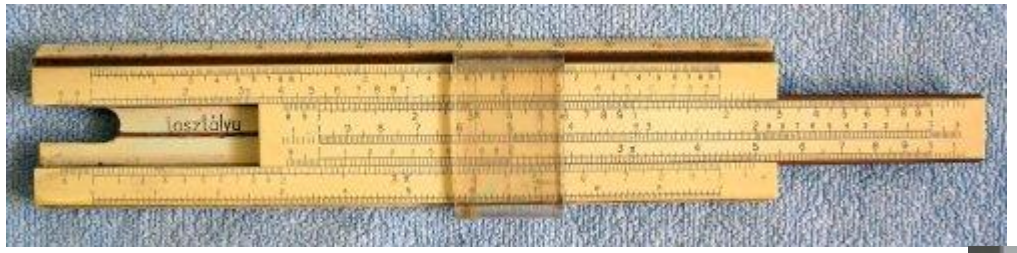
2. fejezet

Hardver

Grafikus Processzorok Tudományos Célú Programozása

A számítógépek története



- Mechanikus „számítógépek”:

A számítógépek története

Mechanikus „számítógépek”:
egyetlen célfeladat sokszori, parametrikus megoldása

- Periodikus jelenségek jóslása
  (csillagászat, ár-apály)
- Aritmetikai, polinom műveletek, harmonikus sorok
- Később: differenciál-egyenletek megoldása

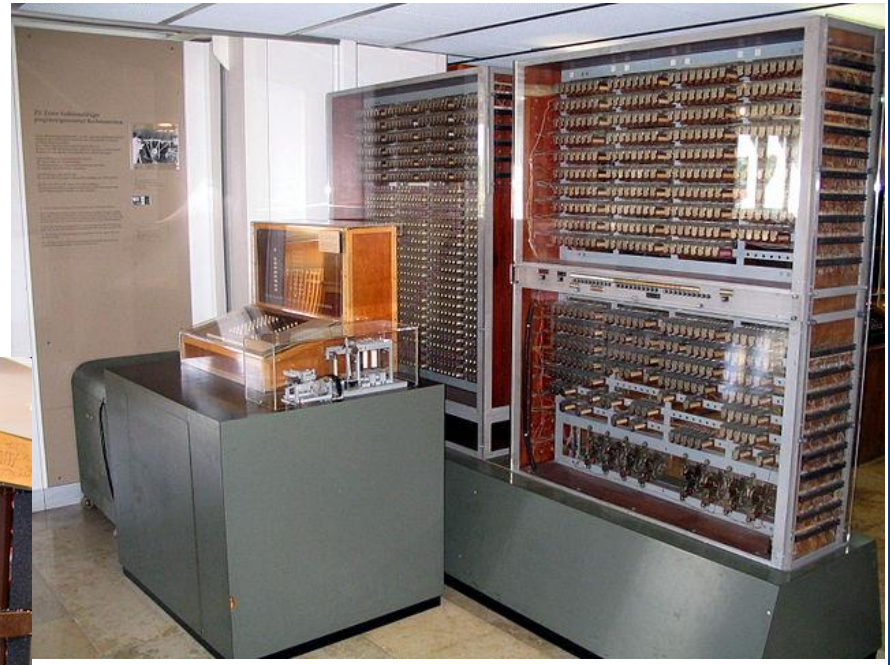
A számítógépek története

- Elektro-mechanikus analóg eszközök:

  nagy dinamikus rendszerek modellezése hadi alkalmazások…

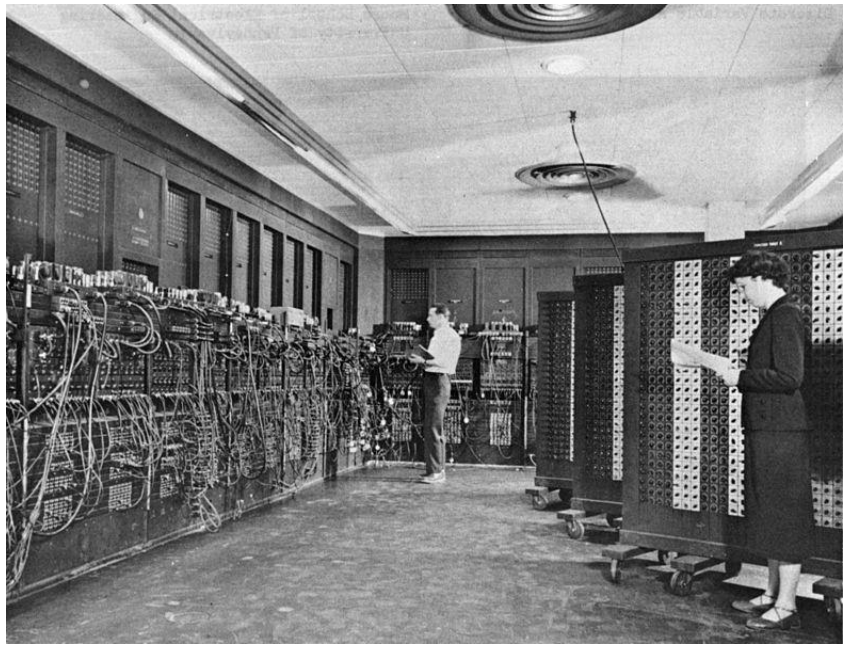
  Hálózat analizátorok
  Z1, Z3

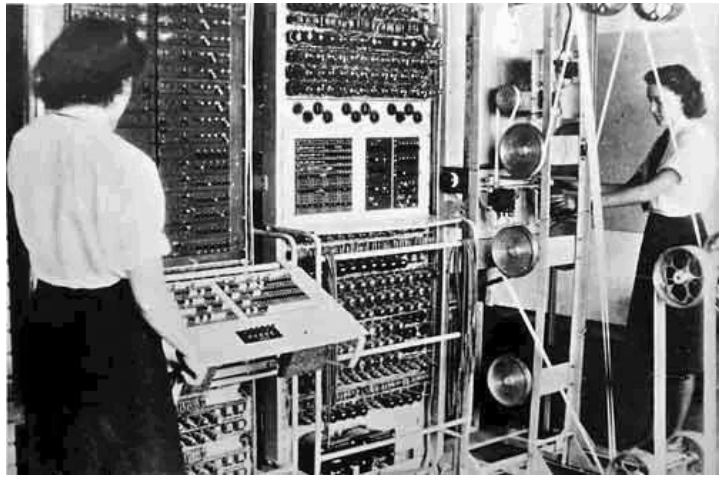
  titkosítás:
  ENIGMA - Bombe



A számítógépek története

- Első programozható számítógépek:
  Colossus
  ENIAC
  EDVAC



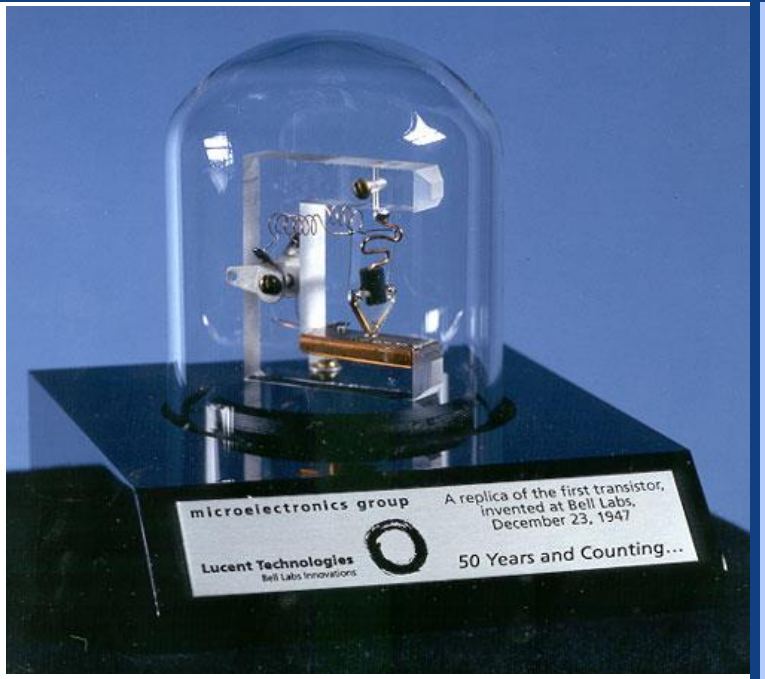


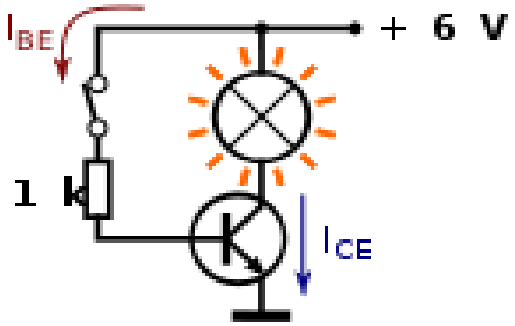
Technológia: vákuum csövek

A számítógépek története

- A tranzisztor (1947):

