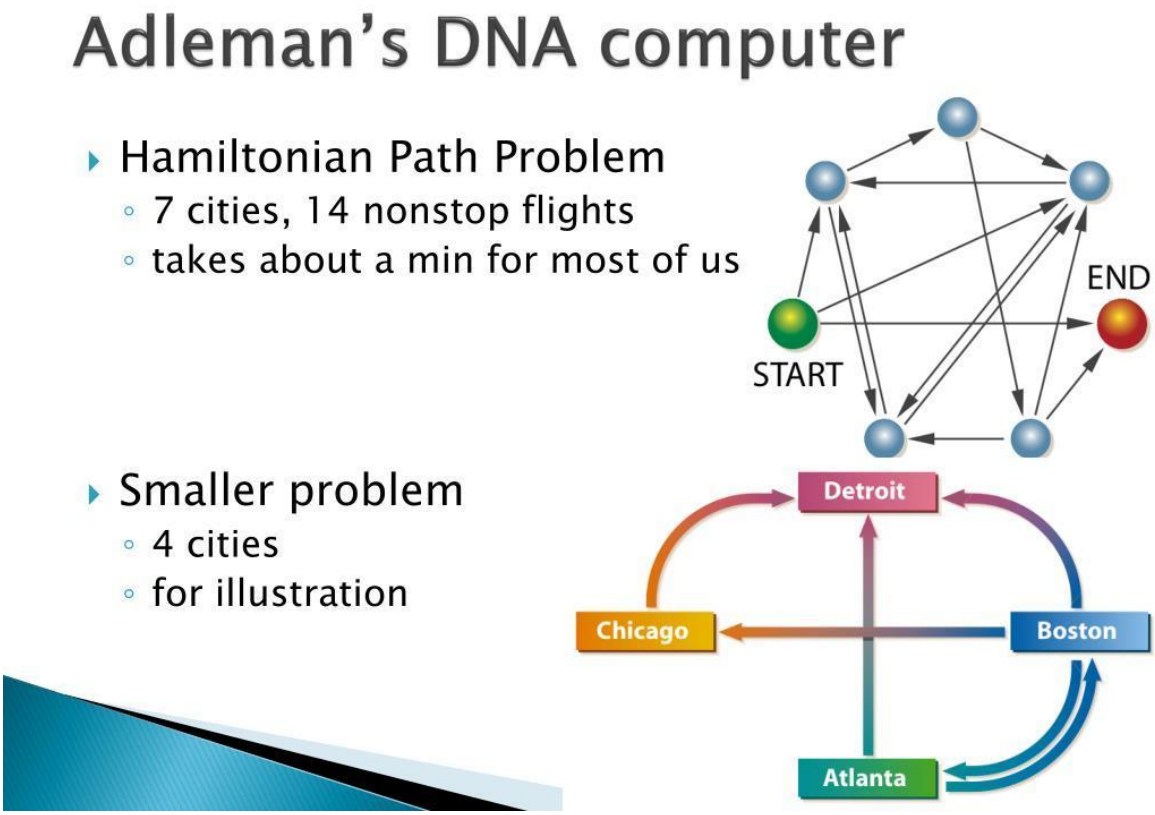
Egy kémcsőben elképzelhetetlenül sok molekula reagálhat egyszerre.

Ez azonban nem jelenti azt, hogy a DNS-számítógép általánosan gyorsabb lenne egy elektronikus számítógépnél. A molekulák előállítása, a reakciók végrehajtása és különösen az eredmény kiolvasása lassú és laboratóriumi műveleteket igényelhet.

A DNS-számítás ezért egyelőre inkább különleges kutatási terület, mint a szilícium közvetlen versenytársa.

De történetileg rendkívül fontos gondolatot bizonyított:

A számítás nem feltétlenül elektronikus folyamat.



Ábra: *Leonard Adleman 1994-es kísérlete megmutatta, hogy DNS-molekulák segítségével is végezhető számítás. A molekulák hatalmas száma természetes párhuzamosságot biztosít.*

Mit tanulhatunk ezekből a kísérletekből?

Első pillantásra a párhuzamos processzor, az agyat utánzó chip, a fénnyel működő áramkör és a DNS-számítógép között alig van kapcsolat.

Pedig ugyanarra a kérdésre keresnek választ:

Van-e jobb módja egy adott számítás elvégzésének, mint a hagyományos processzor?

A válasz egyre inkább az, hogy **attól függ, mit akarunk kiszámítani**.

Módszer	Alapötlet	Különleges előny
Párhuzamos számítás	sok számítóegység egyszerre	nagy számítási teljesítmény
Neuromorf	az idegrendszer utánzása	energiahatékonyság
Optikai	fotonok használata	gyors adatátvitel, párhuzamosság
DNS	molekuláris reakciók	óriási párhuzamosság és adatsűrűség
Kvantum	kvantummechanikai állapotok	speciális kvantumalgoritmusok

És **pontosan innen érdemes átlépni a következő fejezetbe**.

A párhuzamos, neuromorf, optikai és DNS-számítógépek ugyan nagyon különböznek a hagyományos PC-től, de a kvantumszámítógép még radikálisabb lépést tesz:

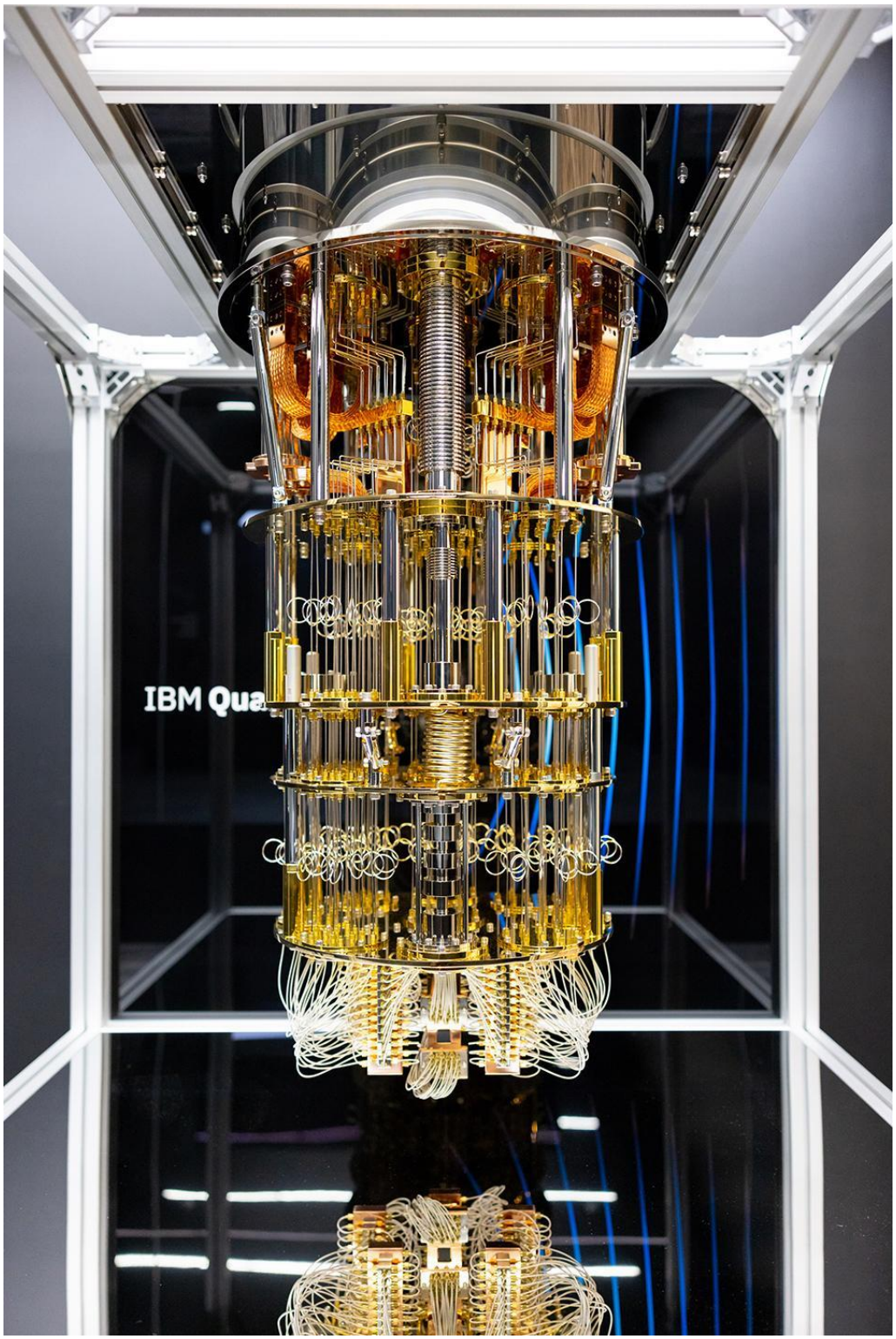
magukat a kvantummechanika törvényeit használja számításra.

