Rajzolja fel a digitális számítógép Neumann-féle modelljének ***blokkvázlatát***, ***sorolja*** fel a modell működését meghatározó ***alapelveket!***

Mi ***különbözteti*** meg egymástól a memóriában tárolt ***utasításokat és adatokat*** egymástól?

Sorolja fel milyen tényezőktől függ egy számítógép teljesítménye!

Magyarázza el mi az előnye, illetve a hátránya az általános, illetve a speciális célú regiszter-használatnak!

Sorolja fel, és néhány szóval jellemezze az utasításrendszer tervezési szempontjait!

Írja be, hogy ***négy-címes számítógép*** utasításai esetén mit tartalmaznak az egyes ***mezők***!

OP. Kód				

Magyarázza el, hogyan lehetett ebből ***2 címes*** megoldást létrehozni!
Adja meg milyen új ***utasítástípust***, illetve milyen speciális célú *regisztert* alkalmaztak, hogy ebből ***2 címes*** megoldást hozzanak létre
Utasítástípus:.....
Regiszter.....***Feladata:***.....

Magyarázza el, hogyan lehet a négycímes utasításkészletű számítógépeknél alkalmazott megoldásból 3, 2, 1, 1.5 címes megoldást kialakítani!

Ismertesse mit jelent a többkomponensű címzés, adjon egy lehetséges példát!

Milyen többkomponensű címzési mód használható előnyösen egy tömb elemeinek az elérésére, és ez hogyan állítja elő az effektív címet?
Címzési mód:.....***Effektív cím***=.....

Mi a stack frame (verem keret) alkalmazásának előnye?

Miben és miért különbözik egy Pascal, illetve egy C programnyelv stack frame implementációja?

Milyen többkomponensű címzési módot alkalmaznak a stack frame esetén, mi ennek az előnye?

Ismertesse a CISC, illetve a RISC utasításkészlet jellemzőit!

Jellemezze néhány szóval az alábbi elven kialakított utasításrendszereket

CISC		***RISC***
	utasítás	
	Címzési mód	

Ismertesse milyen módszereket alkalmaznak a számítógép teljesítményének növelésére!

Sorolja fel az utasítás végrehajtás gyorsításának módszereit!

I***smertesse*** a RISC processzoroknál alkalmazott elveket!

Ismertesse a processzoroknál alkalmazott PIPE LINE elvét!

Milyen utasítás egymásra hatási problémák léphetnek fel a PIPE LINE alkalmazásakor?

Mi a különbség a lappangási idő, az újraindítási idő, illetve az utasítás-áteresztő képesség között?

Egy Pipe-line-t alkalmazó processzor ***három elemi*** műveletvégzőt tartalmaz. *Az* első elemi műveletvégző végrehajtási ideje ***50ns,*** a második ***30ns*** a harmadik ***50ns*** a részeredmény áttöltéshez szükséges időt elhanyagoljuk.

Hány ns alatt hajtódna végre ***egy*** utasítás pipe line nélkül? T_{ut}=.....ns

Hány ns alatt hajtódik végre ***három utasítás*** a Pipe-line működésekor? ***T***=.....ns

Mekkora az újraindítási idő a fenti esetben ? ***Tui***=.....ns

a) Rajzolja be a mellékelt ábrába ***öt utasítás pipe-line*** elven történő megvalósítását, ha feltételezzük, hogy minden utasítást ***három elemi műveletvégző*** dolgoz fel és ezek elemi művelet-végrehajtási ideje egyenlő.
b) Adjon egy lehetséges (gyakori) részművelet feladatot a három elemi műveletvégző egységnek (EMV)

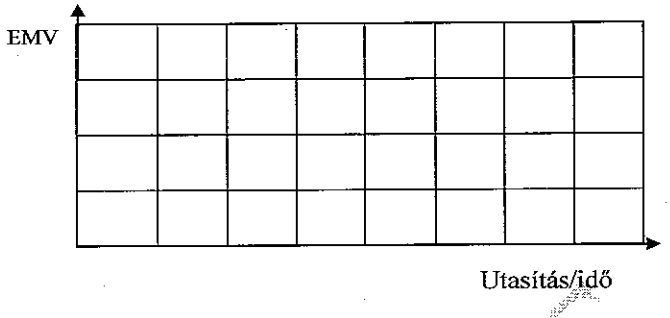
1EMV:.....

2EMV:.....

3EMV:.....

Hány ns egy utasítás végrehajtási ideje?

.....ns



c) Hány ns alatt hajtódik végre az öt utasítás ha az elemi műveletvégzők végrehajtási ideje ***egyenként 40ns*** (az áttöltéshez szükséges időt elhanyagoljuk)?

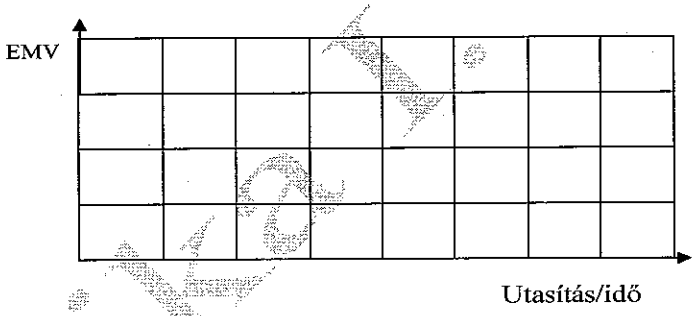
.....ns

a) Rajzolja be a mellékelt ábrába ***négy utasítás pipe-line*** elven történő megvalósítását, ha feltételezzük, hogy minden utasítást ***három elemi műveletvégző*** dolgoz fel.
b) Adjon egy lehetséges (gyakori) részművelet feladatot a három elemi műveletvégző egységnek (EMV)

1EMV:.....

2EMV:.....

3EMV



Hány ns alatt hajtódik végre ***egy***, illetve ***négy*** utasítás, ha az első elemi műveletvégző végrehajtási ideje ***45ns,*** a második ***30ns*** a harmadik ***40ns*** a részeredmény ***áttöltéshez illetve kiadásához szükséges idő 5ns***?

Egy utasítás:.....ns

Négy utasítás:.....ns

A pipe-line feldolgozás egyik problémája az utasítás egymásra hatás. ***Hogyan*** oldja meg a ***i386***-os mikroprocesszor a ***procedúrális*** utasítás egymásra hatást?
Hogyan oldja meg a ***486***-os mikroprocesszor a ***procedúrális*** utasítás egymásra hatást?
Miben különbözik ettől a ***Pentium megoldása***?

.....

Mit jelent a feldolgozási egymásrahatás, és hogyan küszöbölhető ki?

Mit jelent az adat egymásrahatás, és hogyan küszöbölhető ki?

Milyen módszerrel csökkenthető az elemi műveletvégző egységek időben egyenlőtlen terheléséből adódó probléma?

Rajzolja fel és röviden ismertesse, a processzor tehermentesítésének blokkvázlatát társprocesszorral (8086-8087), illetve DMA vezérlővel!

Magyarázza el hogyan működik időben átlapolva a processzor és a matematikai társprocesszor?

Hogyan tudja a 8086 processzor az eredményt, annak felhasználása előtt, kivárni?

Neumann-alapelvnek megfelelő számítógépekre vonatkozó alábbi kijelentések közül jelölje **x**-szel az igaz állítás(oka)t és - jellel a hamis(ak)at!

*Pontozásnál **minden jó jelölés +0,5** pont, **minden hibás jelölés -0,5 pont**, eredő >=0*

A utasítást és az adatot külön memóriában tárolja, így azok, külön sínen gyorsabb elérésűek, és a hely alapján egyértelműen azonosíthatók.	
A CPU egy –már meglévő- utasításkészlet gyorsabb implementálása (emulálása), érdekében mindig huzalozott vezérlő egységet tartalmaz.	
Az indirekt és az indexelt címzés alkalmazása előnyösen alkalmazható összetett adatszerkezetek kezelésénél.	
Négycímes utasításkészletnél nincs szükség vezérlésátadó utasításra (pl.: feltétel nélküli ugró utasításra).	
A RISC elvű processzoroknál az összetett utasítások megvalósítására gyakran mikroprogramozott vezérlőegységet alkalmaznak.	
Pipe-line alkalmazásakor az egymás után következő fokozatok (elemi műveletvégzők) között négy átmeneti tárolót kell alkalmazni.	

