2. A CPU

2.1. 1. A leckében szereplő ismeretek

A lecke tartalma:

- A CPU feladata
- A CPU részei (vezérlő egység, aritmetikai-logikai egység, regiszterek, gyorsítótár)

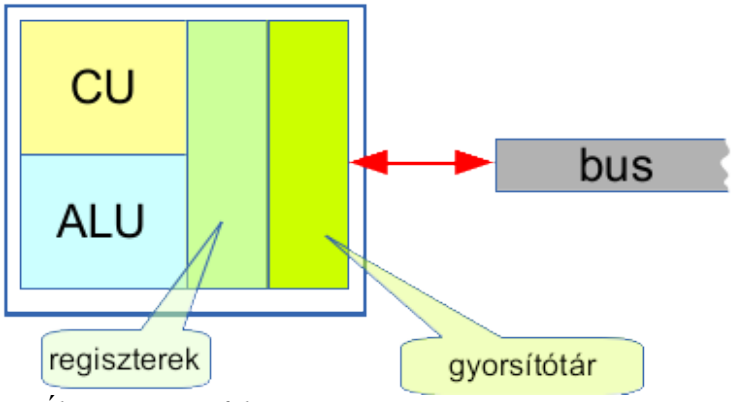
2.2. A CPU főbb felépítése

Az 5. ábra a CPU elvi felépítését mutatja a legfontosabb elemekkel. A CPU valójában a Neumann-féle számítógép elemeiből hármat tartalmaz:

CU = Control Unit: *Vezérlő egység*, amely a számítógép működését ténylegesen vezérlő áramkörökből áll.

ALU = Arithmetical–Logical Unit: *Aritmetikai-logikai egység*, amely a számításokat végzi.

Akkumulátor: már igazából nem olyan kiemelt szerepű, de a legtöbb CPU-ban megtalálható, azonban nem önálló elemként.



5. Ábra: A CPU felépítése

Lássuk az ábra alapján tehát a CPU elemeit!

2.2.1. A Vezérlő egység (Control Unit: CU)

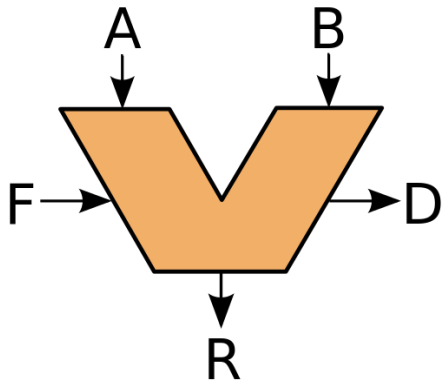
A *Vezérlő egység* a CPU azon áramköri része, amely az utasításkódban levő megfelelő bitek alapján a CPU további részeinek – egyes esetekben a CPU-n kívül található egységeknek is – megfelelő vezérlőjeleket továbbít, amely alapján azok az egységek valamilyen *műveletet hajtanak* végre. Ez lehet adat beolvasása az operatív memóriából egy regiszterbe, egy regiszter tartalmának egy memóriacímre való kiírása, vagy valamilyen matematikai művelet végrehajtása a regiszterekben található adatokkal.

2.2.2. Aritmetikai-logikai egység (ALU)

A 6. ábrán látható az ALU sematikus ábrája, amit szakkönyvekben szokás használni, ahol a CPU belső felépítését tárgyalják. Az A és B betűk a két bemenetet mutatják, míg az R a kimenetet. Ezen felül bemenő feltételek, és a művelet végrehajtása során keletkezett eseményeket (hibák például) mutatja az F és D betű. Az ábra értelmezhető egy bites ALU-ként is, amely esetben a D a keletkezett *átvitelt*, az F pedig az előző helyiértékről érkező *átvitelt* jelzi, de több bites ALU esetén is van bemenő feltétel és kimenő állapotjelző. Az F-be mindkét esetben beleértendő még a vezérlő egységtől

érkező *vezérlőjel* is, amely meghatározza, hogy az ALU-nak milyen műveletet kell A-n és B-n végrehajtania.

Maga az *Aritmetikai-logikai egység* végzi a CPU-ban a számításokat és a logikai műveleteket. A számítások a kettes számrendszernek köszönhetően csupán az *összeadásra*, *kettővel szorzásra* (azaz egy helyiértékkel balra léptetésre) – esetleg kettővel való egész osztásra, azaz egy helyiértékkel jobbra léptetésre –, valamint *-1-gyel való szorzásra* korlátozódnak, mivel az osztás kivételével minden visszavezethető ezekre. Ezen felül az ALU-nak még tudnia kell két számot összehasonlítva megmondania, hogy egyenlők-e vagy valamelyik nagyobb (ekkor a választ a D-ben kapja meg a CPU), illetve *logikai bitműveletek*et elvégezni, mint az ÉS, VAGY és NEM.