21. A kvantumszámítógép

A számítógépek történetében eddig szinte minden nagy ugrásnál ugyanaz történt: **egyre gyorsabban, egyre kisebb eszközökkel végeztük ugyanazokat az alapvetően klasszikus műveleteket**. A relé helyét átvette az elektroncső, azt a tranzisztor, majd az integrált áramkör; egy processzorból sokmagos processzor, GPU és hatalmas párhuzamos rendszer lett.

A kvantumszámítógép azonban más.

Nem egyszerűen gyorsabb alkatrészekből építünk számítógépet, hanem **más fizikai törvényeket használunk a számításhoz**. A kvantummechanika különös jelenségei – a szuperpozíció, az összefonódás és az interferencia – magának a számításnak az eszközeivé válnak.



Ábra: *Egy mai kvantumszámítógép belseje inkább emlékeztet fizikai laboratóriumra, mint személyi számítógépre. A látványos, csillárszerű szerkezet jelentős része a kvantumprocesszor rendkívül alacsony hőmérsékleten tartására szolgál.*

21.1. Bit és qubit

A hagyományos számítógép legkisebb információs egysége a **bit.**

Értéke:

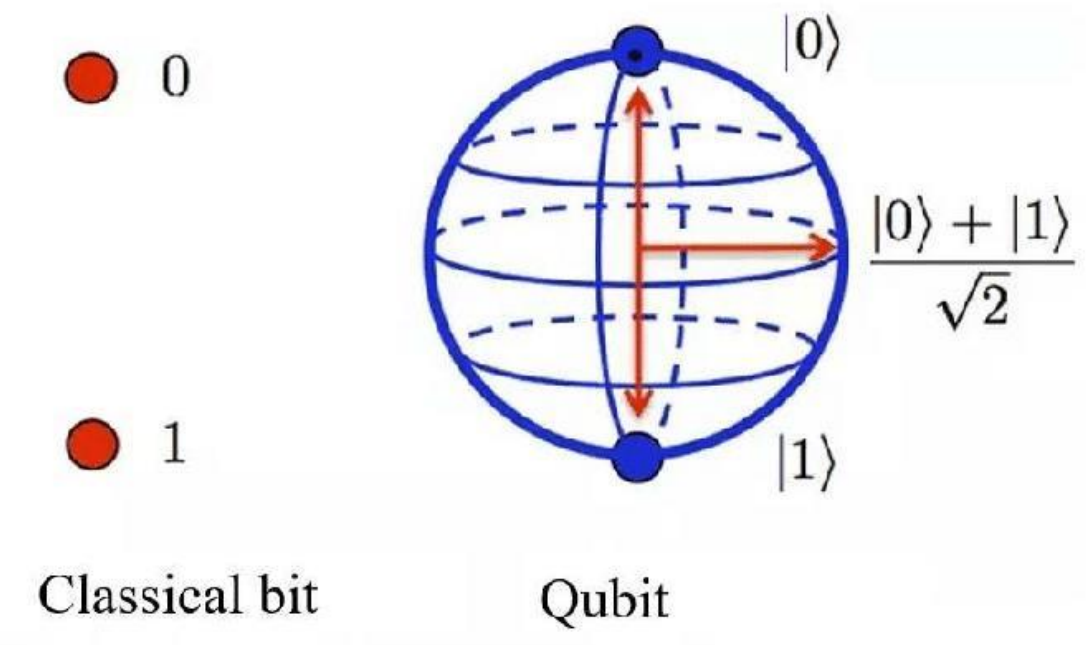
0 vagy 1.

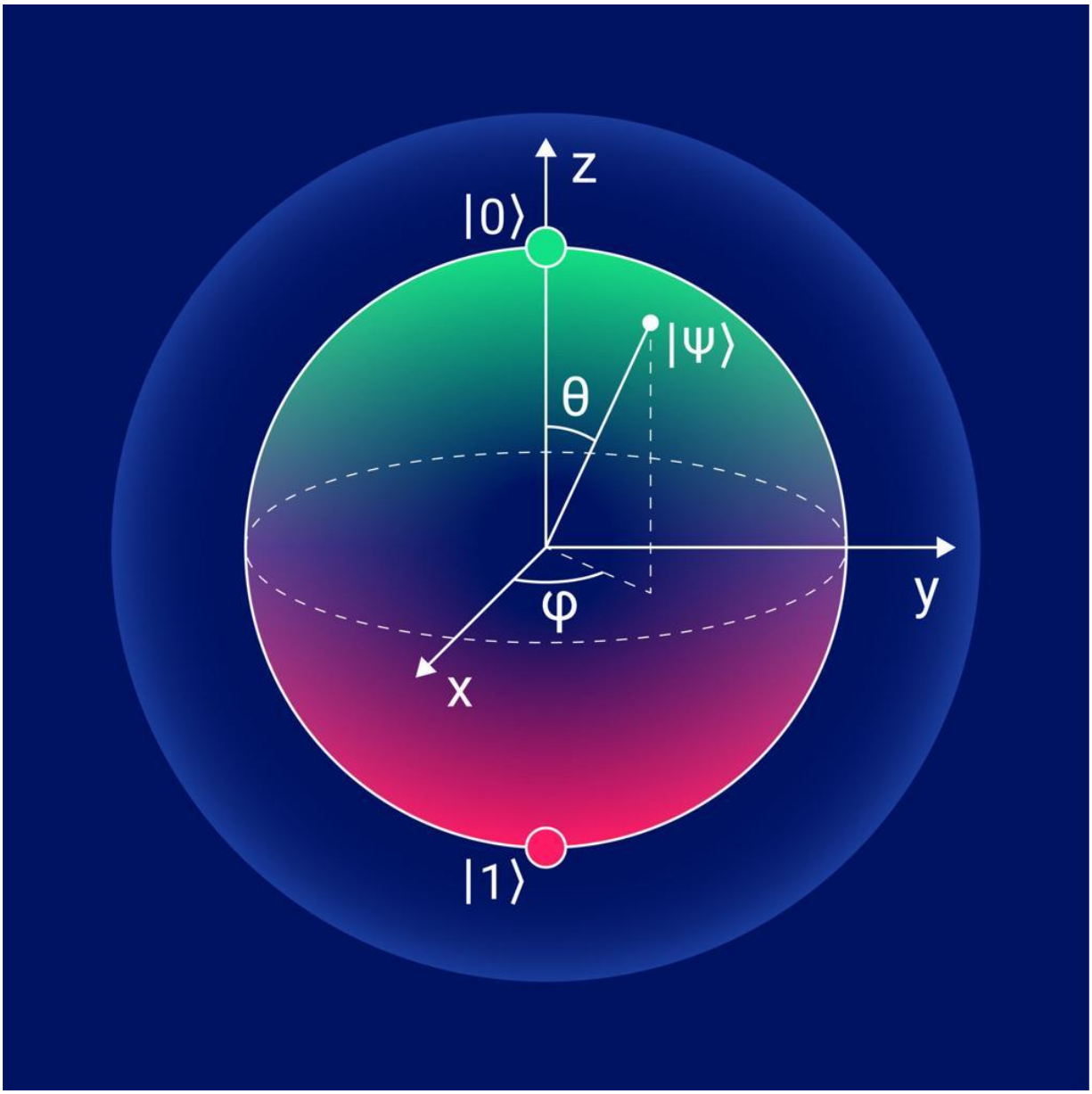
A processzorban ezt két jól megkülönböztethető fizikai állapot képviseli.

