1. Tétel

Processzorok és Memóriák

Tartalom

[Neumann elvek: 3](#_Toc160884794)

[Neumann architektúra: 4](#_Toc160884795)

[A processzor felépítése és feladata: 5](#_Toc160884796)

[A CPU feladata: 5](#_Toc160884797)

[A CPU felépítése: 6](#_Toc160884798)

[A szuperskalár architektúrák: 7](#_Toc160884799)

[Az utasítások és a regiszterek szerepe: 8](#_Toc160884800)

[Utasítások 8](#_Toc160884801)

[Regiszterek 9](#_Toc160884802)

[A regiszterek mérete 9](#_Toc160884803)

[Túlcsordulás 9](#_Toc160884804)

[Alulcsordulás 10](#_Toc160884805)

[Számábrázolási pontosság 10](#_Toc160884806)

[Adat-, cím-, és vezérlőbusz 10](#_Toc160884807)

[Memóriák típusai 11](#_Toc160884808)

[Csak olvasható memória (ROM) 11](#_Toc160884809)

[Írható és olvasható memóriák 11](#_Toc160884810)

[RAM (Random Access Memory) 11](#_Toc160884811)

[Memóriák osztályozása 12](#_Toc160884812)

# Neumann elvek:

**Teljesen elektronikus működés:**

Régebben ez elektroncsövekkel, aztán tranzisztorokkal, manapság pedig integrált áramkörökkel valósul meg. Az elektromos működés előnyei közé tartozik a gyorsaságuk, ami felülmúlja a mechanikus komponenseket, valamint a megbízhatóságuk. Könnyebben lehet csökkenteni helyigényüket és könnyebben lehet őket skálázni és újra programozni.

**Kettes számrendszer használata:**

A kettes számrendszer használata előnyös a számítástechnikában, mivel fizikai szempontból könnyebb elektronikusan megvalósítani, például tranzisztorok vagy áramkörök állapotainak ábrázolásával. A tranzisztorok könnyen kezelhetőek az alacsony és magas feszültséggel, így lehetővé teszik az egyszerű bináris (1 és 0) állapotok reprezentálását, ami az összes számítógépes számítás és adatfeldolgozás alapja.

**Belső memória használata:**

A belső memória fontos a számítógép számára, mert lehetővé teszi gyors és hatékony adatelérést, programok folyamatos végrehajtását, valamint egyszerűbb rendszertervezést. Manapság ez szinte természetes.

**Tárolt program elve:**

A tárolt program elve segít abban, hogy a programokat dinamikusan lehessen változtatni, mert az utasítások és az adatok egyaránt ugyanabban a memóriában vannak tárolva.

Ez lehetővé teszi, hogy a programok manipulálják saját utasításaikat és adatokat futás közben, ami rugalmasabbá teszi azokat. A programok így képesek arra, hogy a tárolt adatok és utasítások alapján dinamikusan változzanak, ami fontos lehet például az alkalmazkodásban vagy a felhasználói igények kielégítésében.

**Soros utasítás-végrehajtás:**

Soros utasítások használata esetén az utasításokat egyenként hajtja végre a processzor, ami egyszerűséget és könnyű kezelhetőséget eredményez. Ezáltal a programok olvashatóbbak és érthetőbbek, könnyebben karbantarthatóak és hibák kijavíthatóak. Ezenkívül a soros utasítások lehetővé teszik a műveletek előrejelzését és optimalizálását, ami javíthatja a program teljesítményét.

**Univerzális felhasználhatóság:**

Az univerzális felhasználhatóság elve fontos volt a Neumann-elvben, mert lehetővé tette, hogy a számítógépek széles körben alkalmazhatóak legyenek különböző feladatokhoz és környezetekben.

# Neumann architektúra:

A Neumann-architektúra egy számítógép-tervezési elv, mely szerint az adatokat és az utasításokat ugyanabban a memóriában tárolják, és egy központi feldolgozó egység (CPU) végzi azok feldolgozását. Ez a modell alapja a modern számítógépek működésének, és lehetővé teszi a programok végrehajtását, valamint az adatok kezelését a számítógép által.