A be és kimenő adatokat a gyorsabb elérés érdekében az aritmetikai-logikai (ALU) egységben tárolja	
DMA vezérlő alkalmazása esetén a ki/bemenő adatok a ALU-n keresztül olvashatók be/írhatók ki a memóriába.	
Multitask-os rendszereknél a fizikai és a virtuális processzor összerendelést a ko-processzor végzi.	
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	
A stack frame (verem keret) alkalmazásakor a bemenő paraméterek helyének felszabadítása (keret lebontása) mindig a függvényt hívó program feladata.	
A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában.	

Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	
A utasítást és az adatot külön memóriában tárolja, az utasítást és az adatot a memóriában a tárolás formátuma különbözteti meg.	
A CISC elvű számítógépekben az utasítások nem azonos méretűek és rendszerint több óraciklus alatt hajthatók végre, s ez előnyös a pipe line alkalmazásánál.	
Az ENTER és a LEAVE utasítás az x86-os processzornál a verem keret (stack frame) alkalmazását támogatja.	
A RISC processzoroknál az aritmetikai utasítások operandusai vagy regiszterben, vagy a memóriában találhatók.	
A térbeli és az időbeli lokalitási elvek miatt gyorsító tárakat (cache) csak az utasítások tárolására használhatnak.	

Az utasításokat memóriában tárolja, az adatokat perifériából kapja.	
Az utasításokat és az adatokat bináris formában tárolja.	
Az utasításokat és az adatokat a memóriában csak a program algoritmusa különbözteti meg.	
Az utasításhoz és az adathoz külön-külön cím- és adatbusz tartozik.	

A utasítást és az adatot külön memóriában tárolja, az utasítást és az adatot a memóriában a tárolás formátuma különbözteti meg.	
A RISC elvű processzoroknál a gyorsabb működés elérésére huzalozott vezérlő egységet alkalmaznak.	
RISC elvű processzoroknál a két-vagy többkomponensű címzés (pl.: bázisregiszteres és indexelt) előnyösen alkalmazható összetett adatszerkezetek kezelésénél.	
Kétcímes utasításkészletnél nincs szükség vezérlésátadó utasításra(pl.: feltétel nélküli ugró utasításra).	
A CPU egy –már meglévő- utasításkészlet gyorsabb implementálása (emulálása), érdekében mikroprogramozott vezérlő egységet tartalmazhat.	
Pipe-line alkalmazásakor az egymás után következő műveletvégzők közé, a működési idő különbség miatt, átmeneti tárolót lehet alkalmazni.	

Az utasításokat és az adatokat az op. memóriában csak a program algoritmusa különbözteti meg.	
Az utasítást és az adatot külön memóriában tárolja, így az külön sínen gyorsabb elérésű, és a hely alapján egyértelműen azonosítható.	
Indexelt címzésnél az effektív címet az utasításban lévő címrészből és egy regiszter tartalmából állítja elő.	
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	
A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben bázis-relatív címzést alkalmaznak.	
A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában.	

A utasítást és az adatot az operatív memóriában a tárolás formátuma és a helye különbözteti meg..	
A RISC elvű processzoroknál a gyorsabb működés elérésére huzalozott vezérlő egységet alkalmaznak.	
Az indirekt és az indexelt címzés alkalmazása előnyösen alkalmazható összetett adatszerkezetek kezelésénél.	
Kétcímes utasításkészletnél nincs szükség feltétel nélküli ugró utasításra, illetve vezérlésátadó utasításra.	
A CPU egy –már meglévő- utasításkészlet gyorsabb implementálása (emulálása), érdekében mikroprogramozott vezérlő egységet tartalmazhat.	
Pipe-line alkalmazásakor az egymás után következő műveletvégzők közé, a működési idő különbség miatt, átmeneti tárolót lehet alkalmazni.	

Az eredeti Neumann modellnél a kombinált BE/KI meneti egység az ALU-val nem, csak a memóriával tudott közvetlenül információt cserélni.	
A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben bázis-relatív címzést alkalmaznak.	
A RISC elvű processzoroknál a gyorsabb működés elérésére mindig mikroprogramozott vezérlő egységet alkalmaznak.	
Egycímes utasításkészletnél az utasítás címrésze a következő utasítást tartalmazó memória helyét adja meg.	
A többkomponensű címzési módok előnyösen alkalmazhatók összetett adatszerkezetek kezelésére.	
Ha az utasításkészlet tartalmaz I/O utasítást akkor annak végrehajtására mindig önálló I/O processzort kell alkalmazni.	

Az egy címes utasításkészlet csak egy operandust használhat.	
A három címes utasításkészlet egyik címe az eredmény helyét jelöli ki.	
Kétkomponensű címzésnél az effektív címet az utasításban lévő címrészből és egy regiszter tartalmából állítja elő.	
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	
A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában.	
A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben indexregiszteres többkomponensű címzést alkalmaznak.	

Pipe-line alkalmazásakor az egymás után következő fokozatok (elemi műveletvégzők) között átmeneti tárolókat kell alkalmazni.	
A gyorsító tár (cache) a virtuális tár és az operatív memória közötti átvitel sebességét növeli meg.	
Memória átlapolás (memory interleave, memória beékelés) esetén egy páros című bájt és a közvetlenül utána következő páratlan című ugyanabban a memóriatömbben (bank) található.	
Egycímes utasításkészletnél az egyik operandus mindig a veremmemóriában található.	
A CISC elvű processzoroknál az összetett utasítások megvalósítására gyakran mikroprogramozott vezérlőegységet alkalmaznak.	
A tárkezelő egység (memory management unit, MMU) laphibát (page fault) jelez, ha egy felhasználói módú program az operációs rendszer adataihoz próbál hozzáférni.	

A CPU a -már meglévő- utasításkészlet gyorsabb implementálása(emulálása), érdekében gyakran mikroprogramozott vezérlő egységet tartalmaz.	
az utasításokat és adatokat a memóriában csak a program algoritmusa különbözteti meg.	
Az adatok könnyű, flexibilis kezelése érdekében sokféle, bonyolult, többkomponensű címzési módokat valósíthat meg.	
Az utasításokat és az adatokat bináris formában tárolja.	
Legalább háromcímes utasításkészletet alkalmaz.	
Az operandusok címzéséhez kevés egyszerű címzési módot alkalmaz, a memóriában található operandusokhoz csak LOAD(olvasás) és STORE(írás) típusú műveletet(címzést használ).	

Az eredeti Neumann modellnél a kombinált BE/KI meneti egység az ALU-val nem, csak a memóriával tudott közvetlenül információt cserélni.	
A RISC elvű processzoroknál a gyorsabb működés elérésére mikroprogramozott vezérlő egységet alkalmaznak.	
Pipe-line alkalmazásakor, ha az egymás után következő fokozatok (elemi feldolgozó egységek) száma 2n, a teljesítmény(utasításáteresztő képesség) maximum 4n-szeresére nőhet.	
Egycímes utasításkészletnél a cím a következő utasítás helyét adja meg.	
A többkomponensű címzési módok előnyösen alkalmazhatók összetett adatszerkezetek kezelésére.	
Az I/O processzor, az átadott kezdőcímtől, a memóriában tárolt utasításokból álló perifériakezelő algoritmust önállóan, a CPU-val párhuzamosan hajtja végre.	