A kvantumszámítógép alapegysége a **qubit – quantum bit**, vagy kvantumbit.

Qubit többféle fizikai rendszerből készíthető. Használhatunk például szupravezető áramköröket, ionokat, semleges atomokat vagy fotonokat.

A lényeg nem az, hogy miből készül, hanem hogy olyan kvantumrendszer legyen, amelynek állapotát létre tudjuk hozni, manipulálni és végül meg tudjuk mérni.





Ábra: *A klasszikus bit két állapot valamelyikében van: 0 vagy 1. A qubit állapotát ezzel szemben a két alapállapot kvantummechanikai kombinációja írhatja le.*

21.2. Szuperpozíció

A kvantummechanika egyik legszokatlanabb jelensége a **szuperpozíció.**

Leegyszerűsítve gyakran azt mondjuk, hogy a qubit „egyszerre lehet 0 és 1”. Ez szemléletes, de könnyen félrevezető megfogalmazás.

Pontosabban: a qubit állapota a **0 és az 1 kvantummechanikai szuperpozíciója** lehet. A két lehetőséghez amplitúdó tartozik, és ezekből adódnak a méréskor tapasztalható valószínűségek.

Amikor ugyanis megmérjük, már **0-t vagy 1-et kapunk**.

Több qubit esetében az állapottér rendkívül gyorsan nő. Egy n qubites rendszer állapotának klasszikus leírásához általában $\mathbf{2^n}$ **amplitúdó** szükséges.

50 qubitnél:

$$\mathbf{2^{50} \approx 1{,}13 \times 10^{15}}$$

amplitúdóról beszélünk.

Ez már jelzi, honnan eredhet a kvantumszámítás különleges ereje – de ebből még nem következik, hogy egyszerűen „minden lehetőséget egyszerre kiszámolunk és kiolvasunk”.

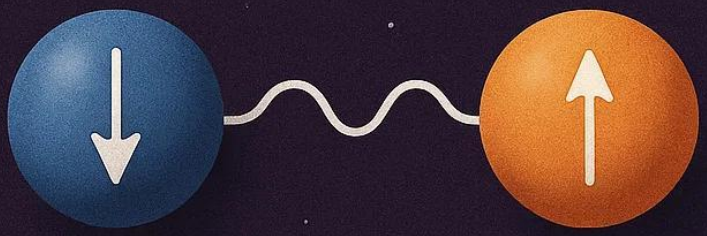
21.3. Összefonódás

Még furcsább jelenség az **entanglement – kvantum-összefonódás**.

Két vagy több qubit olyan közös kvantumállapotba kerülhet, amelyet már nem lehet egyszerűen az egyes qubitek különálló állapotaira bontani.

Ilyenkor a rendszer egésze hordozza az információt.

QUANTUM ENTANGLEMENT



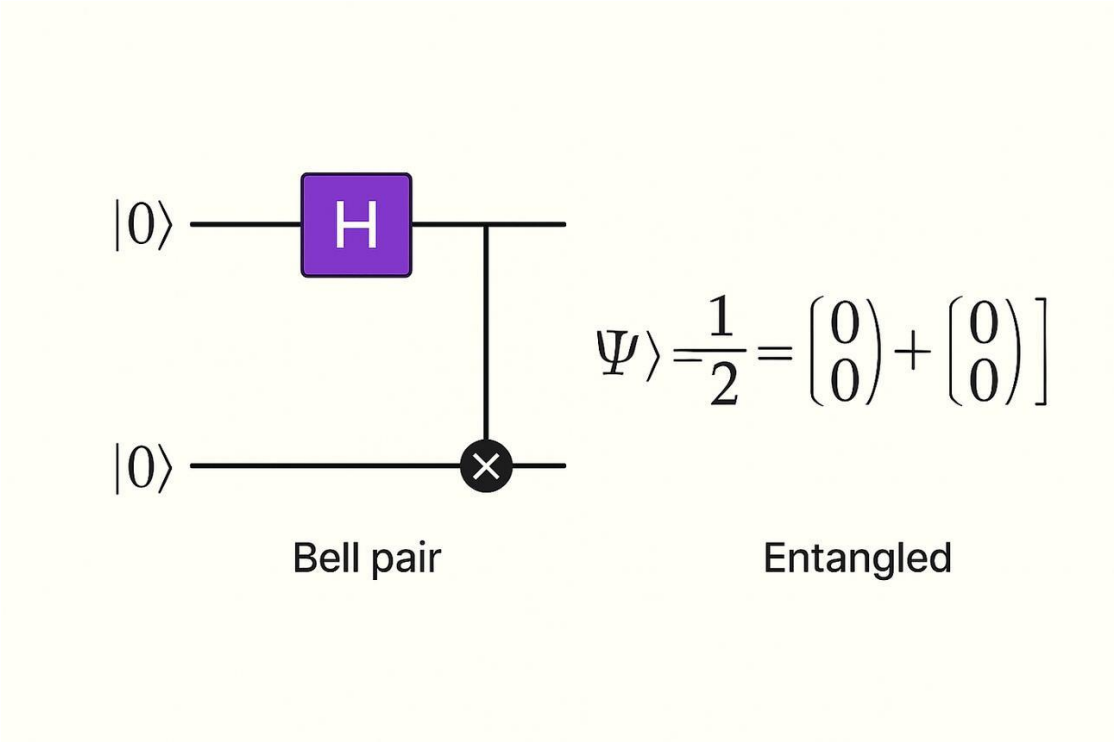
BELL STATES

$$|00\rangle + |11\rangle - \frac{1/2}{2}$$

$$|00-\rangle - |11\rangle - \frac{1/2}{2}$$

$$|01\rangle + |10\rangle - \frac{1/2}{2}$$

$$|01\rangle - |10\rangle - \frac{1/2}{2}$$



Ábra: *Összefonódott qubitek esetén az egyes részek helyett a teljes rendszer közös kvantumállapota válik alapvetővé.*

Az összefonódás a kvantumalgoritmusok és a kvantumkommunikáció egyik kulcsfontosságú erőforrása.