  Olcsó,
  tömegesen előállítható,
  megbízható
  flexibilis

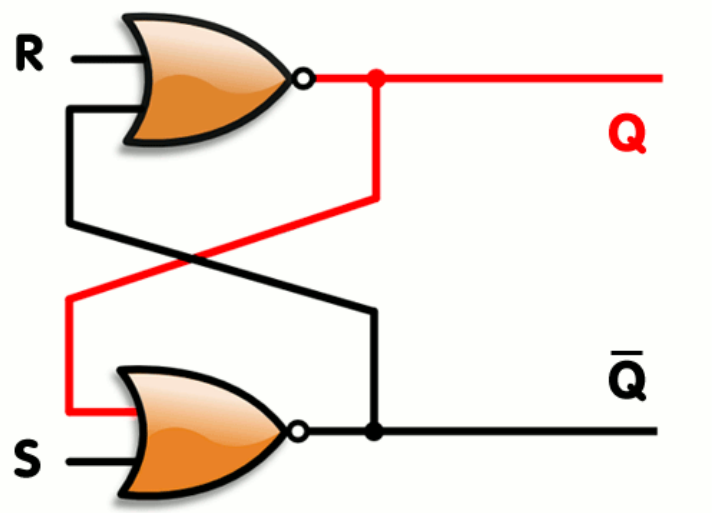


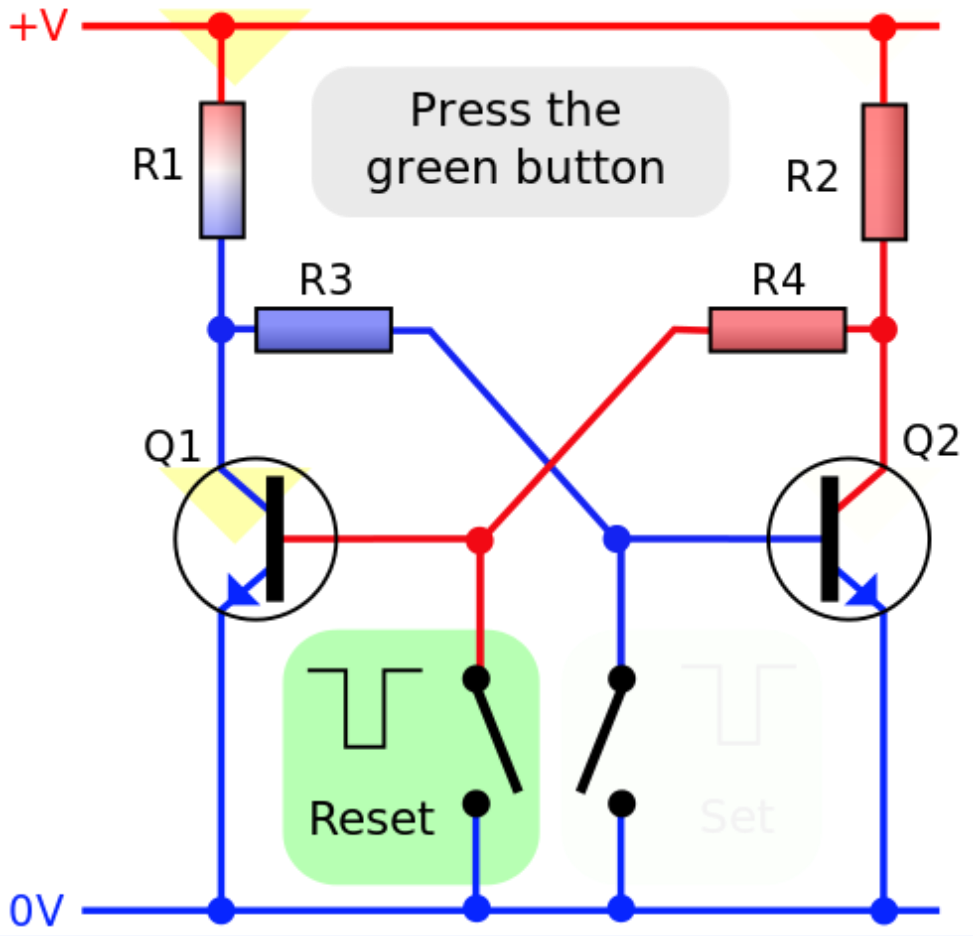


Első tranzisztor alapú számítógépek

A számítógépek története

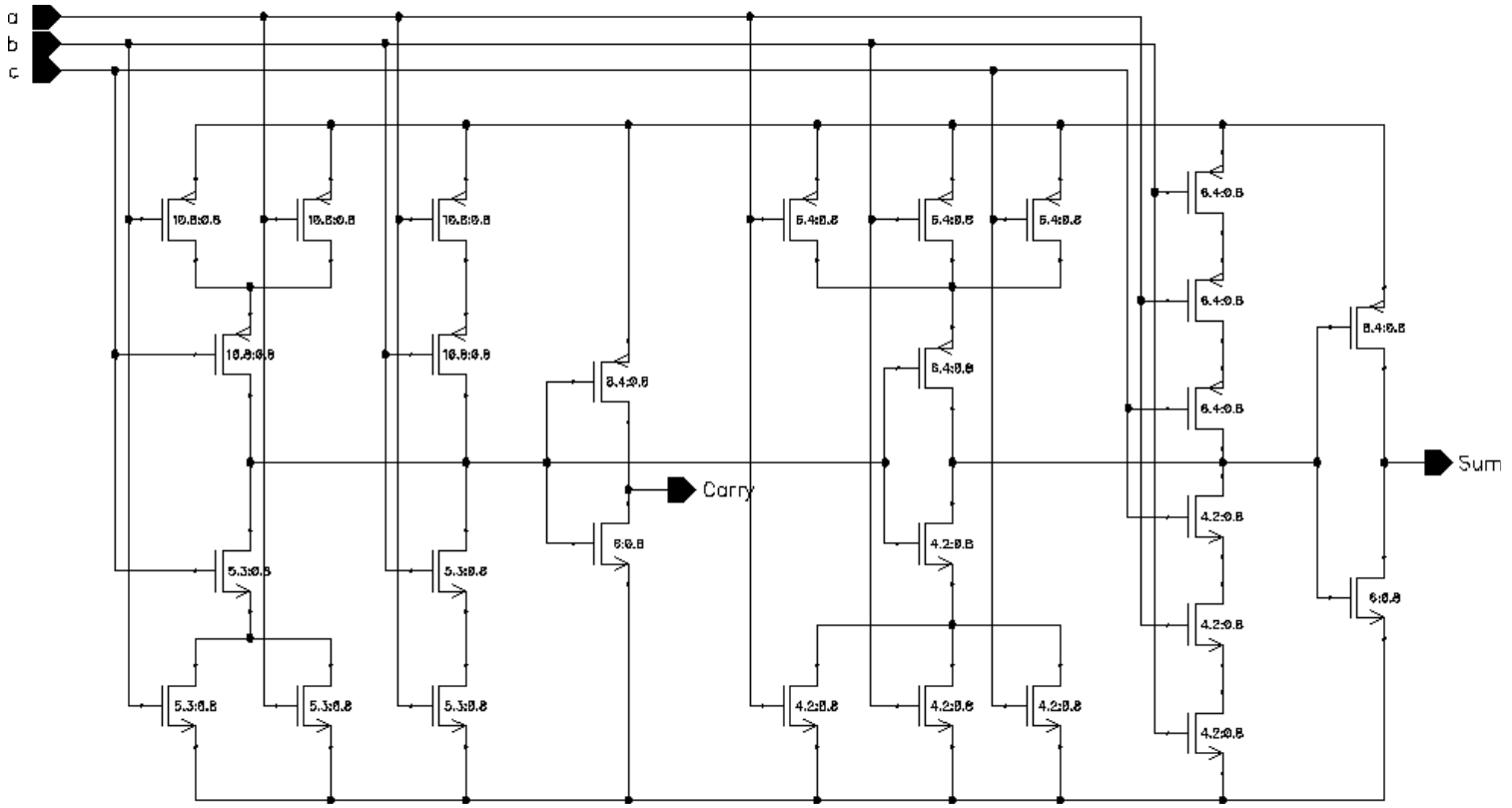
- A legegyszerűbb memória tranzisztorokból: flip-flop:





A számítógépek története

- Összeadás tranzisztorokkal:

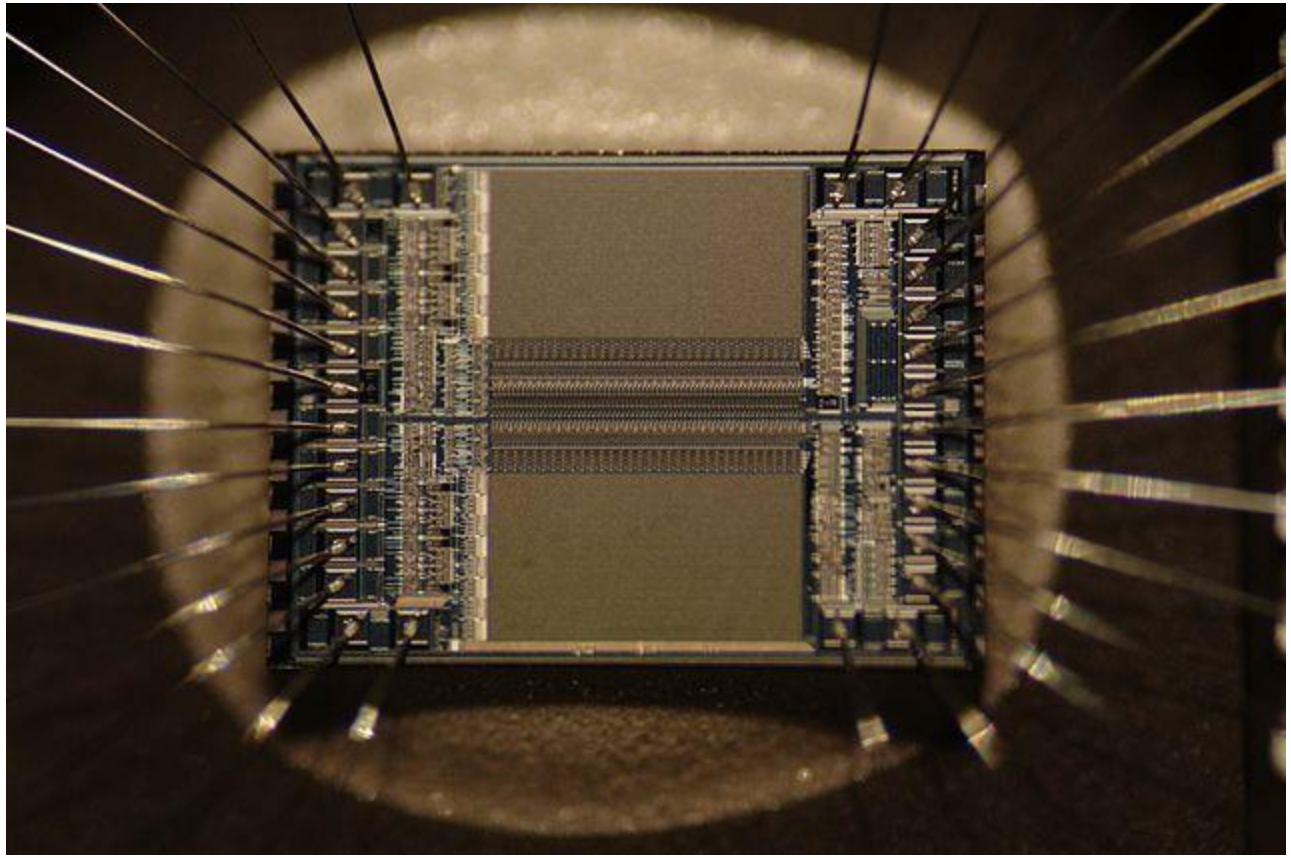


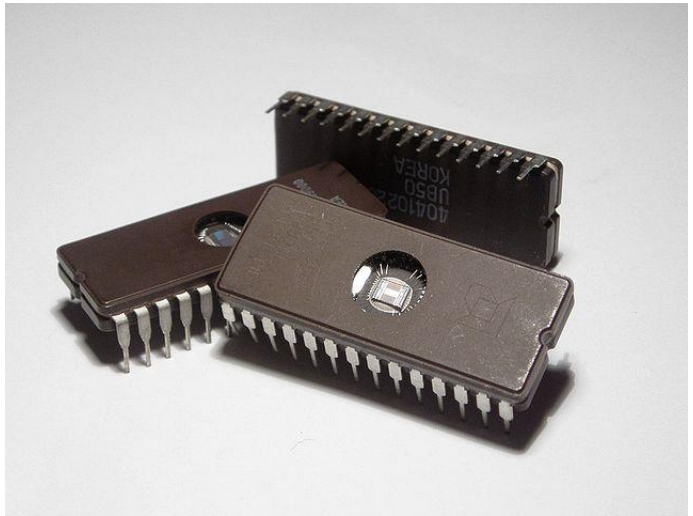
A számítógépek története

- Regiszter:
  egy bit sorozatot tároló áramkör, ami egyben hozzáférhető a többi komponens számára. Jellemző: hány bit van benne (szélesség)
- Busz:
  egy kapcsolat, amin keresztül a bitsorozatok elérhetővé válnak az alkatrészek között. Fajtái: soros / párhuzamos
- Multiplexing: ha egy erőforrás korlátozott, akkor több felhasználási módnak esetleg osztoznia kell rajta:
  pl. két busz egy vezetéken, pl.: időosztással
- Órajel: szinkronizációs jel, hogy a különböző áramköri részek összehangoltan működjenek.

