# 1. Projektmunka

Információ Technoloógia

**Készítették: Bodó László és Jakus Máté**

## Weboldal tartalma:

A CPU (központi feldolgozóegység) más néven processzor, ill. mikroprocesszor, a számítógép „agya”, azon egysége, amely az utasítások értelmezését és végrehajtását vezérli, félvezetős kivitelezésű, összetett elektronikus áramkör. Míg a processzor fogalma már korábban ismert volt, a mikroprocesszor megjelenését csak a félvezetős technológia és az integrált áramkörök kifejlesztése tette lehetővé, az 1970-es évek elején. A processzor alatt általában mikroprocesszort értünk, régebben a processzor sok különálló áramkör volt, ám a mikroprocesszorral sikerült a legfontosabb komponenseket egyetlen szilíciumlapkára integrálni.

ALU: (Arithmetic and Logical Unit) ALU: (Arithmetic and Logical Unit – Aritmetikai és Logikai Egység). A processzor alapvető alkotórésze, ami alapvető matematikai és logikai műveleteket hajt végre.Sebessége növelhető egy koprocesszor (FPU, Floating Point Unit, lebegőpontos műveleteket végző egység) beépítésével. Az FPU korábban külön részegység volt, manapság a processzorok mindegyike beépítve tartalmazza.

AGU: (Address Generation Unit) AGU: (Address Generation Unit) - a címszámító egység, feladata a programutasításokban található címek leképezése a főtár fizikai címeire és a tárolóvédelmi hibák felismerése.

CU: (Control Unit) CU: (Control Unit a.m. vezérlőegység vagy vezérlőáramkör). Ez szervezi, ütemezi a processzor egész munkáját. Például lehívja a memóriából a soron következő utasítást, értelmezi és végrehajtatja azt, majd meghatározza a következő utasítás címét.

Regiszter (Register)

A regiszter a processzorba beépített nagyon gyors elérésű, kis méretű memória.  
A regiszterek addig (ideiglenesen) tárolják az információkat, utasításokat, amíg a processzor dolgozik velük.  
A mai gépekben 32/64 bit méretű regiszterek vannak. A processzor adatbuszai mindig akkorák, amekkora a regiszterének a mérete, így egyszerre tudja az adatot betölteni ide.  
Például egy 32 bites regisztert egy 32 bites busz kapcsol össze a RAM-mal.  
A regiszterek között nem csak adattároló elemek vannak (bár végső soron mindegyik az), hanem a processzor működéséhez elengedhetetlenül szükséges számlálók, és jelzők is. Ilyen például :

utasításszámláló, (PC=program counter, IP=instruction pointer) ami mindig a következő végrehajtandó utasítás címét,utasításregiszter (IR=instruction register), mely a memóriából kiolvasott utasítást tárolja. E kód alapján határozza meg a vezérlőegység az elvégzendő műveletetflagregiszter, amely a processzor működése közben létrejött állapotok jelzőit (igaz, vagy hamis)veremmutató (SP = Stack Pointer)és az akkumulátor, (AC) amely pedig a logikai és aritmetikai műveletek egyik operandusát, majd az utasítás végrehajtása után az eredményt tartalmazza.

Buszvezérlő

Buszvezérlő: A regisztert és más adattárolókat összekötő buszrendszert irányítja. A busz továbbítja az adatokat.

Cache

A Cache modern processzorok fontos része a cache (gyorsítótár). A cache a processzorba, vagy a processzor környezetébe integrált memória,  
ami a viszonylag lassú rendszermemória-elérést hivatott kiváltani azoknak a programrészeknek és adatoknak előzetes beolvasásával, amikre a végrehajtásnak közvetlenül szüksége lehet.  
A mai PC processzorok általában két gyorsítótárat használnak, egy kisebb (és gyorsabb) első szintű (L1) és egy nagyobb másodszintű (L2) cache-t. A gyorsítótár mérete ma már megabyte-os nagyságrendű.

A processzor tokozása

Tokozáson a processzor külső burkát, érintkezőinek kialakítását értjük.