Einstein híresen idegenkedett a jelenségtől, de az összefonódás nem puszta elméleti furcsaság: kísérletek sora igazolta, és ma már laboratóriumi technológiákban tudatosan létrehozzák és használják.

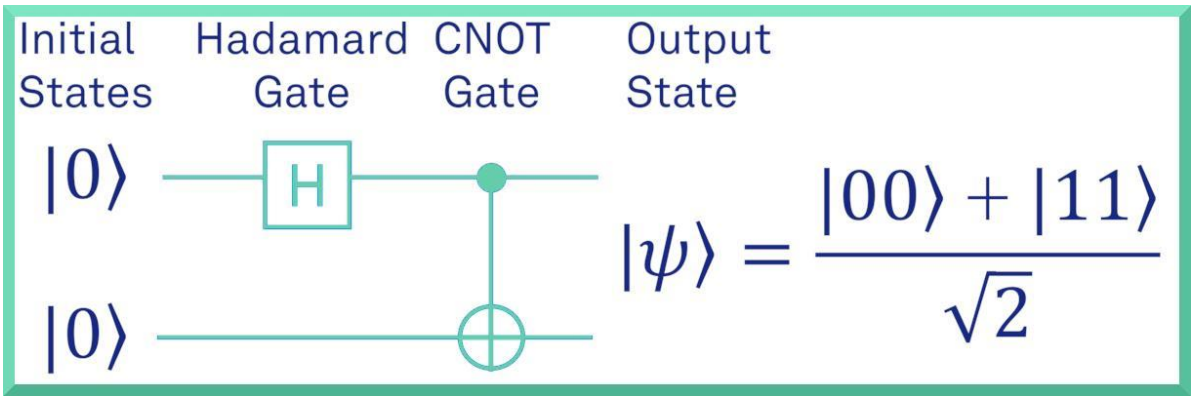
21.4. Kvantumkapuk

A klasszikus számítógépben logikai kapuk dolgoznak:

AND, OR, NOT, XOR...

A kvantumszámítógépben ezek megfelelői a **kvantumkapuk**.

Ilyen például a Hadamard-kapu, amely szuperpozíció létrehozására használható, vagy a CNOT-kapu, amely két qubit között hozhat létre megfelelő körülmények között összefonódást.



Ábra: *Egyszerű kvantumáramkör. A vízszintes vonalak a qubiteket, a rajtuk elhelyezett jelek a kvantumkapukat ábrázolják. A számítás végén a qubiteket megmérjük.*

A kvantumprogram tehát első látásra meglepően hasonlít egy elektronikus kapcsolási rajzhoz:

qubitek → kvantumkapuk → mérés

A különbség az, hogy itt a kapuk kvantumállapotokat alakítanak át.

21.5. Kvantumalgoritmusok

És itt következik a legfontosabb rész.

Önmagában attól, hogy létrehozunk sok qubitből álló szuperpozíciót, még nem kapunk hasznos számítógépet.

Olyan algoritmust kell készíteni, amely a kvantumállapotokat úgy manipulálja, hogy az **interferencia** a számunkra hasznos eredmények amplitúdóját erősítse, a nem kívánt eredményekét pedig gyengítse.

A kvantumalgoritmus tehát nem egyszerűen párhuzamos számítás.

A legismertebb kvantumalgoritmusok közé tartozik **Shor faktorizáló algoritmusa** és **Grover keresési algoritmusa**.

És fontos: nem minden problémára ismerünk olyan kvantumalgoritmust, amely lényegesen jobb a legjobb klasszikus módszernél.

21.6. Shor algoritmusa

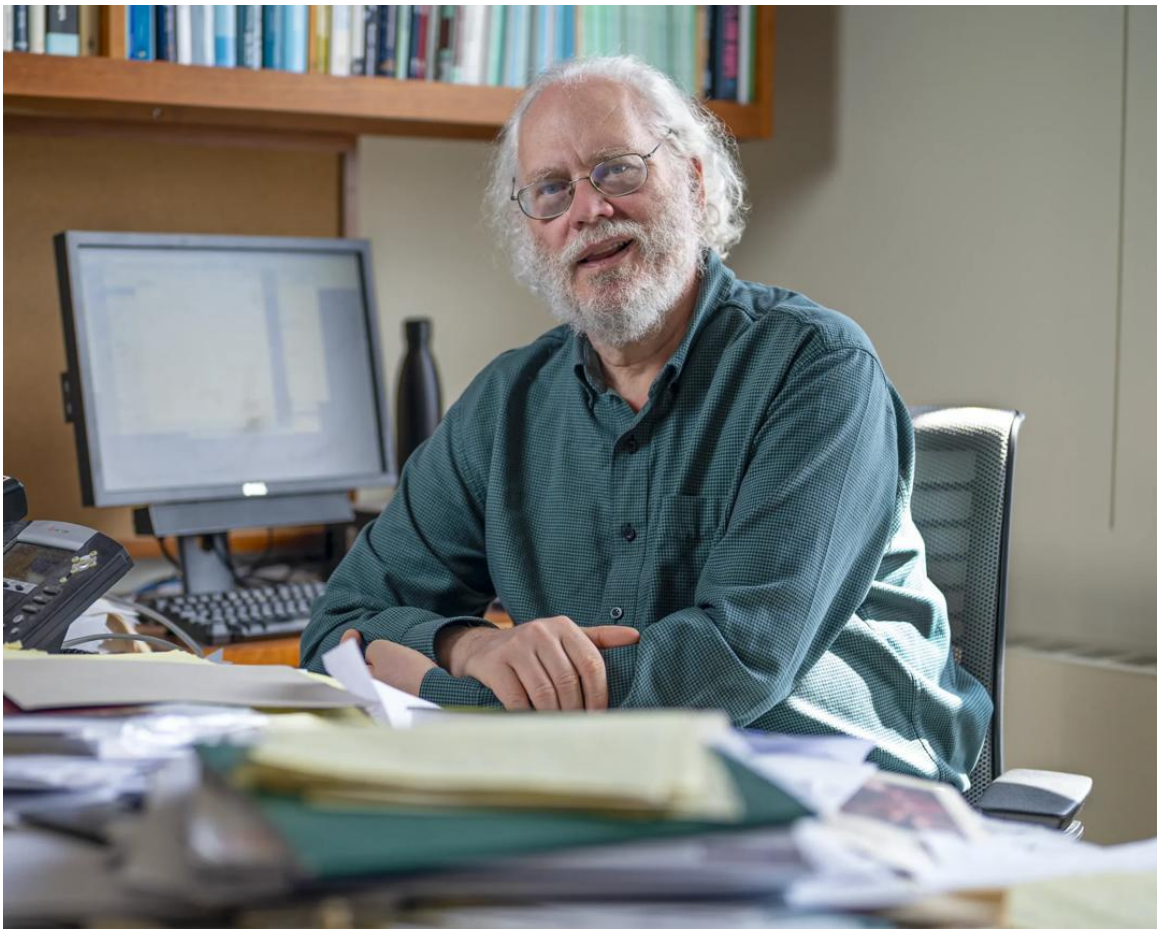
1994-ben **Peter Shor** olyan kvantumalgoritmust mutatott be, amely nagy egész számok prímtényezőkre bontását elméletileg drámaian gyorsabban végezheti el, mint az ismert klasszikus algoritmusok.