Négycímes utasításkészlet esetén a program következő utasítását az éppen végrehajtott jelöli ki.	
Kétcímes utasításkészlet esetén az eredmény mindig az akkumulátorban keletkezik.	
Kétcímes utasításkészlet esetén mindig kell vezérlésátadó (ugró) utasítás.	
Bázisregiszteres memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	
A RISC processzoroknál egy ciklus alatt végrehajtható utasításokat használnak, mert ez elősegíti a pipe-line szervezést.	
A stack frame alkalmazása esetén a szubrutinok (függvények) lokális változóinak mindig a szubrutint hívó program foglal helyet.	

A három címes utasításkészlet egyik címe az eredmény helyét jelöli ki.	
Az egy címes utasításkészlet csak egy operandust használhat. .	
Kétkomponensű címzésnél az effektív címet az utasításban lévő címrészből és egy regiszter tartalmából állítja elő.	
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	
A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben bázis-relatív címzést alkalmaznak.	
A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában.	

Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	
A RISC elvű processzoroknál a gyorsabb működés elérésére decimális aritmetikát alkalmaznak.	
Pipe-line esetén az utasítás egymásra hatás egyik fajtája a procedúrális egymásra hatás. Ennek elkerülésére a Pipe-line-t mindig teljesen kiürítik és újra töltik.	
Egycímes utasításkészletnél nincs szükség vezérlésátadó utasításra (pl.:feltétel nélküli ugró utasításra).	
A DMA egység az utasításkészletben szereplő, de a CPU-ban nem megvalósított utasításokat hajtja végre a memóriában tárolt szubrutinok felhasználásával.	
A többkomponensű címzési módok hátránya, hogy nem alkalmazhatók összetett adatszerkezetek kezelésére.	

Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	
A RISC elvű processzoroknál a gyorsabb működés elérésére huzalozott vezérlő egységet alkalmaznak.	
Pipe-line alkalmazásakor, ha az egymás után következő fokozatok (elemi feldolgozó egységek) száma n, a teljesítmény maximum n-szeresére nőhet.	
Egycímes utasításkészletnél a cím az egyik operandus helyét adja meg.	
A többkomponensű címzési módok hátránya, hogy nem alkalmazhatók összetett adatszerkezetek kezelésére.	
A társprocesszor az utasításkészletben szereplő, de a CPU-ban nem megvalósított utasításokat (pl.:aritmetikai) önállóan, a CPU működésével párhuzamosan, hajthatja végre.	

Pipe-line alkalmazásakor az egymás után következő két utasítás azonos típusú rész-műveleteit (két fetch, két dekódoló, stb.) azonos időszeletben egyszerre dolgozzák fel.	
Pipe-line esetén az utasítás egymásra hatás egyik fajtája a procedúrális egymásra hatás. Ez kiküszöbölhető, ha négy utasításon belül nincs két vezérlésátadó utasítás.	
RISC elvű processzoroknál a gyorsabb működés érdekében nem használnak mikroprogramozott vezérlő egységet.	
A CISC elvű processzoroknál csak LOAD és STORE típusú adatmozgató utasításokat valósítanak meg.	
A kétcímes utasításkészletnél az egyik cím az operandusok címe a másik az eredmény címe.	
A stack frame (verem keret) a stack-ként (verem) felhasználható memóriaterület elejét és a végét jelöli ki a memóriában.	

Négycímes utasításkészlet esetén mindig szükség van akkumulátor regiszterre.	
Közvetlen operandusú (immediate) címzésnél az operandust az utasítást követő memóriahely tartalmazza.	
A veremmutató (SP) tartalma x86 mikroprocesszornál aritmetikai utasításokkal módosítható, s ez felhasználható pl.: stack frame-né a lokális változó helyének lefoglalásával.	
Az ENTER és a LEAVE utasítás a verem keret (stack frame) alkalmazását támogatja.	
A RISC processzoroknál az aritmetikai utasítások operandusai vagy regiszterben vagy a memóriában találhatók.	
A gyorsítótár (cache) tartalma hosszabb ideig eltérhet az operatív memória megfelelő rekeszeinek tartalmától.	

Négycímes utasításkészlet esetén nincs szükség vezérlés átadó (pl.:ugró) utasításra.	
Négycímes utasításkészlet esetén a program következő utasítását mindig az éppen végrehajtás alatt lévő jelöli ki.	
A RISC processzoroknál egy ciklus alatt végrehajtható utasításokat használnak, mert ez elősegíti a pipe-line szervezést.	
A CISC elvű processzoroknál csak LOAD és STORE típusú adatmozgató utasításokat valósítanak meg, mert ezek egyszerű címképzésűek és gyorsak.	
Pipe-line alkalmazásakor, ha az egymás után következő fokozatok (elemi feldolgozó egységek) száma n a teljesítmény minden esetben n- szeresére nő.	
Pipe-line esetén a feldolgozási egymásra hatás kiküszöbölhető, ha megtöbbszörözik a szükséges elemi feldolgozóegységek számát.	

Kétcímes utasításkészlet esetén a program következő utasítását az éppen végrehajtott jelöli ki.	
Kétcímes utasításkészlet esetén az egyik cím az eredmény helyét, míg a másik cím a következő utasítás helyét jelöli ki.	
Pipe-line esetén a procedúrális egymásra hatás kiküszöbölhető, ha a további műveletek feldolgozását felfüggesztik a feltételes vezérlés átadás feltételének kidolgozásáig.	
Pipe-line alkalmazásakor az egymás után következő fokozatok (elemi feldolgozó egységek) között mindig átmeneti tárolókat kell alkalmazni.	
A RISC elvű processzoroknál csak LOAD és STORE típusú adatmozgató utasításokat valósítanak meg, mert ezek egyszerű címképzésűek és gyorsak.	
A CISC elvű processzoroknál az összetett utasítások megvalósítására gyakran mikroprogramozott vezérlőegységet alkalmaznak.	

Az eredeti Neumann modellnél az utasításokat és az adatokat az operatív memóriában a tárolás helye és a tárolás formátuma különbözteti meg.	
A stack frame (verem keret) alkalmazásakor a keretet az algoritmus kódja és a hozzá tartozó adatok elválasztására használják.	
Pipe-line alkalmazásakor az egymás után következő négy utasítás azonos típusú részműveleteit (pl.: négy fetch, négy dekód., stb.) azonos időpillanatokban dolgozzák fel.	
Kétcímes utasításkészletnél a két cím a két operandus helyét adja meg.	
RISC elvű processzoroknál a gyors működés és az egyszerű címzés miatt csak LOAD és STORE típusú memória-referens utasításokat valósítanak meg.	
A CPU egy-már meglévő- utasításkészlet gyorsabb implementálása (emulálása) érdekében mikroprogramozott vezérlőegységet tartalmazhat.	

Az eredeti Neumann modellnél a kombinált BE/KI meneti egység az ALU-val nem, csak a memóriával tudott közvetlenül információt cserélni.	
Az utasításokat és az adatokat az operatív memóriában azonos formában tárolja, így azokat csak a program algoritmusa különbözteti meg.	
A RISC elvű processzoroknál a gyorsabb működés elérésére mindig mikroprogramozott vezérlő egységet alkalmaznak.	
A RISC elvű processzoroknál egyetlen ciklus alatt végrehajtható, egyforma hosszúságú utasítások kialakításával előnyösen alkalmazhatóa gyorsításra a PIPE-LINE elv.	
Egycímes utasításkészletnél a cím a következő utasítás helyét adja meg.	
A többkomponensű címzési módok előnyösen alkalmazhatók összetett adatszerkezetek kezelésére.	

Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	
A utasítást és az adatot fizikailag mindig külön álló memóriában tárolja, így az külön sínen gyorsabb elérésű, és a hely alapján egyértelműen azonosítható.	
A RISC elvű processzoroknál a gyorsabb működés érdekében nem alkalmaznak mikroprogramozott vezérlő egységet.	
A RISC elvű processzoroknál az egyszerű címzés miatt csak LOAD és STORE típusú műveletekkel érik el a memóriában lévő adatokat.	
Kétcímes utasításkészletnél nincs szükség vezérlésátadó utasításra(pl.: feltétel nélküli ugró utasításra).	
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	

A ***RISC***-alapelvnek megfelelő számítógépekre vonatkozó alábbi kijelentések közül jelölje x-szel az igaz(ak)at és - jellel a hamis(ak)at!

A CPU-nál a gyorsabb működéshez szükséges egyenlő számú órajelciklust igénylő utasítások érdekében nem alkalmazzák a pipe-line elvet.	
Az utasítást memóriában tárolja, az adatokat mikroprogram vezérlésű gyors perifériák tárolják.	
Az adatok könnyű, flexibilis kezelése érdekében bonyolult, többkomponensű címzési módokat valósít meg.	
Az utasításokat és az adatokat bináris formában a memóriában tárolja, ezér ezeket csak a program algoritmusa különbözteti meg.	

Mikroprogramozott vezérlő egységet tartalmaz.	
Az utasítást memóriában tárolja, az adatokat perifériából kapja.	
Az utasításhoz és az adathoz külön-külön cím- és adatbusz tartozik.	
Az utasításokat és az adatokat bináris formában tárolja.	

A CPU a -már meglévő- utasításkészlet gyorsabb implementálása(emulálása), valamint a gyorsabb működés érdekében mikroprogramozott vezérlő egységet tartalmaz.	
Négycímes utasításkészletet alkalmaz.	
Az utasításokat és az adatokat bináris formában tárolja.	
Az általában egyetlen ciklus alatt végrehajtható, egyforma hosszúságú utasítások kialakítása miatt, előnyösen alkalmazható a PIPE-LINE elv a gyorsításra.	
az utasításokat és az adatokat a memóriában csak a program algoritmusa különbözteti meg.	
Az operandusok címzéséhez kevés egyszerű címzési módot alkalmaz, a memóriában található operandusokhoz csak LOAD(olvasás) és STORE(írás) típusú műveletet(címzést) használ.	

Magasszintű programozási nyelveknél függvényhívás (szubrutinhívás) implementálásakor a bemenő paraméterek átadására, illetve a lokális változók tárolására a ***verem keretet*** (stack frame) alkalmazzák. Milyen címzési módot használnak a ***bemenő paraméterek*** és a ***lokális változók elérésére***, mi a módszer előnye?
Címzési mód:.....

Előnye:.....

Egy ***Pascal*** programban adott a következő ***függvény***:

function f(i,j,k:integer):integer;
var m,n,:integer;
begin .....
end;

8086-os processzornál az ***f(3,1,5)*** függvényhívás után ***rajzolja*** fel (írja be a memória rekeszek tartalmát) a ***verem keretet*** (stack frame), feltételezve, hogy hívás előtt az SP a jelölt helyre mutat (az integer 16 bites)! ***Jelölje*** be a felépített keret bázisát (BP). ***Jelölje*** be a ***stack pointer*** helyét a függvény végrehajtása alatt, valamint a visszatérés után!
Milyen címzési módot használnak a ***bemenő paraméterek*** és a ***lokális változók elérésére***, mi a módszer előnye?
Címzési mód:.....

Előnye:.....

Mem.cím(hexa)	
EFA02	
SS:SP →EFA00	
EF9FE	
EF9FC	
EF9FA	
EF9F8	
EF9F6	
EF9F4	
EF9F2	
EF9F0	
EF9EE	

Egy ***Pascal*** programban adott a következő ***függvény***:

function f(i;j:integer):integer;
var m:integer;
begin .....
end;

8086-os processzornál a fenti függvényt meghívtuk az aktuális paraméterekkel. A mellékelt ábrán a memóriának az a része látható amelybe a verem keret használatakor szokásos szerkezet (stack frame) is felépült (az integer 16 bites). Az SP a keret felépítése után a végrehajtás alatt a bejelölt helyre mutat. Adja meg milyen aktuális paraméter értékekkel hívták meg a függvényt.

i=.....H ***j***=.....H

Adja meg a ***keret aktuális*** (fügvény végrehajtása alatti) hexa értékét.

Keret:.....H

C nyelvű függvény esetén a bemenő paraméterek helyének felszabadítása a hívó program feladata.

Miért?.....

C nyelvű függvény esetén a bemenő paramétereket fordított sorrendben kell a frame-be írni.

Miért?

Mem.cím(hexa)	
EFA02	**1221H**
EFA00	**1300H**
EF9FE	**0013H**
EF9FC	**0001H**
EF9FA	**IP**
EF9F8	**BP**
EF9F6	**0A00H**
SS:SP →EF9F4 végrehajtás alatt	**0A01H**
EF9F2	**1221H**
EF9F0	**0012H**
EF9EE	**0BC2H**

Egy ***Pascal*** programban adott a következő ***függvény***:

function f(k;l:integer):integer;
var m:integer;
begin .....
end;

8086-os processzornál a fenti függvényt meghívtuk az aktuális paraméterekkel. A mellékelt ábrán a memóriának az a része látható amelybe a verem keret használatakor szokásos szerkezet (stack frame) is felépült (az integer 16 bites). Az SP a keret felépítése után a végrehajtás alatt a bejelölt helyre mutat. Adja meg,hogy a ***k*** paraméter milyen aktuális értékével hívták meg a függvényt.

k=.....H

Mi az ***m*** lokális változó pillanatnyi értéke ***m***=.....H

Mi a stackpointer értéke közvetlenül a visszatérés után?

SS:SP=.....H

Milyen címzési módot használnak a ***bemenő paraméterek*** és a ***lokális változók elérésére***, mi a módszer előnye?

Címzési mód:.....

Előnye:.....

Mem.cím(hexa)	
EFA02	**1221H**
EFA00	**1500H**
EF9FE	**0027H**
EF9FC	**0002H**
EF9FA	**IP**
EF9F8	**BP**
EF9F6	**2A00H**
SS:SP →EF9F4 végrehajtás alatt	**0A01H**
EF9F2	**1221H**
EF9F0	**0012H**
EF9EE	**0BC2H**

Ismertesse a memóriák hierarchikus felépítését!

Mit jelent a térbeli, illetve az időbeli lokalitási elv? Adjon egy-egy példát, ami alátámasztja, illetve ami sérti az előbbi elveket!

Ismertesse a lokalitási elvet , mit jelent a ***találati arány*** (Hit rate)?

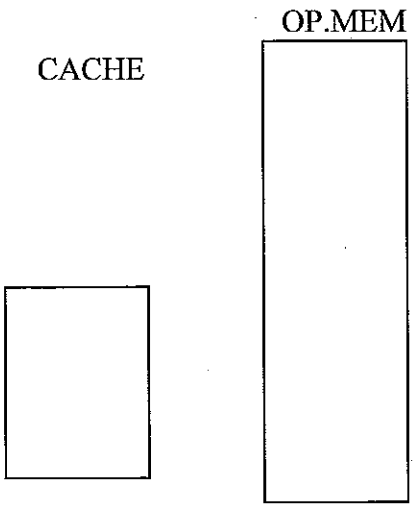
Mi a cache szerepe?

Rajzolja fel egy direkt szervezésű cache ***blokkvázlatát***!

Rajzolja fel a ***direkt*** szervezésű cache ***leképzésének*** és ***működésének blokkvázlatát!***

Leképzés OP. MEMÓRIA CÍM

Működés blokkvázlata:



Röviden ismertesse a ***leképzés elvét*** és a ***működést***

Mi a ***leképzés előnye*** és a ***hátránya, hogyan csökkentik*** az utóbbit?

Használhatnak LRU blokk-csere (block replacement policy) stratégiát?

Válasz:.....

Indoklás.....

Hogyan ***változna*** a komparátorok száma kétutas direkt leképzés esetén?

Változás:.....

