Arhitektúrák jegyzet

Neumann arhitektúra

- AC (n+m bit) – Akkumulátor regiszter: általános célú regiszter, aritmetikai/logikai műveletekhez szüksége operandusok egyikét, valamint azok eredményét tárolja
- AR (n+m bit) – Adat regiszter: adatátvitelt biztosít a memória és a KFE belső alkotóelemei között
- PC (m bit) – Programszámláló (counter): mindenkor a következő utasítás címét tárolja
- CR (m bit) – Cím regiszter: a memória fele közvetített (utasítás vagy operandus) címet tárolja
- UR (n+m bit) – Utasítás regiszter: az utasítás dekódolása a feladata, annak felosztása utasítás kódra és operandus címre
- ALÁ – Aritmetikai - Logikai Áramkör: egy vagy kétoperandusos logikai és aritmetikai műveleteket végrehajtó egység
- VEZ – Vezérlő egység: egy többállapotú automata, mely az egész KFE működését összehangolja. Az általa létrehozott kontrolljelek (piros nyilak a tömbvázlatban) vezérlik az összes többi részegység működését.

Vezérlő egység

A vezérlőegység szerepe az utasítás előhívása a memóriából, az utasítás értelmezése, a megfelelő vezérlőjelek előállítása és kiküldése az adatfeldolgozó egységnek.

A vezérlőegység **két funkciója** azonosítható:

- Az utasítások szekvencia előállítása (z utasítások sorba állítása) (azon metódusok, amelyek kijelölik a következő végrehajtandó utasítást)
- Az utasítások értelmezése (metódusok a dekódolt utasítás vezérlőjeleinek előállítására)

A utasítés szekvencia előállítása

Az esetek nagy többségében egy program valamely utasítását az eggyel nagyobb memóriacímen levő utasítás követi. Az előbbi érvből adódóan, a legegyszerűbb megoldás az utasítássorozatok előállítására egy programszámlálónak az alkalmazása, amely az aktuális végrehajtandó utasítás dekódolása után automatikusan inkrementálódik eggyel. Ebben az esetben nem szükséges az utasításkódban meghatározni (megadni) a következő utasítást, csak speciális esetekben, mint: ugróutasítás, függvényhívás, megszakítás. Az ugrási utasítás esetében a vezérlőegység biztosítja a programszámláló párhuzamos feltöltését az utasításkódban meghatározott címmel.

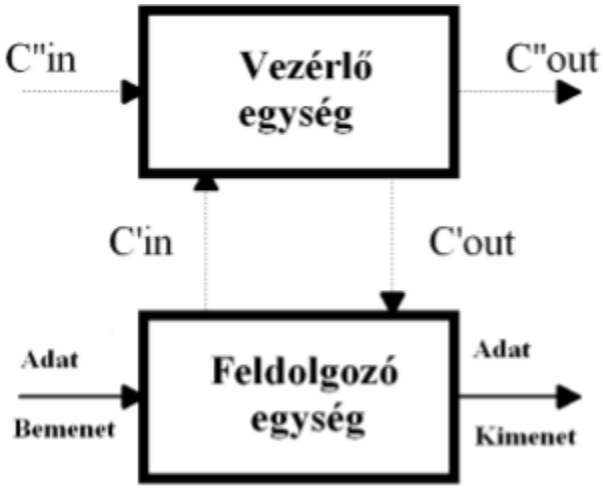
Amikor függvényhívás (szubrutin hívás) történik, a szubrutin utasítást követő utasítás címét a vezérlőegység automatikusan elmenti a verembe, a programszámlálóba betöltődik a szubrutin kezdőcíme. A visszatérés a szubrutinból (RETURN) utasítás esetében, a vezérlőegység kiolvassa a verem tetejéről az elmentett (a szubrutin után következő) utasítás címét és betölti a programszámlálóba (PC).

Az utasítások értelmezése

Négy különböző típusú jelet lehet megkülönböztetni:

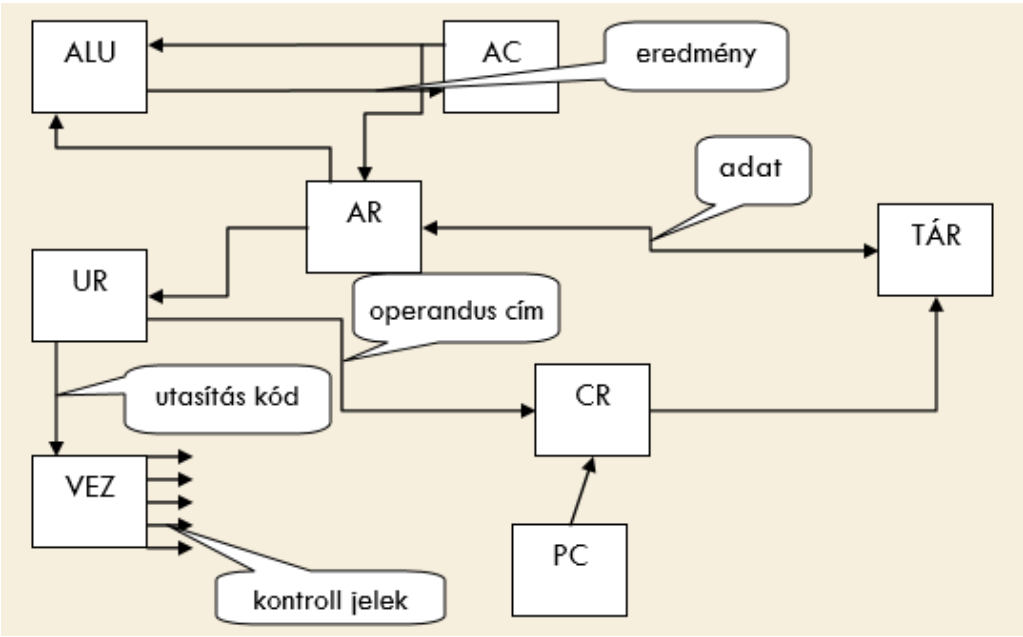
- C''in- a vezérlőegységtől vagy (ha létezik) a felügyelő egységtől bejövő jelek (START, STOP, valamint más egységekkel való szinkronizálásra használt jelek)
- C''out – más vezérlőegységek fele kimenő jelek, amelyek jelzik a központi feldolgozó egység állapotát (éppen feldolgozás alatt van egy utasítás, utasítás végrehajtása véget ért stb.)
- C'in – a feldolgozó egységtől beérkező jelek amelyek direkt módon befolyásolják a kimeneti jeleket (hibajelzések, nullával való osztás, állapotbitek, túlcsordulás
- C'out – kimenő jelek, amelyek ellenőrzik (vezérlik) az adatforgalmat és kijelölik a megfelelő műveletet a vezérlőegységben

A fő feladata a vezérlőegységnek a C'out típusú kimenő jelek (adatforgalmat vezérlő jelek) előállítása. Az imént említett jelek az alábbi egyszerű rajzon vannak szemléltetve.



A belső sínrendszer megvalósítása

- kapuk segítségével
- tri-state kimenettel rendelkező regiszterek segítségével



Függvényhívási utasítás végrehajtásához szükséges CPU alegységek és azok feladatai

Egyszintes függvényhívás esetén a CALL utasítás elmenti a **PC** aktuális értékét az N regiszterbe (**verem**), majd végrehatja az ugrást az operandus cím mezőbe kódol memória címre, ahol a meghívott függvény első utasítása helyezkedik el. A RETURN utasítás az N regiszterből kiemeli az elmentett értéket és visszatölti azt a PC-ba, tehát folytatódik a CALL előtt abbahagyott program futtatása.

Többszintes (pl. M=8 szintes) függvényhívás implementálásához szükségünk lesz M darab regiszterre (verem), így lehetővé válik az is, hogy egy bizonyos függvény más függvényt hívjon meg. Az N regiszter ez esetben egy mutató lesz a következő szabad M-edik regiszterre.

Ha a Verem értékeit nem a CPU belső regisztereiben, hanem a RAM-ban szeretnénk tárolni, annak mérete ez által jóval megnövekedhet, de az utasítások végrehajtása is bonyolódik, mivel egy újabb memória hozzáférésre lesz szükség akkor, amikor veremmel dolgozunk. A SP (Stack Pointer) a verem mutató, továbbra is a CPU egy belső regisztere lesz, fel-le számláló, párhuzamos betöltésű vagy resetelhető.

Az utasítás-formátum hatása a processzorok utasítás-készlet architektúra (ISA – Instruction Set Architecture) szintjének tervezésére

- Utasítás forma (címzés)

Utasítás kód	Operandus cím
n	m

- utasítás készlet – szükség lesz a CALL (eljárás hívás), RETURN (visszatérés az eljárás végén) és Set SP (Stack Pointer – Verem mutató beállítása) utasításokra
- Architektúra (pl. egy db. Akkumulátor központú felépítés), ki kell egészítsük egy N regiszterrel – a legegyszerűbb esetben - ahova a visszatérési címet tároljuk el (PC értékét az ugrás előtt) vagy egy m méretű kis, verem típusú memóriával, ha többszintű (m) függvényhívási lehetőséget szeretnénk.

SZÜKSÉGES VEREM TÍPUSÚ MEMÓRIA

Az operatív tárak szervezése. Lapszervezésű virtuális tár

Egy címzéstartomány, más néven virtuális memória meghatározása képezi a virtuális címzés alapját. Ez nem más, mint egy másodlagos memória (háttértár) és sokkal nagyobb, mint egy operatív tár.