6. Ábra: Az ALU sematikus rajza (forrás: Wikipédia)

A valós számokkal való számolásra szolgáló FPU (floating point unit) valójában az ALU továbbfejlesztésének tekinthető CPU. A *grafikus segédprocesszorok* (GPU) belsejében szintén vannak ALU egységek, amelyek a grafikus számításokat végezhetik el, ráadásul egyszerre többet is, hiszen többen vannak. Ám a CPU-ban általában csak a fent felsorolt alapműveletek elvégzésére képes ALU található, esetleg több, hogy az utasítások végrehajtása párhuzamosítással gyorsítható legyen.

A CPU-k túlnyomó többségében az ALU két bemenete (A és B az ábrán) a CPU valamelyik regisztere lehet csak, ahogyan a kimenete is (R az ábrán) csak valamelyik *regiszter* lehet. Ez a megszorítás jelentősen leegyszerűsíti a processzor *áramköreit* és az *utasításkészletét* is. Az ilyen processzorok esetében tehát először mindig be kell tölteni a megfelelő regiszterekbe az adatokat, majd az eredményt szintén külön utasítással lehet a regiszterből a memóriába másolni. Egyes processzorokban az ALU kimenete csak egy meghatározott regiszter lehet, ami megfelel az eredeti Neumann-féle számítógép-felépítés *akkumulátor*ának. Más processzorok ezt a megszorítást nem tartalmazzák, amely könnyebb programozást, de valamivel bonyolultabb áramkört jelent.

2.2.3. Regiszterek

Belső *adattárolók*, zömében valódi adatok tárolására szolgálnak, de egy részük a CPU működésével kapcsolatos állapotjelzőket tárol. A regiszterek egyike valójában az *akkumulátor*, ha van egyáltalán kiemelt regiszter. Egyes CPU-k bármely regiszter tartalmát képesek a memóriába küldeni, ezeknél nincs akkumulátor szerepű regiszter, másoknál valamelyik regiszter rendelkezik azzal a kiváltsággal, hogy akkumulátorként képes viselkedni, azaz a tartalma áttölthető a memóriába.

Minden CPU-ban más regiszterek vannak, de ezek a regiszterek néhány funkcionális csoportba oszthatók:

1. **Tényleges adatregiszterek** vagy **általános regiszterek**, amelyek tartalmával az ALU számításokat végez. Ezek azok a regiszterek, ahova a CPU a memóriából az adatot betölti, hogy számításokat végezhessen rajtuk. A legtöbb CPU csak ezekkel a regiszterekkel tud számolni közvetlenül, a memória tartalmával nem. Nyilván mivel ez gyorsabban elérhető minden egyéb tárolónál, ezért ez igazából nem jelent hátrány. Ellenkezőleg: ezzel kikényszeríthető a programozóktól a sebesség-hatékony megoldás, szemben az esetleg állandóan a memória és a CPU közötti adatmozgást alkalmazó számítási módokkal.
2. **Adminisztratív regiszterek**, amelyek a CPU működését befolyásolják. Legfontosabb közülük a minden CPU-ban valamilyen megnevezéssel megtalálható **utasításszámláló** regiszter, amely mindig a következőnek betöltendő utasítás *memóriacímét* tárolja. Később a memóriá-

nál látni fogjuk, hogy egy CPU-t úgy kell legyártani, hogy áram alá helyezésekor ebben a regiszterben az a cím jelenjen meg, amelyről induláskor az első utasításkódot fogja kapni a CPU. Ide tartoznak a *címregiszterek*, mint például az *utasításszámláló*, illetve a *veremregiszter*.

3. **Állapotjelzők**: Olyan regiszterek, amelyek egy-egy bitje valamilyen esemény bekövetkezését jelzik. Ilyen jelző (*flag*) például a számítás során történt *túlcsordulás* vagy a kivonás utáni nulla eredmény, negatív vagy pozitív eredmény jelzésére szolgáló bit. Ezek az állapotjelzők egyes utasítások működését befolyásolják. Ilyen módon lehet valamilyen feltétel esetén a végrehajtást más utasításra küldeni, mint abban az esetben, ha a feltétel nem teljesült.