Három elterjedt fajtája van:

[LGA-tokozás](file:///I:\!!!suli\dec5ig%20beadando\Dinamikus\CPU.html#LGA)[PGA-tokozás](file:///I:\!!!suli\dec5ig%20beadando\Dinamikus\CPU.html#PGA)[SECC-tokozás](file:///I:\!!!suli\dec5ig%20beadando\Dinamikus\CPU.html#SECC)

LGA-tokozás

az előző kialakításokkal szemben a tűsor az alaplapon helyezkedik el, míg a processzoron csak úgynevezett érintőpadok találhatóak. PGA-tokozás itt a csatlakozók a négyzet alakú tok alján helyezkednek el. Ezen belül is lehet: CPGA, azaz kerámiatok PPGA műanyag tok

SECC-tokozás a tok inkább egy kazettára hasonlít, az érintkezők (tűk) az alján vannak.

Mi a virtualizáció?

Egyszerűen megfogalmazva:

Szoftveralapú (azaz virtuális), változata egy eszköznek, infrastruktúrának:-számítógépnek-tárolóezközöknek-hálózatnak és hálózati eszköznekIlletve absztrakt, menedzselt változata egy:-platformnak vagy-alkalmazásnak.

Egyelőre az első részre, azaz az eszközvirtualizációra koncentrálunk. Ez lényegében virtuális gépek futtatását jelenti virtuális hardveren és virtuális hálózatban.

Egy virtuális környezet leírásakor az alábbi fogalmakat használjuk:

-gazda (host)-vendég (guest) / VM-hypervisor-VMM

Protection Rings

Az Intel-implementáció (x86-architektúrákra)

A gyűrűk hardver-alapú privilégiumszintek.

A UNIX 0. gyűrűje három részre bomlik

0: OS kernel – rezidens rész1: OS – dinamikusan memóriába töltött részek, eszközmeghajtók2: OS – eszközmeghajtókhoz való hozzáférés, ha több alkalmazás akar azeszközmeghajtóhoz, memóriához férni3: User space alkalmazások

A gondolat az volt, hogy a felhasználói alkalmazások 3 → 2 → 1 → 0 irányban beszélgetnek a kernellel, de léteznek direkt kernelhívások is. A sok réteg lassulást okoz.

A mai Windows és Linux a 0-t (kernel space) és a 3-at (user space, userland) használja. Azeszközmeghajtók a kernel részei, így azok 0-ban futnak.

Protection Rings

Szoftveres virtualizáció

A vendég user space alkalmazásaival nincs gond, lévén a gazda hypervisora a kódot natívan futtatja a processzoron. A problémát a vendég kernel ténykedésének elfogása okozza.

Teljes szoftveres emuláció

Rendszerint valósidejű újrafordítást követően futhat a vendég rendszer kódja. A feladat összetett, az eredmény lassú, a felhasználó boldogtalan.

Paravirtualizáció

A vendég OS módosítását igényli. A hardverhozzáférést vagy privilegizált processzormódot igénylő hívások a hypervisorhoz „térülnek el”.  
A teljesítmény jó, a felhasználó boldog – ha rendelkezésre áll módosított OS.

A való világ

A virtualizációs platformok ring0-ban futnak. A vendégrendszerből igyekeznek ring3-ban futtatni amit lehet, amit pedig nem lehet, azzal ügyeskednek.  
A jegyzetünkben tárgyalt VirtualBox nevű virtualizációs platform például a vendég ring0 részeit ring1-ben futtatja, és így amikor a vendég OS igyekezne privilegizált hozzáférést kívánó műveletet végrehajtani, a ring0-ban futó hypervisor beleszólhat.

Virtualizált gép Ring Protection

Az elkapott művelet kerülhet virtuális eszközhöz, vagy újrafordíttatik. Sok probléma megoldása nem magától értetődő, a virtualizációs platform futtathat például saját disassemblert,  
vagy például in-situ patchelést használ – akár tárolva az eredményeket.

Lévén 2015 óta gyakorlatilag minden számítógép támogatja a hardveres virtualizációt, a VirtualBox a 6.1-es verzióval (2019 végén) kivezette a szoftveres virtualizáció lehetőségét.

A virtualizáció előnyei  
jobb erőforrás-kihasználtságalacsonyabb költségekagilitáscsökkenő downtime

Mi fut a felhőben?