Indoklás:.....

.....

.....

Rajzolja fel egy két-utas direkt leképzésű (set asszociativ) cache blokkvázlatát! Ismertesse a működést, valamint a megoldás előnyét és hátrányát!

A cache kezelés egyik kérdése, hogy mikor hozzunk be egy blokkot a cache-be. Sorolja fel és egy mondatban ismertesse az itt használatos stratégiákat (fetch policy)!
Sorolja fel és ismertesse a cache blokkcsere stratégiákat!

A cache szervezésnél ***igény szerinti, előrelátó és szelekzív blokk- behozatali*** (fetch) stratégiákat alkalmazhatnak. Adjon **legalább két** példát a **szelektív** stratégiára!

A cache kezelés egyik kérdése ***a blokkbehozalal*** (fetch policy).
Alkalmazható e az ***igény szerinti*** stratégia direrkt, illetve n utas direkt leképzésnél?
Válasz:.....

Indoklás:.....

Mit jelent az előrelátó stratégia?:.....
Milyen szelektív behozatali stratégiát valósítmeg a 486-os, illetve a Pentium processzor?

486:.....

Pentium:.....

Az operatív memória 32MB, a cache 64KB, a blokkméret 128 byte
Adja meg (***rajzolja fel***) a cím egyes részeit és ***határozza*** meg bitenkénti elhelyezkedésüket ***asszociatív***, illetve ***négyutas direkt*** (set asszociatív) leképzés esetén!

Sorolja fel és egy-egy mondatban ismertesse a ***cache blokkcsere*** stratégiákat

Lapszervezésnél hasonló blokkcsere stratégiákat használnak, mégis mi az ***alapvető*** különbség a cache-blokk-, illetve a lapcsere megvalósítása között?

Egy rendszer operatív memóriája 50ns hozzáférési idejű. A 256KB méretű ***két utas set asszociatív*** szervezésű cache memória 10ns hozzáférési idejű. A találati arány (HIT RATE) 90%.
Számítsa ki mekkora a felhasználó által látott átlagos(látszólagos) memória hozzáférési idő (a cache blokk betöltési idejét figyelmen kívül hagyjuk)!

Egy rendszer operatív memóriája 50ns hozzáférési idejű. A 512KB méretű ***négy utas set asszociatív*** szervezésű cache memória 10ns hozzáférési idejű. A találati arány (HIT RATE) 90%.
Számítsa ki mekkora a felhasználó által látott átlagos (látszólagos) memória hozzáférési idő, ha a blokk betöltési idő 100ns? Ta=.....ns

Egy **négy utas set asszocitatív** vezérlést alkalmazó gyorsító tár(cache) adatai a következők: a blokkméret 64 byte, a teljes gyorsító tár összesen 4096 blokkot tartalmaz. A vezérlőtárakban (TAG) 1 bittel jelezzük a bejegyzés érvényességét.Az operatív memória címe 32 bites.

Hány bites az offset (eltolás)?
Milyen szervezésű (szószám x bitszám) egy vezérlő (TAG) tár?
Hány TAG komparátort tartalmaz a cache?

Az operatív tárhoz egy ***4 utas*** **direkt leképzésű cache** kapcsolódik. Az operatív tár byte-os szervezésű, a cím 32 bites. A teljes cache összesen 512 blokkot tárol, egy blokk 256 byteból áll.

Hány bites a TAG a cache-ben?
Hány bites a CBA (cache block address)?
Hány TAG-komparátort tartalmaz a cache?

Az operatív tárhoz egy **teljesen asszociatív cache** kapcsolódik. Az operatív tár byteos szervezésű, a cím 24 bites. A cache 128 blokkot tárol, egy blokk 16 byteból áll.

Hány bites a TAG a cacheben?
Hány TAG-komparátort tartalmaz a cache?

Egy ***négy utas set asszocitatív*** vezérlést alkalmazó gyorsító tár (cache) adatai a következők: a blokkméret 64 byte, a teljes gyorsító tár öszesen 4096 blokkot tartalmaz. A vezérlőtárakban (TAG) 1 bittel jelezzük a bejegyzés érvényességét. Az operatív memória címe 32 bites.

Hány bites az offset (eltolás)?
Hány bites a CBA(blokkazonosító) mező?
Hány TAG komparátort tartalmaz a teljes cache?

Hogyan ***változhatna a találati arány*** ha a fenti cache-ben két utas direkt leképzést alklamaznánk? ***Indokolja*** a választ!
Változás:.....
Indoklás:.....

Komparátorok száma:.....-ról(ről) .....-ra(re)
Komparátor(ok) bitszélessége.....bitről.....bitre

Egy teljesen asszocitatív vezérlést alkalmazó gyorsítótár adatai a következők: a blokkméret 128 byte, a gyorsítótár mérete 256 Kbyte, a hozzáférési idő 10 ns. A vezérlőtárban 1 bittel jelezzük a bejegyzés érvényességét. A számítógép címsínje 32 bites, az operatív memória hozzáférési ideje 75 ns, a találati hibák aránya 8%.

Hány bites az offset (eltolás)?
Milyen szervezésű (szószám x bitszám) a vezérlőtár?
Mennyi a memória átlagos elérési ideje?

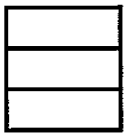
A ***fenti*** cache szervezésnél ***LRU, LFU,FIFO, RANDOM blokkcsere stratégiákat*** alkalmazhatnak.

Mit jelent az ***LFU*** stratégia?.....
.....

Használható e LRU blokkcsere sratégia ***direkt*** leképzésű cache-nél ? ***Indokolja*** a választ!.....

Az operativ tárhoz egy **direkt leképzésű cache** kapcsolódik. Az operatív tár byte-os szervezésű, a cím 32 bites. A teljes cache összesen 512 blokkot tárol, egy blokk 256 byteból áll.

Hány bites a TAG a cache-ben?
Hány bites a CBA (cache block address)?
Hány TAG-komparátort tartalmaz a cache?



Az operativ tárhoz egy ***2 utas*** **direkt leképzésű cache** kapcsolódik. Az operatív tár byte-os szervezésű, a cím 32 bites. A teljes cache összesen 1024 blokkot tárol, egy blokk 256 byteból áll.

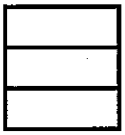
Hány bites a TAG a cache-ben?
Hány bites a CBA (cache block address)?
Hány TAG-komparátort tartalmaz a cache?

Az operatív tárhoz egy **direkt leképzésű cache** kapcsolódik. A behozatali stratégia igény szerinti, az írási stratégia write- through. Az operatív tár byte-os szervezésű, a cím 32 bites. A teljes cache összesen 512 blokkot tárol, egy blokk 256 byteból áll.

Hány TAG-komparátort tartalmaz a cache?
Hány bites a TAG komparátor?
Hány bites a CBA (cache block address)?

Az operatív tárhoz egy **direkt (közvetlen) leképzésű cache** kapcsolódik. Az operatív tár byte-os szervezésű, a cím 32 bites. A teljes cache 128KB méretű, egy blokk 64 byteból áll.

Hány bites a TAG a cache-ben?
Hány bites a CBA (cache block address)?
Hány TAG-komparátort tartalmaz a cache?

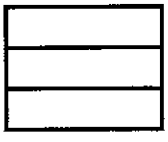


Egy **négy utas set asszocitatív** vezérlést alkalmazó gyorsító tár(cache) adatai a következők: a blokkméret 256 byte, a teljes gyorsító tár mérete 128Kbyte, a hozzáférési idő 10 ns. A vezérlőtárban(TAG) 1 bittel jelezzük a bejegyzés érvényességét és a cache vezérlő write through with write allocate írási stratégiát alkalmaz. A számítógép címsínje 32 bites, az operatív memória hozzáférési ideje 75 ns, a találati arány HR=92%)

Hány bites az offset (eltolás)?
Milyen szervezésű (szószám x bitszám) egy vezérlő (TAG) tár?
Mennyi a memória látszólagos átlagos elérési ideje?