A számítógépek története

- Integrált áramkör



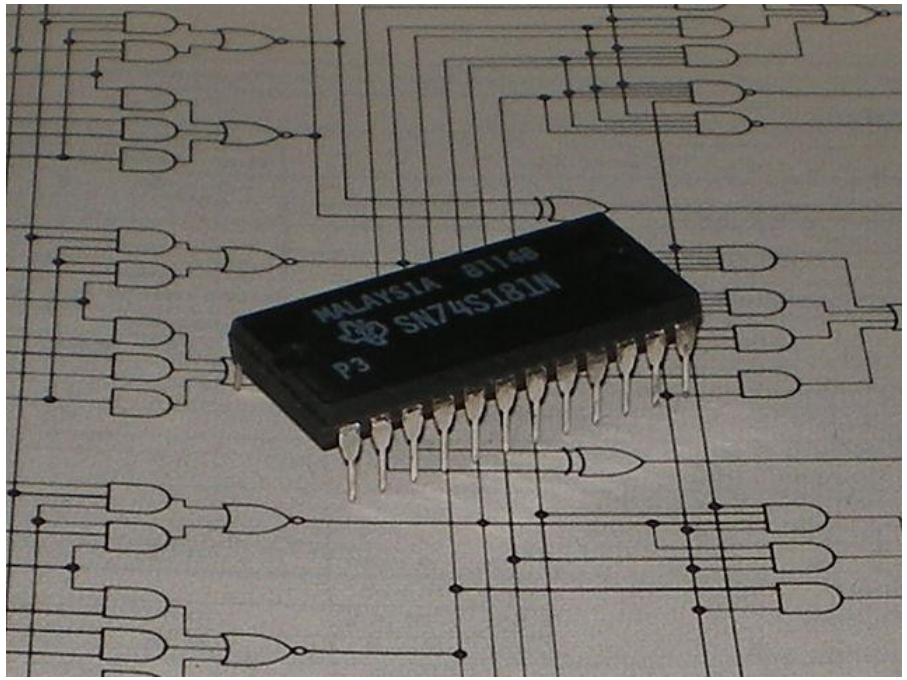


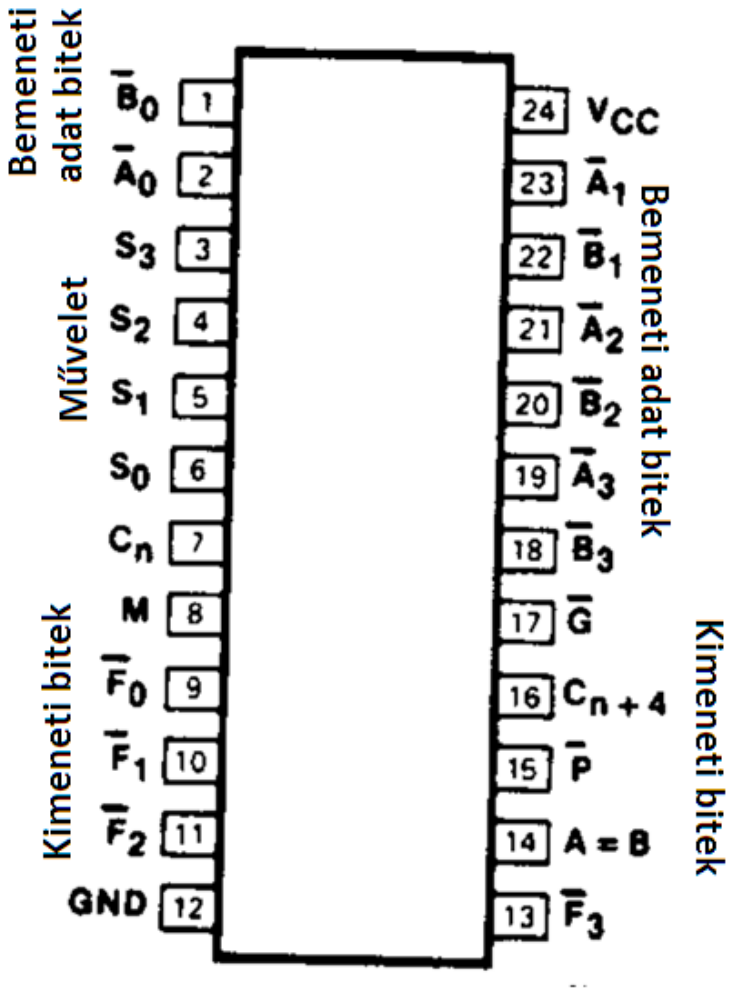
Gyártástechnológia:

Fotolitográfia

Jellemző: csíkszélesség

A mikroprocesszorok története

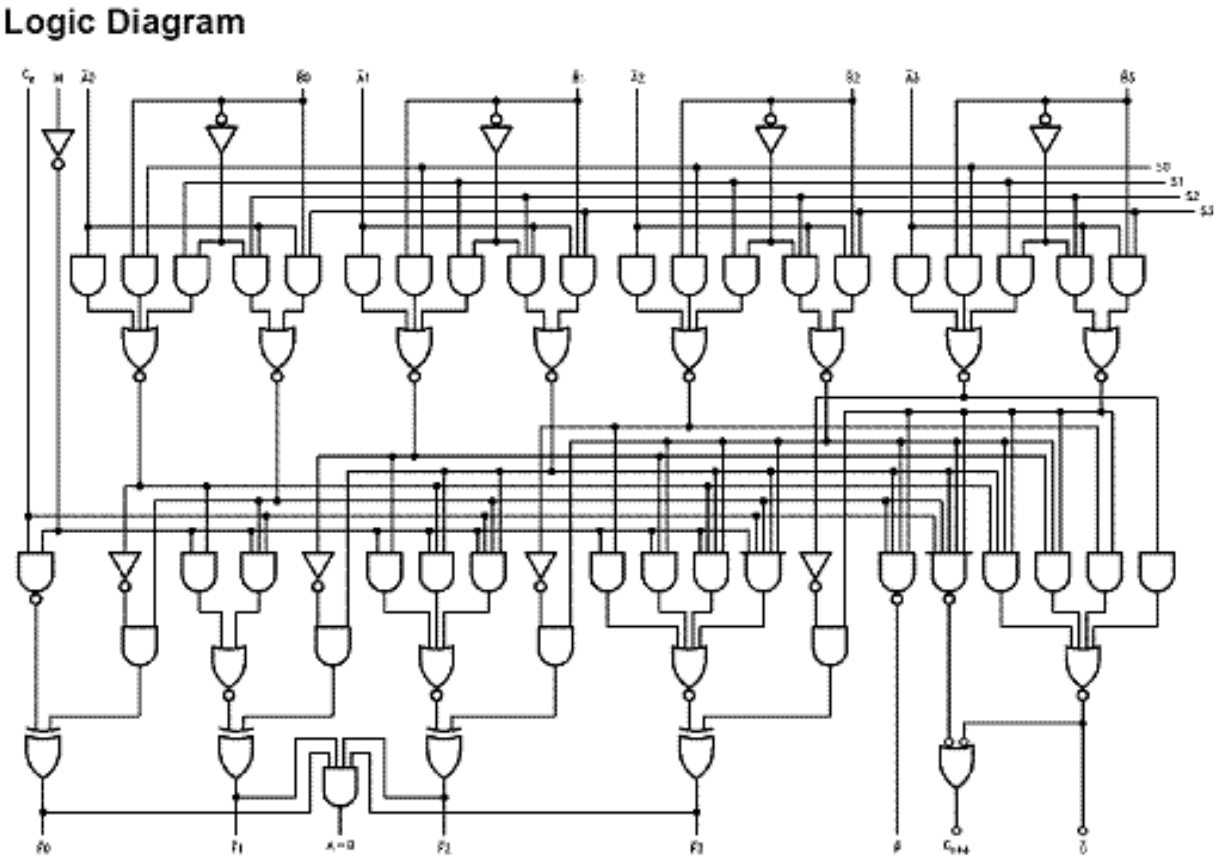




- Egyszerű 4 bit-es ALU (Arithmetic Logic Unit): 74181

A mikroprocesszorok története

- Egy egyszerű ALU (Arithmetic Logic Unit): 74181



$S3$	$S2$	$S1$	$S0$	$M=1$	$M=0$ $CIN=1$	$M=0$ $CIN=0$
0	0	0	0	$\overline{A}$	A	$A+1$
0	0	0	1	$\overline{A\|B}$	$A\|B$	$A\|B+1$
0	0	1	0	$\overline{A}B$	$A\|\overline{B}$	$A\|\overline{B}+1$
0	0	1	1	0	-1	0
0	1	0	0	$\overline{AB}$	$A+A\overline{B}$	$A+A\overline{B}+1$
0	1	0	1	$\overline{B}$	$(A\|B)+A\overline{B}$	$(A\|B)+A\overline{B}+1$
0	1	1	0	$A\oplus B$	$A-B-1$	$A-B$
0	1	1	1	$A\overline{B}$	$A\overline{B}-1$	$A\overline{B}$
1	0	0	0	$\overline{A}\|B$	$A+AB$	$A+AB+1$
1	0	0	1	$\overline{A\oplus B}$	$A+B$	$A+B+1$
1	0	1	0	B	$(A\|\overline{B})+AB$	$(A\|\overline{B})+AB+1$
1	0	1	1	AB	$AB-1$	AB
1	1	0	0	1	$2*A$	$2*A+1$
1	1	0	1	$A\|\overline{B}$	$(A\|B)+A$	$(A\|B)+A+1$
1	1	1	0	$A\|B$	$(A\|\overline{B})+A$	$(A\|\overline{B})+A+1$
1	1	1	1	A	$A-1$	A

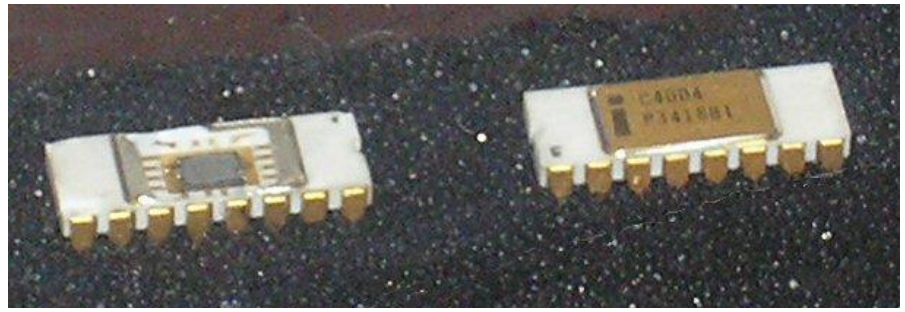
A mikroprocesszorok története

- Utasítás:
  elemi művelet, amit bemenetként az integrált áramkör megkap
- Micro-op:
  a legtöbb utasítást ezek után még szét kell szedni, ezek a micro-op-ok.
  Pl.: címzés, indirekt címzés, több argumentum stb.
- Cím:
  azonosító, ami alapján a memóriában tárolt értékek közül
  kiválaszthatunk egyet.
- Indirekt címzés:
  a cím nem a kódban egy fix szám,
  hanem egy regiszterből olvasandó érték.

