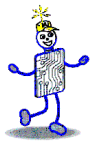


7. fejezet:
A CPU és a memória

The Architecture of Computer Hardware, Systems Software & Networking:
An Information Technology Approach

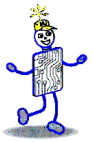
4th Edition, Irv Englander

John Wiley and Sons ©2010

Fordította és kiegészítette: Végh János

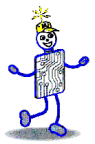
PowerPoint slides authored by Wilson Wong, Bentley University

PowerPoint slides for the 3rd edition were co-authored with Lynne Senne, Bentley College

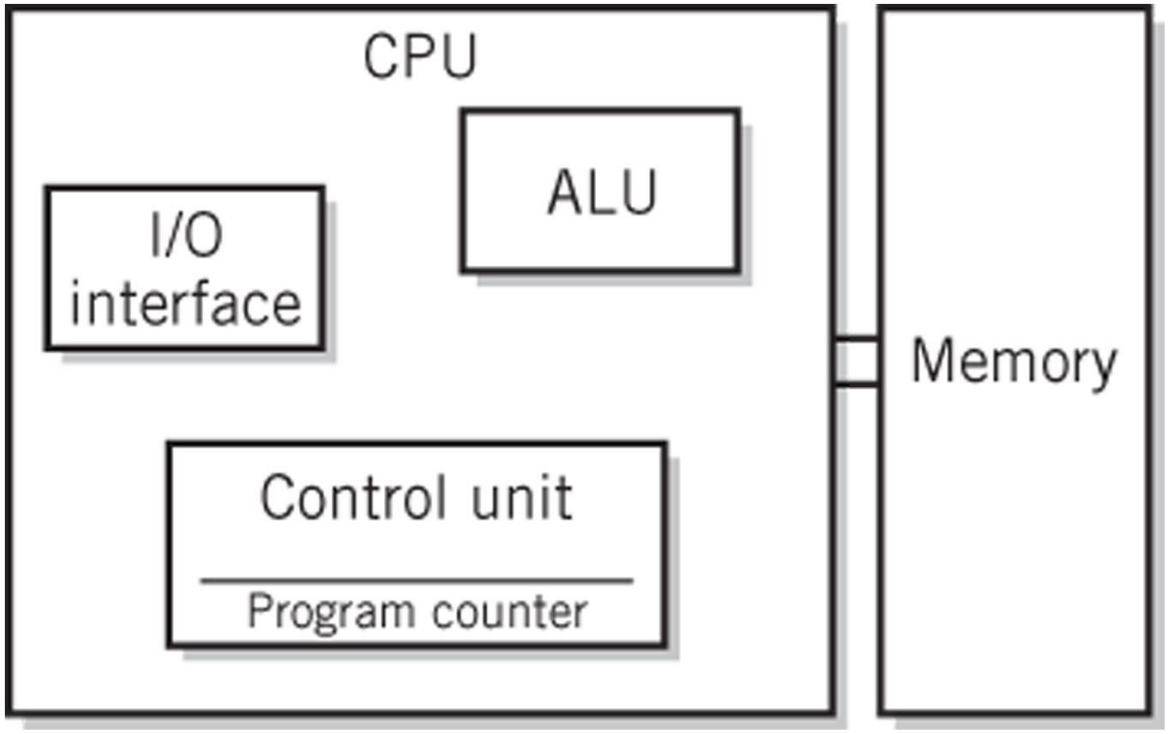


CPU: A főbb alkotórészek

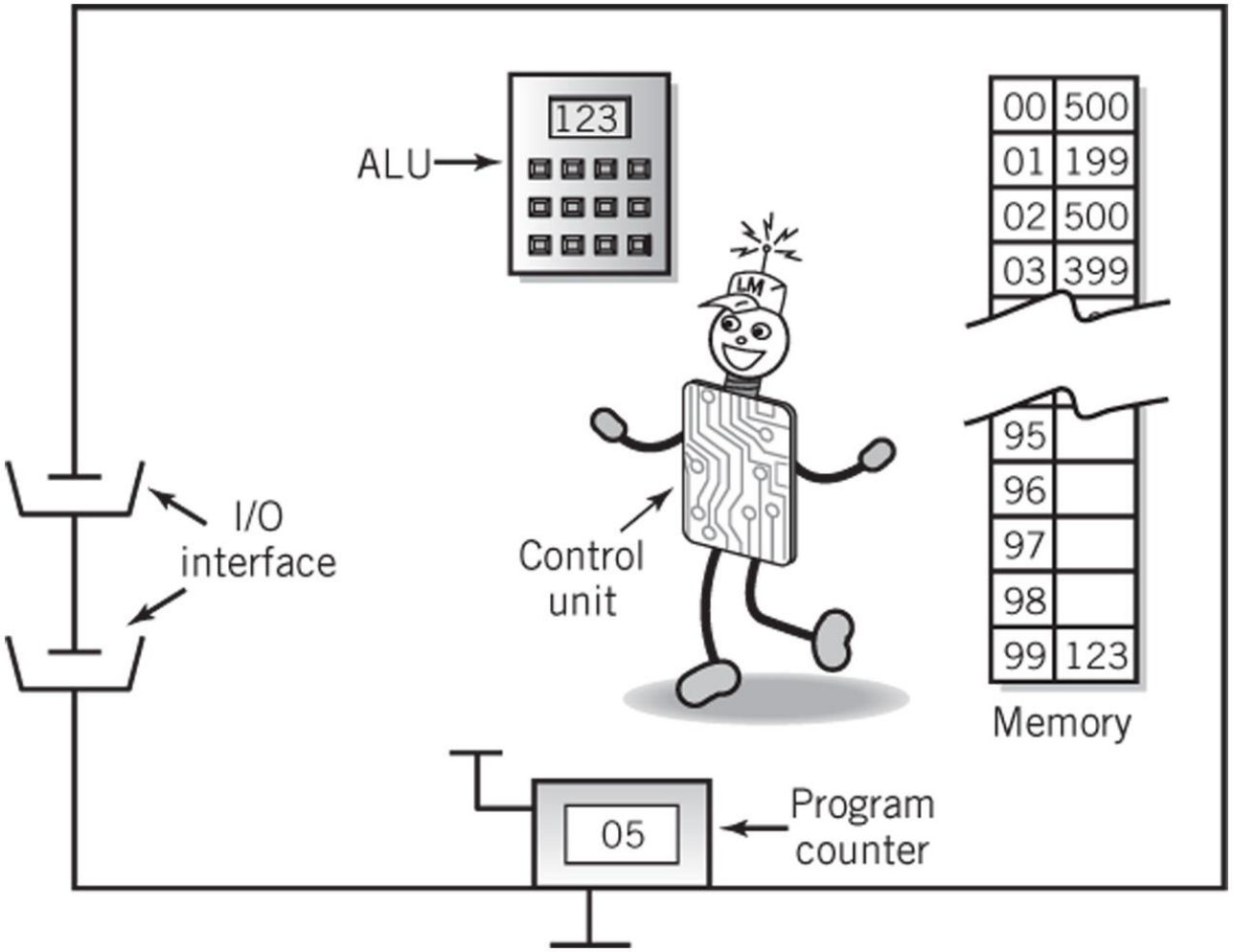
- *ALU* (arithmetic logic unit)
  - Számítási műveleteket és összehasonlításokat végez
- *CU* (vezérlő egység)
  - Utasítás elővételi/végrehajtási ciklus
    - Előveszi a program utasításokat és utasításokat küld az ALU-nak
    - Adatokat mozgat a CPU regiszterek és más hardver komponensek között
  - AI-komponensek:
    - *Memory management unit:* felügyeli az adatok és utasítások elővételét a memóriából
    - *I/O Interface:* néha összevonják az előbbivel, *Bus Interface Unit* néven

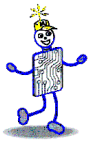


A rendszer blokkdiagramja



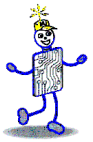
A „kis ember számítógép”





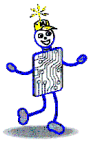
A regiszter fogalma

- Speciális célú, kicsi, állandó tárterület a CPUn belül
- Közvetlenül a vezérlőegység kezeli
- Huzalozott, speciális funkciót lát el
- Bitben vagy bájtban megadott méretű (nem MB, mint a memória)
- Adatot, címet vagy utasítást tárolhat
- Hány regiszter van az LMC-ben?
- Mik a regiszterek az LMC-ben?