Egy adott CPU regiszterei mindig meghatározott számú bit tárolására képesek. Gyakorlatilag amikor azt mondjuk, hogy a CPU 32, 64 vagy 128 bites, akkor azt mondjuk meg, az *egyes regiszterek hány bit méretű adat tárolására képesek*. Ez persze meghatározza, hogy mekkora pontosságú számokkal tud számolni. De ugyanígy meghatározza a CPU által *megcímezhető memória méretét* is, hiszen azok a regiszterek is, amelyek memóriacímeket tárolnak (amelyek közül csak az egyik az utasításszámláló) ennyi bitet tárolnak. Amennyiben például a processzor 32 bites, akkor 2^{32} byte, azaz 4 Gbyte memóriát képes megcímezni. Ugyanígy egyébként a beolvasott utasításkód is egy speciális regiszterbe kerül, amelynek mérete meghatározza a lehetséges utasítások számát.

Ezt az értéket, vagyis a regiszterek méretét nevezzük **szóméret**nek, hiszen a CPU ilyen méretű *adat-szavakkal* képes dolgozni.

Minél nagyobb egy CPU szómérete, annál bonyolultabb áramköröket igényel és annál több energiát fogyaszt, éppen ezért például az egyszerű mikrovezérlőkben használt CPU-k esetén a szóméret 4, esetleg 8 bit, ami arra a célra elég. A személyi számítógépekben ma használatos szóméret már 64 bit (az ezredfordulón még csak 32 bit volt), de egyes célszámítógépekben nem ritka a 128 bites szóméret sem.

2.2.4. Gyorsítótár

A CPU fejlesztők az áramkörök miniatürizálását a sebesség növelésére, míg a memória fejlesztők a kapacitás növelésére fordítják. Ennek eredményeként a sebességkülönbség a CPU és a memória működése között egyre nő. Emiatt folyamatosan nő az a probléma, hogy *a memória lassabban küldi az adatokat* – és a programutasításokat – a CPU felé, mint amilyen ütemben annak szüksége van rá.

Emiatt jelent meg a processzorokban a **gyorsítótár**, amely segítségével a CPU *előbb beolvassa* a memóriából annak tartalmát, mint hogy szükség lenne rá. Ez persze önmagában még jelentene megoldást, hiszen attól, hogy hamarabb érkezik meg az adat, még ugyanolyan lassan érkezik meg.

Viszont a gyorsítótár valójában a memóriának egy olyan *lenyomatát* tartalmazza, amelyre várhatóan a közeljövőben a leginkább szükség lesz. Ez azt jelenti, hogy nem csak az kerül ide, amire szükség van, hanem az azt követő byte-ok is, amelyre feltehetően szükség lesz. Így a processzor akkor is adatot tölt a memóriából, amikor egyébként mással foglalkozik. Ráadásul nem kell minden adatot külön elkérni, hanem lehet olyan kérést küldeni a memória felé, hogy egy adott címtől például egy 512 byte méretű blokknyi adatot küldjön el. Így valamivel csökken a buszon az adatforgalom, tehát hamarabb megérkezhetnek az adatok.

Ráadásul megfigyelhető tény, az úgynevezett *lokalitás elve*, amely szerint ha egy adott byte-ot kért a CPU a memóriától, akkor azt követően várhatóan a környezetét fogja ezután kérni. Azok előreolvasása esetén pedig ezekre már nem kell várni, míg a memóriából megérkeznek, hanem rögtön használhatók. Ha pedig ugyanazt az adatot többször kell elérni, akkor csak az első alkalommal kell betölteni a memóriából, míg a további alkalmakkor már csak a gyorsítótárhoz kell hozzáférni. Ez jelenti az igazi gyorsulást a CPU működésében.

A gyorsítótár angol neve a *cache*. Több fajtája is van már, amelyek egyre közelebb vannak a CPU többi részéhez, de egyre kisebb méretűek, mivel egyre drágább azokat előállítani.

2.3. A CPU feladata és működése

A CPU a **Central Processing Unit**, magyarul *Központi Feldolgozó Egység* rövidítése. Ez a számítógép legfontosabb központi egysége, hiszen ez hajtja végre a programot. Maga a program nem más, mint *egyszerű utasítások sorozata*. Ezek az utasítások a másik fontos központi egységben, az *operatív memóriában* tárolódnak. A CPU innen kapja egymás után az *utasításkódokat* (kettes számrendszerben kódoltak ezek is), és sorban végrehajtja őket.