Ez első pillantásra matematikai érdekességnek tűnhet.

Csakhogy a széles körben használt **RSA nyilvános kulcsú titkosítás biztonsága** szorosan kapcsolódik a nagy számok faktorizálásának klasszikus nehézségéhez.

Egy kellően nagy, hibajavított kvantumszámítógép ezért elvileg veszélyeztethetné az RSA-t.

Shor algoritmusa óriási lendületet adott a kvantumszámítógépek kutatásának: hirtelen volt egy rendkívül fontos probléma, amelynél a kvantumgép nem egyszerűen kicsit gyorsabbnak ígérkezett.



Ábra: *Peter Shor 1994-ben publikált algoritmusa megmutatta, hogy egy megfelelő kvantumszámítógép bizonyos matematikai problémákban alapvetően felülmúlhatja a klasszikus számítógépeket.*

21.7. Miért olyan nehéz kvantumszámítógépet építeni?

A kvantumállapotok rendkívül érzékenyek.

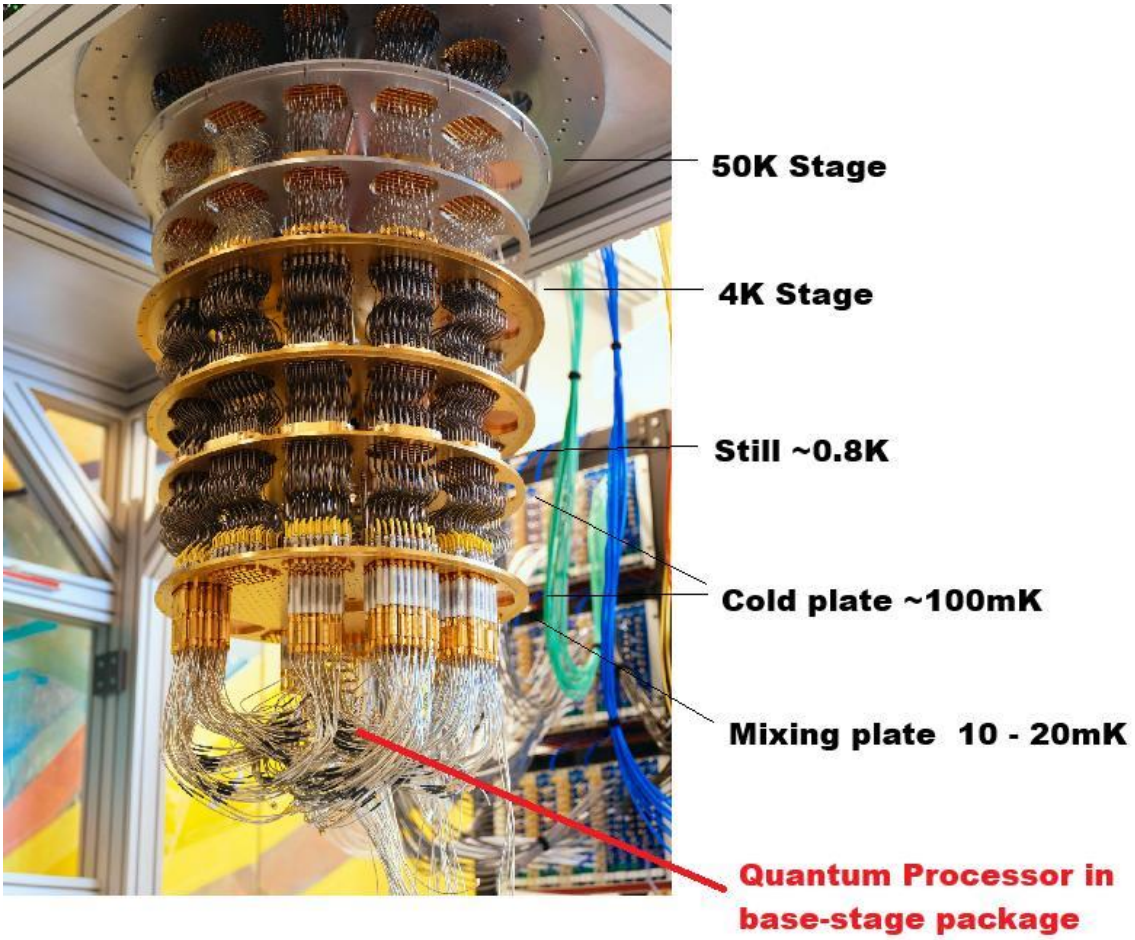
A környezet hőmozgása, elektromágneses zaj, anyaghibák és számos más zavaró hatás tönkreteheti a gondosan létrehozott kvantumállapotot.

Ezt nevezzük **dekoherenciának.**

Ezért bizonyos kvantumprocesszorokat az abszolút nulla fok közelében működtetnek.

0 kelvin = −273,15 °C.

A szupravezető qubiteket használó rendszerek tipikusan néhány millikelvines tartományban dolgoznak.



És még ez sem old meg mindent.

A qubitek hibáznak.

Ezért **kvantum-hibajavításra** van szükség. Egyetlen megbízható, úgynevezett logikai qubit előállításához a technológiától és hibaaránytól függően sok fizikai qubitre lehet szükség.

Ez az egyik oka annak, hogy óriási különbség van aközött, hogy egy gépnek „sok qubitje van”, és aközött, hogy hány **jó minőségű, hasznos logikai qubitet** tud működtetni.

21.8. Mire használható?

A kvantumszámítógép legígéretesebb alkalmazásai olyan problémák, amelyek szerkezete jól illeszkedik a kvantummechanikai számításhoz.

Különösen fontos lehet:

- kvantumrendszerek szimulációjában,
- molekulák és új anyagok vizsgálatában,
- egyes kriptográfiai problémákban,
- bizonyos keresési feladatokban,
- egyes optimalizálási és matematikai problémákban.

Talán a legtermészetesebb alkalmazás éppen a **kvantummechanika szimulációja**.

Richard Feynman már az 1980-as évek elején rámutatott: a kvantumrendszereket klasszikus számítógépen rendkívül nehéz szimulálni.

Miért ne használjunk hozzájuk olyan számítógépet, amely **maga is kvantummechanikai rendszer?**



Ábra: *A kvantumszámítógépek egyik legígéretesebb alkalmazási területe molekulák és anyagok kvantummechanikai viselkedésének szimulációja.*

21.9. Mire nem használható?

A kvantumszámítógépet gyakran úgy mutatják be, mintha minden számítási feladatot elképesztően felgyorsítana.

Ez tévedés.

Nem lesz értelme kvantumszámítógépen levelet írni, táblázatot kezelni, honlapot böngészni vagy hagyományos operációs rendszert futtatni csak azért, mert „kvantum”.

A kvantumgyorsulás **probléma- és algoritmusfüggő.**

Ráadásul az eredmény kiolvasása is korlátozott. Hiába írható le egy kvantumrendszer állapota rengeteg amplitúdóval, ezeket nem tudjuk egyszerűen mind kiolvasni: a mérés klasszikus eredményt szolgáltat.

Ezért hibás az a népszerű magyarázat, hogy: „A kvantumszámítógép egyszerre kipróbál minden megoldást, ezért azonnal megtalálja a helyeset.”