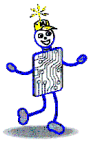
Regiszterek

- A regiszterek használata
  - „Scratchpad”, az éppen végrehajtás alatt levő program számára
    - Gyakran használt vagy gyorsan elérendő adatokat tartalmaz
  - A CPUról és az éppen végrehajtás alatt levő programról tárol információkat
    - A következő programutasítás címe
    - Külső eszközöktől származó jelek
- Általános célú regiszterek
  - *A felhasználó által használható regiszterek*
  - Közbülső adatokat, pl. ciklusszámlálót tárol
  - Az LMC kalkulátorával egyenértékű
  - A mai CPUk tipikusan pár tucat regisztert tartalmaznak



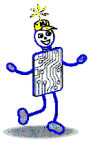
Speciális célú regiszterek

- *Program Count Register* (PC)
  - Utasítás mutató (instruction pointer) a másik neve
- *Instruction Register (IR)*
  - A memóriából elővett utasítást tárolja
- *Memory Address Register (MAR)*
- *Memory Data Register (MDR)*
- *Status Registers*
  - A CPU és az éppen végrehajtódó program állapota
  - *Jelzőbitek (Flags)* (egy bites logikai változók) olyan feltételek nyomon követésére, mint az aritmetikai átvitel és túlcsordulás, tápfeszültség hiba, belső hiba



Regiszter műveletek

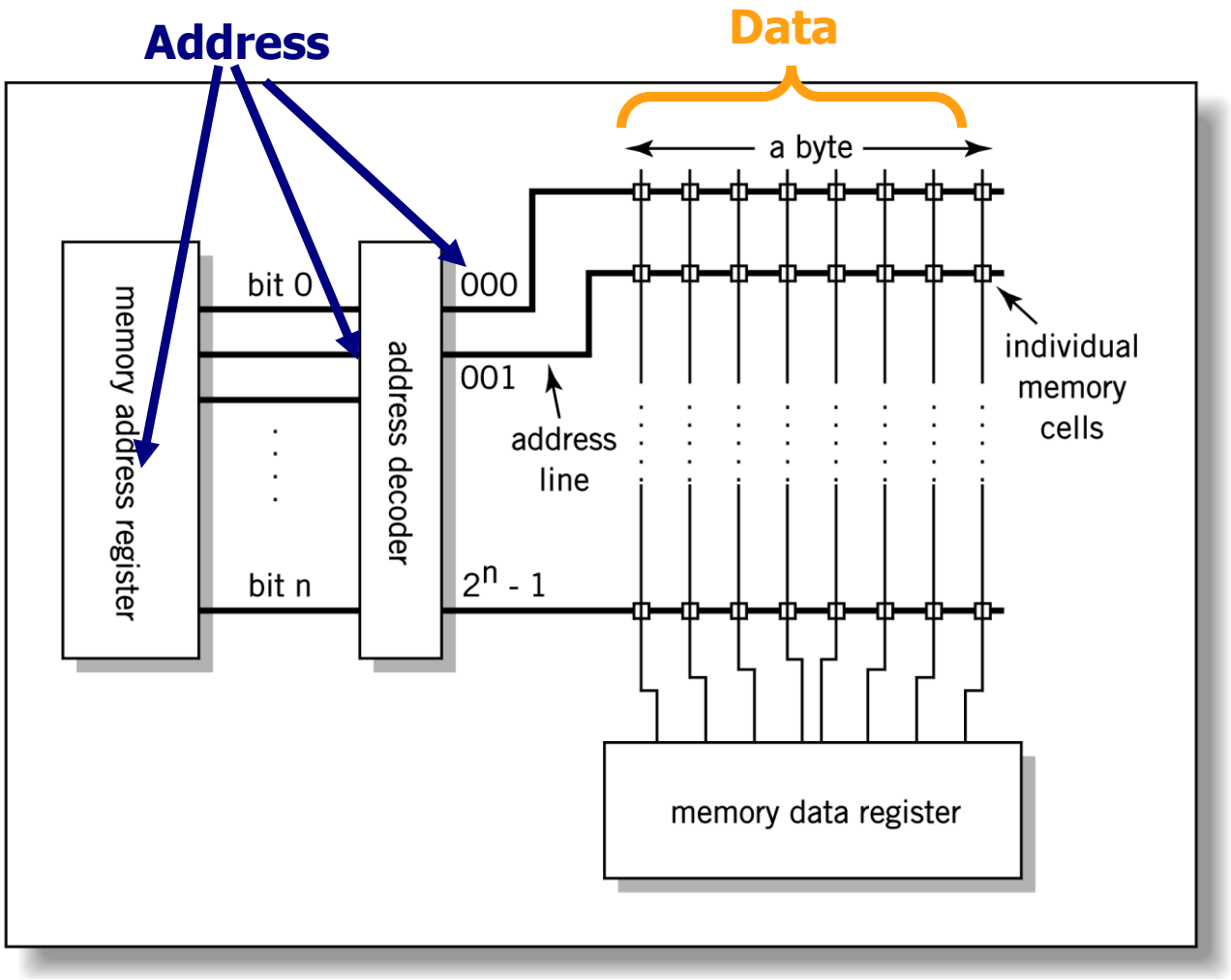
- Más helyről (regiszterekből és memóriából) származó adatok tárolása
- Összeadás és kivonás
- Adatok léptetése és forgatása
- Feltételek vizsgálata, pl. zero vagy pozitív



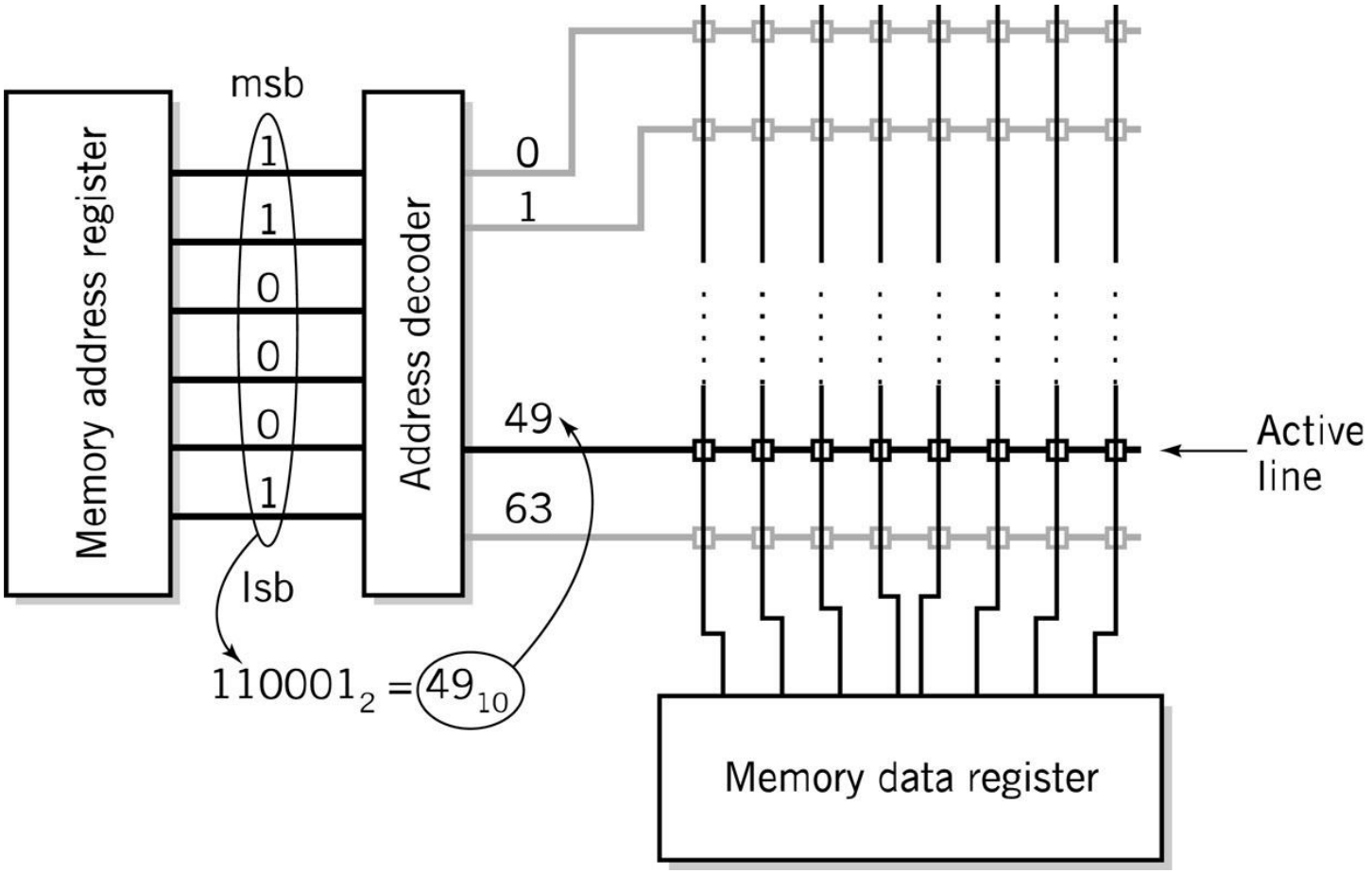
Memória műveletek

- Mindegyik memória helynek egyedi címe van
- Az utasításból a cím a MAR-ba másolódik, ami alapján megtalálható a memória hely
- A CPU eldönti, hogy írás vagy olvasás
- Átvitel a MDR és a memória között
- Az MDR kétirányú regiszter

