7. Fejezet
A processzor és a memória

Hardver és Szoftver rendszerek architektúrája:

Egy Információ Technológiai Szemlélet

3. kiadás, Irv Englander

John Wiley and Sons Ó2003

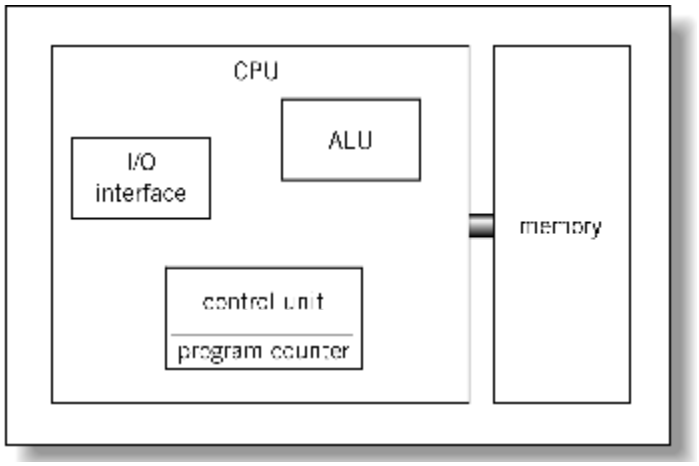
Wilson Wong, Bentley College

Linda Senne, Bentley College

A CPU három fő része

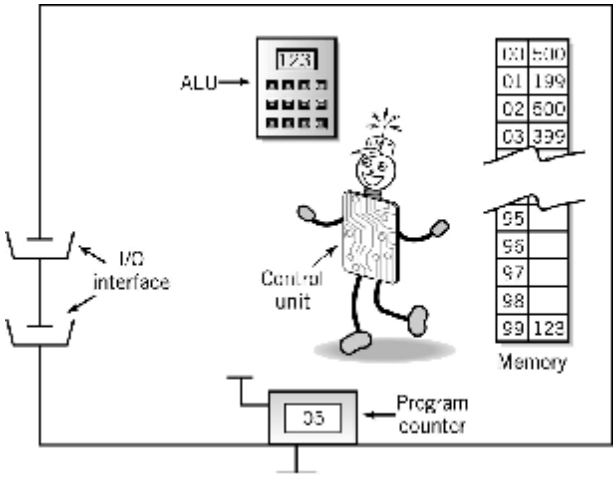
- *ALU (Arithmetic/Logic Unit = Aritmetikai/Logikai Egység)*
  - Számítási (aritmetikai és logikai), ill. összehasonlító műveleteket hajt végre a tárolt adatokon (regiszterek tartalma változik)
- *CU (Control Unit = Vezérlő Egység)*
  - Vezérli az utasítás-végrehajtást (fetch/execute ciklusokat)
  - Funkciók:
    - A CPU regisztereiben az adatokkal végez műveleteket (nem változnak meg az adatok)
    - Beolvassa a program utasításait és parancsokat ad az ALU-nak
  - Részei:
    - *Memory Management Unit (Memória Vezérlő Egység):*
      - Felügyeli a „fetch”-elését, az utasításoknak és adatoknak
    - *I/O Interfész*
      - néha egybeépítik a memória vezérlővel, melyet *Bus Interface Unit*-nak nevezünk
- *Regiszterek*
  - Példa: *Program counter (PC – program számláló)* vagy instruction pointer (utasítás mutató) meghatározza a következő utasítást a futtatáshoz

Számítógép sematikus felépítése



7-3

Little Man Computer (LMC)



7-4

Regiszterek

A regiszterek tulajdonságai

- Kicsi, *permanens (nem múlékony)* tárolóhelyek a CPU-n belül, amiket a számolások részeredményeinek tárolására használ
- A Vezérlő Egység (CU) vezérli őket
- *Speciális használatra* vannak tervezve
  - *általában mindegyik adott feladatot lát el*
- Tároló méretük néhány bit, ill. byte
- Tárolhat adatot, címet és utasítást
- Hány regisztere van az LMC-nek?

