A gyakorlat anyaga két részből áll:

- az első részben megismerkedünk a merevlemezek particionálásával kapcsolatos ismeretekkel,
- a második részben megvizsgáljuk a mikroprocesszor működését egy egyszerű mintapéldán keresztül. (Gyakorlatvezetőknek célszerű előzetesen az Intel honlapján is megnézni ezt a feladatot, hogy jobban tudják rajzzal, vagy mutogatással segíteni a megértést. Web címet lásd lentebb.)
- Egyszerű CPU emulátor: https://schweigi.github.io/assembler-simulator/

1. Merevlemezek particionálása

1.1 A particionálás fogalma: a merevlemez tárolóterületének felosztása több, önálló fájlrendszer tárolására alkalmas részre, melyek önállóan formázandók és saját meghajtóazonosító betűvel (pl. C:, D:, E:, stb) érhetők el. Nincs meghajtóazonosítója és nem kell formázni a boot-manager partíciót (lásd később).

1.2.A particionálás szükségessége, indoka:

- Kényelmi szempont: szeretnénk az adatfájljainkat a programfájljainktól, vagy az operációs rendszer állományait az alkalmazói programoktól elkülöníteni, vagy adatainkat több, lemezkezelő programokkal (pl. formázó, töredezettségmentesítő, stb programok) önállóan kezelhető részekre osztani.
- Az operációs rendszer miatti kényszermegosztás: nagyméretű merevlemezek tartalmát egyes operációs rendszerek nem képesek egyben elérni. Például a FAT (File Allocation Table, fájl foglalási táblázat) fájrendszert használó merevlemezpartíciók nem lehetnek 2 Gbyte-nál nagyobbak, ezért pl. egy 5.1 Gbyte kapacitású merevlemezt két FAT partícióra kell osztanunk.
- A merevlemezterület jobb kihasználása: FAT fájlrendszerre formázott partíciók alapegységeinek nagysága szektorokban mérve függ a formázott partíció méretétől. Nagyobb partíciókon az alapegységek nagyobb lemezterületeket jelentenek. Mivel a lemezterület legkisebb lefoglalható/felszabadítható területrésze egy alapegységnyi, és mivel általában az állományok vége nem tölti ki teljesen azt az alapegységet, amelyre kerül (sok fájl esetén a kihasználtság 50 százalék körüli), ezért a foglaltnak nyilvánított alapegységek egy része ki nem használt területeket is tartalmaz. Nagyméretű merevlemezeknél ez a kihasználatlan terület elérheti a partíció 20 százalékát is. Amennyiben kisebb partíciókra osztjuk a merevlemezünket, ez a veszteség csökken.
- Több operációs rendszert akarunk választhatóan üzemeltetni a számítógépünkön, pl. WINDOWS98, LINUX, WinXP, stb. Ilyenkor legalább annyi partícióra kell tagolni a merevlemezt, ahány különböző fájlrendszert rendszert (mely oprendszer függő) akarunk használni. Ez a szám korlátozott, kb.4.
- Többféle fájlrendszert akarunk használni: egyes operációs rendszerek (pl. WinXP, Linux) többféle fájlrendszer párhuzamos kezelésére is képesek. (Általában a FAT és a saját speciális fájlrendszerük.) Mivel egy partíció csak egyféle fájlrendszerre formázható, több fájlrendszerhez több partícióra van szükség. Az egyes fájlrendszer-féleségek nem egyformán előnyösek különböző méretű partíciók esetén: a FAT kisméretű, kb 500Mbyte határig gyorsabb és takarékosabb, mint a speciális saját fájlrendszerek (NTFS = New Technology File System, FAT32 = File Allocation Table 32).

1.3. A particionálás eszköze: az FDISK segédprogram, mely az operációs rendszerek része. Egyes operációs rendszerek telepítési folyamatának része a beépített lemezparticionálás (pl. WinXP). Létezik már particionált lemezt a tartalom sérülése nélkül az üres területen tovább particionálni képes program is (pl. Partition Magic, ill. a LINUX segédprogramja). A particionálás általában a merevlemez

teljes tartalmának elvesztésével jár, ezért előtte a mentés kötelező. Az aktív partíció beállítására is használható az FDISK.