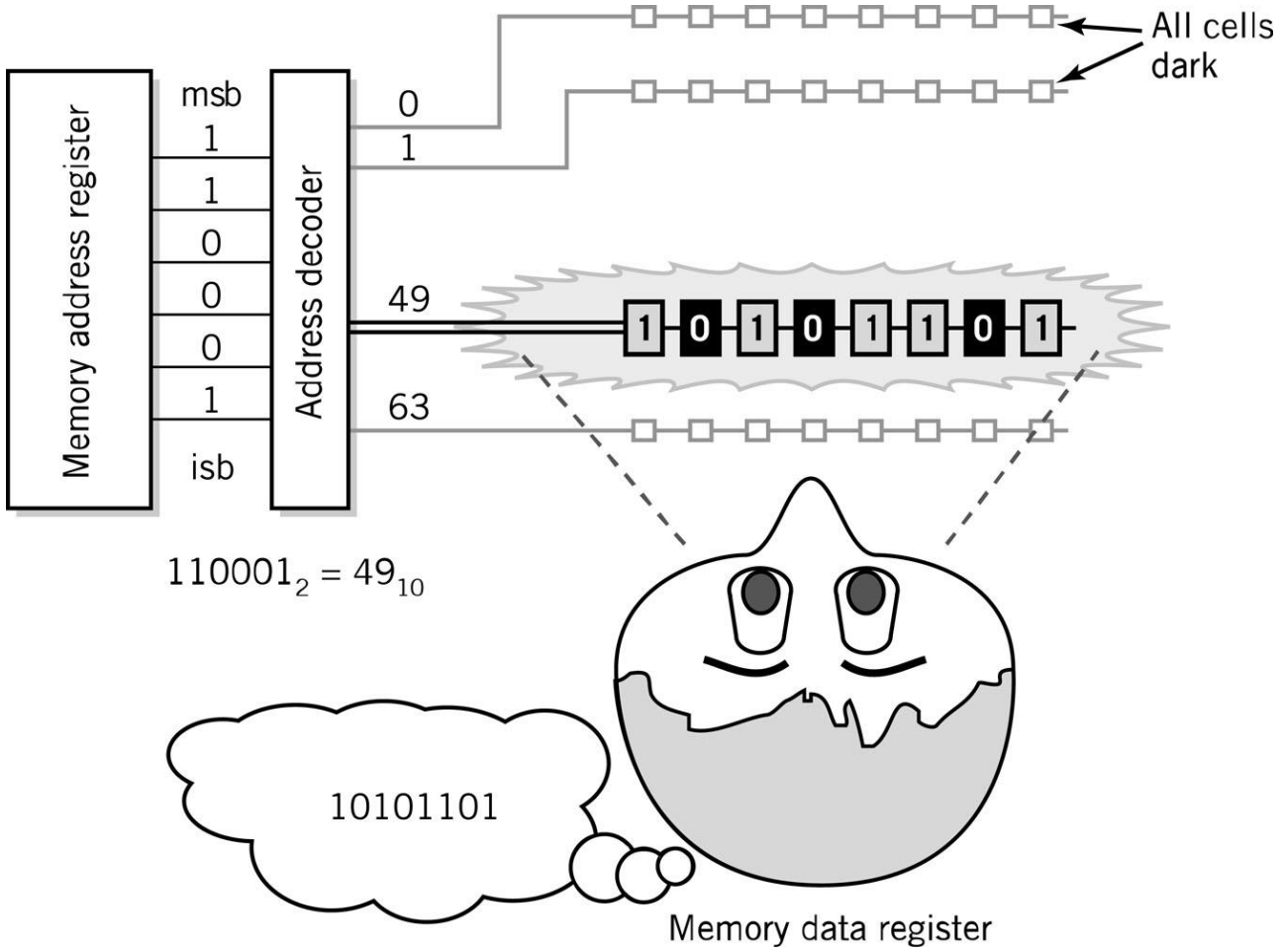
Kapcsolat a MAR, MDR és a memória között



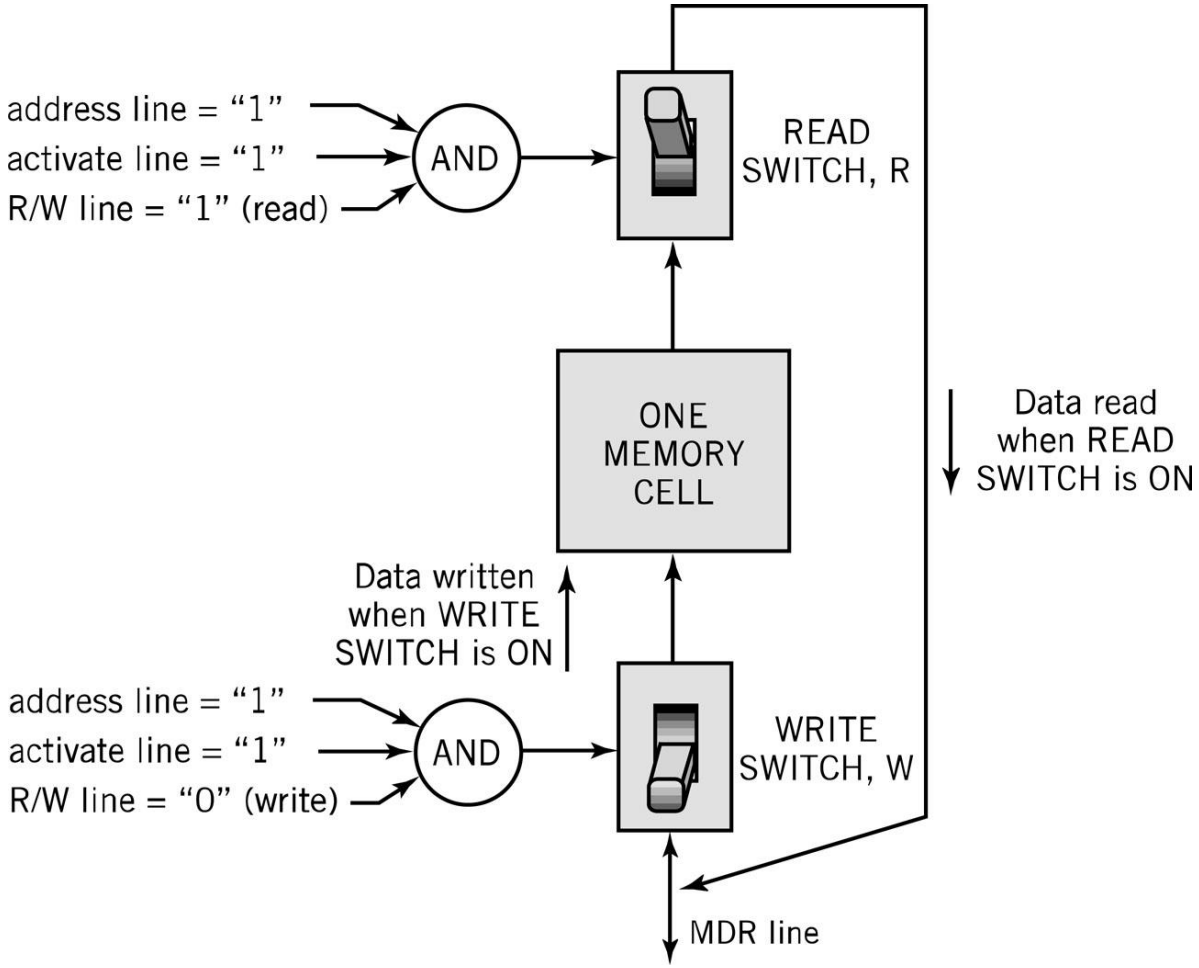
MAR-MDR példa

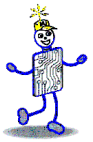


A memória vizuális megfelelője



Egyedi memória cellák

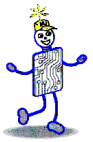




Memória kapacitás

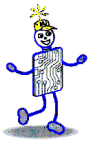
Két tényező határozza meg

1. A MAR bitjeinek száma
   - LMC = 100 (00-tól 99-ig)
   - 2^K ahol K = a regiszter szélessége bitekben
2. Az utasítás címrészének mérete
   - 4 bit 16 helyet tesz lehetővé
   - 8 bit 256 helyet tesz lehetővé
   - 32 bit pedig 4,294,967,296 vagy 4 GB helyet