**Feldolgozó egység (CPU)**:

Tartalmazza az aritmetikai-logikai egységet (ALU), ami végzi az aritmetikai és logikai műveleteket, valamint a processzor-regisztereket, melyek rövidtávú adatok tárolására szolgálnak.

**Vezérlő egység (CU):**

Ide tartozik az utasítás-regiszter és a program-számláló, amelyek szabályozzák az utasítások végrehajtását és a programfolyam irányítását.

**Memória:**

Tárolja az adatokat és az utasításokat, amelyekre a feldolgozó egység hivatkozik.

**Külső tárolóeszköz:**

Ide tartozik a külső merevlemez, amely hosszú távú adattárolást tesz lehetővé.

**Bemeneti és kimenetieszközök (I/O):**

Ezek az eszközök lehetővé teszik a kommunikációt a felhasználó és a külső eszközök között, mint például a billentyűzet, az egér és a monitor.

# A processzor felépítése és feladata:

## A CPU feladata:

A processzor, vagy központi feldolgozó egység (CPU), az a számítógép komponens, amely az összes alapvető számítási és irányítási feladatot végzi. A CPU a számítógép "agyának" tekinthető, mivel felelős az utasítások értelmezéséért, a műveletek végrehajtásáért és az adatok kezeléséért.

A CPU a memóriából olvassa ki az utasításokat, majd ezeket dekódolja és végrehajtja. Az utasítások végrehajtása során a CPU elvégzi az aritmetikai, logikai, memóriakezelési és irányítási műveleteket. Emellett a CPU irányítja a számítógép működését, kommunikál a perifériákkal, kezeli az interrupciókat és a programfutást.

**A processzor fő feladatai közé tartozik:**

**Utasítások végrehajtása:**

A CPU olvassa és hajtja végre az utasításokat, amelyeket a számítógép operációs rendszere vagy a felhasználó által futtatott programok küldenek neki.

**Adatfeldolgozás:**

A processzor az adatokat feldolgozza a különböző aritmetikai és logikai műveletek végrehajtásával, mint például az összeadás, kivonás, szorzás és osztás.

**Utasításdekódolás:**

A CPU az utasításokat dekódolja és megfelelően végrehajtja azokat, figyelembe véve az utasítások és az adatok közötti kapcsolatot.

**Memóriakezelés:**

A processzor kezeli az adatokat és az utasításokat a memóriában, beleértve azok betöltését, tárolását és átvitelét.

**Irányítás és vezérlés**:

A CPU irányítja a számítógép működését, például a perifériákkal és a memóriával való kommunikációt, valamint az események és utasítások sorrendjének irányítását.

## A CPU felépítése:

**Vezérlőegység / Control unit (CU):**

A vezérlőegység irányítja és összehangolja a CPU működését.

Feladatai közé tartozik az utasításdekódolás, az utasítások végrehajtásának ütemezése és az eredmények visszajelzése.

**Aritmetikai-logikai egység / Arithmetic logic unit (ALU):**

Az ALU végzi el a különböző aritmetikai és logikai műveleteket, mint például az összeadás, kivonás, szorzás, osztás, logikai és bitenkénti műveletek.

Az ALU képes kezelni egyszerre többféle műveletet, és a mérete és a funkciói a processzor architektúrájától függően változhatnak.

**Címgeneráló egység / Address generation unit (AGU):**

Felelős a memóriacímek kiszámításáért és előkészítéséért.

**Memóriakezelő egység / Memory management unit (MMU):**

Ez az egység felelős a virtuális és fizikai memória közötti leképezésért, az elérési jogosultságok kezeléséért és az oldalazás vagy szegmentálás technikák támogatásáért a számítógépes rendszerekben.

**Regiszterek:**

A regiszterek olyan kis memóriablokkok a processzorban, amelyek rendkívül gyors hozzáférést biztosítanak az adatokhoz és az utasításokhoz.