7-6

Regiszterek

- Regiszterek használata
  - „Scratchpad” („vázltfüzet”) az aktuálisan futó program számára
    - A gyakran használt, ill. gyorsan elérendő adatokat tárolja
  - A CPU és a futó programra vonatkozó információt tárolja
    - A következő program utasítás címe
    - Külső eszközökből érkező jelek
  - általános, ill. speciális célú regisztereket különböztetünk meg
- **Általános célú regiszterek**
  - *Felhasználó számára láthatóak*
  - Megszakítási és adat értékeket tartalmaz
  - Megegyezik az LMC számológépével
  - Minden processzorban van néhány tucat

Speciális célú regiszterek

- *Program Count Register = Program számláló regiszter (PC)*
  - Instruction pointernek (utasítás mutatónak) is hívjuk
- *Instruction Register = Utasítás regiszter (IR)*
  - Memóriából származó utasításokat tartalmaz
- *Memory Address Register = Memória Cím Regiszter (MAR)*
- *Memory Data Register = Memória Adat Regiszter (MDR)*
- *Status Registers = Állapot Regiszterek*
  - A processzor és a futó program állapotát tartalmazza
  - *Flag*-eket *(jelzőbit)* (egybites logikai változókat) tartalmaz,
    - nyomon követhessük a számítógép működését
    - túlcsordulás, ill. carry, elektromos kimaradás, egyéb belső hibák stb.

Regiszter műveletek

- Más helyekről származó értékek tárolása (regiszterek és memóriák)
- Összeadás és kivonás
- Bit műveletek (Shift, Rotate – Eltolás, Felcserélés)
- A regiszter tartalmának ellenőrzése, pl.: nulla vagy pozitív

7-9

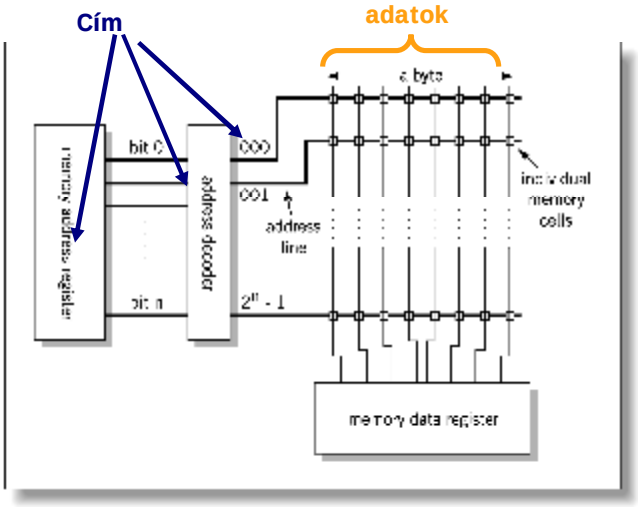
Memória

A memória működése

- Minden memóriatartomány egyedi címmel rendelkezik
- Az utasítás címe a Memória Cím Regiszterbe kerül, amely megtalálja a helyét a memóriában
- A processzor határozza meg, hogy tároljon vagy olvasson adatot
- A Memória Adat Regiszter és a memória közötti szállítási helyet igényel
- A Memória Adat Regiszter kétirányú regiszter