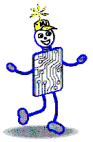
RAM: Random Access Memory

- *DRAM (Dynamic RAM)*
  - Gyakrabban használt, olcsó, kisebb teljesítmény igényű, kisebb hőleadású, kisebb méretű
  - Nem állandó (Volatile): frissíteni kell (újratölteni) kb. msec-onként
- *SRAM (static RAM)*
  - Gyorsabb és drágább, mint a DRAM
  - Nem állandó (Volatile)
  - Kis mennyiségben használatos *cache memory* minőségben, nagy sebességű memória elérésre



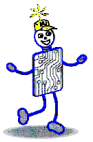
Állandó (Nonvolatile) Memória

- *ROM*
  - Read-only Memory
  - Olyan szoftvert tartalmaz, ami várhatóan nem változik a rendszer életideje alatt
- EEPROM
  - Electrically Erasable Programmable ROM
- *Flash Memory*
  - Gyorsabb, mint a diszk, de drágább
  - Töltés injektálással tárolja az adatbiteket
  - A RAM-hoz képest lassúbb az újraírási ideje
  - Jól használható hordozható állandó tárolóként

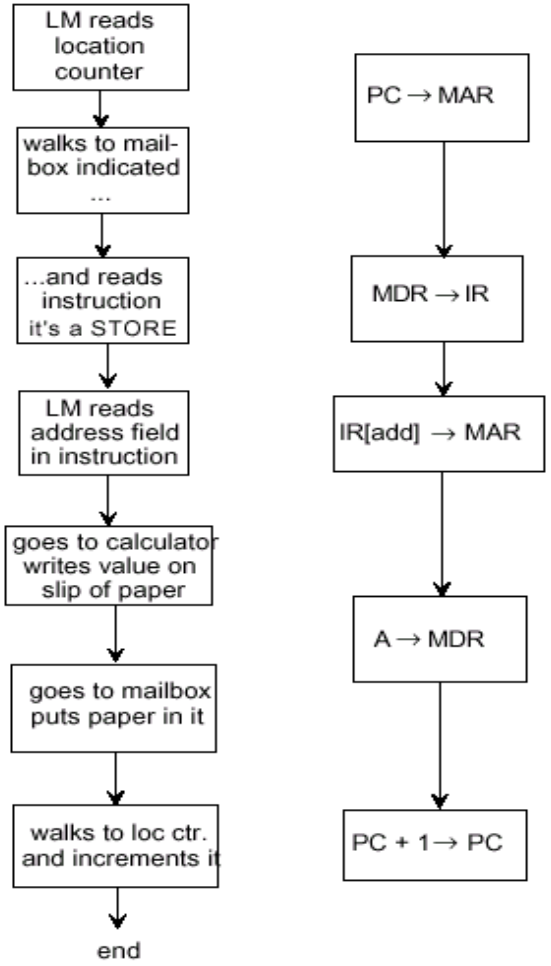


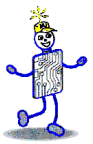
Fetch-Execute Ciklus

- Két ciklusú folyamat, mivel az utasítás és az adat egyaránt a memóriában vannak
- *Fetch* Az utasítás megtalálása és értelmezése, betöltés a memóriából regiszterbe, jelzés az ALU-nak
- *Execute*
  - Végrehajtja az utasítás által szükségesnek tartott műveleteket
  - Mozgatja/átalakítja az adatokat



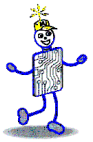
Az LMC és a CPU utasítás elővételi és végrehajtási ciklusa





Az adat elővétel elővételi/végrehajtási ciklusa

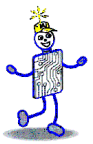
1. PC → MAR	Átviszi az utasítást a PC-ből a MAR-ba
2. MDR → IR	Átírja az utasítást az IR-be
3. IR[address] → MAR	Az utasítás cím részét betölti a MAR-ba
4. MDR → A	Az aktuális adatot beírja az akkumulátorba
5. PC + 1 → PC	Megnöveli a Program Counter-t



Az adattárolás elővételi/végrehajtási ciklusa

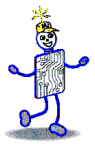
1. PC → MAR	Átírja a címet a PC-ből a MAR-ba
2. MDR → IR	Átírja az utasítást az IR-be
3. IR[address] → MAR	Az utasítás címrészét betölti a MAR-ba
4. A → MDR*	Az akkumulátorból adat másolódik az MDR-be
5. PC + 1 → PC	A Program Counter megnövelődik

*Figyeljük meg, miben tér el LOAD és STORE esetén a #4 lépés



Az ADD elővételi/végrehajtási ciklusa

1. PC → MAR	Átírja a címet a PC-ből a MAR-ba
2. MDR → IR	Átírja az utasítást az IR-be
3. IR[address] → MAR	Az utasítás címrészét betölti a MAR-ba
4. A + MDR → A	Az MDR tartalmát hozzáadja az akkumulátor tartalmához
5. PC + 1 → PC	A Program Counter megnövelődik



LMC Fetch/Execute

SUBTRACT

PC → MAR

MDR → IR

IR[addr] → MAR

A – MDR → A

PC + 1 → PC

IN

PC → MAR

MDR → IR

IOR → A

PC + 1 → PC

OUT

PC → MAR

MDR → IR

A → IOR

PC + 1 → PC

HALT

PC → MAR

MDR → IR

BRANCH

PC → MAR

MDR → IR

IR[addr] → PC

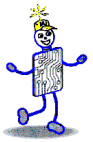
BRANCH on Condition

PC → MAR

MDR → IR

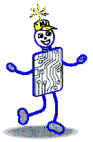
If condition false: PC + 1 → PC

If condition true: IR[addr] → PC



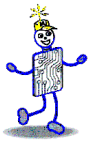
Busz

- Az a fizikai összeköttetés, ami lehetővé teszi a számítógéprendszeren belül adatok eljuttatását egyik helyről a másikra
- Elektromos vagy optikai vezetékek csoportja, amely jeleket szállít egyik helyről a másikra
  - Vezeték vagy áramköri lapra nyomtatott sáv
  - *Line:* a busz egyes vezetékei
- 4 féle jel
  1. Adat (Data)
  2. Címzés (Addressing)
  3. Vezérlő jelek (Control signals)
  4. Tápellátás (Power) (esetleges)



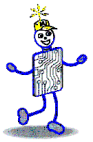
Busz jellemzők

- A különálló vezetékek száma
- Az egyidejűleg szállítható adatszélesség bitekben
- Címző képesség
- A busz egyes vonalai egyediek vagy megosztottak
- Adatátviteli sebesség (throughput) - bits per second
- A végpontok közötti távolság
- A csatlakoztatható eszközök száma és típusa
- A szükséges vezérlés típusa
- A kitűzött cél
- Tulajdonságok és képességek



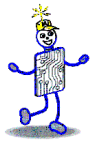
Buszok osztályozása

- Párhuzamos és soros buszok
- Az átvitel iránya szerint
  - Simplex – egyirányú
  - Fél duplex – kétirányú, de nem egyidőben
  - (Teljes) duplex – egyidejűleg két irányú
- Az összekapcsolás módszere szerint
  - Point-to-point – egy forrásból egy nyelőbe
    - Kábelek – point-to-point buszok amelyek külső eszközhöz kapcsolódnak
  - Multipoint bus – also broadcast bus or multidrop bus
    - Több pontot kapcsol egyetlen másikhoz