A mikroprocesszorok története

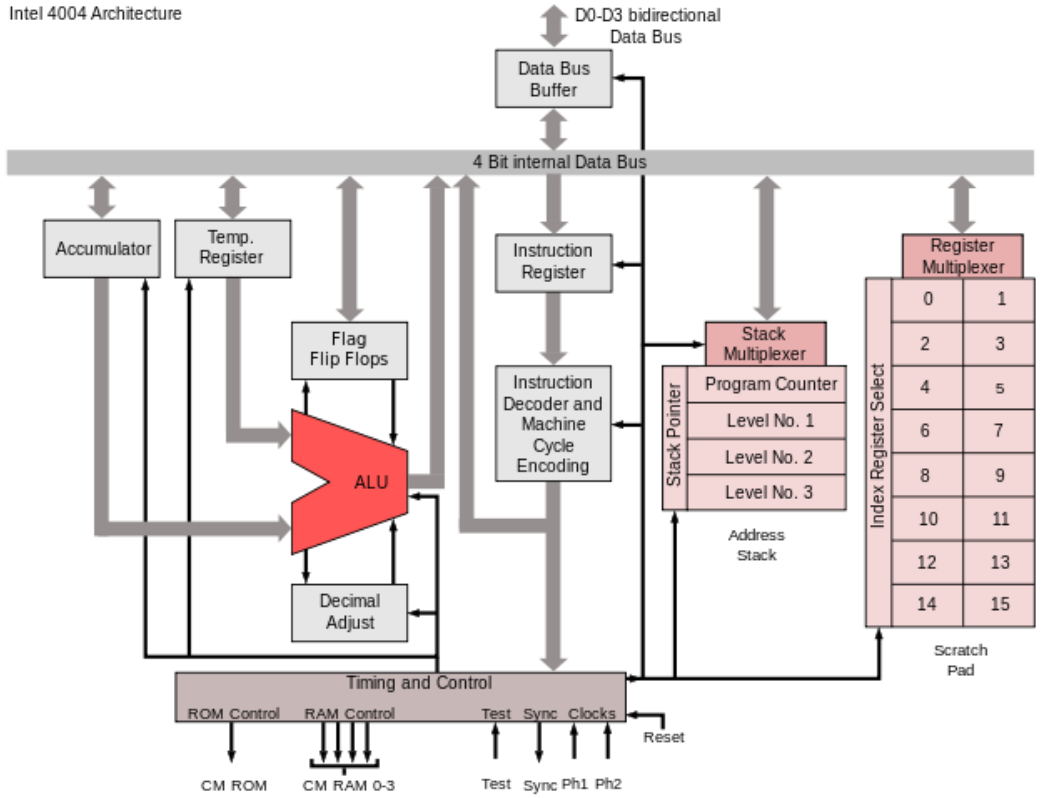
- Az egyik első mikroprocesszor:
  Intel 4004 (60 $, 1971) kb. 2300 tranzisztor

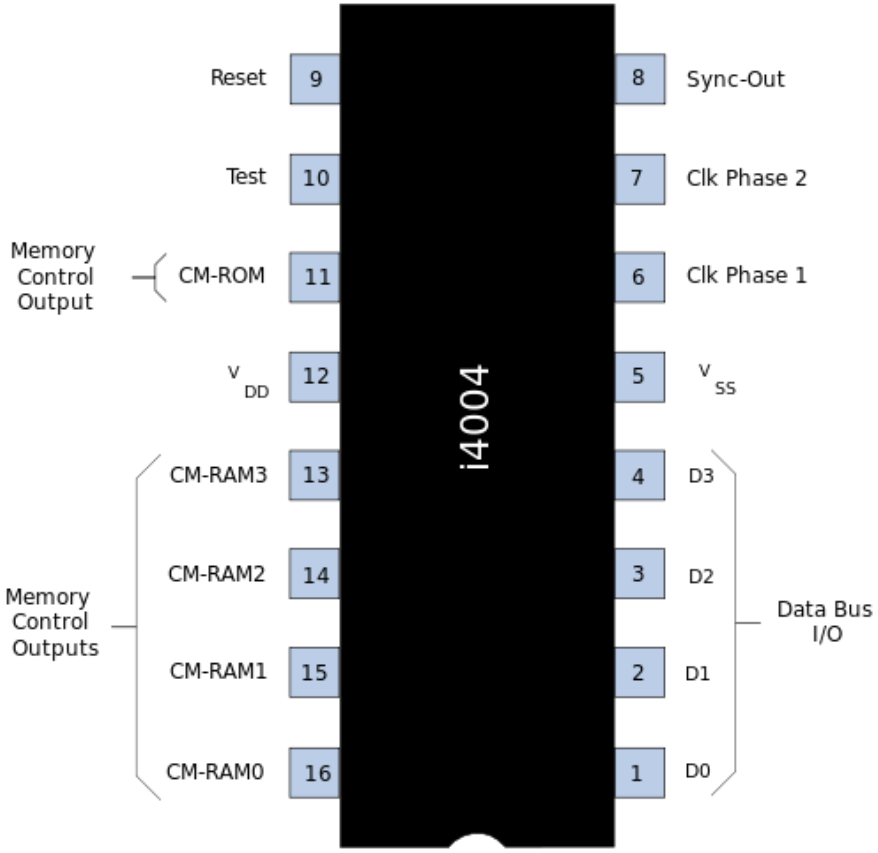
- 4/8 bit-es működés
- Csíkszélesség 10 μm
- Órajel: 740 kHz
- 50 000-100 000 utasítás / sec
- 640 byte-ot tudott megcímezni
- Nagyon egyszerű utasításkészlet



A mikroprocesszorok története

- Az egyik első mikroprocesszor: Intel 4004 (60 $, 1971)





A mikroprocesszorok története

- Programcounter / Instruction pointer:
  Speciális regiszter,
  ami a programkódban jelöli az aktuális végrehajtási helyet
- Stack (verem):
  Speciális memória,
  amit a függvényhívások argumentumainak és a visszatérési cím tárolására használnak
- Flag-ek:
  Speciális állapotokat tároló bitek,
  amik implicit bemenetei egyes utasításoknak.
  Pl: Eq flag, compare, feltételes ugrás

A mikroprocesszorok története

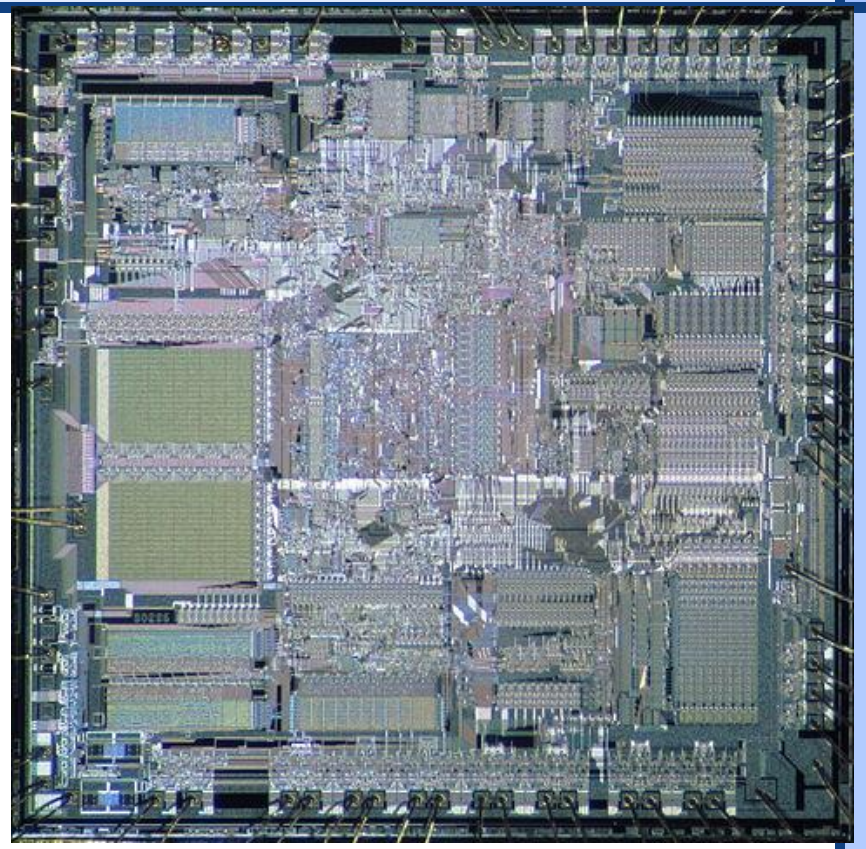
- Utasítás készlet

Intel 4004 Instructions Set

INSTRUCTION	MNEMONIC	BINARY EQUIVALENT 1st byte	2nd byte	MODIFIERS
No Operation	NOP	00000000	-	none
Jump Conditional	JCN	0001CCCC	AAAAAAAA	condition, address
Fetch Immediate	FIM	0010RRR0	DDDDDDDD	register pair, data
Send Register Control	SRC	0010RRR1	-	register pair
Fetch Indirect	FIN	0011RRR0	-	register pair
Jump Indirect	JIN	0011RRR1	-	register pair
Jump Uncoditional	JUN	0100AAAA	AAAAAAAA	address
Jump to Subroutine	JMS	0101AAAA	AAAAAAAA	address
Increment	INC	0110RRRR	-	register
Increment and Skip	ISZ	0111RRRR	AAAAAAAA	register, address
Add	ADD	1000RRRR	-	register
Subtract	SUB	1001RRRR	-	register
Load	LD	1010RRRR	-	register
Exchange	XCH	1011RRRR	-	register
Branch Back and Load	BBL	1100DDDD	-	data
Load Immediate	LDM	1101DDDD	-	data
Write Main Memory	WRM	11100000	-	none
Write RAM Port	WMP	11100001	-	none
Write ROM Port	WRR	11100010	-	none
Write Status Char 0	WR0	11100100	-	none
Write Status Char 1	WR1	11100101	-	none
Write Status Char 2	WR2	11100110	-	none
Write Status Char 3	WR3	11100111	-	none
Subtract Main Memory	SBM	11101000	-	none
Read Main Memory	RDM	11101001	-	none
Read ROM Port	RDR	11101010	-	none
Add Main Memory	ADM	11101011	-	none
Read Status Char 0	RD0	11101100	-	none
Read Status Char 1	RD1	11101101	-	none
Read Status Char 2	RD2	11101110	-	none
Read Status Char 3	RD3	11101111	-	none
Clear Both	CLB	11110000	-	none
Clear Carry	CLC	11110001	-	none
Increment Accumulator	IAC	11110010	-	none
Complement Carry	CMC	11110011	-	none
Complement	CMA	11110100	-	none
Rotate Left	RAL	11110101	-	none
Rotate Right	RAR	11110110	-	none
Transfer Carry and Clear	TCC	11110111	-	none
Decrement Accumulator	DAC	11111000	-	none
Transfer Carry Subtract	TCS	11111001	-	none
Set Carry	STC	11111010	-	none
Decimal Adjust Accumulator	DAA	11111011	-	none
Keybord Process	KBP	11111100	-	none