Talán még emlékszünk a prezentációból, egy felhős rajzra, de ha a kedvenc internetes keresőnkkel a „linux servers mostly cloud meme” képet keresünk, akkor ezer variációban mondják el nekünk a lényeget,  
de mi inkább arra keressük a választ, hogy milyen munkaterhelést miért szokás a felhőbe vinni.

Asztronauta 1: Várjunk, ezek mind Linux Szerverek?

Asztronauta 2: Mindíg is azok voltak

Amikor nem éri meg a földön maradni

elavuló adatközpontrövid ideig szükséges eszközöktervezhető költségsürgős erőforrásigénymessze lévő felhasználók

Amikor kényelmetlen a földön maradni

PaaS-okhiányzó szakrételemnem éri meg a megoldás egyéni megvalósításával küzdenikönnyebb kikényszeríteni az irányelvek betartásátskálázódás, hibatűrés

A felhő ellenében

Adatbiztonsági megfontolásokTörvényi megfelelőségLegyen kilépési tervMinél inkább kihasználjuk a felhőt (felhő-natív alkalmazások), annál bonyolultabb otthagyniAz egyszerű szerverigényeket olcsóbban megkapjuk, ha nem a nagy felhőszolgáltatókhoz mennénk

Konténerizáció

A konténerizáció olyan megoldás, amelyben nem virtuális gépeket, hanem egyes alkalmazásokat futtatunk egy gazdagépen egymástól elszigetelve.  
A konténerek kevesebb erőforrást igényelnek, mint a virtuális gépek, és karbantartásuk és lényegesen egyszerűbb, hiszen nincs újabb operációs rendszer az alkalmazás és a fizikai gép között.

Mi teszi lehetővé?

A konténerizáció 2000 körül kezdődött, csak akkor még nem hívták konténerizációnak, hanem virtualizációs környezetnek (virtualization environment, VE).  
Sok cég közreműködésének köszönhetően a fő Linux-kernel integráns részéve váltak azok az eszközök, amelyek lehetővé teszik a konténerizációt. Ezek lényegében:

névterek (önálló fájlrendszer, PID-ek, UID-k, hálózat, …)cgroup-ok (memória, CPU, disk I/O korlátozása)kernel capability-k (a root hozzáférés szétbontása elemekre)

A mai konténerizáció-megvalósítások (LXC, runC, CoreOS rkt, Mesos, Docker) mindegyike ezekre épül.  
A konténerek futtatásának natív lehetősége Windowsban a Windows Server 2016-ban jelenik meg (Docker platform használatával), hasonló elvek alapján.

A Docker és a virtuális gépek viszonya

A konténerek működés szempontjából nem jelentenek virtuális környezetet!  
Használatuk mégis sok szempontból hasonló

önálló izolált környezetet adnakönálló (bár a legtöbb esetben lecsupaszított) operációs rendszer az alapjuk

Mi a különbség? A konténerek

kisebbekgyorsabbak (kernel ugyanaz)a fájlrendszerük maradandóan nem módosítható (de ez nem is cél)

## Terv

IT ismeretek témakörből weblap készítése amelyekhez képeket csatoltunk

## Feladatok megosztása

Jakus Máté: képek keresése a tartalomhoz, css kód megírása

Bodó László: tartalom keresése, html kód megírása

## Tapasztalatok (önreflexió)

Jakus Máté: Elősször is szerintem Lacival jól lehet közösen dolgozni. Azt vettem észre magamon a feladat készítése alatt, hogy viszonylag jól ment, de mivel mindketten dolgozunk iskola mellett így sajnos nehéz volt összeegyeztetni időpontot ami mindkettőnknek jó. De ahol elakadtam ott Laci tudott segíteni ahol pedig Ő akadt el ott tudtam én segíteni neki. Viszont mivel sajnos kevés időm volt és neki is így ennyit sikerült összehoznunk ami szerintem jólsikerült és Laci szerint is.

Tartalom

[1. Projektmunka 1](#_Toc89535881)

[Weboldal tartalma: 2](#_Toc89535882)

[Terv 5](#_Toc89535883)

[Feladatok megosztása 5](#_Toc89535884)

[Tapasztalatok (önreflexió) 6](#_Toc89535885)