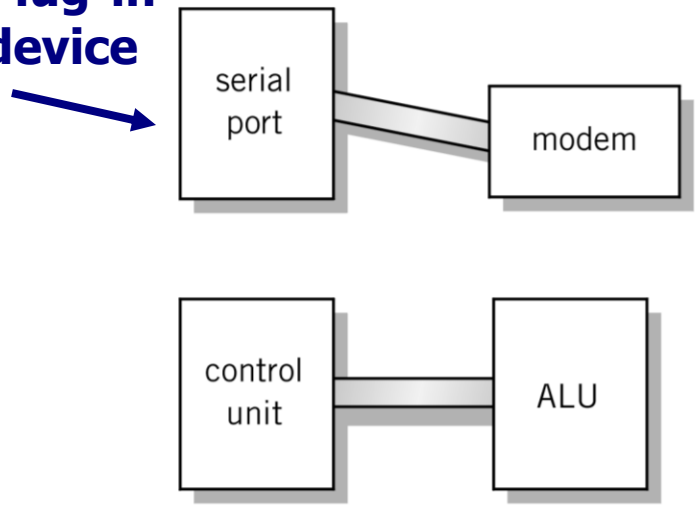
Párhuzamos és soros buszok

- Párhuzamos
  - Nagy átviteli sebesség, mivel a szó valamennyi bitjét egyidejűleg visszük át
  - Drága és nagy helyigényű
  - Elektromágneses interferenciákra érzékeny, ami korlátozza sebességét és hosszát
  - Általában kis távolságokra használják, mint pl. CPU buszok és számítógép alaplapok
- Soros
  - Egyszerre csak 1 bitet visz át
  - Egy adatvezeték pár és néhány vezérlő vonal
  - Sok alkalmazás esetén az átviteli sebesség nagyobb mint párhuzamos busz esetén, az elektromágneses interferencia hiánya miatt

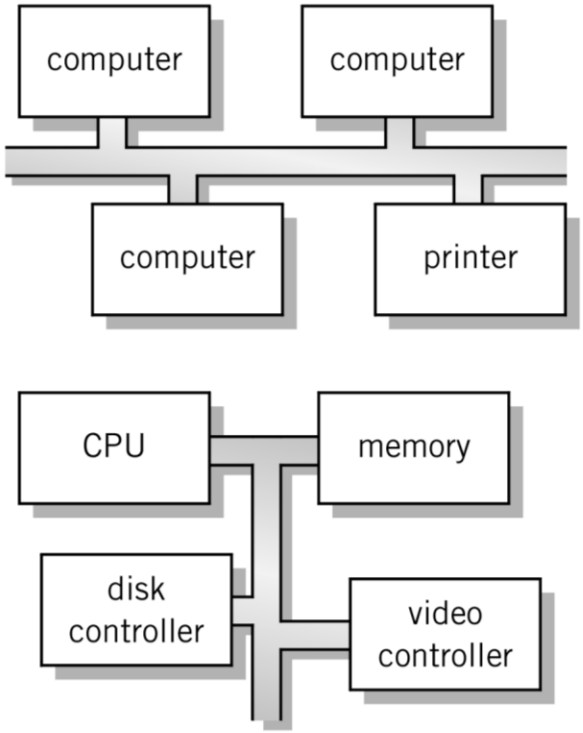


Point-to-point vs. Multipoint

Plug-in device



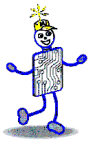
examples of point-to-point buses



examples of multipoint buses

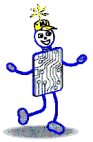
Broadcast bus Example: Ethernet

Shared among multiple devices



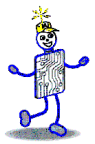
Az utasítások osztályozása

- Adatmozgató (load, store)
  - A leggyakoribb, nagy flexibilitású
  - Memóriára és regiszterekre alkalmazható
  - Mekkora a *word* mérete? 16? 32? 64 bits?
- Aritmetikai
  - Operátorok **+ - / * ^**
  - Egészek és lebegő pontosok (Integers and floating point)
- Logikai (boolean)
  - Általában tartalmazza az **AND, XOR és NOT** utasításokat
- Egyoperandusú manipuláló utasítások
  - Negálás, csökkentés, növelés, nullázás

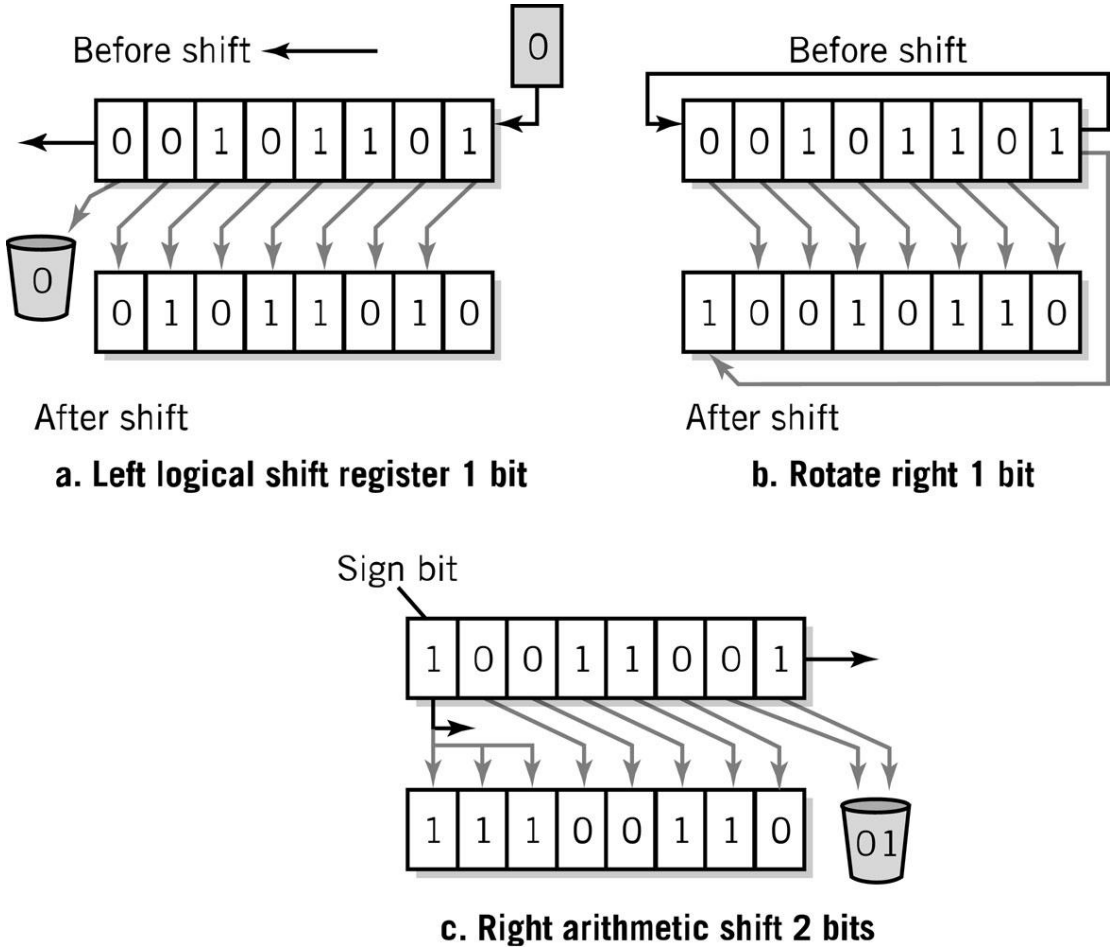


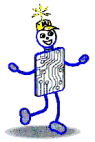
További utasítás osztályozások

- Bit manipuláló utasítások
  - Feltétel vizsgáló jelzőbitek
- Eltolás és forgatás
- Program vezérlés
- Verem kezelő utasítások
- Több adatos utasítások
- I/O és számítógép vezérlés