A regiszterek közé tartoznak a munkavégző regiszterek (általában az ALU-hoz tartoznak), az utasítás regiszterek (tárolják az aktuálisan végrehajtott utasításokat) és az általános célú regiszterek (tárolják az átmeneti adatokat).

**Cache memória**:

A cache memória egy kis, nagy sebességű memória, amely közvetlenül a CPU és RAM között helyezkedik el. Fő célja az, hogy tárolja és **gyorsan** hozzáférhetővé tegye a **gyakran** használt adatokat és utasításokat a CPU számára.

Amikor a CPU adatot vagy utasítást keres, először a cache memóriában keresi meg. Ha az adat vagy az utasítás megtalálható a cache-ben, a CPU gyorsan hozzáférhet hozzá, és nem kell a lassabb RAM memóriában keresnie. Ha az adatot vagy az utasítást nem találja meg a cache-ben, a CPU kénytelen a lassabb fő memóriába fordulni.

A cache memória elrendezése általában több szintből áll. A leggyorsabb és legkisebb szintű cache, amely közvetlenül a CPU-ban található, a **L1** cache. Ezt általában követi a nagyobb és lassabb **L2** cache, majd esetenként még egy nagyobb és lassabb **L3** cache is lehet jelen a processzor körül.

A cache memória használata jelentősen növelheti a rendszer teljesítményét, mivel csökkenti az adatokhoz való hozzáférés idejét, különösen azoknál az alkalmazásoknál, amelyek gyakran ugyanazokat az adatokat használják. A cache hatékony működtetése és kezelése nagyban hozzájárul a számítógépek és más rendszerek hatékonyabb működéséhez.

**Órajel:**

Az órajel generálja az impulzusokat, amelyek szinkronizálják a CPU működését.

Az órajel impulzusai meghatározzák a CPU működési sebességét, ami kritikus a teljesítmény és a stabilitás szempontjából.

# A szuperskalár architektúrák:

A szuperskalár architektúra egy olyan processzor tervezési megközelítés, amely lehetővé teszi, hogy egy órajel alatt több utasítást hajtson végre. Ehhez több **végrehajtási egységet** és **pipelinet** alkalmaz, hogy párhuzamosan futtassa az utasításokat. Ezáltal növeli a processzor teljesítményét és kihasználtságát.

A **végrehajtási egységek** (execution units) olyan hardveres komponensek a CPU-ban, amelyek felelősek az utasítások végrehajtásáért. Ezek az egységek különböző típusú műveleteket végezhetnek el, például aritmetikai (pl. összeadás, kivonás), logikai (pl. AND, OR), ugrási (pl. ágazások kezelése), vagy memóriakezelési műveleteket (pl. adatok betöltése vagy mentése a memóriába). A szuperskalár architektúrákban több végrehajtási egység lehet jelen, és ezek párhuzamosan dolgoznak, hogy egyszerre több utasítást tudjanak végrehajtani, ami növeli a CPU teljesítményét és hatékonyságát.

Pl: ALU, MMU

A **pipelining** egy olyan technika a számítógépek tervezésében, ahol az utasítások végrehajtását több egymást követő lépésre bontják, és ezeket a lépéseket egymást követően, párhuzamosan végzik. Ennek lényege, hogy az egyes utasításokat kisebb részekre osztják, majd ezeket a részeket különböző feldolgozási egységekben, vagy lépésekben hajtják végre. Így az egyes lépések között az utasítások folyamatosan áramlanak, ami növeli a processzor kihasználtságát és sebességét. A pipelining a modern processzorokban és CPU-kban széles körben alkalmazott technika a teljesítmény növelése érdekében.

# Az utasítások és a regiszterek szerepe:

## Utasítások

Az utasítások azok a parancsok vagy műveletek, amelyeket a CPU végrehajt a számítógépes programokban. Az utasítások meghatározzák, hogy milyen műveleteket kell elvégezni a CPU-nak, például adatok mozgatása, aritmetikai műveletek végrehajtása vagy vezérlési döntések meghozatala.