Egy **két utas set asszocitatív** vezérlést alkalmazó gyorsító tár(cache) adatai a következők: a blokkméret 128 byte, a teljes gyorsító tár öszesen 2048 blokkot tartalmaz. A vezérlőtárakban (TAG) 1 bittel jelezzük a bejegyzés érvényességét.Az operatív memória címe 32 bites.

Hány bites az offset (eltolás)?
Milyen szervezésű (szószám x bitszám) egy vezérlő (TAG) tár?
Hány TAG komparátort tartalmaz a cache?



A leggyakrabban használt cache-írási stratégia a ***write through*** stratégia. ***Mi történik*** ennél a stratégiánál egy byte írásakor, ha a hivatkozott adat blokkja bent van a cache-ben?

.....

Használható-e ez a stratégia lapszervezésű ***virtuális tárkezelés*** esetén? ***Indokolja*** a választ!

.....

Indoklás:.....

Rajzolja fel cache blokk behozatalnál, a gyorsítás érdekében, alkalmazott memória átlapolás (memória interleaving) ***blokkvázlatát! Magyarázza el a működését!***
Adja meg kb. mekkora gyorsítást érnek el egy nyolcszoros átlapolásnál.....
Használható- e a módszer direkt leképzés esetén?.....
Indoklás:.....

Rajzolja fel a tömbkapcsolásos elven működő memória-bővités blokkvázlatát! Röviden magyarázza el a működést, sorolja fel előnyeit és hátrányait!

Milyen tárkapacitás problémát old meg a ***Tömbkapcsolásos*** memóriakezelés?

Milyen követelményt támaszt ez a megoldás a tömbkapcsolót vezérlő, illetve a megszakítás kezelő programokkal szemben?

Rajzolja fel az ***Indexelt leképzés*** elvén működő memóriakezelés blokkvázlatát*!* Röviden magyarázza el a működést, sorolja fel előnyeit és hátrányait!

Indexelt leképzés esetén a logikai cím ***16*** bites, amelyből a legmagasabb helyértékű ***4*** bit az index. A fizikai memória mérete ***1Mbyte***.Számítsa ki mekkora az indexregisztertömb mérete, ha vezérlésre 2 bitet alkalmaz! Adott egy ***4KB*** (1000H) kezdőcímre címfolytonosan lefordított ***12KB*** méretű program. A fizikai memóriában a ***512KB..520KB*** (80000H-81FFFH) és az utolsó ***8KB*** tartományban (FE000H-tól) van szabad memóriahely. Írja fel az indexregiszter-tömb ***programhoz tartozó*** regisztereinek sorszámát és a címrészének az értékeit!

Indexelt leképzés esetén a logikai cím ***16*** bites, amelyből a legmagasabb helyértékű ***4*** bit az index. A fizikai memória mérete ***1Mbyte***. Számítsa ki mekkora az indexregiszter-tömb mérete, ha vezérlésre ***2*** bitet alkalmaz! Adott egy ***1000H kezdőcímre*** (4KB) címfolytonosan lefordított ***12KB*** méretű program. A fizikai memóriában az ***512KB..520KB*** (80000H-81FFFH) és az utolsó ***4KB*** tartományban (FF000H-tól) van szabad memóriahely. Írja fel az indexregisztertömb ***–a program futásakor felhasznált–*** regisztereinek sorszámát hexadecimális tartalmát, ha a legfelső helyértéken alkalmazott vezérlő bitek értéke ***01***!

Indexregisztertömb mérete (regiszter x bitű)

.....X.....

Regisztersorszám	**Hexadecimális érték**
.....	.....
.....	.....
.....	.....

Rajzolja fel az ***Indexelt leképzés*** elvén működő memóriakezelés blokkvázlatát a következő adatok felhasználásával: ***logikai*** cím ***16*** bites, amelyből a legmagasabb helyértékű ***4*** bit az ***index, vezérlésre 2*** bitet alkalmaz. A fizikai memória mérete ***1Mbyte***!
Az ***előző*** **leképzésnél** adott egy ***1000H*** kezdőcímre címfolytonosan lefordított ***7KB*** méretű program. A fizikai memóriában az ***512KB.-516KB*** (80000H-80FFFH) és az utolsó ***4KB*** tartományban (FF000H-tól) van szabad memóriahely. Írja be az indexregisztertömb ***programhoz tartozó*** regisztere(i)nek sorszámát és a tartalmának ***hexa*** értéke(i)t, feltételezve, hogy a legmagasabb helyértéken lévő vezérlő bitek értéke ***10***!***Mit kell biztosítani,és hogyan,*** ha a regisztertömböt író és a megszakításkezelő programok logikailag és fizikailag az első 4kB-on helyezkednek el?

.....

.....

Logikai cím

15 0

OP. Memória

D7 D0

00000-00FFFH
.....
.....
80000-80FFFH
.....
FF000-FFFFFH

Egy 16 bites logikai és egyben fizikai címmel rendelkező rendszerben (pl.: 8085) indexelt leképezésű (index regiszter tömböt alkalmazó) memóriaszervezéssel 1Mbyte-ra kell bővíteni a fizikai memória méretét. Az index regiszter-tömb 32 regisztert tartalmaz. Vezérlésre (adminisztrációra) 3 bitet használunk)

Hány bites az offset (eltolás)? ☐
Hány bites a regiszter tömb egy regisztere? ☐
Hány blokkot (lapot) tartalmaz a teljes memória? ☐
Maximálisan hány blokk (lap) lehet egyszerre aktív? ☐

Egy indexelt leképezést alkalmazó számítógép címsínje 16 bites, ebből a 3 MSB az index. Az operatív memória 256 kByte méretű.
Milyen az indexregiszter tábla szervezése (szó x bit) ha vezérlési célokra 2 bit szükséges? ☐
Ha egy 1980h byte hosszú programrészt a hexa 8080h címre fordítunk le, hányas indexű indexregiszter(ek) tartamát kell beállítani? ☐
Milyen (hexa) értéket kell ebbe az indexregiszterbe irni ha a fizikai memóriában 2A000h-2BFFFh tartományban van üres hely a program számára? ☐

Egy indexelt leképezést alkalmazó számítógép logikai címe 16 bites, ebből a 4 MSB az index. Az operatív memória 512 kByte méretű.
Milyen az indexregiszter tábla szervezése (szó x bit)
ha vezérlési célokra 2 bit szükséges? ☐
Ha egy 0E80h byte hosszú programrészt a hexa 8080h címre fordítunk le, hányas indexű indexregiszter(ek) tartamát kell beállítani a program futtatásához? ☐

Milyen (hexa) értéket kell ebbe az indexregiszterbe irni ha a fizikai memóriában 7A000h-7AFFFh tartományban van üres hely a program számára és az 11 értékű vezérlési célú bitek a legmagasabb helyértéken vannak? ☐

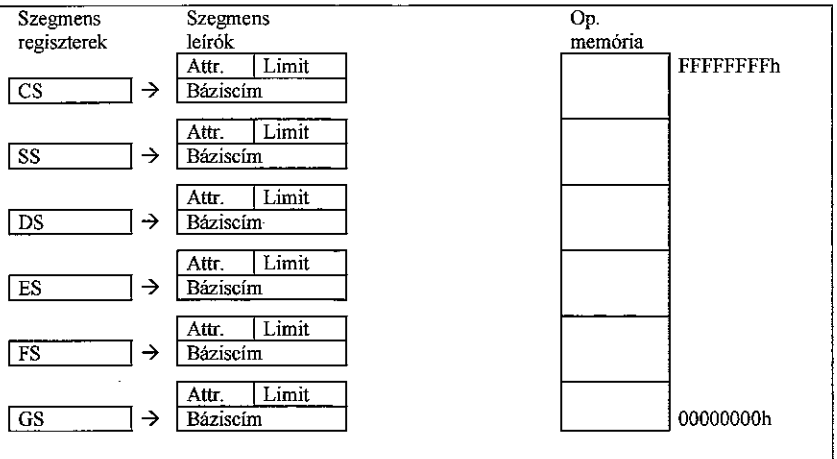
Adott egy címfolytonosan 0 kezdőcímre lefordított 48kB méretű program. A processzor 16 bites címet ad. Az indexelt leképezéssel kibővített fizikai memóriában a 0… 32 KB és a 64… 84 KB tartományban van szabad memóriahely. Az indexregiszter tábla 32db regisztert tartalmaz.
Kell-e változtatni a lefordított felhasználói programon? Indokolja a választ!