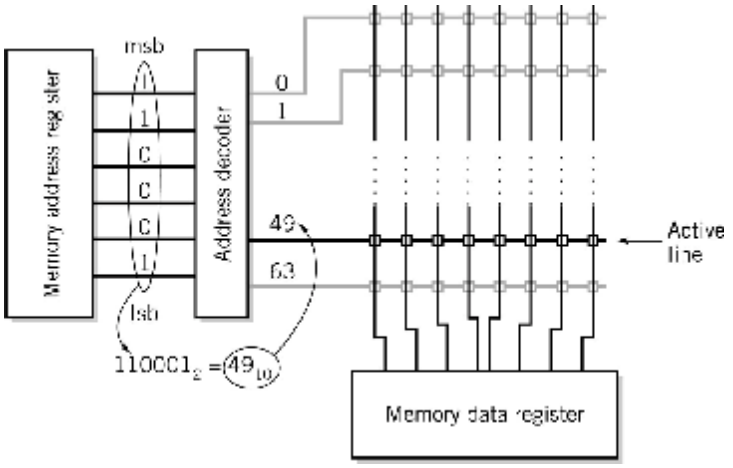
7-11

Kapcsolatok a MAR, a MDR és a memoria között

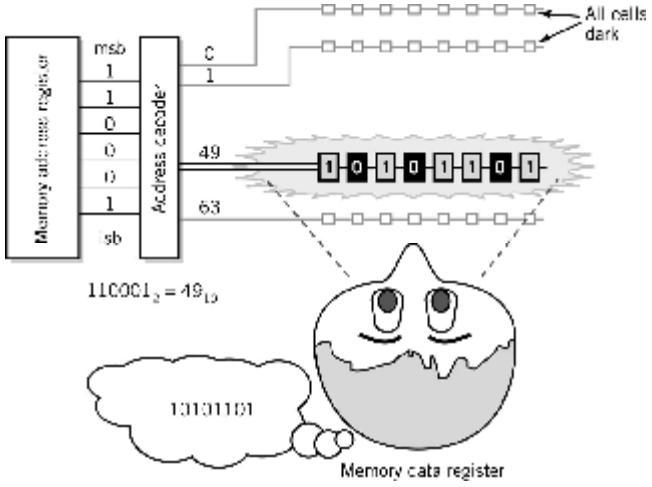


7-12

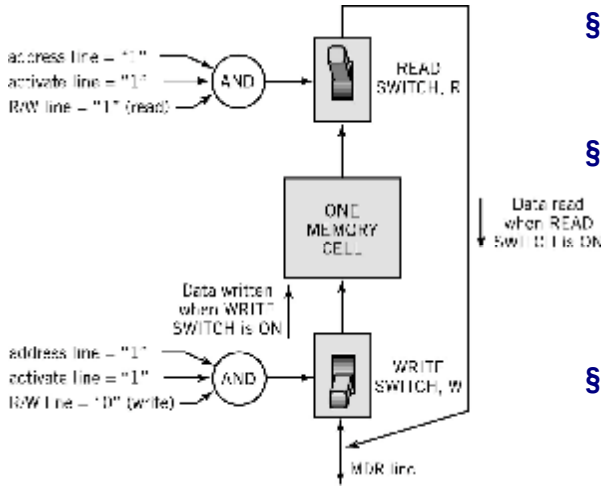
MAR-MDR példa



A memória működésének vizuális leírása



Egy memória cella működése



- Két kapcsoló:
  - Olvasás (Read)
  - Írás (write)
- Két vezérlő áramkör (AND), három vezérlő jellel:
  - address line
  - activate line
  - R/W line
- Információt tároló memória cella

Memória-kapacitás

- Két tényező határozza meg
  1. A MAR bit-jeinek száma
     - LMC = 100 (00 - 99)
     - 2^K, ahol K = a regiszter szélessége bit-ekben
  2. Az utasításban szereplő cím rész (operandus) mérete
     - 4 bit 16 helyet tudunk megkülönböztetni
     - 8 bit enged 256 helyet tudunk megkülönböztetni
     - 32 bit 4,294,967,296 vagy 4 Giga helyet tudunk megkülönböztetni
- A memória mérete lényegesen befolyásolja a számítógép teljesítményét
  - A kevés memória arra kényszeríthetheti a processzort, hogy 50%-alatt dolgozzon

RAM: Random Access Memory (Véletlen elérésű memória)

- *RAM*
  - eredetileg mágnesezhető tároló cellák, ma DRAM és SRAM
- *DRAM (Dynamic RAM=Dinamikus memória)*
  - Ma a legáltalánosabban használt, olcsó
  - Volatilis tároló: feszültség kikapcsolásával elvész a tartalma
  - Tartalmát hamar elveszti : másodpercenként kb. 1000-szer kell újraírni a tartalmát az értékek tárolásához
- *SRAM (Statikus RAM)*
  - Gyorsabb és drágább, mint a DRAM (más gyártási technológiája)
  - Volatilis tároló: feszültség kikapcsolásával elvész a tartalma
  - Keveset használnak belőle a **cache** memóriában, a gyors eléréshez

7-17

ROM - Read Only Memory (Csak olvasható memória)