1.4. Partíciófajták: primary (elsődleges) és extended (kiterjesztett) partíciók. A primary partíciók tipikusan operációs rendszerek számára célszerűek. A primary partíciók az egyetlen C: meghajtójelzésen osztozkodnak, azaz mindegyik C: jelű lesz, amiből az is következik, hogy egyidejűleg csak egy lehet aktív, elérhető, használható közülük, egymással nem tudnak kommunikálni, állományokat cserélni. Primary partíciót igényel az IBM Boot Manager-e is, amely lehetővé teszi a különböző partíciókon levő operációs rendszerek közötti választást (az aktív partíció rejtett kijelölése) a rendszer betöltődése (boot) elején egy menüvel. A primary partíciókat, amennyiben operációs rendszert telepítünk rá, általában a merevlemez első 1024 cilinderén belül kell elhelyezni. Az extended partició általában a primary partíciókat követően a merevlemez végén kerül kialakításra. Az extended partíción egy, vagy több logikai meghajtót alakíthatunk ki a partícionálás során, amelyeket később önállóan formázhatunk, tetszőleges fájlrendszerre. Mindegyik logikai meghajtónak saját meghajtóazonosítója van, pl. D:, E:, F:, stb. Ezeken a meghajtókon lévő állományokat egyidejűleg elérhetjük, egyikről a másikra, vagy a C: meghajtóra/ról másolhatjuk (feltéve, hogy azonos fájlrendszerűek). Több operációs rendszer kivételként telepíthető logikai meghajtóra is. Egy merevlemezt max. négy partícióra oszthatunk, amely lehet max. 4 primary és 0 extended, vagy max. 3 primary és 1 extended partíció egyetlen merevlemez esetén. Két merevlemez esetén a számok megduplázódnak.

Példa egy merevlemez particionálására:

```
   ----------------------------------|
   |   primary (WINXP)      C:       |           primary partició
   | --------------------------------|
   |   primary (WinME)      C:       |           primary partició
   | --------------------------------|
   |   logikai meghajtó     D:       |  \
   |---------------------------------|   \
   |   logikai meghajtó     E:       |    \     extended partició
   |---------------------------------|    /
   |   logikai meghajtó     F:       |   /
   ----------------------------------|  /
```

1.5. Boot Manager-es indítás: több operációs rendszert tartalmazó merevlemezről történő rendszertöltés esetén alkalmazhatjuk pl. az IBM OS/2 operációs rendszerhez mellékelt FDISK programmal létrehozható Boot Managert. Ebben az esetben a merevlemez legelején egy 1Mbyte-os Boot Manager partíciót kell létrehoznunk, majd a további primary partíciókba telepíthetjük a kívánt operációs rendszereket. Ilyen esetben a kívánt operációs rendszert egy, a rendszertöltés legelején betöltődő menüről választhatjuk. (A Windows operációs rendszerek a második oprendszer felinstallálásakor automatikusan létrehozzák a boot menüt.)

1.6. Meghajtóazonosítók automatikus kiosztása a rendszer indításakor (Mapping): ez a funkciója az aktuálisan töltött operációs rendszernek arra szolgál, hogy az operációs rendszer által kezelt fájlrendszerrel (fájlrendszerekkel) formázott logikai meghajtókhoz (és csak azokhoz) meghajtóazonosítókat rendeljen. Például egy WinXP-s NTFS rendszerre formázott logikai meghajtóhoz nem fog meghajtóazonosítót rendelni egy WINDOWS98 operációs rendszer, az azon lévő állományok láthatatlanok lesznek a WINDOWS98-ból.

```
  |-----------------------------------|
  |   primary (WIN98)    C:           |   primary partició, aktív
  | ----------------------------------|
  |   primary (WinXP)                 |   primary partició, most nem
elérhető
  | ----------------------------------|
  |   FAT16       D:                  |   elérhető logikai meghajtó
  |-----------------------------------|
  |   NTFS                            |   most nem elérhető logikai meghajtó
  |-----------------------------------|
  |  FAT32        E:                  |   elérhető logikai meghajtó
  |-----------------------------------|
```

A meghajtóazonosítók operációsrendszer töltésekori automatikus kiosztása, hozzárendelése azzal a veszéllyel járhat, hogy pl. egy második merevlemez gépbeszerelése után az első merevlemez logikai meghajtóinak azonosítói megváltoznak és a rajtuk levő programokban található, meghajtóazonosítót is tartalmazó útvonalmegadások, állományhivatkozások hibássá válnak és a programok nem működnek. A korszerű operációs rendszerek, mint pl. a WinXP, lehetővé teszik a meghajtók betűjelének megváltoztatását, így ez a gond elkerülhető.