Ismertesse a lapszervezésű virtuális tárkezelés elvét, rajzolja fel a megoldás vázlatát! Hogyan csökkenthető a lapleíró táblához szükséges memória mérete? Hogyan gyorsítható a leírók elérése?

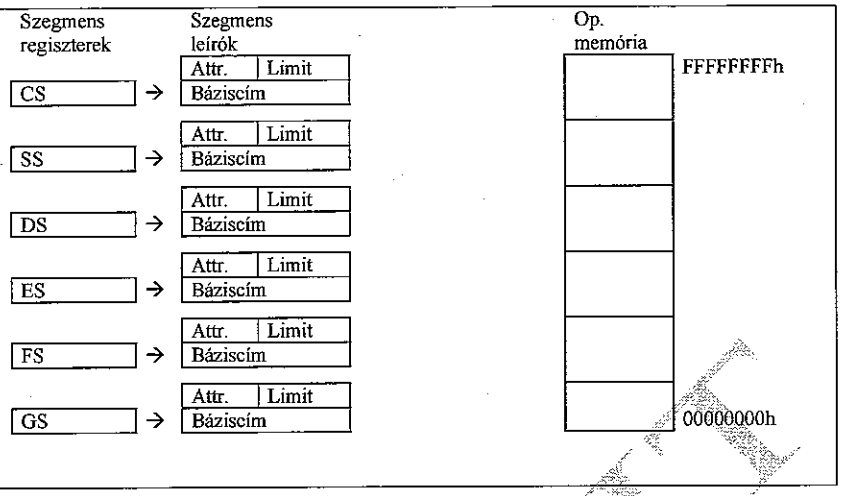
Hasonlítsa össze a lapszervezésű és a szegmensszervezésű virtuális tárkezelés előnyeit- hátrányait!

Milyen memóriaképeket biztosíthat a 386-os mikroprocesszor?

A mellékelt ábrába **rajzolja be** hová mutatnak az egyes szegmensleírók értékei **386-os** mikroprocesszor **struktúrálatlan** (lineáris) memória-modell (flat modell) esetén ha a rendszerben 4GB op. memória található.
A szegmensleírók báziscíme által mutatott memóracím értékére **folytonos vonallal** és nyíllal mutasson, a limit helyére **szaggatott vonallal** és nyíllal mutasson.



A mellékelt ábrába **rajzolja be** hová mutatnak az egyes szegmensleírók értékei **386-os** mikroprocesszor **struktúrált** memória-modell (protected multisegment modell) esetén ha a rendszerben 4GB op. memória található.
A szegmensleírók báziscíme által mutatott memóracím értékére **folytonos vonallal** és nyíllal mutasson, a limit helyére **szaggatott vonallal** és nyíllal mutasson.



Rajzolja fel a ***386***-os mikroprocesszor ***logikai-fizikai*** cím transzformációjának a ***vázlatát, protected*** módban ***bekapcsolt lapszervezés*** esetén!
Milyen ***megoldásokat*** használ a processzor a transzformáció ***végrehajtásának*** gyorsítására?
Mi a ***legfontosabb előnye*** a szegmentált lapszervezésnek az egyszerű szegmensszervezéssel szemben, és ***mi a hátránya***?
Előnye:.....
Hátránya:.....

A 386-os mikroprocesszor példáján magyarázza el, hogy miért előnyös a kétlépcsős lapszervezésű memóriakezelés az egylépcsőshöz képest! Mi a hátránya és ezt hogyan csökkenti a 386-os?

Milyen ***hátránnyal*** járna az ***egylépcsős*** laptábla alkalmazása?

Hogyan ***csökkenti*** a 386-os mikroprocesszor a ***kétlépcsős laptábla*** alkalmazásának ***hátrányát*** (indoklja a válaszokatt)?

A ***386***-os mikroprocesszor bekapcsolt lapszervezés esetén a 32bites lineáris címből kétlépcsős laptábla szervezéssel állítja elő a fizikai címet. A lapméret 4Kbyte. A laptábla könyvtár elejére a CR3regiszter tartalma mutat. Egy lapleíró bejegyzés 32bitet tartalmaz.

Minimálisan mekkora op. memória kell a kétlépcsős laptábláknak?

Mekkora lenne a laptábla mérete egylépcsős szervezéssel?

Mi a kétlépcsős szervezés ***hátránya*** az egylépcsőssel szemben?

Hány értékes bitet kell a CR3-nak tartalmaznia?

Ismertesse az i386 mikroprocesszor lapokra vonatkozó védelmét!

Egy 48 KB méretű program futtatásához ***mekkora*** méretű ***lineárisan összefüggő*** szabad memóriaterület kell ***szegmensszervezésű***, illetve ***lapszervezésű*** virtuális tárkezelés esetén? ***Indokolja*** a válaszokat (feltételezzük, hogy a laphiba lekezelése után az utasítás folytatható)!

386-os mikroprocesszornál egy ***248 KB*** méretű program ***futtatásához*** mekkora méretű ***lineárisan összefüggő*** szabad memóriaterület kell ***KI,*** illetve ***BE***-kapcsolt lapszervezésű virtuális tárkezelés esetén? ***Miért?***

386-os mikroprocesszornál, egy konkrét memóriahivatkozásnál a ***szegmensleíró felhasználói*** (3) privilégium szintet jelez. A ***laptábla könyvtár*** (directory) bejegyzés ***user***, a laptábla bejegyzés pedig ***szisztem*** beállítású. Mi történik? (***Indokolja*** a választ!)

A 386-os processzornál egy memória típusú szegmens leírója tartalmazza a szegmens kezdőcímét, a hosszát és az atribútumait. **Miben különbözik** ettől (mit tartalmaz) egy **CALL GATE** leírója?

Ismertesse, hogy multitaszkos rendszerben mit értünk virtuális processzoron!

Mit nevezünk statikus illetve dinamikus feladathozzárendelésű multiprocesszoros rendszernek?

Milyen hardver támogatás szükséges multiprocesszoros rendszereknél a kölcsönös kizárás megvalósítására?

Milyen támogatást nyújt az I386-os mikroprocesszor a taszkváltáshoz? Ismertesse az ehhez szükséges adatstruktúrát (TSS) és a taszkváltás lépéseit!

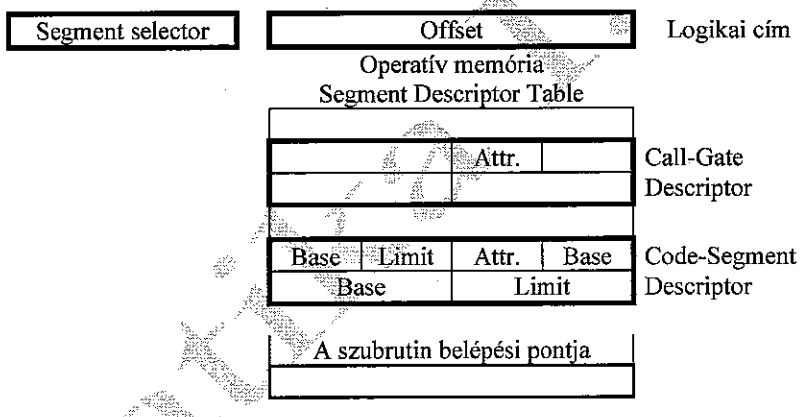
Sorolja fel az i386-os mikroprocesszor privilégium szintjeit, és a hozzájuk kapcsolódó elérési szabályokat!