A virtuális memória egyenlő méretű lapokra (oldalakra) oszlik. Az operatív memóriában egy laptáblázat található, mely minden oldalról információkat tartalmaz: az oldal címét az operatív memóriában – ha volt töltve – és az állapotbiteket (a lap hozzáférési jogait, a lap operatív memóriában való jelenlétét, a tárolt lokációk értékeinek változtatott vagy változatlan állapotát, a lapok elérésének sorrendjét).

Egy virtuális cím (a felhasználó által látható) lapcímből és egy eltolásból áll. A processzor a Memóriakezelő Egységen keresztül (MMU - Memory Managment Unit), kikeresi az lapszámnak (lapcímnek) megfelelő információkat. Ha a lap az alapmemóriában található, kiolvassa a lap kezdőcímét

és hozzáadva az eltolást, kialakítja a kért információ fizikai címét. Ugyanakkor módosítja a lapok elérhetési sorrendjét mutató állapotbiteket. Ha a lap nincs jelen az operatív tárban, akkor beolvasódik a háttértárból, kiegészítve a laptáblázatot is.

Ugyanakkor belső megszakítás (Exception) jön létre. Az is lehetséges, hogy az alapmemóriában nincs elegendő tér a lap betöltésére. Ebben az esetben a legrégebben használt lap kicserélődik a kért lappal. Ha ebben nem történtek változtatások, bemásolódik a másodlagos memóriába. A lapok mérete 1K-4K között szokott változni.

Vektorizált megszakításrendszert alkalmazó I/O adatátvitelek végrehajtásának lépései

Az I/O műveletekkel kapcsolatos alapfogalmak

I/O kapcsolatok kezelése: a perifériális eszközök kapcsolatát a processzorral az eszközvezérlőkben található regiszterek segítségével oldják meg. Minden adatforgalom, parancskiküldés illetve állapotlekérdezés ezeken keresztül valósul meg. A kapcsolatok kezelésére az I/O portok szolgálnak. Ezeket a:

- parancsregiszter
- az állapotregiszter és
- az I/O regiszter alkotja.

Megszakításos I/O esetében a cél az, hogy processzor idejét felszabadítsuk az átviteli feladatok alól. Az átvitel kezdetén a processzor jelzi az I/O eszköz számára az átvitelre vonatkozó indítási igényt. Az eszköz ezután az átvitel kezdetére alkalmas időpontot a processzor felé küldött megszakítási kérelmével jelzi.

Vektorizált megszakítások

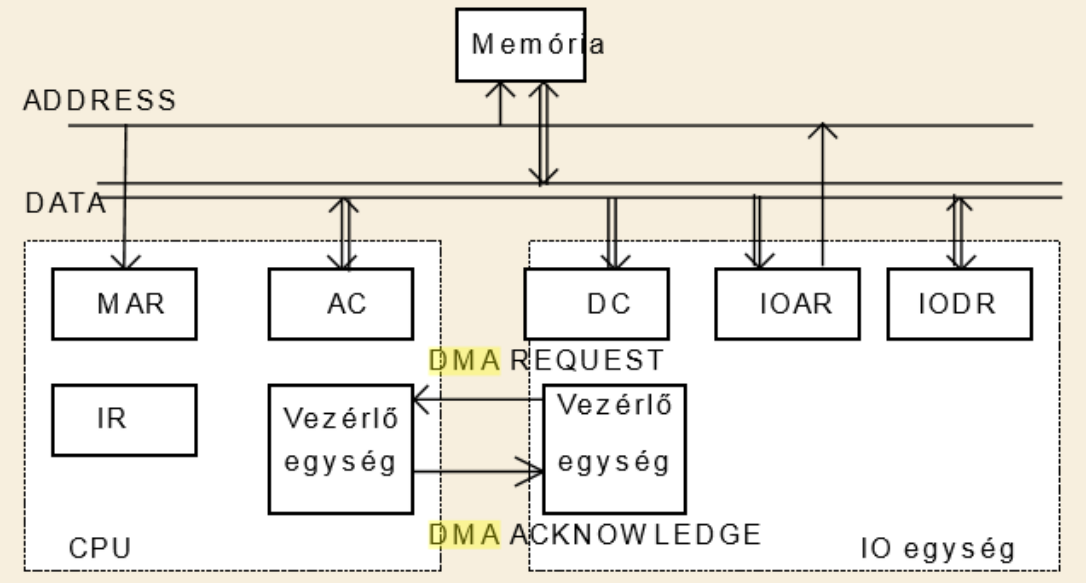
A perifériák által küldött adatmennyiség lecsökkentésének érdekében, a címküldés helyettesíthető megszakítás vektor küldésével. Ennek alapján a CPU kiolvassa a megszakítás vektor táblázatból (mely a központi memóriában foglal helyet) a lekezelő rutin címét. Ennek a módszer az az előnye, hogy a megszakításokat lekezelő rutinok lecserélhetőek. Általában több eszköz fog használni egy un. megszakítás vezérlő áramkört, mely a megszakítás vektorok és a prioritásdekódolás elvégzéséért felelős.