A mikroprocesszorok története

- Újabb mikroprocesszorok:
  1972: 8008 – igazi 8-bites működés
  1974: 4040 – Interruptok
  1974: 8080 – első x86-os processzor, 2MHz, 6 μm, 6000 tranzisztor, 8/16 bites működés, 3-4 $
  1976: 8086 – 10MHz, 3 μm, függvényhívás (push, ret), prefetch, lebegőpontos co-processzor 8087, utasítások itt.
  1982: 80186 – 25 MHz, de az utasítások is hatékonyabbak
  1982: 80286 – 25 MHz, jobb cím dekódolás, protected mode, multi tasking



A mikroprocesszorok története

- Interrupt:
  Speciális, idő-kritikus kezelésű külső jel hatására az aktuális program félbeszakítása és egy másik rövid program lefuttatása a jel fogadására, majd visszatérés
- Szegmentált memória:
  Darabokban címezhető csak a memória, egy regiszter párossal, esetenként implicit műveletekkel
- Lineáris memória:
  Folytonos címzés egyetlen számként
- Prefetch:
  Amikor a processzor gyorsabb, mint a memória, előre célszerű betölteni az utasításokat, adatokat
- Cache:
  A fő memóriából betöltött kisebb adat, vagy utasítások, amiket a processzor gyorsabban elér

A mikroprocesszorok története

- Real mode:
  A program, ami fut, mindenhez hozzáfér: memória, I/O, perifériák, stb.

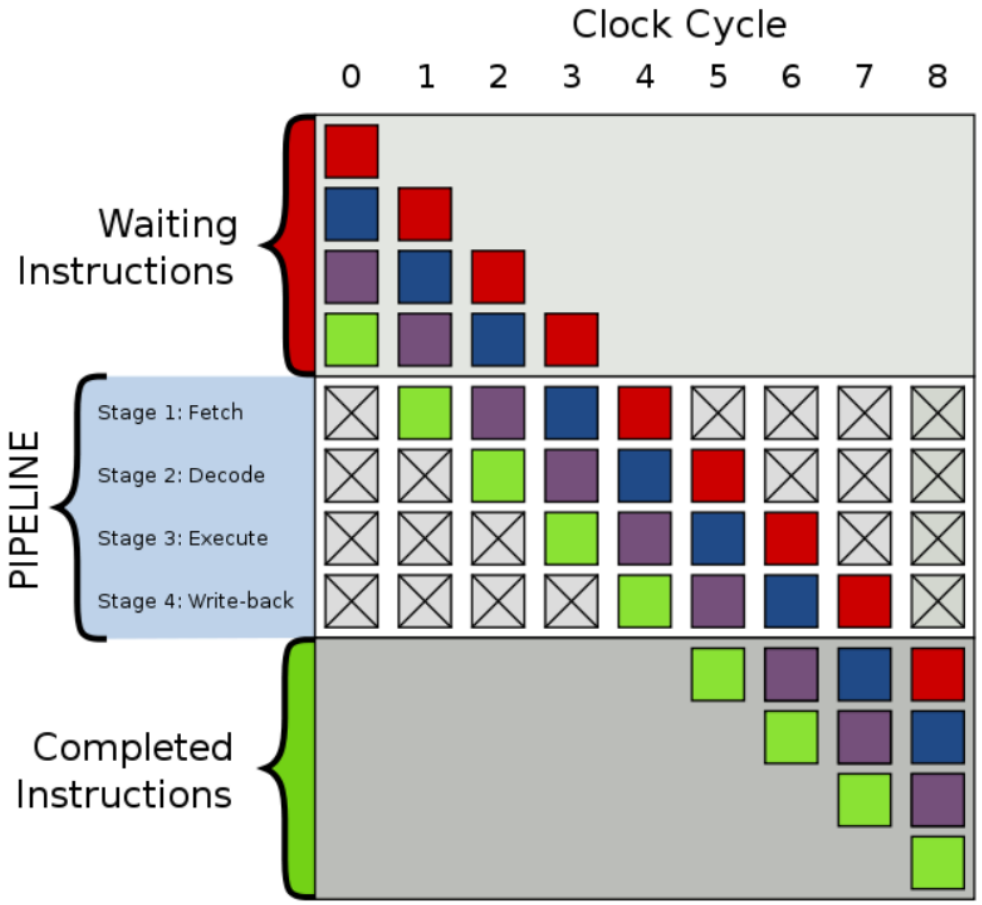
- Protected mode:
  Védelmi funkciók (ringek), privilegizált utasítások (I/O), direkt hw hozzáférés korlátozása, data exec, BIOS védelem,
  Multi task támogatás, stb…

A mikroprocesszorok története

- Pipeline: Az utasítások végrehajtásának „futószalagja”

  Latency (késlekedés):
  Mennyi ideig tart, amíg átér az utasítás a pipeline-on

  Throughput (Áteresztés):
  Mennyi utasítást / lépést tud egyszerre elvégezni a futószalag



A mikroprocesszorok története

1986: 80386 – 40 MHz 1 μm, 32 bit, többféle szám típus, lapozás, 8086 virtualizáció, lineáris memória modell

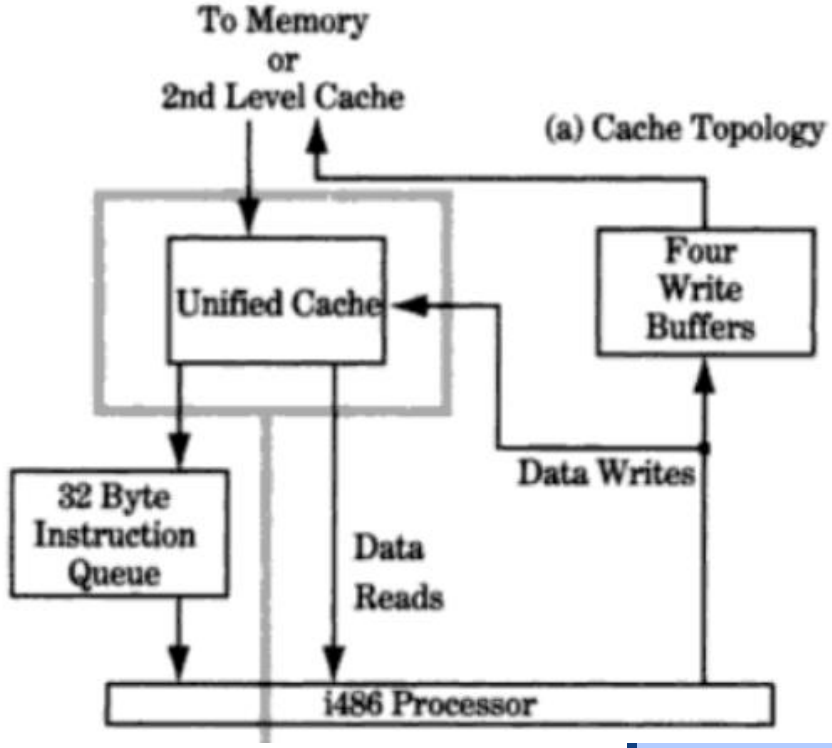
1989: 80486 – 150 MHz 0,6 μm, integrált FPU, egyesített caching, első atomic utasítás (CMPXCHG)

A mikroprocesszorok története

- Virtualizáció:
  A futó programok úgy látják, mintha másik hardveren, vagy szoftver (OS)-en futnának

- Atomic utasítás: (bővebben még később)
  Olyan elemi utasítás, ami garantált, hogy nem szakítható félbe másik szál által. Ezeken keresztül valósíthatóak meg egyes szinkronizációs, hozzáférés vezérlési feladatok szálak között.

- Egyesített cache:
  Mind az utasításokat, mind az adatokat előre be kell olvasni, és célszerű tárolni, minél közelebb a processzorhoz.



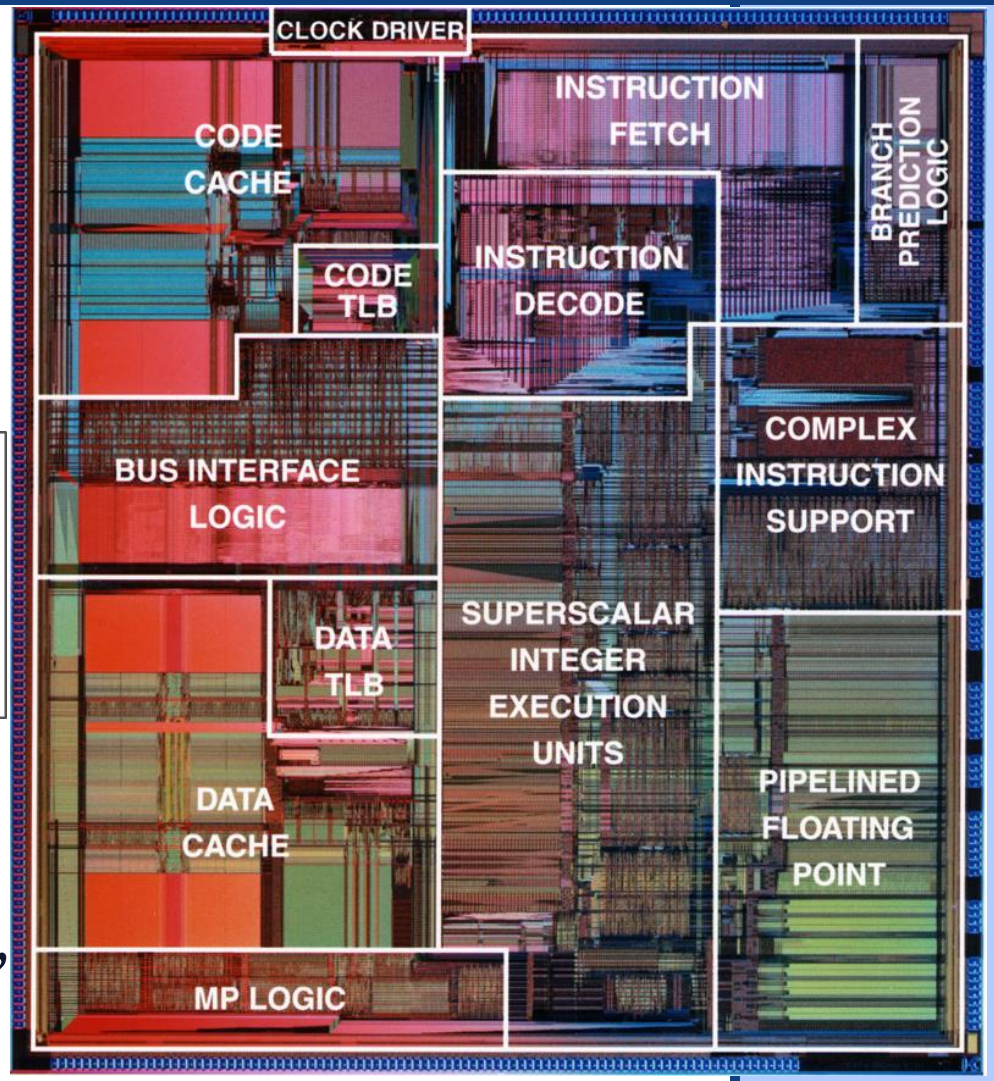
A mikroprocesszorok története

1993: Pentium P5 – Super scalar architektúra, Branch prediction, számos utasítás sokkal gyorsabb, nagyobb busz sávszélesség, MMX
UV pipeline: egyes utasításokból kettő is végrehajtható egyszerre