- Stabil, nem volatilis tároló olyan programok tárolásához, amelyek nem változnak
- Mágneses magmemória
- EEPROM
  - Electrically Erasable Programmable ROM (Elektromosan törölhető és programozható ROM)
  - Lassabb és kevésbé rugalmasan használható, mint a FLASH ROM
- *Flash ROM*
  - Gyorsabb, mint a merevlemezek, de sokkal drágább is
  - Felhasználási területek:
    - BIOS: BOOT program, diagnosztikai programok
    - Digitális kamerák

7-18

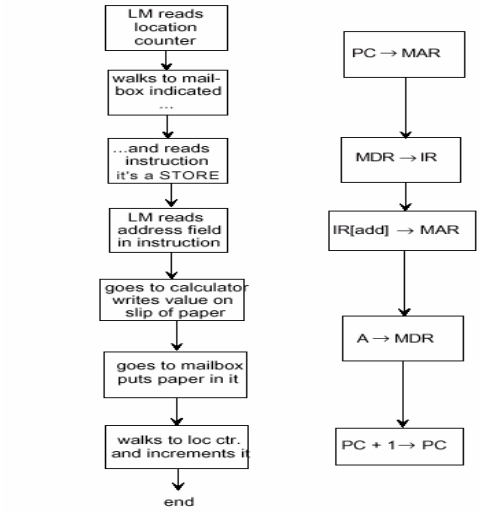
CPU utasítások végrehajtása

Fetch-Execute ciklus

- A CPU műveletek végrehajtása két ciklusra osztjuk (két memória hozzáférésre)
  - mind a utasítás kódja, mind az az adat, amin a utasítást végrehajtjuk a memóriában van tárolva
- *Fetch*
  - Megkeressük, ill. dekódoljuk az utasítást, a memóriából a regiszterekbe töltjük és jelzést küldünk az ALU-nak
- *Execute*
  - Végrehajtjuk azt a műveletet, amit az utasítás kódol
  - Adatokat mozgatjuk, ill. módosítjuk

7-20

LMC vs. CPU
Fetch és Execute ciklus



7-21

A LOAD utasítás Fetch/Execute ciklusa

1. PC -> MAR	Az adatokat a PC-ből a MAR-ba mozgatja
2. MDR -> IR	Az adatokat az IR-be mozgatja
3. IR(address) -> MAR	Az utasítás cím része (operandus) betöltődik a MAR-ba
4. MDR -> A	Az aktuális adatokat a tároló regiszterbe mozgatja
5. PC + 1 -> PC	A programszámláló növelése

7-22

A STORE utasítás Fetch/Execute ciklusa

1. PC -> MAR	Az adatokat a PC-ből a MAR-ba mozgatja
2. MDR -> IR	Az adatokat az IR-be mozgatja
3. IR(address) -> MAR	Az utasítás cím részletei betöltődnek a MAR-ba
4. A -> MDR*	A tároló regiszterből az MDR-be mozgatja az adatokat
5. PC + 1 -> PC	A programszámláló növelése

* Vegyük észre, hogy a 4. pont hogyan változott meg a LOAD utasításhoz képest! Mikor történik a memória használata a két esetben?

7-23

Az ADD utasítás Fetch/Execute Ciklusa

1. PC -> MAR	Az adatokat a PC-ből a MAR-ba mozgatja
2. MDR -> IR	Az adatokat az IR-be mozgatja
3. IR(address) -> MAR	Az utasítások cím részletei betöltődnek a MAR-ba
4. A + MDR -> A	Az MDR tartalma a tár tartalmához adódik
5. PC + 1 -> PC	A programszámláló növelése

7-24

LMC utasításainak Fetch/Execute ciklusai

Kivonás	**Bemenet**	**Kimenet**	**Szünet**
PC **à** MAR	PC **à** MAR	PC **à** MAR	PC **à** MAR
MDR **à** IR	MDR **à** IR	MDR **à** IR	MDR **à** IR
IR[addr] **à** MAR	IOR **à** A	A **à** IOR	
A – MDR **à** A	PC + 1 **à** PC	PC + 1 **à** PC	
PC + 1 **à** PC			

Ugró (elágazás) utasítás	**Feltételes (elágazás) ugró utasítás**
PC **à** MAR	PC **à** MAR
MDR **à** IR	MDR **à** IR
IR[addr] **à** PC	Ha a feltétel hamis: PC + 1 **à** PC
	Ha a feltétel igaz: IR[addr] **à** PC