2. A mikroprocesszor működése

Az alábbi példán bemutatjuk a mikroprocesszor működését. Forrás: Intel homepage. (http://www.intel.com/education/mpuworks/glossary.htm) (Úgy látom, 2003-ra a link megszűnt..., DL)

Az erősen egyszerűsített mintapélda processzorának felépítése az alábbi ábrán látható:

```
  |----------------------------------------   |-----------------------------
---------------   |----------------|
  |            Aritmetikai-logikai            |             Vezérlőegység
|   Előrendező   |
  |                  egység                   |
|     egység     |
  |----------------------------------------   |-----------------------------
---------------   |                |
  |               Regiszterek                 |             Dekódoló egység
|                |
  |                                           |
|                |
  |----------------------------------------   |-----------------------------
---------------   |----------------|
  |               Adat cache                  |                Busz egység
|   Utasítás     |
  |               X,Y,Z,W,...                 |
|      cache     |
  |----------------------------------------   |-----------------------------
---------------   |----------------|
/\
                   Output
```

```
/  \
                      \    /
Input
                       \ /
```

Az egyes részek:

Busz egység: az a hely, ahol az utasítások áramlanak a memória és a mikroprocesszor között.

Utasítás cache: gyorsítótár, amelyben a soronlevő utasítások egy része tárolódik azért, hogy a mikroprocesszornak ne kelljen minden utasításért a számítógép központi memóriájához fordulnia. Ez a gyors utasításelérés a feldolgozást gyorsítja, mivel az utasítások már oda vannak készítve az Előrendező egységhez, amely azokat a feldolgozáshoz megfelelő sorrendbe rendezi.

Előrendező egység: az aktuális utasításokra, vagy végrehajtandó feladatokra alapozva ez az egység dönti el, hogy mikor vegye az utasításokat az Utasítás cache-ből, és mikor a számítógép központi memóriájából. Amikor az utasítások betöltődnek, az előrendező egység számára a legfontosabb feladat hogy meggyőződjön arról, hogy az utasítások megfelelően vannak sorba állítva a Dekódoló egységbe való küldéshez.

Dekódoló egység: egyszerűen dekódolja, vagy lefordítja a komplex gépi kódú utasításokat olyan egyszerű alakra, amelyet az Aritmetikai-logikai egység (ALU) és a Regiszterek közvetlenül tudnak kezelni. Ez a feldolgozást még hatékonyabbá teszi.

Vezérlőegység: ez az egyik legfontosabb része a mikroprocesszornak, mivel a teljes folyamatért ez felel. A Dekódoló egység utasításaiból kiindulva vezérlőjeleket hoz létre, amelyek megmondják az Aritmetikai-logikai egységnek és a Regisztereknek, hogyan működjenek, mivel végezzék el a feladatot és az eredménnyel mit kezdjenek. A Vezérlőegység biztosítja, hogy minden a megfelelő helyen és időben történjen.

Aritmetikai-logikai egység (ALU): a csipben zajló feldolgozás utolsó állomása. Az ALU a csip okos része, amely végrehajtja az olyan parancsokat, mint az összeadás, kivonás, szorzás, osztás. Kezeli a logikai utasításokat is, mint pl. az OR, AND és NOT. A Vezérlőegység mondja meg az ALU-nak, mit csináljon, azután elviszi az adatokat az ALU közeli tartozékából, a Regiszterekből, hogy befejezze a feladatot.

Regiszterek: olyan kis tárolóterületek adatok számára, amelyeket az ALU használ a Vezérlőegységtől kapott feladatainak végrehajtása közben. Az adatok jöhetnek az Adat cache-ből, a központi memóriából, vagy a Vezérlőegységből és a Regiszterek speciális részeiben tárolódhatnak. Ez az ALU számára az adatok elérését gyorssá és hatékonnyá teszi.

Adat cache: Az Adat cache szorosan együtt dolgozik a feldolgozó társegységekkel, az ALU-val és a Regiszterekkel, valamint a Dekódoló egységgel. Ez az a hely, ahol a Dekódoló egységből speciálisan címkézett adat tárolódik az ALU általi későbbi felhasználás céljára és ahol a végső eredmények előkészülnek a számítógép különböző részeibe való szétosztásra.

Központi memória: Ez egy nagy adattároló a számítógépen belül, de a processzoron kívül. A Központi memória küldhet adatokat vagy utasításokat az Előrendező egységbe, amely gyakran az Utasítás cache-ben tárolja ezeket későbbi felhasználásra.

Végeztessük el a mikroprocesszorral a következő kis feladatot: 2+3= ?