1995: Pentium P6 – speculative exec., out-of-order exec., Physical Address Extension

- Pentium Pro: conditional move
- Pentium III: SSE

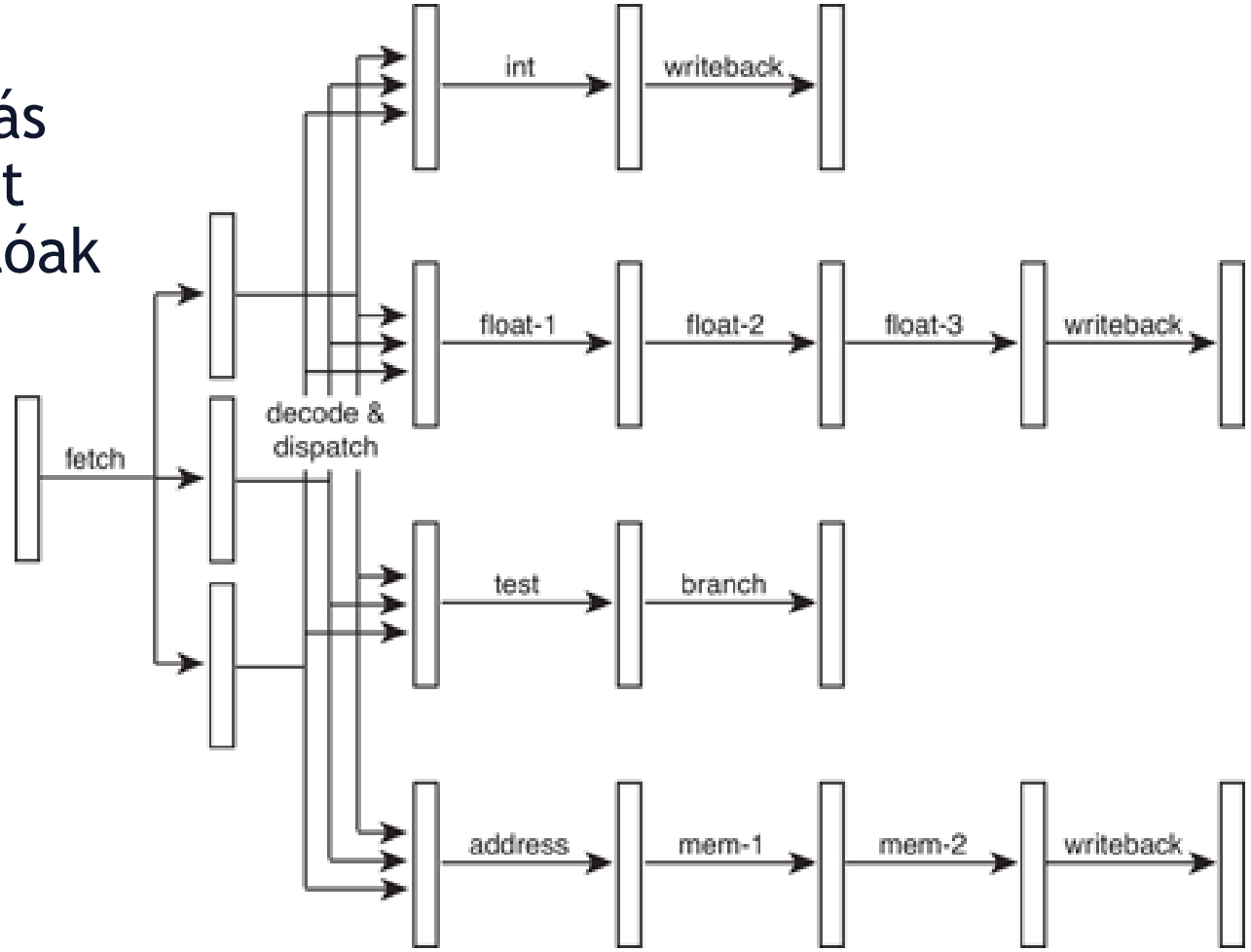
2000: NetBurst – Pentium 4, 20 (!) elemű pipeline, exec trace cache, hyper-threading, több mag, SSE2, SSE3, SSSE3 (FMA), hardveres virtualizáció, 64 bit



A mikroprocesszorok története

- Superscalar architektúra:
  Különböző típusú műveletek más feldolgozókra kerülnek, de pont ezért egyszerre is végrehajthatóak

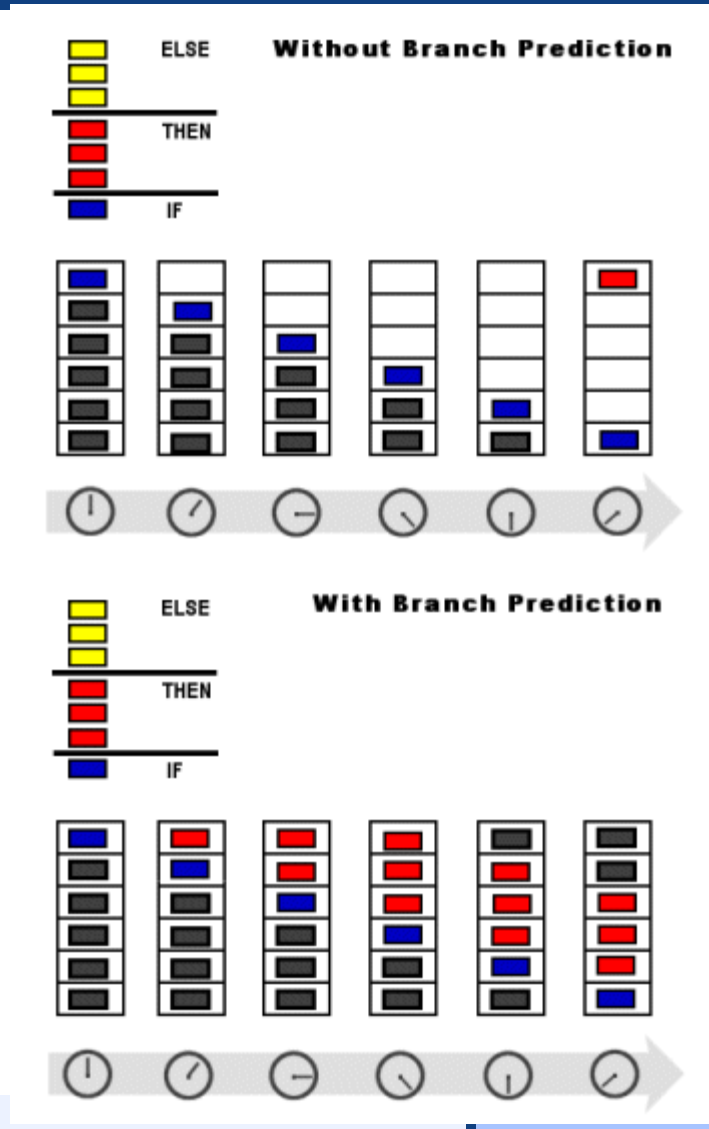
  utasítás szintű párhuzamosság



A mikroprocesszorok története

- Branch prediction:
  Mit töltsön be a fetch lépésben a processzor, ha elágazás van a programban?
  A Branch predictor megpróbálja kitalálni, hogy melyik ág jöjjön, ha téved, akkor rossz esetben az egész pipeline-t üríteni kell!

- Speculative execution:
  Inkább előre elvégeznek több műveletet,ami esetleg nem kell, csak hogy kevesebb gond legyen az elágazásokkal



A mikroprocesszorok története

- Out-of-order execution:
  Az éppen elérhető adatok alapján végrehajtható utasításokat dolgozza fel, esetlegesen eltérve az eredeti utasítás sorrendtől,
  ha ez nem okoz változást a végeredményben
- Hyper-Threading:
  Minden fizikai processzor két logikai processzorként látszik és fogad utasításokat, így a SuperScalar feldolgozók többféle munkát kapnak, és összességében gyorsabbak tudnak lenni. Sok előny és hátrány…
- Translation look aside buffer:
  A növekvő virtualizációk miatt egyre többször kell a címeket transzformálni.
  A Cache miss-ek egyik oka, ha az adat a cache-ben van, de a cím nincs!
  A TLB ezt a problémát segít megoldani: cacheli a dekódolt címeket
  egy hash alapú adatbázisban

A mikroprocesszorok története

- 2006: Intel Core – kisebb fogyasztás, macro-op fusion, SSE4
- 2007: Penryn – SSE4.1
- 2008: Nehalem – SSE4.2, integrált PCI-E, DMI vezérlők, két szintű branch prediction
- 2010: Westmere – AES titkosítás
- 2011: Sandy Bridge – AVX, hw video dekódolás, közös L3 cache az integrált grafikusprocesszorral,
  2011: Ivy Bridge – tri-gate tranzisztorok, Trusted Execution Technology, hardveres véletlen szám generátor

A mikroprocesszorok története

- 2013: Haswell – AVX2, FMA3,
  Transactional Synchronization Extensions
- 2014: Broadwell – ADX (arbitrary precision integer arithm.)
- 2015: Skylake – AVX-512, SHA titkosítás, memória védelem
  egyre több video formátum és titkosítás gyorsítása, utasítások és memória hozzáférések optimalizálása