7-25

Buszok

Busz

- Fizikai kapcsolat, ami lehetővé teszi az adatátvitelt a számítógépben két pont között
- Elektromos csatlakozások csoportja a két csomópont közötti jelátvitelhez
  - *Line (vonal)*: csatlakozási pont a buszon
- A jeleknek négy fajtáját különböztetjük meg
  1. Adat (alfa-numerikus, numerikus, utasítások stb.)
  2. Cím
  3. Vezérlő jelek
  4. Táp (néha)

Buszok a számítógépben

- Processzor és memória közötti busz
- Az I/O perifériák lehetnek azonos buszon a CPU-val és a memóriával vagy külön buszon
- Ha a (plug-in) I/O elemek a CPU és a memória közötti buszt használják, akkor a fizikai egységet *backplane-nek* hívják
  - *Rendszerbusznak és külső busznak is hívják*
  - Ez jó példa a „*broadcast" buszra*
  - A PC-kben általában ezeket az elemeket egy nyomtatott áramköri lapra az *alaplapra (motherboard)* integrálják
    - ez tartalmazza a CPU-t és összekapcsolja azt a többi komponenssel

Busz jellemzői

- Protokoll
  - A kommunikáció jól dokumentált leírása
  - Világosan megmagyarázza az összes vonal és a vonalakon lévő minden jel jelentését
- Adatátviteli kapacitás:
  - az adat átvitel mértékegysége bit/másodperc
- Adatbusz szélessége:
  - Az egyszerre átvihető adatok szélessége bitekben

7-29

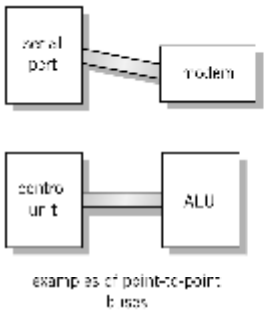
Busz topológiák és üzenetváltás

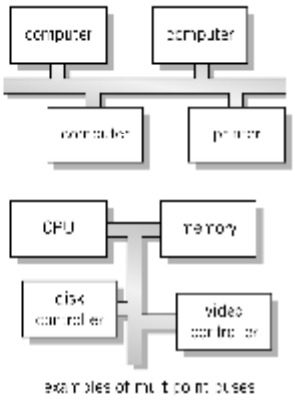
- Pont-pont kapcsolat
- Többpontos kapcsolat
  - „Broadcast busz”: minden elem megkapja a buszra kitett adatokat