Ha ilyen egyszerű volna, szinte minden számítási problémát könnyen megoldhatnánk vele.

Nem így van.

Elérkezett a kvantumelőny korszaka?

A kvantumszámítógépekről évtizedek óta azt halljuk, hogy egyszer olyan problémákat oldhatnak meg, amelyekkel a hagyományos számítógépek nem tudnak megbirkózni. Sokáig azonban jogos volt a kérdés: **mikor jön el ez az „egyszer”?**

2026-ban az IBM és a Chicagói Egyetem kutatói olyan eredményről számoltak be, amely szerintük már megfelel a **kvantumelőny – quantum advantage – alapvető követelményeinek.**

A kísérletben 70 logikai qubitet használtak, és 2415 kétqubites logikai műveletet, valamint 468 úgynevezett T-kaput hajtottak végre. A kvantumszámítógép körülbelül 15 perc alatt végezte el a feladatot, miközben a kutatók szerint a vezető klasszikus szimulációs módszerek számára ugyanennek a számításnak az elvégzése már gyakorlatilag kezelhetetlen erőforrásokat igényelt volna.

De a kísérlet legérdekesebb része talán nem is a sebesség volt.

Honnan tudjuk, hogy igaza van?

A kvantumelőny egyik különös paradoxona, hogy ha egy kvantumszámítógép valóban olyan problémát old meg, amelyet klasszikus számítógéppel már nem tudunk kiszámítani, akkor felmerül egy kellemetlen kérdés:

hogyan ellenőrizzük az eredményt?

Ha egy hagyományos szuperszámítógéppel újra ki tudnánk számítani, akkor éppen azt bizonyítanánk, hogy a probléma mégsem volt elérhetetlen a klasszikus számítástechnika számára.

Az IBM és a Chicagói Egyetem kísérletének egyik fontos újítása éppen ennek a problémának a kezelése volt. A kutatók olyan strukturált kvantumáramkört és hibadetektálási eljárást alkalmaztak, amely lehetővé tette, hogy **statisztikai bizonyossággal alsó korlátot adjanak arra, milyen hűséggel hajtotta végre a gép a számítást**, anélkül hogy a teljes eredményt klasszikus számítógéppel újra kellett volna számolni.

A hibajavítás azért is kulcsfontosságú, mert a qubitek rendkívül érzékenyek környezetükre. A fizikai qubitek hibáiból ezért logikai qubiteket kell létrehozni, amelyek már megbízhatóbban őrzik és dolgozzák fel a kvantuminformációt. A kísérletben alkalmazott eljárás körülbelül tízszeresére csökkentette az effektív logikai hibaarányt a fizikai hibaarányhoz képest.

Nem mindenben gyorsabb

Mindez azonban nem jelenti azt, hogy a kvantumszámítógép mostantól gyorsabb lenne a hagyományos számítógépnél.

Egy szövegszerkesztő, adatbázis, webböngésző vagy táblázatkezelő számára semmi értelme kvantumszámítógépet használni. A kvantumgépek várható előnye speciális problémákban jelentkezhet: bizonyos fizikai és kémiai szimulációkban, optimalizálási feladatokban, anyagtudományban, kriptográfiában és más, különösen nagy számításigényű problémákban.

A 2026-os eredmény ráadásul egy speciálisan megkonstruált, klasszikusan nehezen szimulálható kvantumáramkörre vonatkozott. Ez még nem ugyanaz, mint egy gazdaságilag vagy tudományosan hasznos probléma kvantumszámítógépes megoldása.

A különbség fontos: **kvantumelőnyt demonstrálni nem ugyanaz, mint hasznos kvantumelőnyt elérni.**

Verseny a kvantum- és a klasszikus gépek között

Van még egy érdekes bizonytalanság. Amikor egy kvantumszámítógép új rekordot ér el, a klasszikus számítástechnika kutatói sem maradnak tétlenek. Új algoritmusokat és hatékonyabb szimulációs eljárásokat fejlesztenek, és előfordult már, hogy egy korábban klasszikusan szinte lehetetlennek tartott kvantumkísérletet később sokkal hatékonyabban sikerült klasszikus gépeken szimulálni.

A kvantumelőny ezért nem feltétlenül egyszer átlépett, mozdíthatatlan határvonal.

Inkább **mozgó frontvonal**. A kvantumszámítógépek fejlődnek – és közben a klasszikus számítógépek algoritmusai is fejlődnek.

A számítógép többé nem feltétlenül egyféle gép

Ez az eredmény egy nagyobb technológiai változás része lehet.

A számítástechnika első évtizedeiben lényegében két nagy paradigma versenyzett egymással: az analóg és a digitális számítás. A digitális számítógép elsöprő győzelmet aratott, és évtizedekre szinte egyeduralkodóvá vált.

Most azonban ismét szaporodnak a lehetőségek:

CPU – GPU – AI-gyorsító – analóg és neuromorf processzor – fotonikus számítás – kvantumszámítógép.

Valószínűleg egyik sem fogja egyszerűen leváltani az összes többit.

Sokkal valószínűbb egy **heterogén számítástechnikai világ**, amelyben a különböző feladatokat különböző fizikai elveken működő processzorok végzik. A hagyományos digitális számítógép szervezheti a munkát, a GPU végezheti a párhuzamos számításokat, egy analóg vagy neuromorf gyorsító dolgozhat fel bizonyos AI-feladatokat, egy fotonikus egység kezelheti az optikai műveleteket, a kvantumprocesszor pedig megoldhatja azt a speciális részproblémát, amelyben valóban előnye van.

A jövő szuperszámítógépe ezért talán már nem egyszerűen **egy nagyon gyors számítógép** lesz. Inkább különböző számítási elveket használó gépek együttműködő rendszere.

A kvantumszámítógép történetének legfontosabb pillanata pedig talán nem az lesz, amikor leváltja a klasszikus számítógépet.

Hanem amikor természetessé válik, hogy bizonyos problémák megoldásakor átadjuk neki azt a néhány percnyi munkát, amelyet semmilyen hagyományos gépre nem érdemes rábízni.

21.10. Leváltja-e a hagyományos számítógépet?

Nagy valószínűséggel **nem**.

A kvantumszámítógép inkább különleges gyorsító lehet, ahogy ma a GPU is az.

A számítás jelentős részét továbbra is hagyományos számítógép végzi, és csak azt a speciális részfeladatot küldi a kvantumprocesszornak, amelyre annak működése előnyt jelent.

A jövő számítógépe ezért elképzelhető így:

CPU + GPU + AI-gyorsító + kvantumprocesszor

Mindegyik azt a feladatot végzi, amelyhez a saját architektúrája a legalkalmasabb.

Ebben az értelemben a kvantumszámítógép nem a számítógép történetének végét jelenti.

Újabb specializált számítási eszközt ad a kezünkbe.