Pipeline hosszak

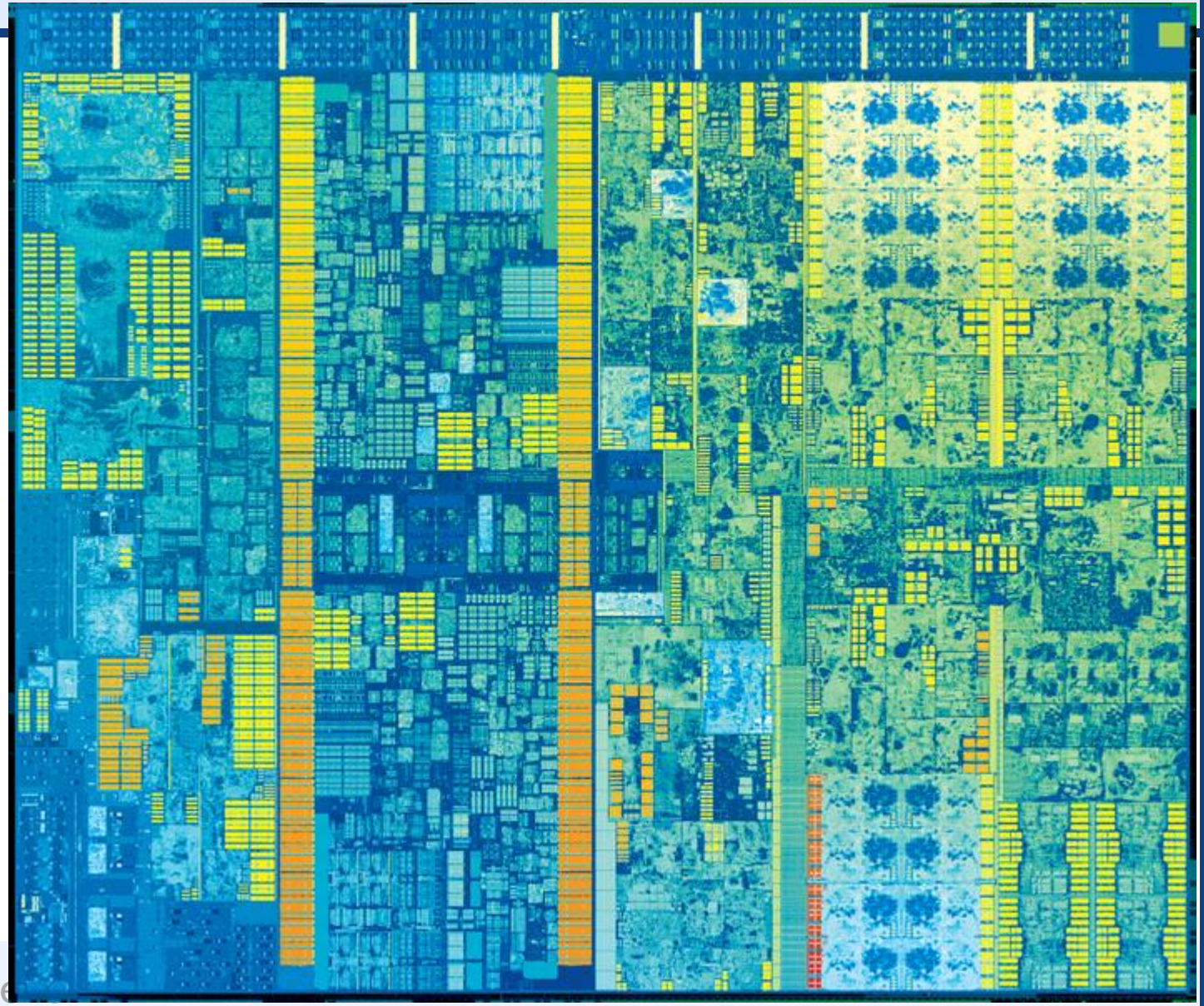
A mikroprocesszorok története

- 2017: Kaby Lake, Coffee Lake – 14 nm frissítés,
  utóbbinál már 6 magos processzorok

- 2018-19: Cannon Lake, Ice Lake – 10 nm,
  AVX kiterjesztés: FMA egész értékeken is

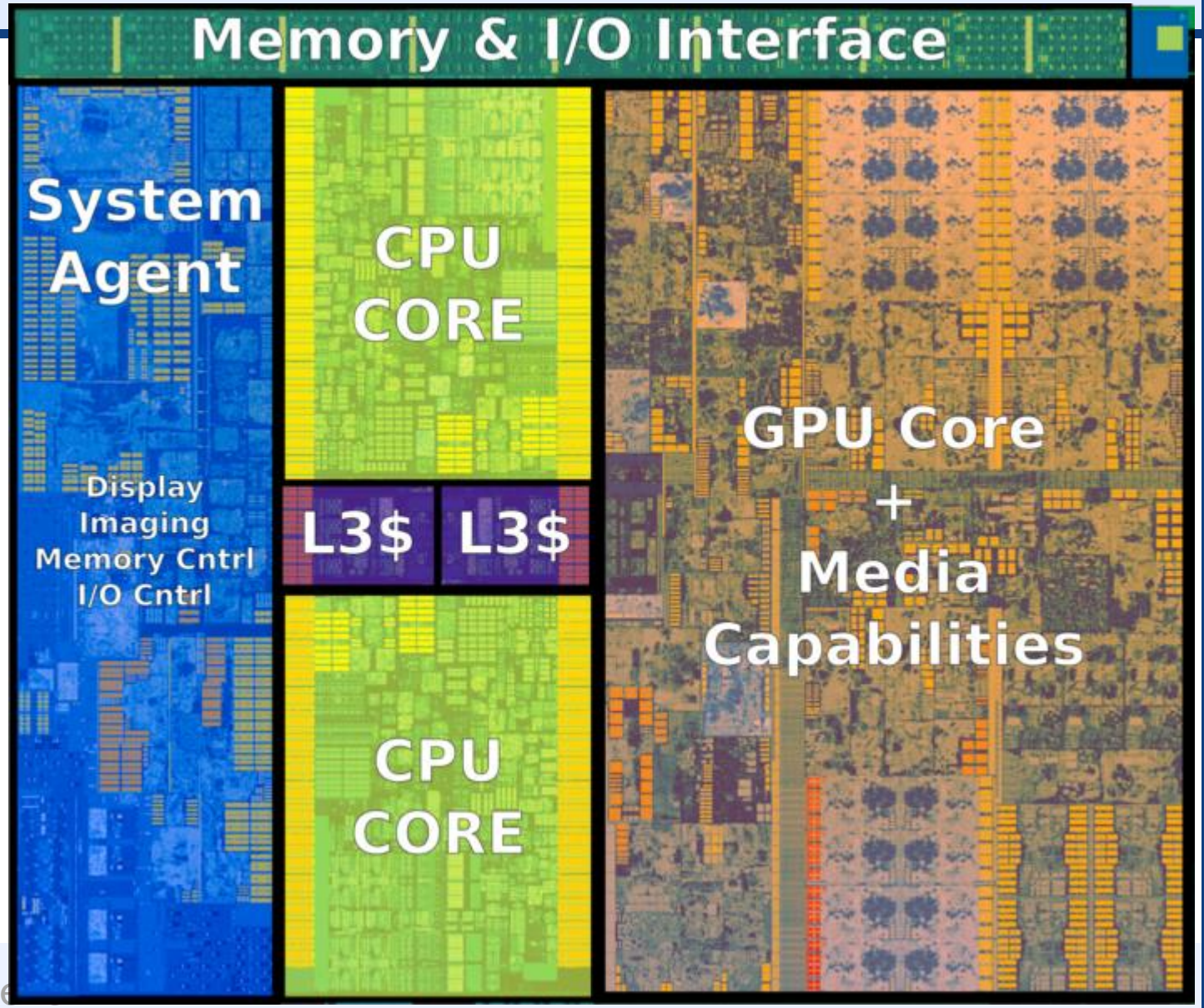
A mikroprocesszorok története

Intel Kaby Lake processzor szerkezete



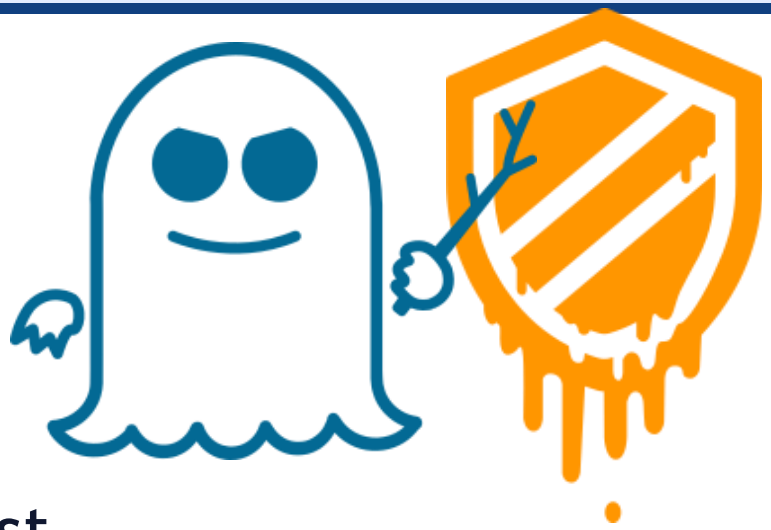
A mikroprocesszorok története

Intel Kaby Lake processzor [szerkezete]



Spekulatív végrehajtási sebezhetőségek

- 2018 elején lett publikus, hogy több igen komoly sebezhetőséget találtak a processzorokban



- A spectre és meltdown sebezhetőségek a spekulatív végrehajtás megkerülésével hardveresen védett adatokhoz ad hozzáférést felhasználói programok számára

- Mivel ezek hardver tervezés szintű problémák, csak szoftverből, költséges memória indirekciókkal lehet védekezni ellenük