7-30

Pont-pont kapcsolat vs. több pontos

Plug-in eszköz





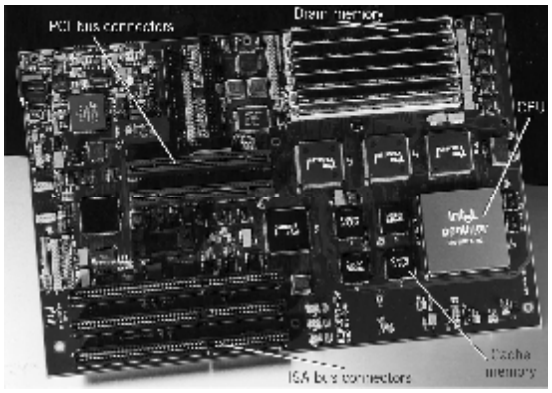
Broadcast bus példa: Ethernet

Az összetett eszközök mindegyike között van kapcsolat

7-31

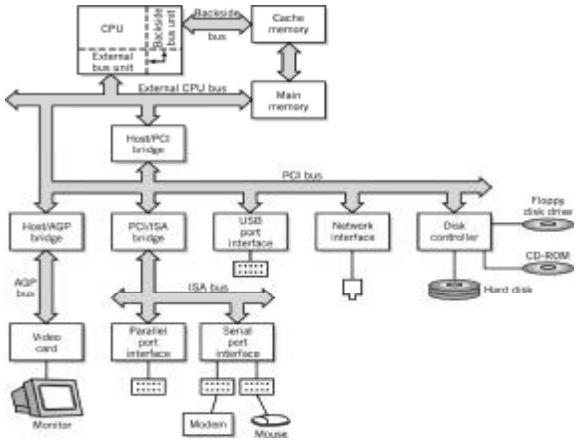
Az alaplap

§ Nyomtatott áramkör, melyen a processzor és egyéb fő elemek találhatók



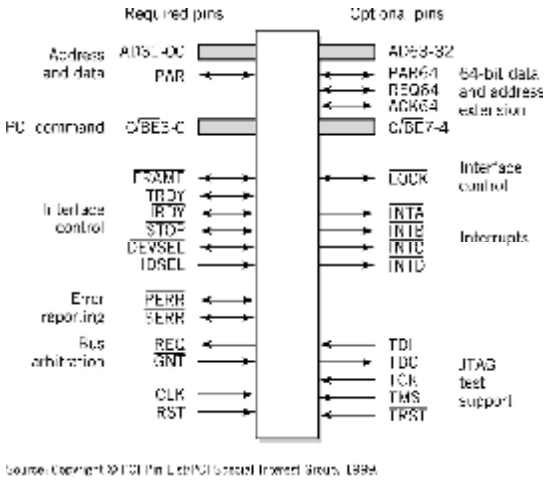
7-32

Összeköttetések egy tipikus PC-ben



Busz interface bridge-ek:
különböző típusú buszokat kapcsolnak össze, átviszik az adatokat az egyik buszról a másikra

PCI bus-on használt jelek



Utasítások

- Utasítás
  - A számítógépet vezérlő parancsok
  - Végrehajtásukkor elektromos jelek hatására a számítógép egyes áramkörei különböző műveleteket hajtanak végre
- Utasítás készlet
  - A tervező által meghatározott funkciók, melyet a processzor végre tud hajtani
  - A különböző architektúrájú számítógépek között különbséget tehetünk a következő szempontok alapján:
    - Végrehajtható utasítások száma
    - Az egy utasítással végrehajtható műveletek bonyolultsága szerint
    - Támogatott (feldolgozható) adattípusok
    - Utasítások formátuma (kötött vagy változó hosszúság)
    - Regiszterek használata
    - Címzés (cím mérete, címzési módok)

7-35

Utasítás részei

- Műveleti kód (OP code):
  - a feladat, amit a számítógépnek végre kell hajtani
- Forrás operandus
- Eredmény operandus

} **Címek**

  - Adat helyének meghatározása (regiszter, memória)
    - Explicit: tartalmazza az utasítás
    - Implicit: alapértelmezés szerinti helyen

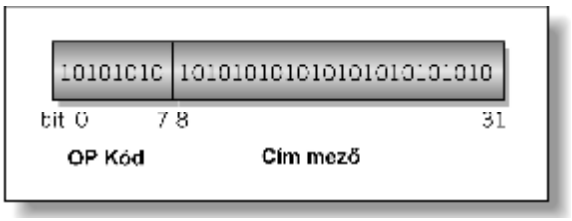
OP code	Forrás operandus	Eredmény operandus

7-36

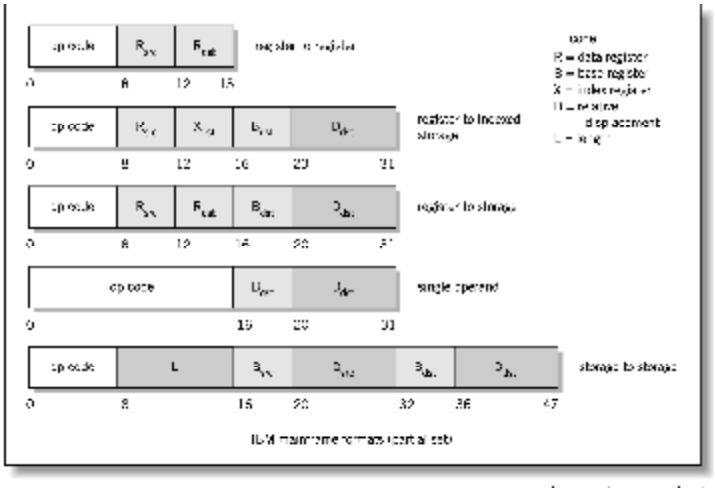
Utasítások formátuma

- *Számítógép-specifikus* leírás, mely meghatározza:
  - műveleti kód hosszát
  - Operandusok számát
  - Operandusok hosszát