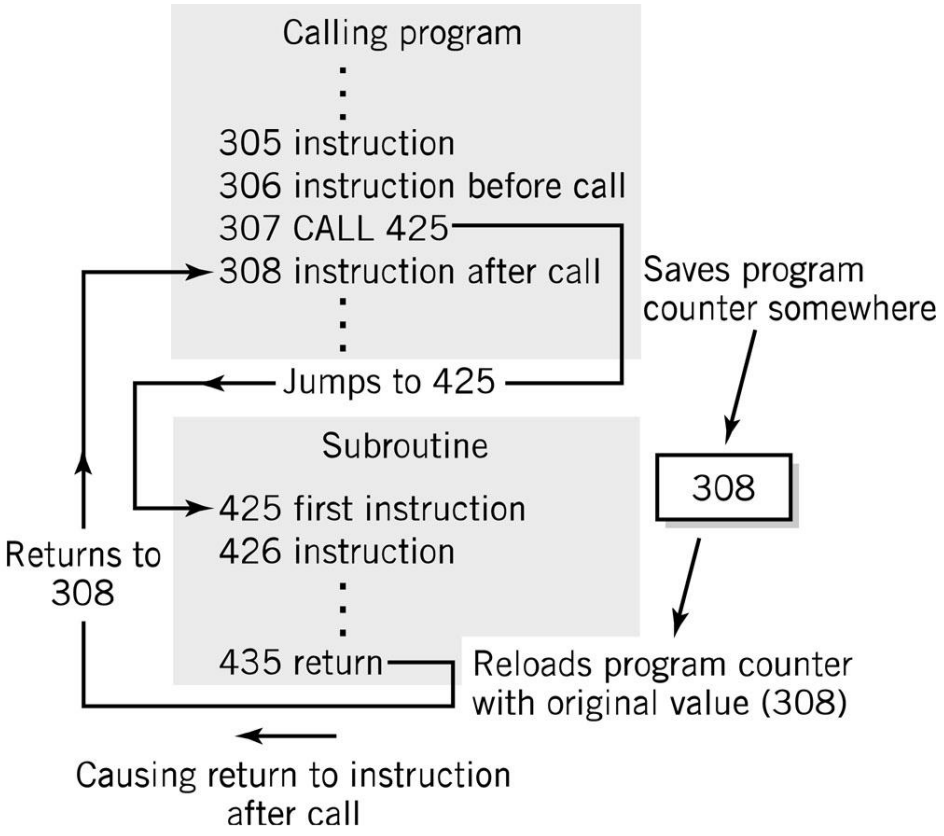
Regiszter eltolás és forgatás

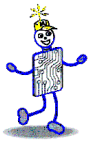




Program vezérlő utasítások

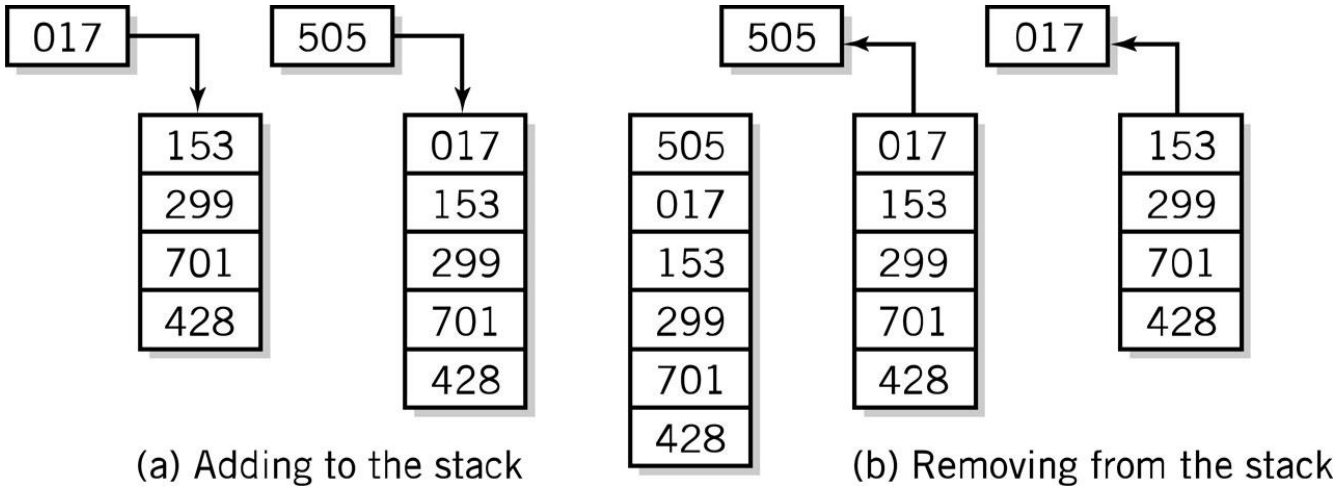
- Program vezérlés
  - Ugrás és elágazás (Jump and branch)
  - Szubrutin hívás és visszatérés (Subroutine call and return)

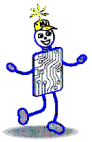




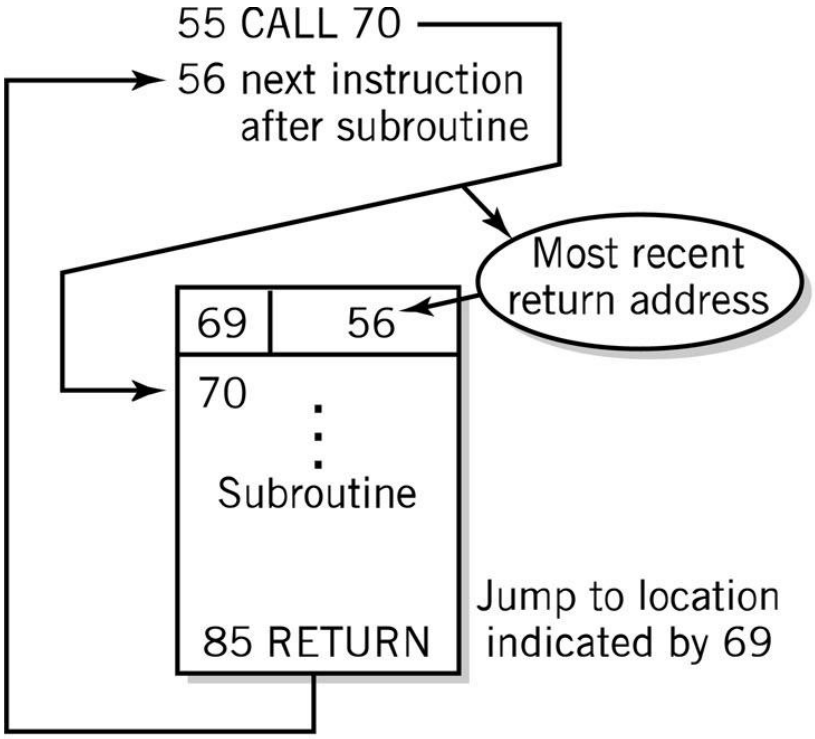
Verem kezelő utasítások

- Verem kezelő utasítások
  - LIFO módszerű információ szervezés
  - Az elemeket a hozzáadás sorrendjéhez képest fordított sorrendben lehet eltávolítani