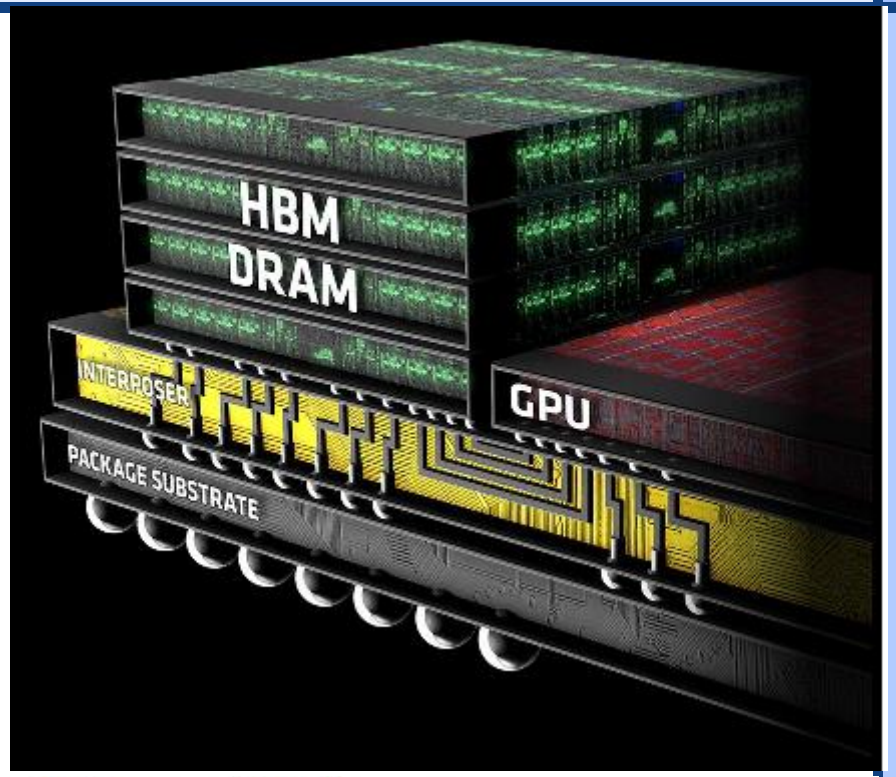
High Bandwidth Memory

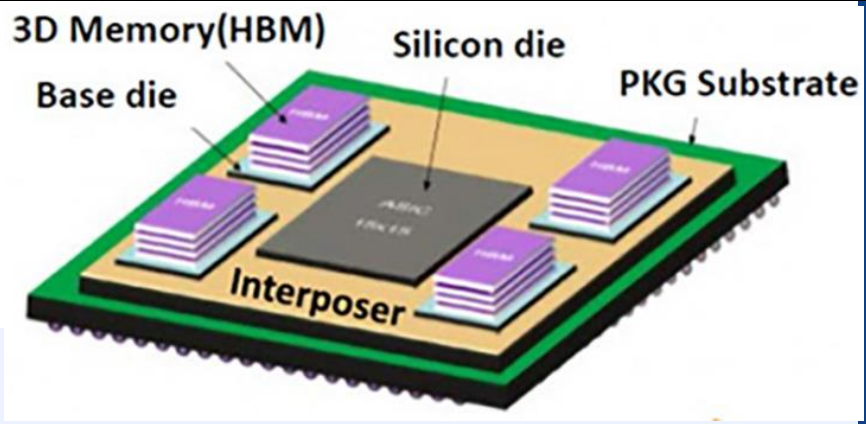
Régen a tranzisztor volt drága, és a vezetékek szinte ingyen voltak

Ma a tranzisztorok kialakítása olcsó, de a vezetékezés egyre költségesebb, mert egyre több helyet foglal

Megoldás: építkezés 3D-ben!

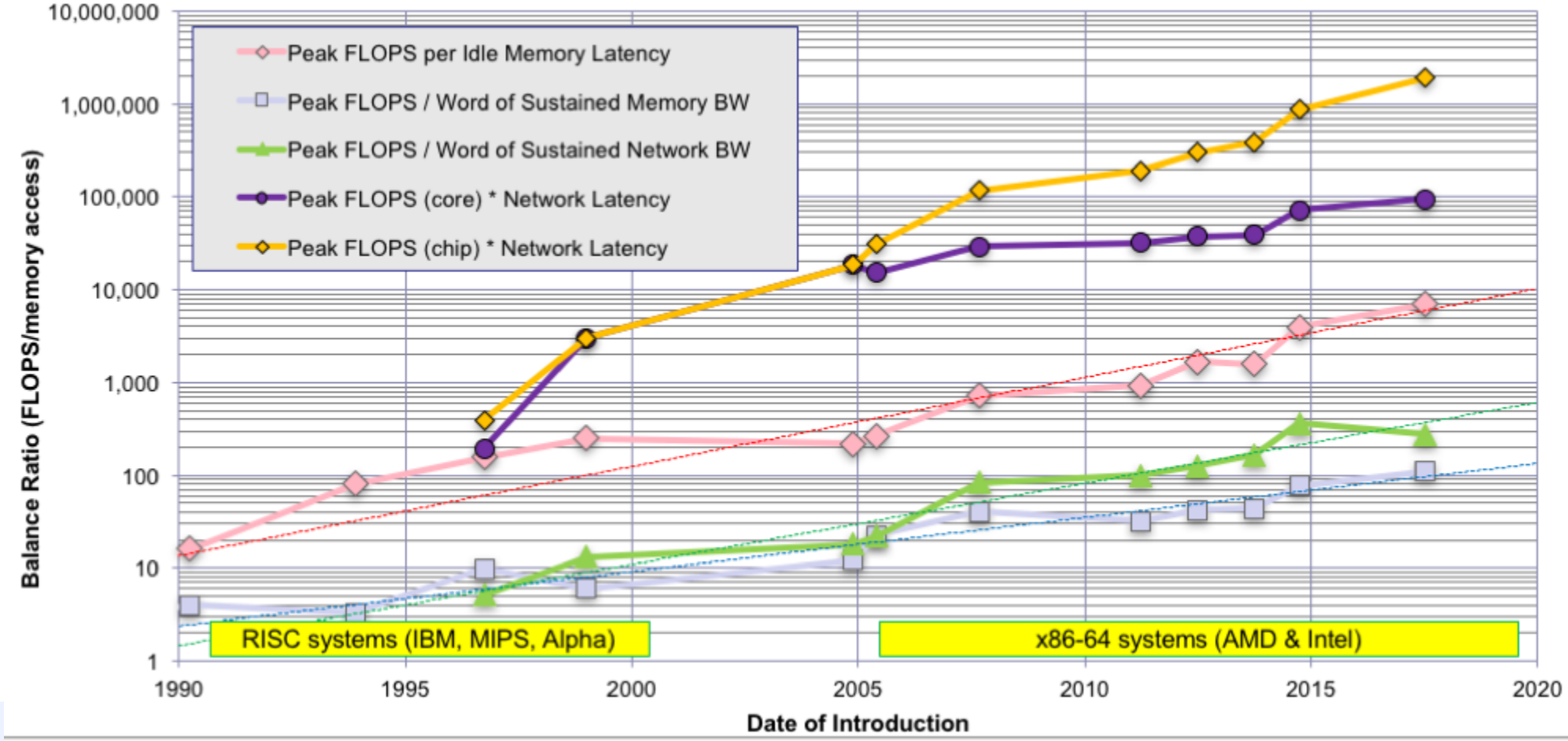
[HBM – High Bandwidth Memory]





Trendek

Az adat elérése (memória, hálózat) egyre lassabb ahhoz képest, hogy a műveleteket milyen gyorsan tudja a processzor végrehajtani



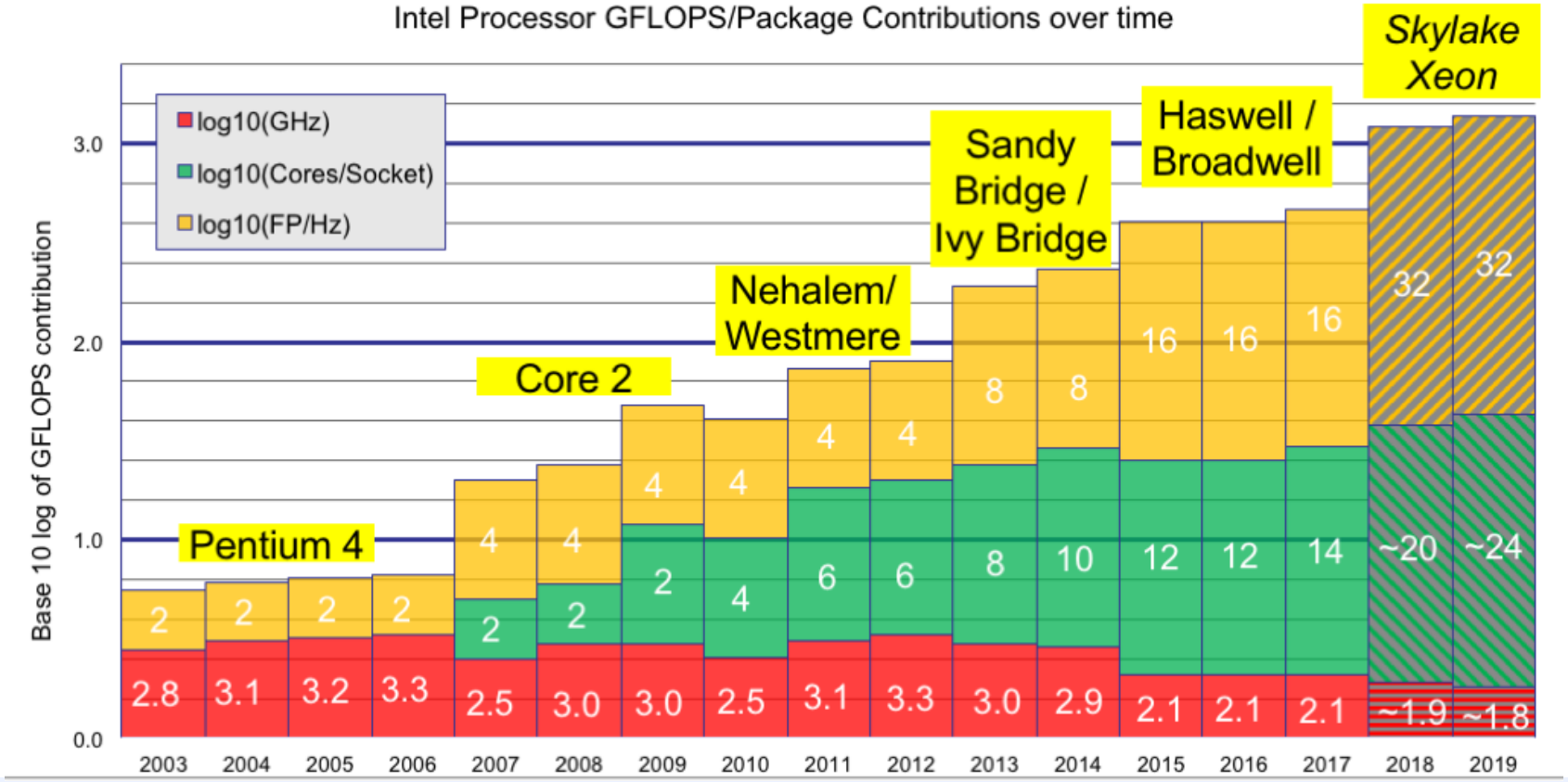
Ezek mind arányok, ideális esetben 1 körül kéne lenniük

Forrás: U. Texas

Trendek

Miért fontosak a vektor utasítások a hardverben?

Mert ezek adják a teljesítmény legnagyobb részét!



Forrás:
U. Texas

Miért drágák a komplex elektronikák?

- A félvezető gyártás erősen statisztikus folyamat, nehéz az egyre kisebb méretek mellett a kémiai kezelést egyenletesen és homogénen megoldani, sokszor fordulnak elő pont hibák, vagy illesztési hibák

- Ezért sok legyártott processzor teljesen hibás, vagy egyes alkomponensek hibásak, különösen egy új gyártási eljárás (pl. csíkszélesség) bevezetésekor.

- A kihozatali arány (jól működő / összes), akár 30%-ra is csökkenhet. A nem tökéletes, de működő darabokat árulják általában az olcsóbb árkategóriákban.

A mikroprocesszorok története

Összefoglalás:

- A modern CPU-k igen bonyolult futószalagon próbálják meg végrehajtani az általunk írt programokat.
- Megpróbálják kitalálni az elágazásokat, hogy milyen adat kell legközelebb, mit lehet egyszerre végrehajtani, stb.
- Mindez igen komplex tervezési kihívásokhoz vezet, de egy operációs rendszer és sok program egyszerre futtatásához ez kell!
- Mindezeket visszafelé kompatibilisen több évtizedre visszamenően…