Az utasításoknak általában van **operandusa** és **operátora**.

**Operandus**: Ez az adat, amelyre az utasítás hat. Az operandus lehet egy érték vagy egy memóriacím, amelyre az utasítás vonatkozik. Például az összeadás utasításnál az operandusok a két összeadandó érték lehetnek.

**Operátor**: Ez az utasítás műveletét vagy parancsát határozza meg. Ez az, amit az operandusokon végezni kell. Például az összeadás utasításnál az operátor az összeadás művelete.

Példaként vegyük az alábbi utasítást: MOV A, B

A MOV az operátor, amely arra utasítja a CPU-t, hogy mozgassa az adatot (operandusokat).

A és B az operandusok, amelyek az adatok, amelyeket az utasítás mozgat. Általában az első operandus a cél, ahova az adatot mozgatják, míg a második operandus a forrás, ahonnan az adatot mozgatják.

Az utasítások egy adott processzor **utasításkészletében** (**Instruction Set Architecture, ISA**) találhatók. Az ISA az a specifikáció vagy interfész, amely meghatározza, hogy a processzor hogyan kommunikáljon a számítógépes rendszer többi részével, és hogy milyen utasításokat ért és hajt végre.

Az utasításkészlet két fő típusra osztható:

**CISC** (Complex Instruction Set Computer): Ebben az esetben az utasításkészlet széles skáláját kínálja a processzor, több összetett utasítással, amelyek egy utasításban sokféle műveletet hajtanak végre. Ezek az utasítások több ciklusban hajthatók végre, ami hosszabb végrehajtási időt eredményezhet. Például az Intel x86 architektúra CISC típusú.

**RISC** (Reduced Instruction Set Computer): Ebben az esetben az utasításkészlet kevesebb, egyszerűbb utasítást tartalmaz, amelyek általában egyetlen ciklusban hajthatók végre. Az ilyen típusú processzorok gyorsabbak és hatékonyabbak lehetnek, mivel az egyszerűbb utasítások miatt egyszerűbb a vezérlés és a végrehajtás. Például az ARM architektúra RISC típusú.

## Regiszterek

A regiszterek olyan kis méretű, gyorsan elérhető tárolók a CPU-ban, amelyek azonnali adatelérést biztosítanak a processzornak. Ezek a regiszterek közvetlenül kapcsolódnak a CPU-hoz, és rövid távú adattárolásra szolgálnak, amelyeket az utasítások feldolgozásához használnak fel. A regiszterek segítenek a CPU-nak hatékonyan és gyorsan végrehajtani az utasításokat, mivel lehetővé teszik az azonnali adatelérést és műveletvégzést.

Általában a következő típusú regisztereket különböztetjük meg:

**Adatregiszterek:**

Az adatregiszterek olyan regiszterek, amelyek az adatok tárolására és kezelésére szolgálnak.

Ezek a regiszterek általában az aritmetikai és logikai műveletek végrehajtásához használatosak, valamint ideiglenesen tárolják az adatokat a feldolgozás során.

**Címzőregiszterek:**

A címzőregiszterek olyan regiszterek, amelyek memóriacímek tárolására és kezelésére szolgálnak.

Ezek a regiszterek segítenek az adatok elérésében a memóriában vagy a külső eszközökön keresztül, például indexelő címzés vagy memóriacímek tárolására.

**Rendszerregiszterek:**

A rendszerregiszterek olyan regiszterek, amelyek a CPU működését és vezérlését szabályozzák.

Ezek a regiszterek lehetnek olyanok, mint a programszámláló (PC), az utasításmutató regiszter (IR), az időzítők és az interrupciókhoz kapcsolódó regiszterek.

### A regiszterek mérete

A regiszterek szélessége az adott architektúrától függ, de általában a következő méretekkel találkozhatunk:

**8-bites, 16-bites, 32-bites, 64-bites**

Ezek a méretek azt jelzik, hogy hány bitet tud tárolni egy adott regiszter. A szélesebb regiszterek több adatot képesek tárolni vagy kezelni egyszerre, ami általában nagyobb feldolgozási kapacitást biztosít a processzornak.