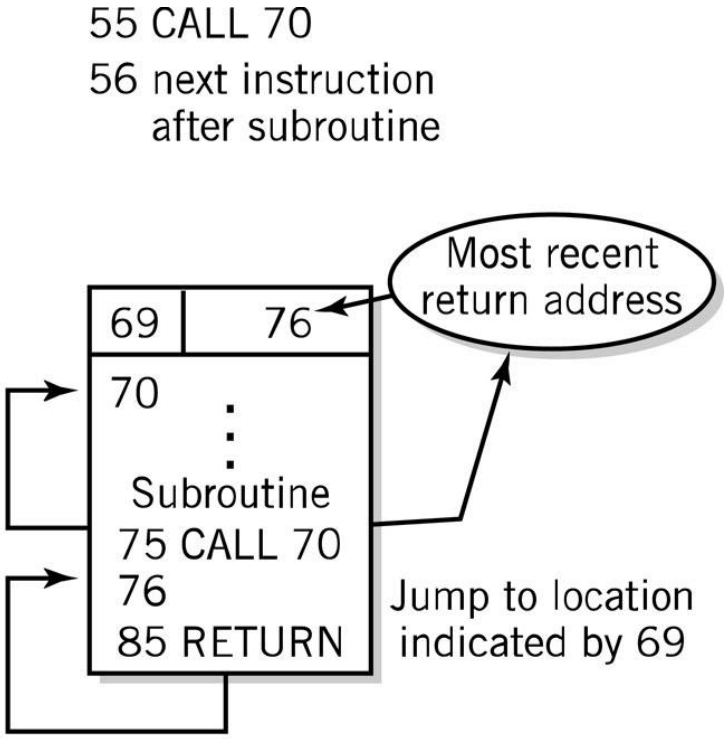




Szubrutin visszatérési cím tárolása rögzített helyen: *Oops!*

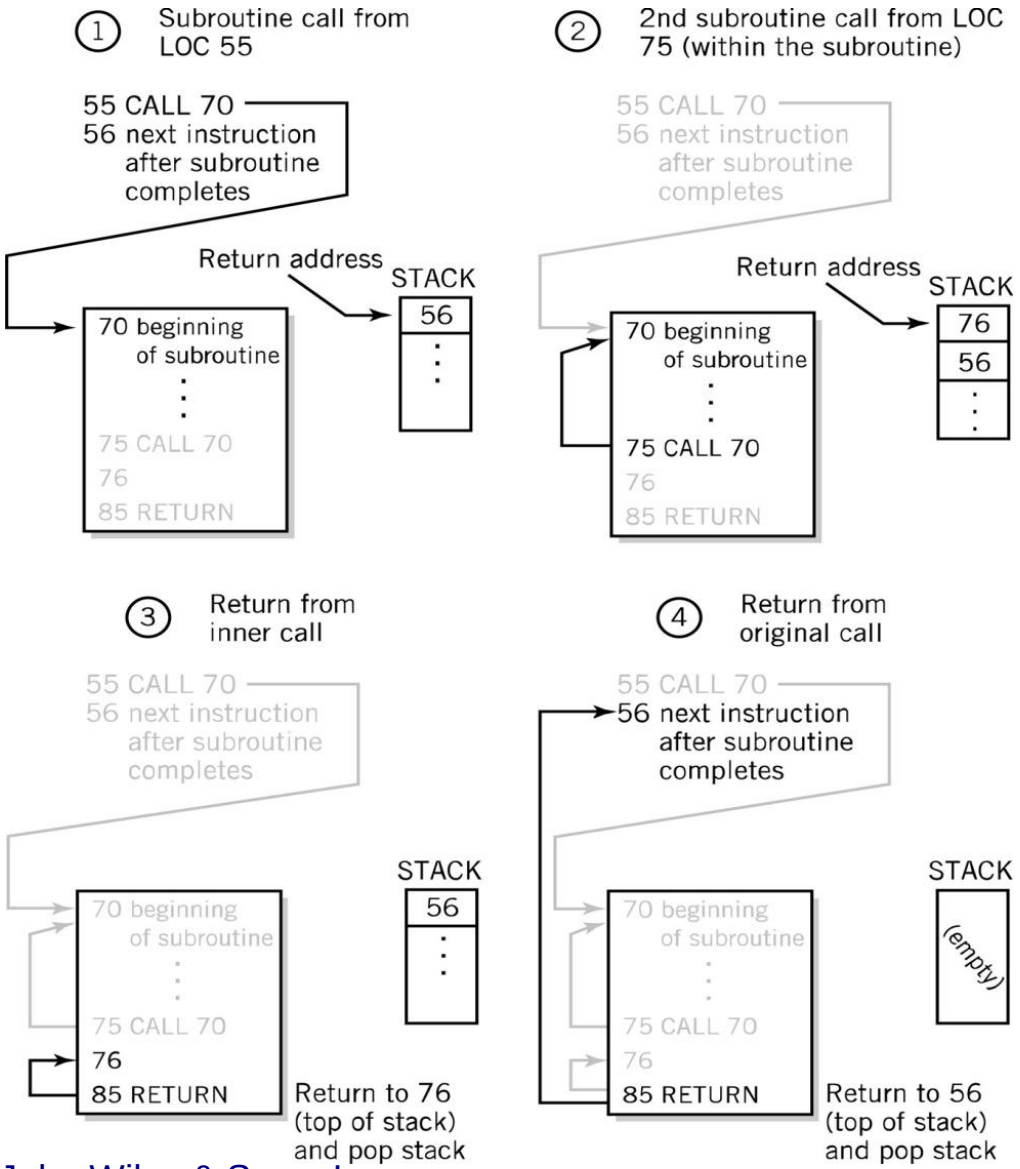


a. Subroutine called from loc.55

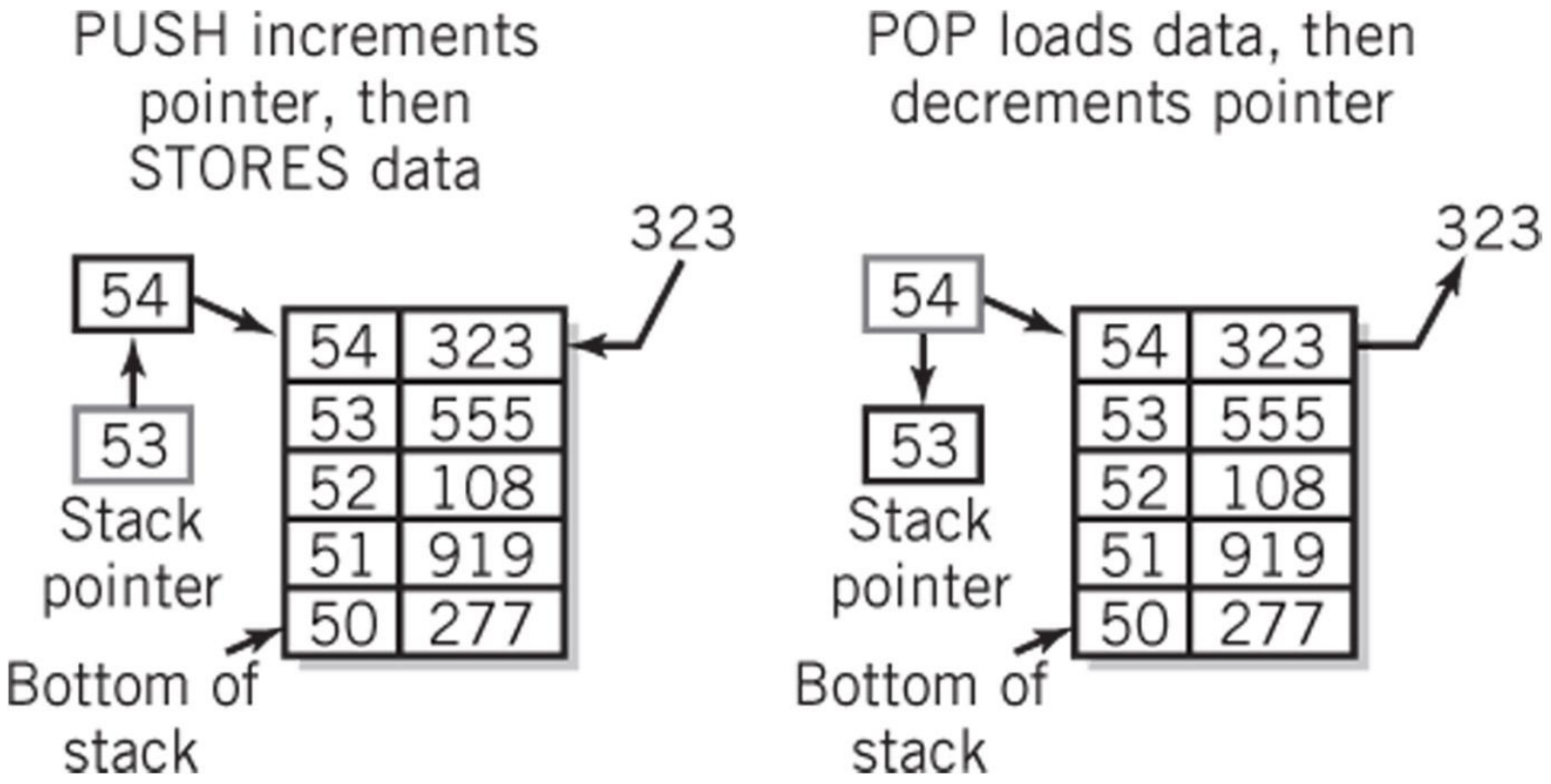


b. Subroutine re-called from 75, within the subroutine

Szubrutin visszatérési cím tárolása veremben

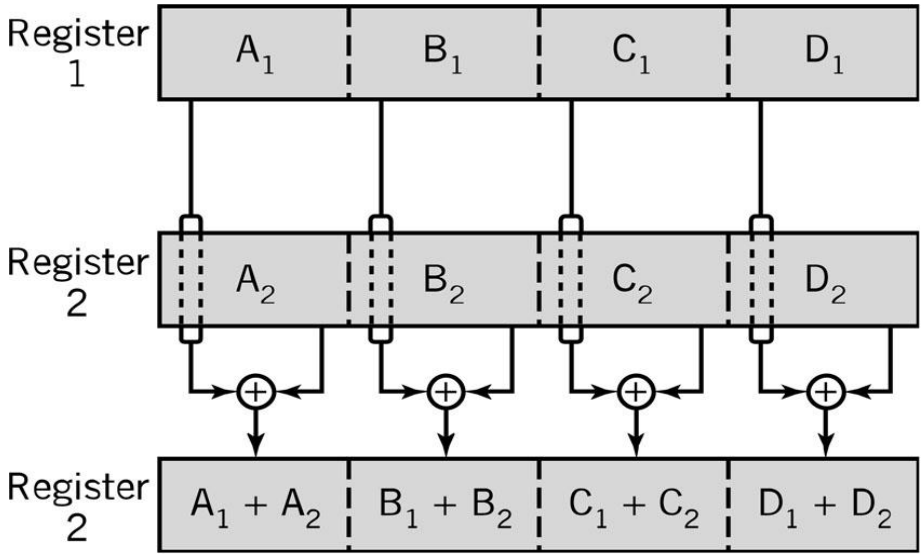


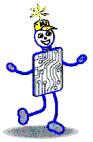
Memória blokk veremtárolóként



Több adatos utasítások

- Egyazon utasítást több adatdarabon egyidejűleg elvégez
  - SIMD: Single Instruction, Multiple Data
  - Elterjedten használt multimedia, *vektor* és tömb feldolgozó alkalmazásokban