2.3.1. A CPU működési ciklusa

Minden CPU alapvetően a következő ciklust ismételgeti:

1. Az *utasításszámlálóban* található *memóriacímről* beolvas egy szót – általában 1 byte-ot, de az újabb CPU-k egyszerre ennél nagyobb egységet is olvashatnak –, amely a következő végrehajtandó utasítás *bináris kódját* tartalmazza.
2. A *vezérlő egység* (CU) ez alapján a kód alapján esetleg további szavak beolvasását kezdeményezi, ha az utasítás hosszabb. A beolvasott byte-ok számával megnöveli egyben a programszámláló értékét is.
3. A vezérlő egység *dekódolja* („értelmezi”) az utasításkódot.
4. Az *utasításkód*, amelyet a CPU beolvasott, meghatározza a CPU további „cselekedetét”. Az értelmezési lépésben a *vezérlő egység* felbontja az utasításkódot kisebb elemekre, amelyek a CPU egyéb elemeinek működését fogják a megfelelő vezérlőjeleken keresztül irányítani. Ennek eredménye lehet további adatok betöltése a memóriából valamelyik regiszterbe, valamelyik regiszter tartalmának a memória egy meghatározott címére írása, vagy egy ALU által végrehajtandó művelet kezdeményezése.
   Alternatívaként előfordulhat, hogy egy *ugró utasítás* érkezett. Ebben az esetben az utasítás végrehajtása a következő utasítás címének meghatározásából áll – amelyhez az ALU szintén szükséges lehet. Ezt a címet kell ilyenkor az *utasításszámláló* regiszterbe helyezni.
5. A ciklus kezdődik ismét az 1. pontnál.

A fenti ciklus akkor indul el, amikor a CPU áramot kap. Viszont ekkor a regiszterek értéke ismeretlen. Kivéve, ha úgy vannak legyártva, hogy mindig egy alapértéket vegyenek fel ilyenkor. Ezzel beállítható egy, az adott CPU-ra jellemző kezdőérték az utasításszámláló regiszterben. Így elérhető, hogy a CPU a számítógép bekapcsolásakor *mindig egy meghatározott memóriacímről kérje az első utasítást*. Ez a számítógép elindításakor szükséges utasításkód első utasítása lesz.

Minden CPU esetén pontosan meghatározott, hogy milyen utasításkódra mit fog csinálni. Ezt a processzor tervezésekor határozzák meg. Egyes CPU-k esetében ez az utasításkód utólag módosítható, mivel a vezérlő egység belsejében egy belső program végzi az utasítások értelmezését és egyszerűbb műveletek sorozatára bontását. Ezt a belső programot hívjuk *mikrokód*nak. A mikrokóddal működő CPU neve **CISC** = **Complex Instruction Set Computer**, míg a mikrokód nélküli, közvetlenül az egyszerű műveletek utasításaival működő CPU neve **RISC** = **Reduced Instruction Set Computer**. A CISC processzorok bonyolultabb műveletekre is rendelkeznek utasítással, azonban ezek végrehajtása hosszabb, hiszen ezeket a mikrokód *alapműveletek hosszabb sorozatára* bontja fel.

A CPU által végrehajtható utasítások halmazát hívjuk **utasításkészlet**nek. Az ilyen utasítások sorozatából áll, tehát a CPU által közvetlenül végrehajtható programot pedig *gépi kód*nak vagy *gépi kódú program*nak.

2.3.2. Az órajel

A digitális elektronikai eszközök számára az *órajel* nagyon fontos, mivel ez *szinkronizálja* az eszközök különböző elemeinek működését.

A CPU esetében az órajel határozza meg *az utasítások végrehajtási sebességét*. Minden számítógép tartalmaz egy belső órát, amely egy adott frekvencián egy szinkronizáló ütemet bocsát ki. Nyilván minél nagyobb ez a frekvencia, annál több utasítást tud a CPU végrehajtani ugyanannyi idő alatt. Ám az órajel sebessége nem állítható be akármilyen frekvenciára. Az egyes elektronikus eszközök működési sebességét a készítésükhöz használt anyag, a belső áramkörük bonyolultsága (a jel végighaladásához szükséges út hossza) egyaránt befolyásolja.

Emiatt tehát minden CPU-nak van egy frekvencia-intervalluma, amelyben működnie kell. Ha az órajel ennél magasabb, akkor képtelen időben befejezni az egyes utasítások végrehajtását. Az intervallumon belüli tényleges frekvencia ugyan változhat, de minél magasabb, a processzor annál jobban melegedni fog. Emiatt, ha túl magasra állítjuk a számítógép órajelét, azzal **a processzor élettartamát rövidítjük meg**, de akár ki is süthetjük azt!