### Túlcsordulás

A regiszterek meghatározott méretűek, ami azt jelenti, hogy csak bizonyos méretű adatokat tudnak tárolni. Ha egy aritmetikai vagy logikai művelet eredményeként az eredmény túl nagy lesz a regiszter által biztosított helyen, akkor túlcsordulás következik be.

A túlcsordulás általában az aritmetikai műveletek végrehajtása során jelentkezik, például összeadás vagy szorzás esetén. Ha az eredmény egy adott regiszter méretén túlmutat, akkor az alacsonyabb rendű bitek általában eldobódnak, és csak a regiszter méretéig maradnak meg.

### Alulcsordulás

Az alulcsordulás hasonló jelenség a túlcsorduláshoz, de az érték az adott regiszter vagy adatméret alsó határának alá esik. Míg a túlcsordulás a felső határt meghaladó értéket jelent, az alulcsordulás az alsó határt alulmúló értéket jelenti.

### Számábrázolási pontosság

A számábrázolási pontosság arra utal, hogy milyen pontosan képes a számítógép reprezentálni a valós számokat.

Általában minél több bitet használ egy adott adattípus vagy formátum, annál nagyobb tartományban és pontossággal képes reprezentálni a valós számokat.

Például, egy 32 bites lebegőpontos számábrázolás több pontossággal képes reprezentálni egy valós számot, mint egy 16 bites lebegőpontos számábrázolás.

A számábrázolási pontosság és a memóriahasználat között általában kompromisszumot kell kötni. Nagyobb pontosság általában több memóriát igényel, míg kisebb pontosságú számábrázolás kevesebb memóriát használ, de kevésbé pontosan reprezentálja a valós számokat.

# Adat-, cím-, és vezérlőbusz

A "busz" a számítógép architektúrájában egy kommunikációs csatorna vagy adatútviteli rendszer, amely összeköti a különböző hardverkomponenseket a számítógép belsejében vagy akár különböző számítógépek között is. Ez a csatorna lehetővé teszi az adatok, utasítások és vezérlőjelek átvitelét a különböző eszközök között.

A busz egy sor elektronikus vezeték vagy nyomtatott áramkör, amelyeken a digitális adatok, például bináris kódok, áramlanak. Ezek az adatok lehetnek utasítások, amelyeket a CPU végrehajt, vagy adatok, amelyek a memóriában tárolódnak vagy a perifériákból érkeznek. A legfontosabbak busz típusok:

**Adatbusz**:

Az adatbusz az adatok átvitelének fő csatornája a rendszerben. Ez a busz lehetővé teszi az adatok továbbítását a processzor, a memória, a perifériák és más rendszerkomponensek között.

**Címbusz**:

A címbusz a memóriacímek, perifériacímek és más címadatok továbbítására szolgál. Ez a busz lehetővé teszi a processzor vagy más vezérlők számára, hogy címeket küldjenek a memóriához vagy perifériákhoz annak érdekében, hogy adatokat olvassanak vagy írjanak beléjük.

**Vezérlőbusz**:

A vezérlőbusz olyan jelzéseket tartalmaz, mint például az írásengedély, az olvasási vagy írási művelet kezdetének jelzése, a megszakítások és a CPU közötti kommunikáció jelzése stb.

# Memóriák típusai

## Csak olvasható memória (ROM)

A ROM (Read-Only Memory) alapvető **rendszerprogramokat**, **firmware**-t és **állandó** **adatokat** tárol az elektronikus eszközökben. Segíti az eszközök bekapcsolását és működését, védi az adatokat és biztosítja a rendszer stabilitását.

A ROM típusai:

**Mask ROM**:

A gyártási folyamat során a chipbe beégetik az adatokat, állandóvá téve azokat.