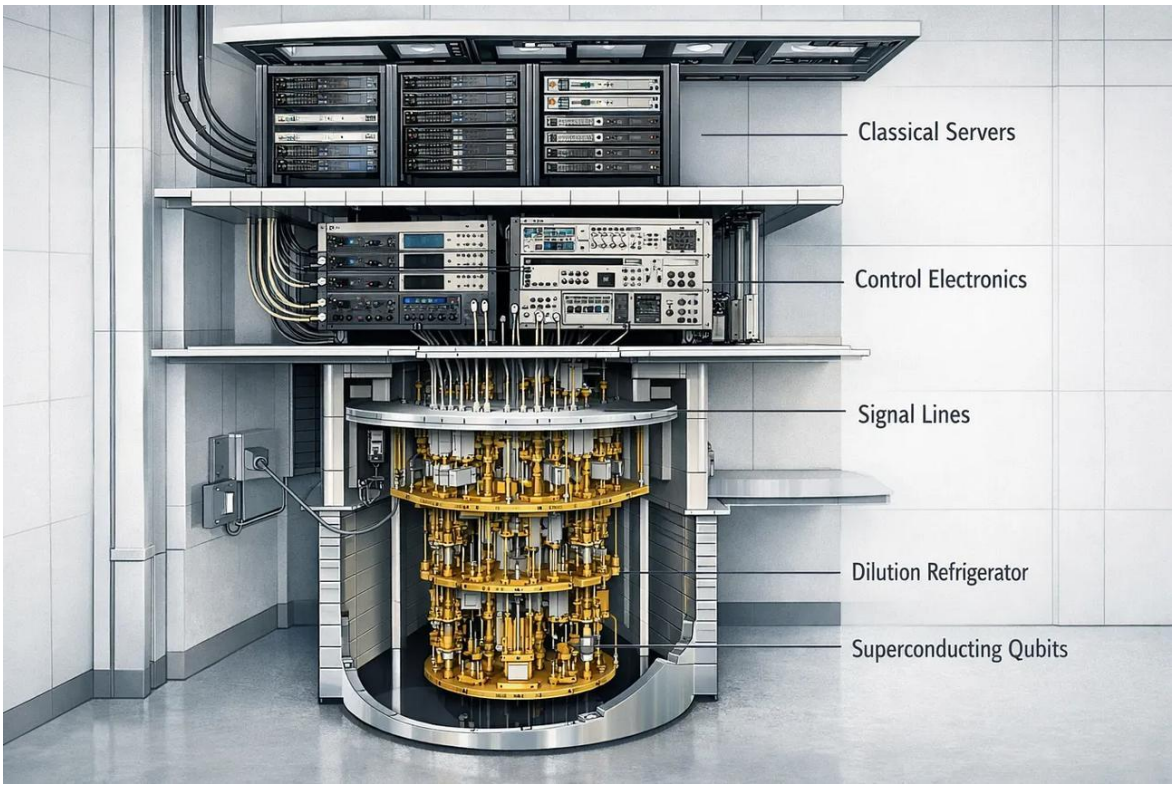
Nem minden számítási problémához kell gyorsabb számítógép

A kvantumszámítógépek fejlődése azt ígéri, hogy bizonyos problémákat olyan sebességgel oldhatunk meg, amely a klasszikus gépek számára elérhetetlen. A mesterséges intelligencia fejlődése azonban egy másik lehetőséget is felvet: mi történik, ha nem a számítógépet tesszük gyorsabbá, hanem **jobb módszert találunk a problémára?**

2026-ban az Anthropic Claude Mythos modellje egy posztkvantum kriptográfiai eljárásban olyan matematikai gyengeséget talált, amelyet korábban éveken át vizsgáló szakértők sem fedeztek fel. Nem óriási számítási teljesítménnyel győzte le a problémát: **jobb támadási módszert talált.**

A jövő számítástechnikájának versenye ezért talán nem egyszerűen arról szól majd, hogy ki rendelkezik a gyorsabb processzorral, nagyobb AI-adatközponttal vagy jobb kvantumszámítógéppel. Egy új szereplő is megjelent:

a gép, amely maga találja ki, hogyan érdemes megoldani a problémát.



A kvantumszámítógép nem egyszerűen nagyon gyors számítógép

Ez talán a kvantumszámítógépekkel kapcsolatos legfontosabb félreértés.

Egy hagyományos processzort könnyű úgy elképzelni, hogy az újabb változat egyszerűen gyorsabban hajtja végre ugyanazokat az utasításokat. A kvantumszámítógép azonban **nem ilyen gyorsabb CPU**.

Más módon reprezentálja és dolgozza fel az információt.

A qubitek szuperpozíciója, összefonódása és a kvantuminterferencia lehetővé tesz olyan algoritmusokat, amelyek **bizonyos speciális problémákat** más számítási úton oldanak meg.

Ezért nem helyes azt kérdezni: **„Hányszor gyorsabb egy kvantumszámítógép a PC-mnél?”**

Helyesebb: **„Van-e ehhez a problémához olyan kvantumalgoritmus, amely előnyt jelent a legjobb klasszikus módszerhez képest?”**

Ha nincs, akkor a kvantumszámítógép akár teljesen haszontalan is lehet az adott feladathoz.

Jó hasonlat lehet a GPU: egy modern GPU bizonyos párhuzamos számításokban elképesztően gyors, de ettől még nem cseréljük le vele egyszerűen a CPU-t.

A kvantumszámítógép ereje nem abban rejlik, hogy mindent gyorsabban csinál, hanem abban, hogy bizonyos dolgokat egészen másképpen tud kiszámítani.

Veszélyben van-e az RSA?

Az RSA biztonságának egyik alapja, hogy két nagyon nagy prímszámot könnyű összeszorozni, de a kapott hatalmas számot klasszikus számítógéppel rendkívül nehéz visszabontani prímtényezőire.

Peter Shor 1994-es kvantumalgoritmusa azonban megmutatta, hogy **egy kellően nagy és hibajavított kvantumszámítógép ezt a problémát sokkal hatékonyabban oldhatná meg**.

Ez nemcsak az RSA-t érinti. Más ma használt nyilvános kulcsú eljárások – különösen az elliptikus görbékre épülők – szintén veszélyeztetettek lennének.

Ez azonban **nem jelenti azt, hogy a mai kvantumszámítógépekkel feltörhető az internet RSA-titkosítása**. Az ehhez szükséges hibatűrő kvantumgép lényegesen meghaladja a jelenlegi rendszerek képességeit.

A veszélyt mégis komolyan kell venni, mert egy ma megszerzett titkosított adatot el lehet tárolni, és esetleg évekkel később visszafejteni. Ezt nevezik:

„harvest now, decrypt later" – gyűjtsd be most, fejtsd vissza később.

Ezért már folyik az átállás **posztkvantum-kriptográfiára**: olyan klasszikus számítógépeken is használható titkosítási és digitális aláírási módszerekre, amelyek a jelenlegi ismereteink szerint a kvantumtámadásoknak is ellenállnak.

2024-ben az amerikai NIST közzétette az első végleges posztkvantum-kriptográfiai szabványokat. Az átállás tehát **nem a távoli jövő problémája – már megkezdődött.**

Epilógus – Ötezer év számolás

Amikor ennek a könyvnek az első oldalain az ősember ujjain számolt, még senki sem sejthette, hogy egyszer több milliárd tranzisztorból álló számítógépek milliárdszor milliárd műveletet végeznek majd másodpercenként. A számolás története azonban nem csupán a gépek története. Minden új eszköz ugyanazt a célt szolgálta: hogy az emberi gondolkodást segítse.

Az első kavicsok, rovások és abakuszok még csak megkönnyítették a számolást. A mechanikus számológépek már átvették a fárasztó műveleteket. A lyukkártyák megtanították a gépeket adatokkal dolgozni. Az elektronika megszületésével a számítás sebessége robbanásszerűen megnőtt, a mikroprocesszor pedig a számítógépet mindenki számára elérhetővé tette.