Persze magának a CPU-nak is lehet lassabb és gyorsabb része. Ezen felül egyes utasítások végrehajtása több műveletet igényelhet. A CPU sebessége nem mondja meg pontosan, hogy másodpercenként hány utasítást fog végrehajtani, csupán azt, hogy hány alapműveletet. Ráadásul a manapság használatos processzorok már képesek párhuzamosan több utasításon is dolgozni: mialatt az első utasítás számítását végzi az ALU, a CU már a következő utasítást dekódolja, a harmadik beolvasását pedig már kezdeményezte is.

Az órajelet és a CPU sebességét is Hz (*hertz*), azaz művelet/másodperc mértékegységben mérjük (az SI váltószámokat használva, hiszen a mai CPU-k sebessége már a 2-4 GHz tartományba esik).

Tudni kell, hogy a szakemberek szerint hamarosan elérik a CPU-k a sebességnövelés felső határát, így a számítási sebesség további növeléséhez más módszereket kell keresni. Ilyen más módszer például a fentebb már említett *csővezeték*, amikor mintegy futószalagon dolgozza fel a CPU az utasításokat. Másik módszer lehet a *párhuzamosítás*, amikor egy foglaltban, vagy akár egy félvezető lapkán több CPU van egymás mellett elhelyezve, amelyek így egyidejűleg több programon, vagy egy program több, egymástól függetlenül végrehajtható ágán képesek dolgozni.

Ez a párhuzamosítás látszólag sérti a Neumann-elvet, de valójában nem, hiszen az egyes utasítások egymás utáni végrehajtási sorrendje nem változik meg, csupán *több Neumann-elvű számítógép működik egyidejűleg a processzorban*.

2.4. CPU-k csoportosítása

A korábbi fejezetekben már szerepelt néhány csoportosítás, most azokat megismételjük és kiegészítjük még néhány eddig nem szerepelt szemponttal.

2.4.1. Utasításkészlet szerinti csoportosítás

Egy processzor lehet RISC, amely esetben csak az alapvető műveletekre van utasítása, és ezekből kell a programozónak összeraknia a bonyolultabb műveleteket. A RISC processzor felépítése egyszerűbb, gyorsabb, viszont sok művelet hosszabb utasítássort igényel.

Lehet egy processzor CISC processzor is, amely esetben a processzorban egy mikrokód dekódolja az utasításokat, és felbontja alapműveletek sorozatára (mint amilyeneket a RISC is használ). Ebben az esetben összetettebb feladatokra is létrehozható utasítás. A mikrokód módosításával a pro-

cesszor utasításkészlete a legyártása után is megváltoztatható. Bonyolultabb áramköröket igényel és egyes utasítások végrehajtása – a több alapművelet miatt – több órajel-ciklust igényel.

Vannak vegyes processzorok is, amelyek az egyszerűbb műveleteket közvetlenül képesek végrehajtani RISC processzorként, míg a bonyolultabb utasításokat a mikrokód bont egyszerűbb utasítások sorozatára. Ez a processzorfajta képes a másik kettő előnyeit egyesíteni, mivel csak a bonyolultabb utasításoknál használja az összetettebb áramköröket. Erre a fajtára példa az Intel Pentium processzor is.

További változatnak tekinthetjük még az ARM processzorokat, amelyek a RISC technológia egyfajta továbbfejlesztését jelentik. Az elnevezése a kifejlesztőjére a Brit *ARM Holdings*ra utal. Jellemzője az alacsony energiaigény és kevesebb hőleadás. Emiatt főképp hordozható eszközökben, mint okostelefonok, laptopok és táblagépek használják. Maga az ARM Holdings nem gyárt processzorokat, csupán azok utasításkészletét és kialakítását fejleszti. Ezeket a *specifikációkat* felhasználva aztán más processzorgyártó cégek készítik el a tényleges processzorokat.

2.4.2. Szóméret szerinti csoportosítás

Igazából nincs értelme csoportosítani, inkább a jellemzőt kell ismerni, hogy mekkora a szómérete a processzornak. Az ezredfordulón 32 bites processzorok voltak elterjedtek a személyi számítógépekben, manapság már 64 bites processzorok vannak bennük. Játékgépekben jellemző a 128 bites processzor is.