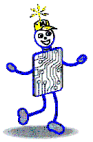
Az utasítások alkotóelemei

- OPCODE: művelet
- Forrás OPERANDUS(ok)
- Eredmény OPERANDUS

} **Címek**

  - Az adat helye (regiszter, memória)
    - Explicit: tényleges szerepel az utasításban
    - Implicit: alapértelmezett értékű

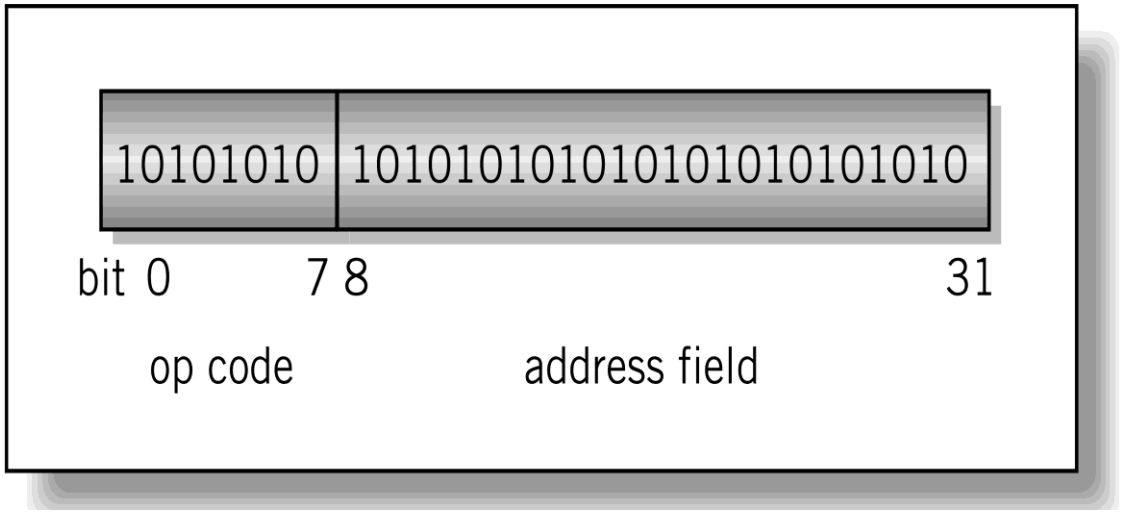
OPCODE	Source OPERAND	Result OPERAND

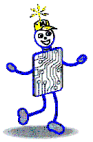


Utasítás Formátum

- *Számítógép-specifikus* minta, amely megadja
  - Az operációs kód hosszát
  - Az operandusok számát
  - Az operandusok hosszát

Simple 32-bit Instruction Format





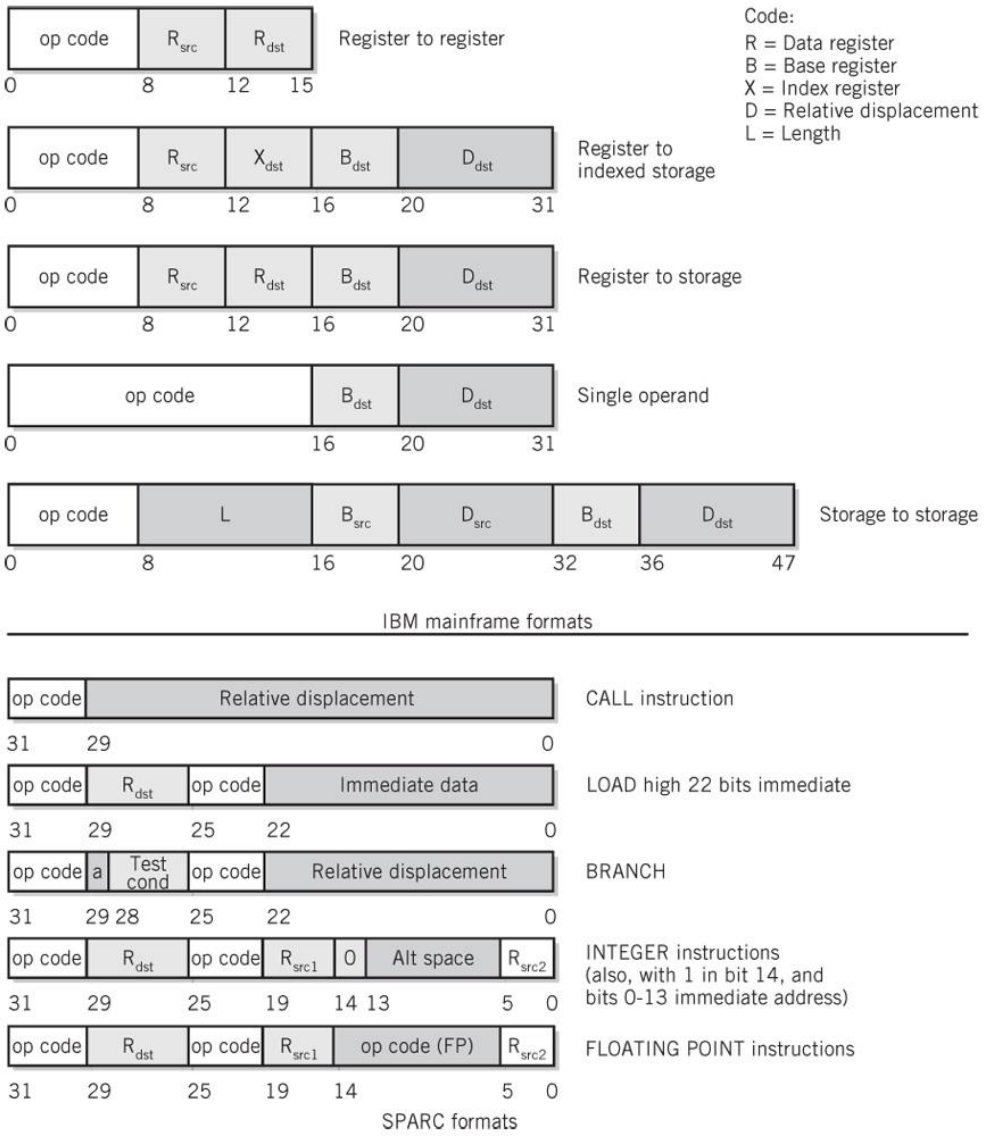
Utasítások

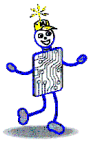
- Utasítás
  - A számítógépnek szóló parancs
  - Hatására elektromos vagy optikai jelek küldődnek speciális feldolgozó áramköröknek
- Utasítás készlet
  - Tervezéskor határozzák meg a processzor által elvégzendő feladatokat
  - Ami különbséget tesz a számítógép architektúrák között
    - Az utasítások száma
    - Az egyes utasítások által végrehajtott műveletek komplexitása
    - A támogatott adattípusok
    - Formátum (megjelenés, rögzített vagy változó hosszúság)
    - Regiszterek használata
    - Címzés (méret, módok)

Az utasítás szó mérete

- Rögzített és változó méret
  - A csővezetékezés nagyrészt kiszűrte a változó hosszúságot használó architektúrákat
- A mostani architektúrák 32 vagy 64 bites szavakat használnak
- Címzési módok
  - Közvetlen
    - Ilyen módot használ az LMC
  - Regiszteren keresztül közvetett
  - Hívják közvetlen, indirekt és indexeltnek is

Utasítás formátum példák





Copyright 2010 John Wiley & Sons