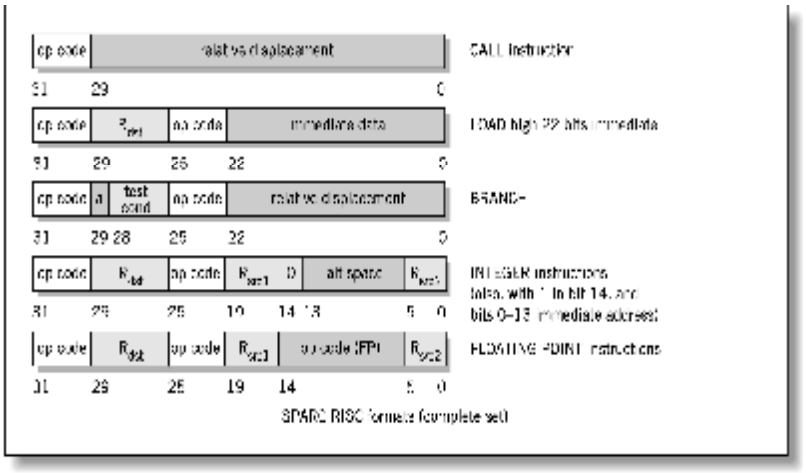
Egy egyszerű 32-bites utasítás formátum



Utasítás formátumok: CISC



Utasítás formátumok: RISC



Utasítások típusai

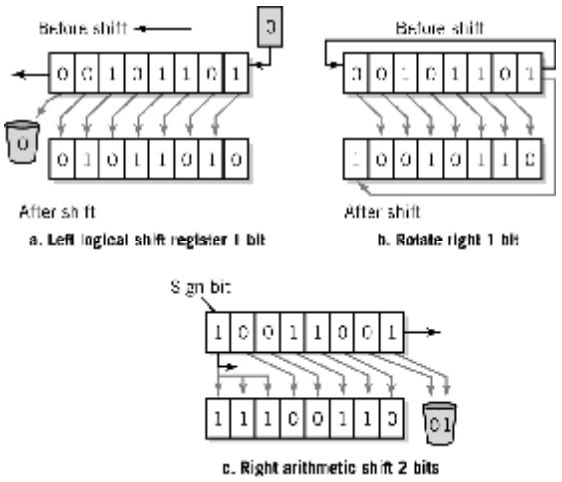
- Adat átviteli műveletek (load: betöltés, store: tárolás)
  - Általánosan használt, változatos megvalósítások
  - A memória és a regiszterek közötti kapcsolat
  - Mekkora a *word* adattípus? 16? 32? 64 bit?
- Aritmetikai utasítások
  - operátorok: **+ - / * ^**
  - Egész és lebegőpontos adatokon is végezhető
- Boolean logikai műveletek és relációs operátorok
  - Relációs operátorok: **> < =**
  - Logikai operátorok: **AND, OR, XOR, NOR,** és **NOT**
- Egyetlen operandusot használó utasítások
  - Negáció, dekrementálás, inkrementálás

Egyéb utasítás típusok

- Bitmanipuláló utasítások
  - Flag-ek (jelzőbitek) a feltételek tesztelésére
- Bitenkénti eltolás, ill. forgatás
- Program vezérlő utasítások
- Verem tár kezelő utasítások
- Összetett adatelemeken végzett utasítások
- I/O elemeket és számítógépet vezérlő utasítások

7-41

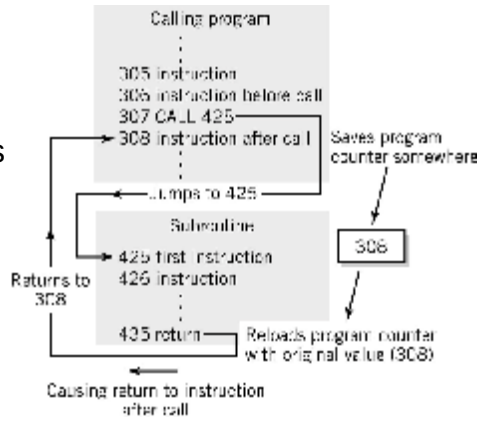
Regiszterben tárolt értékek eltolása és forgatása