2.4.3. Párhuzamosság szerinti típusok

A 32 bites processzorok, amelyeket személyi számítógépekben használtak, gyakorlatilag mind egymagos processzorok voltak, vagyis a processzorban egyetlen CPU volt. A szóhossz változásának első lépése volt a kétmagos processzorok megjelenése, ami gyakorlatilag egy időben jelent meg az első 64 bites PC-processzorokkal. A 64 bites processzoraink már gyakorlatilag mind több magosak.

2.4.4. Gyártó szerinti fajták

Két nagyobb processzorgyártó verseng manapság egymással, de van még néhány kisebb gyár is. A két nagy versenytárs az Intel és az AMD. Az ő processzorai mind az első IBM PC-ben használt Intel 8086-as processzor leszármazottjának tekinthető.[3] Ezek utasításkészlete nagyrészt megegyezik, bár képességeikben nyilván vannak komoly eltérések.

Más géptípusokba természetesen más processzor-gyártók is készítenek CPU-kat. Ilyen volt az IBM és a Motorola cégek közös fejlesztése a PowerPC, amely hosszú ideig például az Apple Macintosh gépcsaládjának CPU-ja volt, de például a koreai Samsung cég is rengeteg processzort gyárt, főleg persze a saját mobiltelefonjaik és táblagépeik számára.

2.5. Processzorok hűtése

Ahogy minden elektronikai eszköz, amely valamilyen feladatot végez, úgy a CPU is **hőt termel működése során**. Ha ezt a hőt nem vezetjük el, akkor a processzor előbb-utóbb túlmelegedne és tönkremenne. Ezért nagyon fontos, hogy a processzor megfelelő hűtést kapjon.

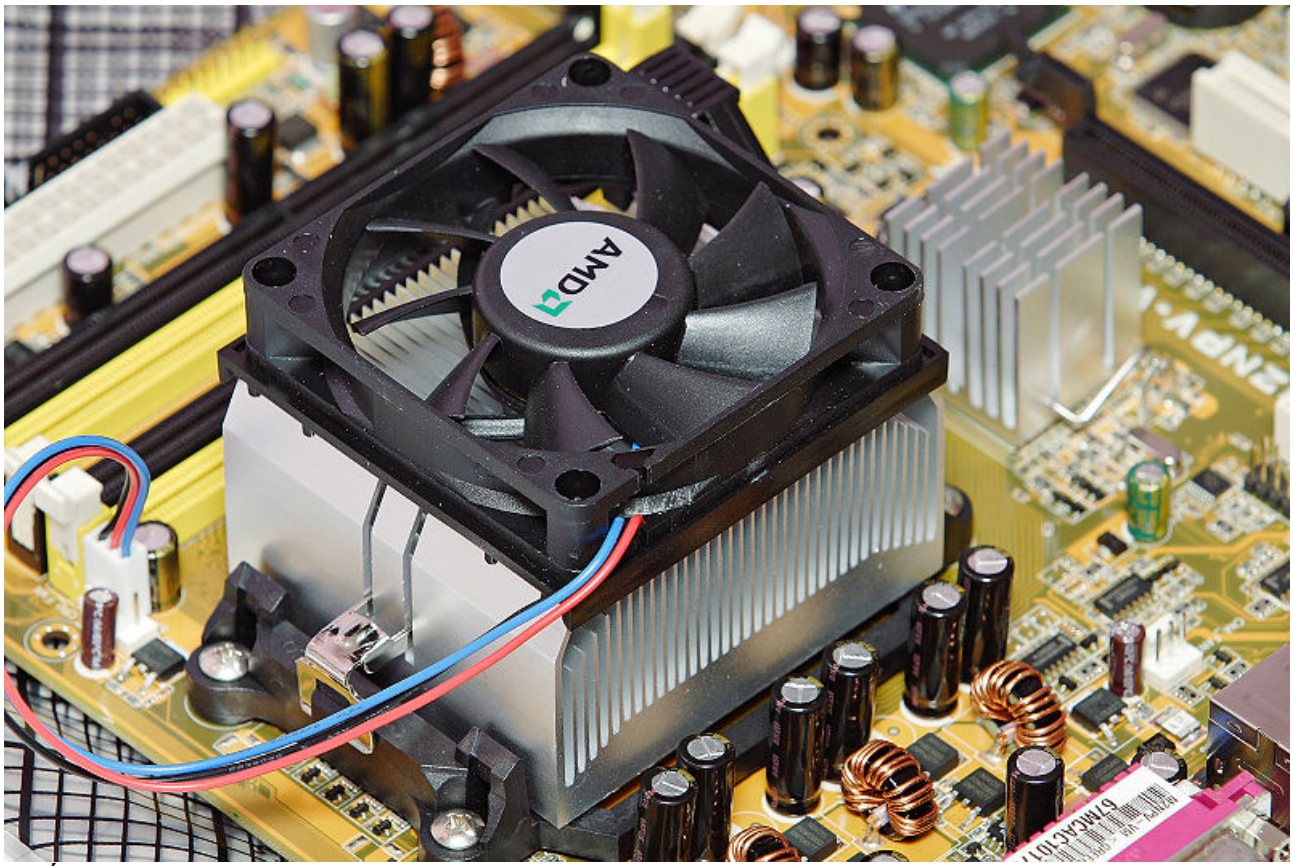
3 *Ez a megállapítás természetesen csak a PC-ként emlegetett számítógépek CPU-jai esetében igaz, mivel mind az Intel, mind az AMD gyárt más típusú processzorokat is.*