**PROM** **(Programmable Read-Only Memory**):

A PROM (Programmable Read-Only Memory) egyszer programozható olvasható memória, amely lehetővé teszi az adatok egyszeri írását a chipbe, de azokat nem lehet utólag módosítani vagy törölni.

**EPROM** **(Erasable Programmable Read-Only Memory):**

Ez a típusú ROM az újraírhatóság képességét adja hozzá a PROM-hoz, mivel az adatokat UV-fény segítségével törölhetők és újra programozhatók.

**EEPROM** **(Electrically Erasable Programmable Read-Only Memory):**

Az EPROM-hoz hasonlóan újraírható, de az adatokat elektronikus törlés útján lehet módosítani, ami sokkal kényelmesebb, mint az UV-fény használata.

**Flash** **ROM**:

Ez egy speciális típusú EEPROM, amely nagyobb adatmennyiségek tárolására alkalmas, és gyorsabban írható és törölhető, mint az EEPROM. Általában alkalmazzák például USB-meghajtókban, SD-kártyákban, és mobiltelefonokban.

## Írható és olvasható memóriák

## 

### **RAM (Random Access Memory)**

A RAM egy olyan memóriatípus, amelyben az adatokat ideiglenesen tárolják a számítógép vagy más elektronikus eszközök működése során. A RAM gyorsan **írható** és **olvasható**, de az adatok **elvesznek**, amikor az eszköz **kikapcsolódik**.

Lehetővé teszi az adatokhoz való **véletlenszerű** **hozzáférést**, ami azt jelenti, hogy bármely adatot könnyen elérhetővé tesz az eszköz számára, anélkül hogy végig kellene olvasnia az összes előtte lévő adatot.

RAM típusok:

**SRAM (Static Random Access Memory)**:

Az SRAM gyors és stabil memória, amely flip-flop cellákból áll. Ezek a cellák **áramlás** **nélkül** megőrzik az adatokat, ami **gyors** hozzáférést tesz lehetővé. Az SRAM **drágább** és **nagyobb** **fogyasztású**, de gyorsabb és kevésbé érzékeny az áramkimaradásra.

Pl: Cache, regiszterek.

**DRAM (Dynamic Random Access Memory)**:

A DRAM az adatokat töltött **kondenzátorokban** tárolja, ami **idővel** **kifolyik**, ezért az adatokat időről időre **frissíteni** kell. Ez a frissítési folyamat időigényesebb, és csökkentheti a DRAM teljesítményét. A DRAM **több** **adatot** tárol ugyanazon a területen, ami nagyobb kapacitást és alacsonyabb költségeket eredményez, de **lassabb** hozzáférést jelent.

Pl: **Operatív memória, grafikus kártyák.**

# Memóriák osztályozása

A memóriák osztályozását számos szempont alapján lehet elvégezni. Néhány fontos osztályozási szempont a következő:

**Fizikai felépítés:**

Félvezető memória: Például az USB-meghajtók és a memóriakártyák, amelyekben az adatokat félvezetőkön tárolják.

Mágneses memória: Például merevlemezek és magnecsík alapú adattárolók, amelyek a mágneses teret használják az adatok tárolására.

Optikai memória: Ilyenek a CD-k, DVD-k és Blu-ray lemezek, amelyek optikai jelenségeket használnak az adatok tárolására.

**Elérés jellege:**

Az elérés jellege azt határozza meg, hogy az adatokhoz milyen módon lehet hozzáférni, például:

Random (véletlenszerű): Az adatokat bármely sorrendben lehet elérni, nincs szükség a sorrend betartására.

Szekvenciális**:** Az adatokat sorrendben kell elérni, a sorrend betartása szükséges.

**Sebesség**:

A memóriák sebessége az adatok olvasásának és írásának sebességét jelenti. Gyakran a memóriák sebességét a következők alapján értékelik:

Latencia: Az idő, amely szükséges az adatok eléréséhez egy adott memóriacellában.

Átviteli sebesség: Az adatok olvasásának és írásának sebessége, általában mértékegységekben, például megabájt vagy gigabájt per másodperc (MB/s vagy GB/s).