7-42

Program vezérlő utasítások

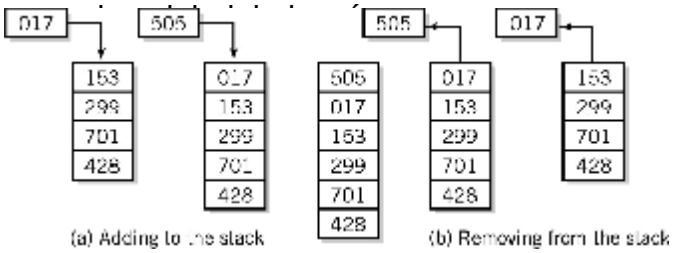
- Program vezérlése
  - Ugrás és elágazás utasítások
  - Szubrutin hívása és visszatérés



7-43

Veremtár kezelő utasítások

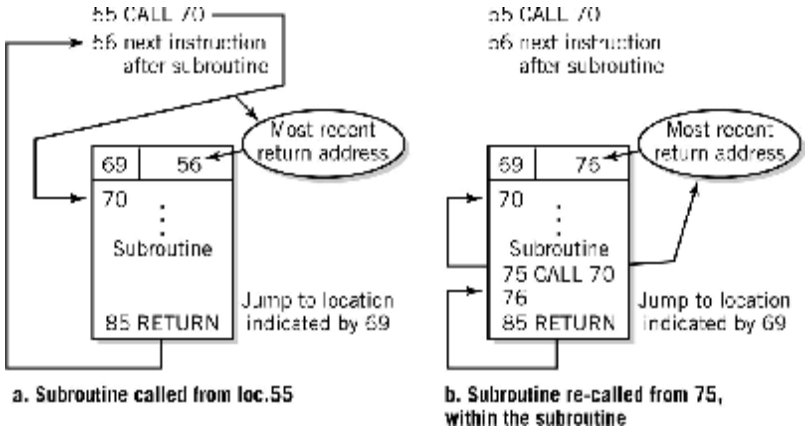
- Veremtár kezelő utasítások
  - LIFO (utolsó be, első ki) működés, információ tárolása
  - Elemek kivétele fordított sorrendben történik, mint



Beszúrás Kivétel

7-44

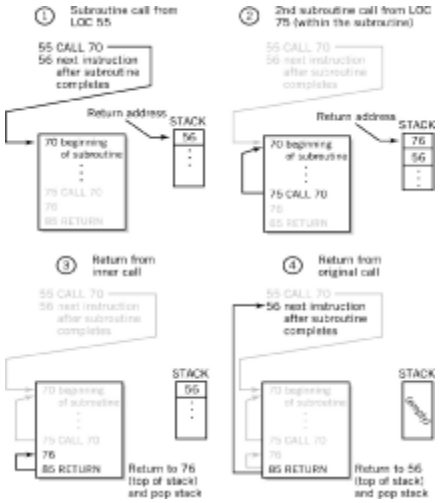
Szubrutin visszatérési címének tárolása állandó helyen : *Oops!*



Hibás működés lehetséges!!

7-45

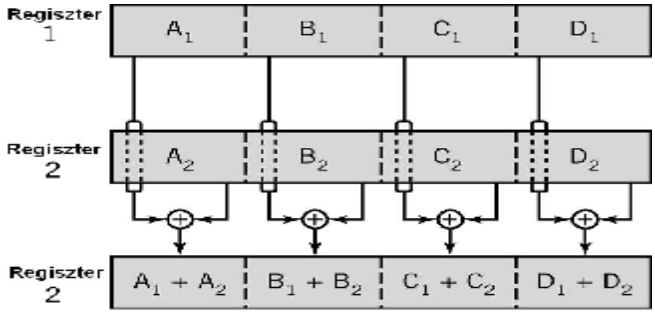
Veremtár a szubrutin visszatérési címének tárolására



7-46

Összetett adatkezelő utasítások

- Párhuzamosan hajt végre egyszerű műveleteket adatok csoportján
  - SIMD: Single Instruction, Multiple Data
    - egy utasítás, több adat
  - Intel MMX™: 57 multimédiás utasítás
  - Általánosan használják vektorok és tömbök feldolgozásakor



7-47