A leggyakoribb megoldás erre a 7. ábrán láthatóhoz hasonló *léghűtés*. A kép alján, az alaplapra szerelve található a CPU, amely fölé egy hozzá képest tekintélyes méretű alumínium (lehet réz is, a cél, hogy jól vezesse a hőt) „kocka” van elhelyezve. Ahogy a képen is látható, ebben mély rések találhatók. Ennek az a célja, hogy minél nagyobbá váljon a felülete. A processzor ennek a fémrácsnak adja át a teljes felületén a hőt egy termikus gél közvetítésével, ami aztán a rácsban levő levegőnek adja tovább. A levegőt a fölé helyezett (a képen fekete műanyag) ventilátor kiszívja a rácsok közül, így a helyére friss, remélhetőleg kisebb hőmérsékletű levegő tud áramlani. További lehetőség még a *folyadékhűtés* is, amit a nagyobb teljesítményű számítógépekben használhatnak. Minél több munkája van a CPU-nak, annál jobban fog melegedni, így annál fontosabb a megfelelően működő hűtése.

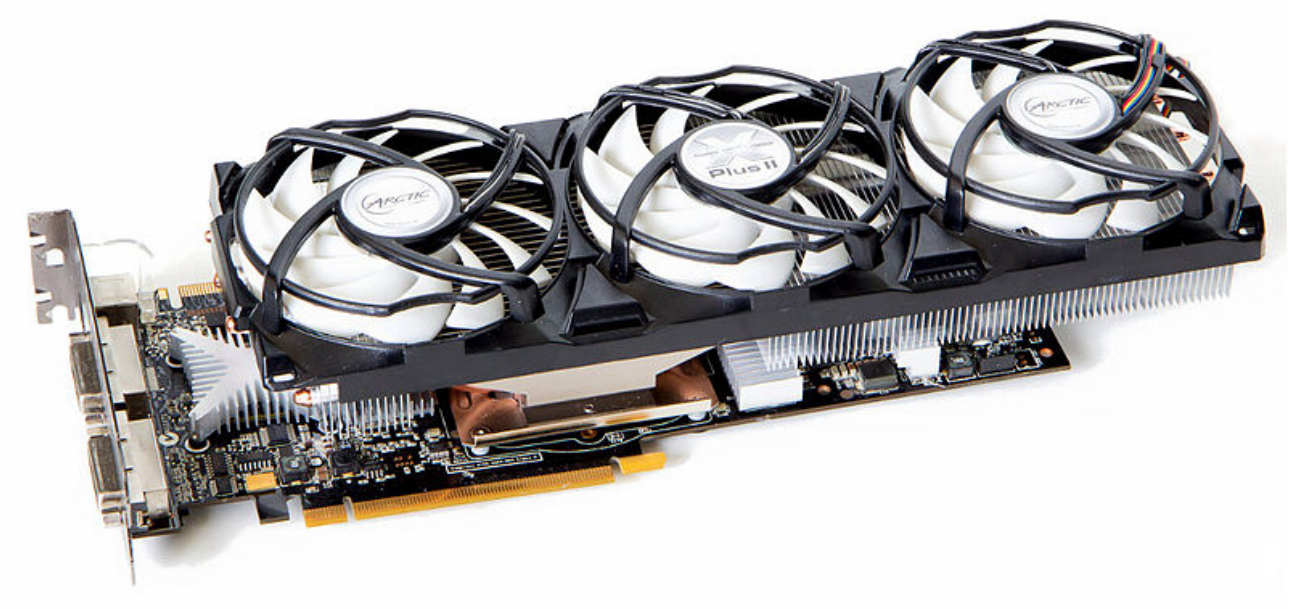
Természetesen az még önmagában nem sokat ér, ha a processzor hűtője működik. A számítógép házából hasonlóképpen szívják ki a meleg levegőt a ház ventilátorai, hogy a helyükre frissebb levegő kerülhessen a ház légnyílásain.