A következő nagy lépés az volt, amikor a számítógép már nem csupán számolt, hanem szöveget írt, képet rajzolt, zenét játszott, kapcsolatot teremtett emberek között, végül pedig segíteni kezdett a tanulásban és a döntések előkészítésében is. A számológépből információfeldolgozó gép, majd tudást kezelő rendszer lett.

A fejlődés üteme szinte felfoghatatlan. Az ókori abakusz évezredeken át alig változott. A mechanikus számológépek fejlődéséhez évszázadokra volt szükség. Az elektronikus számítógépek története alig nyolcvan év, a személyi számítógépeké kevesebb mint ötven, az okostelefonoké alig két évtized. Ma már olyan teljesítményű számítógép lapul a zsebünkben, amely messze felülmúlja azokat a gépeket, amelyekkel egykor az első űrhajókat irányították.

És még mindig nem értünk a történet végére.

A kvantumszámítógépek, a neuromorf rendszerek, az optikai számítástechnika vagy a mesterséges intelligencia újabb fejezeteket nyithatnak. Nem tudjuk, melyik technológia válik meghatározóvá, de egy dolog biztosnak látszik: az ember mindig új eszközöket fog alkotni azért, hogy jobban megértse a világot.

A könyv lapjain végigkísértük a számolás útját az ujjaktól a qubitekig. Megismertük a feltalálókat, mérnököket, matematikusokat és olyan embereket, akik sokszor egész életüket egyetlen ötlet megvalósításának szentelték. Munkájuk nélkül ma egészen más világban élnénk.

Ez a történet azonban nem csupán a híres tudósokról szól. Minden alkalommal folytatódik, amikor valaki bekapcsol egy számítógépet, programot ír, adatot elemez, vagy egyszerűen csak kíváncsian feltesz egy kérdést. A számítástechnika fejlődését mindig az emberi kíváncsiság hajtotta előre.

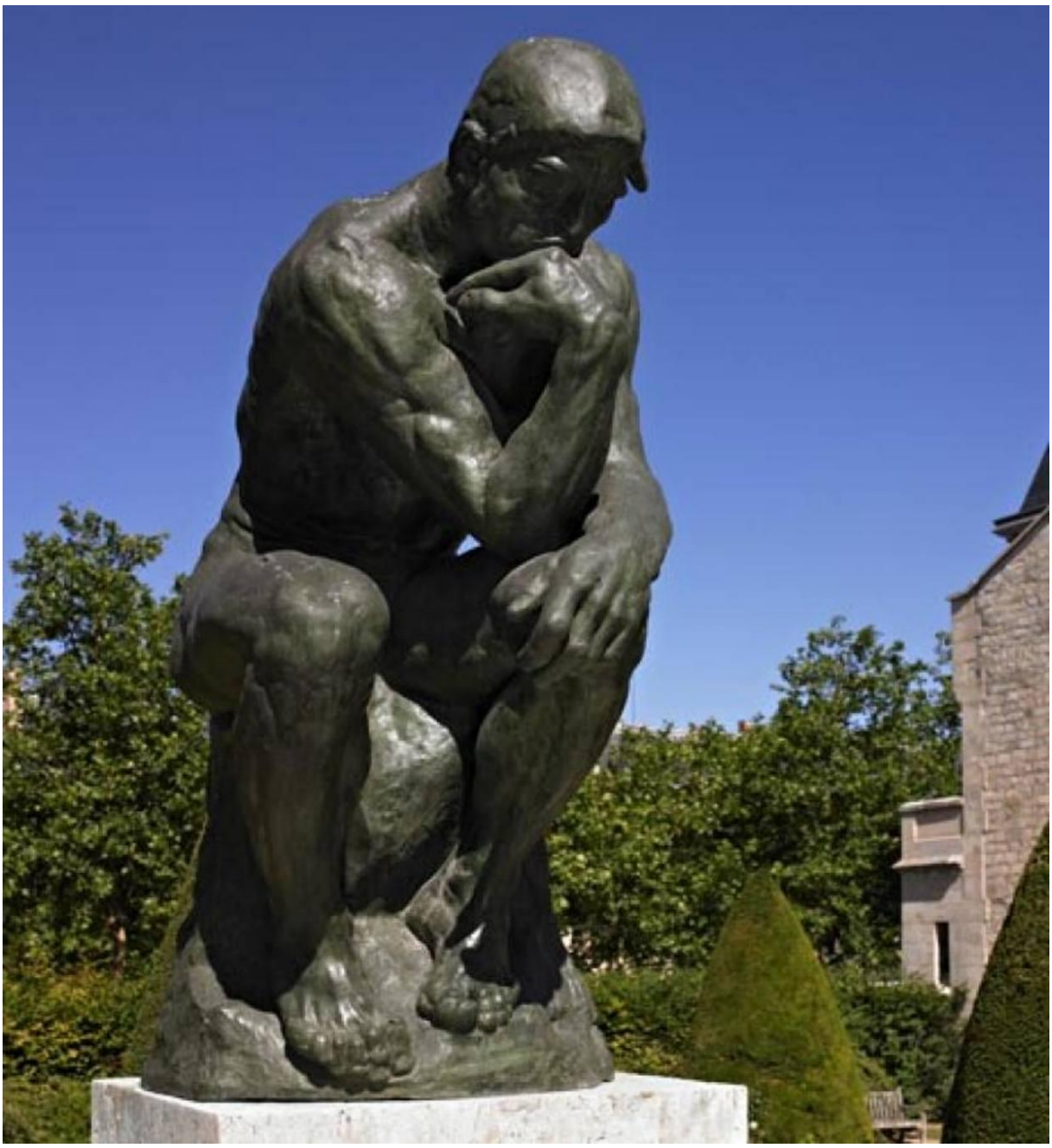
Ötezer évvel ezelőtt az ember megtanult számolni.

Ma már a számítógépek is képesek számolni.

A következő nagy kérdés azonban már nem az, hogy **mit** tudnak kiszámítani a gépek, hanem az, hogy **mi emberek mire használjuk ezt a tudást.**

A számítógépek története valójában nem a gépek, hanem az emberi kíváncsiság, találékonyság és alkotóerő története, amelynek legfontosabb szereplője végig ugyanaz maradt:

a gondolkodó ember.



Auguste Rodin: A gondolkodó (Le Penseur), 1880 körül.

Mi van a dobozban?

A digitális világ felépítése és működése

A második kötet a számítógép belsejébe és a digitális világ kulisszái mögé vezet. Megismerjük, hogyan működnek a processzorok, memóriák és adathordozók, hogyan születnek a programok és operációs rendszerek, miként digitalizálódott kép, hang és film, és hogyan kapcsolódik össze mindez világhálózattá.

A könyv végigvezeti az olvasót a szoftverek, a hálózatok, az internet és a kiberbiztonság, a mesterséges intelligencia és a kvantumszámítógép világán – egészen a jövő lehetőségeiig.

A technika története – emberközelből, képekben elmesélve.

- **2020+** Felhő • AI • IoT • Kvantum
- **2010** Okostelefonok • Közösségi hálók
- **2000** Internet • Böngészők • Wi-Fi
- **1990** Windows • Internet terjedése
- **1980** PC-k • Hálózatok • Játékok
- **1970** Mikroprocesszorok • BASIC
- **1960** Integrált áramkörök
- **1950** Tranzisztorok • Mágneses tárak
- **1940** Elektroncsövek • Relék



ISBN 978-615-01-2345-6