Közvetlen memória-hozzáférést (DMA) alkalmazó I/O adatátvitelek végrehajtásának lépései

Mikor alkalmazzuk a DMA átvitelt?

A közvetlen tárhoz fordulást (DMA)

- a nagyobb sebességű eszközök használata;
- nagyobb tömegű adat átvitele és
- adatblokkos adatátvitel esetén célszerű alkalmazni.



Blokkos átvitel (burst cycle mode)

Ha az átvitel blokkos formájú (például winchester esetén), akkor célszerű a sínt az átvitel teljes időtartamára lekötni.

Processzor oldal

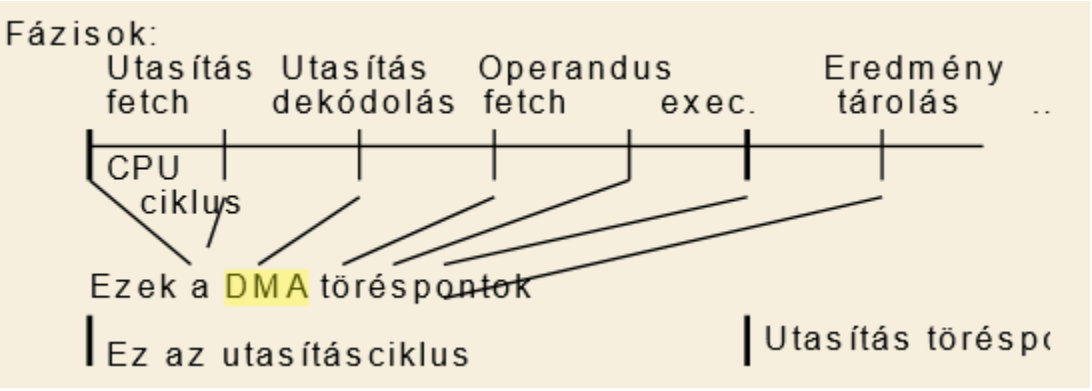
Megvizsgálja az I/O eszköz, hogy tud-e adatot küldeni, ha tud, elindítja a DMA vezérlőt, ha nem, hibavizsgálás.

DMA oldal

- A DMA előkészíti a sínt az adatátvitelre.
- A címet ls az adatot a sínre rakja
- Ha több adatot küldeni, számláló
- Megszakítás a processzor felé, hogy az adatátvitel véget ért

Ciklusloppásos átvitel (cycle stealing)

Ha az átviendő adatok nem blokkos formájúak (például gyorsnyomtató esetén), akkor csak egy-egy adat átvitelére kell igénybe venni a sínt. Ezt az eljárást nevezik ciklusloppásnak, amely tulajdonképpen a sín időosztásos használata a processzorral közösen.



ű

Átviteli folyamat

- Megnézzük, hogy foglalt-e a sín (REQUEST, ACKNOWLDGE)
- Megvalósul a közvetlen adatátvitel
- Amennyiben a DC-t nem dekrementáltuk nullára, de az I/O egység nincs ready állapotban a következő adat küldésére vagy fogadására, visszaadja a vezérlést a CPU-nak
- Ha a DC nullára dekrementálódik, akkor az IO egység ismét lemond a memóriasín vezérléséről. Ez küldhet egy interrupt jelzést a CPU számára. A CPU válaszolhat az IO egység leállításával vagy egy új IO átvitel kezdeményezésével.

Az utasítások végrehajtásának párhuzamosítása csővezeték struktúrák (pipe-line) alkalmazásával.

A lapozás és a szegmentálás megnöveli az utasítások végrehajtásának idejét, mivel több mikrolépésre lesz szükség, valamint több memória hozzáférési ciklusra.

E két módszer alkalmazása előtt, egy utasítás végrehajtása két fázisból állt:

- Kiolvasás és dekódolás;
- végrehajtás;

Virtuális címzés esetén ez a következőképpen alakul.

- címszámítás;
- kiolvasás;
- címszámítás;
- végrehajtás;

Futószalag elv – Pipeline:

Ha gyorsítani akarjuk ezt a folyamatot, akkor tervezhetünk olyan egységeket, amelyek csak egy-egy fázist tudnak elvégezni. Így előre kiszámított címek (adatok) is tárolhatók. Ezeket egy pipe-line struktúrába helyezve, annak feltöltése után, minden ciklusban végrehajtódik egy teljes utasítás.