Ezért ha nem akarjuk tönkretenni a számítógépünket, akkor gondoskodnunk kell arról, hogy mindig jól szellőzzön, és nem szabad a ház nyílásait eltakarni. Szintén fontos, hogy óvjuk a számítógép belsejét a portól, mivel ha a hűtőbordákra por rakódik le, az a hőleadás hatékonyságát jelentősen rontja. Ha pedig a ventilátor telik meg porral, akkor a levegő mozgása lassul le, vagy akár meg is foghatja a túl sok por, így megállhat a forgása, ami egészen biztosan a processzor tönkremeneteléhez fog vezetni.

Amennyiben a számítógépünk minden látszólagos ok nélkül rendszeresen újraindul, az egyik legvalószínűbb ok, hogy a CPU nem kap elég hűtést, például mert a sok por eltömte a hűtőbordákat vagy a ventilátort. Ha ilyen esetben nem tisztítjuk meg sürgősen a hűtőrendszer elemeit, akkor hamarosan új számítógépet leszünk kénytelenek vásárolni!



7. Ábra: Egy tipikus processzor hűtés egy AMD processzor számára (forrás: Wikipédia)



8. Ábra: Három ventilátorral hűtött videokártya, amelynek célja a GPU maximális hatékonyságú hűtése (forrás: Wikipédia)

Természetesen nem a CPU az egyetlen része a számítógépnek, amely érzékeny a túlmelegedésre. A GPU kifejezetten komoly grafikai számításokat igénylő programok futtatása esetén szintén jelentősen fel tud melegedni, amit extra hűtőventilátorokkal lehet elkerülni, mint amilyen a 8. ábrán is látható.

Hasonlóan érzékenyek a túlmelegedésre a merevlemezek is, amelyek működésük során szintén jelentősen felmelegszenek, amely hőt ha nem vezeti el rendesen a hűtésük, akkor bizony az elektronikájukban található félvezetők gyorsan tönkremehetnek – magam részéről két merevlemezt is vesztettem már el így az elégtelen hűtés miatt.

Természetesen a számítógép és az operációs rendszer is képes szabályozni a CPU terhelését, ezzel megelőzve a túlmelegedést. Ennek hatékonysága géptípustól és operációs rendszertől egyaránt függ. Egyes géptípusok esetén még a ventilátor sebességét is lehet az operációs rendszeren keresztül szabályozni, hogy nagyobb melegedés esetén nagyobb legyen a hűtés hatékonysága is. A legegyszerűbb védelem a modern alaplapokban a fentebb már említett védelem, miszerint ha a CPU hőmérséklete egy kritikus értéket elér, akkor egyszerűen lekapcsol a gép. A lekapcsolásnál egy kicsit kevésbé drasztikus megoldás, amikor a CPU egyes részeit – nyilván azokat, amelyek pillanatnyilag nem dolgoznak – lekapcsolja, vagy a CPU sebessége csökken. Az ilyen megoldást – laptopokban, okos-telefonokban és táblagépekben gyakori megoldás ez – angolul a *throttling* kifejezéssel illetik.

Ahogy nő a számítási sebesség, úgy természetesen nő a hőtermelés, így a hűtés hatékonyságát is emelni kell. Ezért a nagyszámítógépek, renderfarmok, szerverfarmok és szuperszámítógépek hűtését különösen komolyan kell venni. Ezek a gépek egyébként is gyakorlatilag állandóan maximális teljesítményen dolgoznak, hiszen csak így éri meg őket üzemeltetni. Például az IBM kifejlesztett ezekhez a gépekhez olyan folyékony hűtésen alapuló rendszert, amelyben folyékony hidrogént használnak.

2.6. A leckéhez felhasznált internetes források

1. http://en.wikipedia.org/wiki/Central_processing_unit
2. http://en.wikipedia.org/wiki/Arithmetic_logic_unit
3. http://en.wikipedia.org/wiki/Processor_register
4. http://en.wikipedia.org/wiki/Control_unit
5. http://en.wikipedia.org/wiki/Computer_cooling
6. http://en.wikipedia.org/wiki/ARM_architecture
7. http://en.wikipedia.org/wiki/CPU_cache