1. lépés: a., A 2-es billentyű megnyomása riasztja a mikroprocesszort (ebben a túlzottan egyszerűsített mintapéldában) és jelez az Előrendező egységnek, hogy kérjen be a számítógép központi memóriájából egy, az új adatra, a 2-re vonatkozó utasítást, mivel nincs semmilyen utasítás az Utasítás cache-ben erre vonatkozóan. b., A számítógép központi memóriájából a Busz egységen keresztül bejön az új utasítás a mikroprocesszorba és tárolásra kerül az Utasítás cache-ben, ahol az X:=2 (az X Adat cache rekesz vegyen fel majd 2-es értéket) utasításként kerül tárolásra. c., Az Előrendező egység ezután kér egy másolatot az Utasítás cache-ben tárolt X:=2 utasításról és elküldi azt a Dekódoló egységnek további feldolgozásra. d., A Dekódoló egységben az X:=2 utasítás lefordítódik, vagy dekódolódik bináris kódra amely továbbítódik a Vezérlő egységhez és az Adat cache-hez, hogy megmondja nekik, mit csináljanak ezen utasítás végrehajtása érdekében. e., Mivel a Dekódoló egység megadta, hogy a 2 értéket további felhasználásra az Adat cache-ben kell tárolni, a Vezérlőegység most végrehajtja az X:=2 utasítást. Ez azt eredményezi, hogy a 2 érték az Adat cache X nevű rekeszébe íródik, ahol további felhasználásra vár.

2. lépés a., Amikor megnyomjuk a 3-as billentyűt, az Előrendező egység kéri az erre az új értékre vonatkozó utasítást a számítógép központi memóriájából és az Utasítás cache-ből. Mivel ilyen utasítást az Utasítás cache-ben nem talál, ezért az utasítást a központi memóriából várja. b., Hasonlóan az X:=2 utasításhoz, az új utasítás betöltődik a számítógép központi memóriájából és eltárolódik az Utasítás cache következő rekeszében, ahol az Y:=3 utasításként jelenik meg. c., Az Előrendező egység ezután átvesz egy másolatot az Utasítás cache-beli Y:=3 utasításról és átküldi a Dekódoló egységnek további feldolgozásra. d., A Dekódoló egységben az Y:=3 utasítás lefordítódik, vagy dekódolódik bináris kódra amely továbbítódik a Vezérlő egységhez és az Adat cache-hez, hogy megmondja nekik, mit csináljanak ezen utasítás végrehajtása érdekében. e., Mivel a Dekódoló egység megadta, hogy a 3 értéket további felhasználásra az Adat cache-ben kell tárolni, a Vezérlőegység most végrehajtja az Y:=3 utasítást. Ez azt eredményezi, hogy a 3 érték az Adat cache Y nevű rekeszébe íródik, és a 2 értékkel együtt további utasításra vár.

3. lépés a., Amikor megnyomjuk a + billentyűt, az Előrendező egység a számítógép központi memóriájából és az Utasítás cache-ből erre az új adatra (a + jelre) vonatkozó utasítást kér, mely csak a központi memóriából jöhet most. b., Amiatt, hogy ez (az összeadás) egy új utasítás, a + betöltődik a központi memóriából a mikroprocesszorba és az Utasítás cache következő szabad rekeszében tárolódik Z:=X+Y (Z legyen egyenlő X+Y) utasításként, jelezve, hogy az összeadás művelete fog majd lezajlani. c., Az Előrendező egység ezután kér egy másolatot az Utasítás cache-beli Z:=X+Y utasításról és átküldi azt a Dekódoló egységbe további feldolgozásra. d., A Dekódoló egységben a Z:=X+Y lefordítódik, vagy dekódolódik és átkerül a Vezérlő- egységbe és az Adat cache-be hogy megmondja nekik, mit csináljanak ezen utasítás végrehajtása érdekében. Az ALU szintén kap egy üzenetet, hogy hajtson majd végre egy ADD (összeadás) műveletet. e., A Vezérlőegység megfejti a kódot és utasítja az ALU-t az ADD művelet végrehajtására, amelyet az el is végez, az Adat cache-ből felküldött X és Y értékekkel. Az ALU azután letárolja az eredményként kapott 5-ös értéket a vele szoros kapcsolatban álló Regiszterek egyik rekeszében.

4. lépés a., Amikor megnyomjuk az = gombot, az Előrendező egység ismét átkutatja az Utasítás cache-t az új adatra, az = jelre vonatkozó új utasítás után, de ott nem találja. b., Az = jelre vonatkozó utasítás a központi memóriából jön a mikroprocesszorba a Busz egységen át és az Utasítás cache-ben kerül eltárolásra, mint 'Print Z' . c., Ezután az Előrendező egység egy másolatot kér az Utasítás cache-ből a 'Print Z' utasításról és átküldi azt a Dekódoló egységbe további feldolgozásra. d., A Dekódoló egységben a 'Print Z' bináris alakra fordítódik vagy dekódolódik amelyet a Vezérlőegység kap meg, hogy tudja, mi a teendője. e., Most, miután a Z értéke már korábban kiszámítódott és a Regiszterben várakozik, a Print parancsnak csak ki kell olvasnia és kiírnia a képernyőre hogy végülis megláthassuk a 2+3= eredményét. A mikroprocesszor megoldotta a feladatot számunkra.