Ismertesse az i386 lapokra vonatkozó védelmét!

Mi a GATE(kapu) szerepe, hogyan biztosítja ezt?

Rajzolja fel a CALL GATE működését jellemző szegmenselérés vázlatát!

A mellékelt ábrába **írja be**, hogy mit tartalmaz a Call-Gate Descriptor és **jelölje** (rajzolja fel), **hogyan történik** a Gate-n keresztüli belépési pont meghatározása (feltételezze, hogy az ábrán cél leíróját adtuk meg).

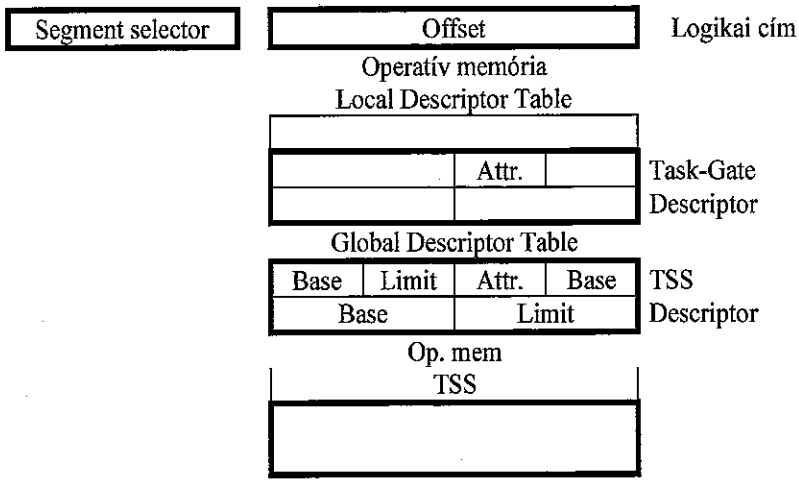


Mi a Task GATE szerepe, hogyan biztosítja ezt?

A 386-os processzornál egy memória típusú szegmens leírója tartalmazza a szegmens kezdőcímét, a hosszát és az atribútumait. **Miben különbözik** ettől (mit tartalmaz) egy **TASK GATE** leírója?

Rajzolja fel a TASK GATE működését bemutató vázlatot!

A mellékelt ábrába **írja be**, hogy mit tartalmaz a ***TASK-Gate*** Descriptor és **jelölje** (rajzolja fel), **hogyan történik** a Gate-n keresztül a task-váltás (TSS elérése).



Multitaszkos rendszernél az alábbi kijelentések közül jelölje x-szel az igaz(ak)at és – jellel a hamis(ak)at!

Minden taszkhoz egy külön fizikai processzor tartozik.	
A processzor taszkváltáskor hardveresen menti a taszk teljes állapotát.	
A i386/486 processzornál minden taszkhoz külön lokális szegmensleíró tábla tartozhat.	
A virtuális és fizikai processzorok időbeli összerendelését külön hardver egység végzi.	
Multitaszkos rendszernél a taszkok állapotát leíró információt taszkváltáskor menteni, illetve cserélni kell.	
Multiprocesszoros rendszereknél, statikus feladat hozzárendelés esetén, egy adott feladatot mindig ugyanaz a processzor lát el.	

Minden fizikai processzorhoz több virtuális processzor tartozhat.	
Minden fizikai processzorhoz külön taszk tartozik.	
A processzor taszkváltáskor részben hardveresen, részben szoftveresen menti a taszk teljes állapotát.	
A i386/486 processzornál minden taszk csak a Globális szegmensleíró táblát használhatja.	
A virtuális és fizikai processzorok időbeli összerendelését ütemező algoritmus végzi.	
A virtuális és fizikai processzorok összerendelését egy taszk állapot leíró felhasználásával végzik.	
Multiprocesszoros rendszereknél, dinamikus feladat hozzárendelésnél esetén egy processzor minden funkciót (feladatot) elláthat.	
A i386/486 processzornál a taszkok korlátlanul újra hívhatják magukat (rekurzív, reentrant).	

Minden taszkhoz egy külön fizikai processzor tartozik.		A virtuális és fizikai processzorok időbeli összerendelését külön hardver egység végzi.	
A processzor taszkváltáskor hardveresen menti a taszk teljes állapotá.t		A i386/486 processzornál a taszkok nem működhetnek re-entrant.	
A i386/486 processzornál minden taszkhoz külön lokális szegmensleíró tábla tartozhat.		A i386/486 processzornál taszkváltáskor a cache tartalmát érvényteleníteni kell.	

Minden taszk-hoz egy külön fizikai processzor tartozik, a taszk-fizikai processzor összerendelést hardver úton az arbiter egység végzi.	
A processzor taszkváltáskor hardveresen menti a taszk teljes állapotát, amelyet a hardveres TSS egység végez.	
A virtuális és fizikai processzorok időbeli összerendelését külön hardver egység végzi.	
A i386/486 processzornál minden taszkhoz külön lokális szegmensleíró tábla és lapleíró könyvtár (directory) tartozhat.	
A i386/486 processzornál a taszkok nem lehetnek újra belépők (re-entrant) mivel minden task-hoz csak egy leíró (TSS) tartozhat.	
A i386/486 processzornál minden új taszk TSS tartalmának betöltése után a CR3 tartalmának megváltozása miatt a lapszervezés TLB-jét realizáló cache tartalmát érvényteleníteni kell.	

Minden taszk-hoz egy külön fizikai processzor tartozik.	
A processzor taszk-váltáskor részben hardveresen részben szoftveresen menti a taszk teljes állapotát leíró információt.	
A taszk-váltást az operációs rendszer (pl.: időosztásos elven működő) ütemezője is kezdeményezheti.	
A i386/486 processzornál minden taszk csak a Globális szegmensleíró táblát (GT) használhatja.	
A i386/486 processzornál minden taszk-hoz annyi taszk állapot leíró szegmens(TSS) tartozhat, ahány helyről hívják ezért önmagát is többször meghívhatja.	
Multiprocesszoros rendszereknél, statikus feladat hozzárendelés esetén, egy adott feladatot (az ezt megoldó taszk-ot) mindig ugyanaz a fizikai processzor végzi el.	
A i386/486 processzornál minden taszk-nak saját lap könyvtár (page directory) táblája lehet, ezért taszk-váltáskor ezt is cserélni kell.	

Minden fizikai processzorhoz több virtuális processzor tartozhat.	
A virtuális és fizikai processzorok időbeli összerendelését külön hardver egység végzi.	
A taszkok állapotát leíró információt taszkváltáskor menteni, illetve cserélni kell.	
A i386/486 processzornál minden taszkhoz csak egy taszk állapot leíró szegmens(TSS) tartozhat, ezért önmagát nem hívhatja meg.	
A i386/486 processzornál minden taszkhoz külön lokális szegmensleíró tábla (LT) tartozhat.	
Dinamikus feladat-hozzárendelésű multiprocesszoros rendszereknél, egy fizikai processzor minden funkciót (feladatot, illetve az ezeket realizáló taszk-ok futtatását) elláthatja.	

Minden fizikai processzorhoz több virtuális processzor tartozhat.	
A virtuális és fizikai processzorok időbeli összerendelését külön hardver egység végzi.	
A task-ok állapotát leíró információt task-váltáskor menteni, illetve cserélni kell.	
A i386/486 processzornál minden task-nak saját lap könyvtár (page directory) táblája lehet, ezért task-váltáskor ezt is cserélni kell.	
A i386/486 processzornál minden task-hoz külön lokális szegmensleíró tábla (LT) tartozhat, ezért task-váltáskor az ennek a leírójára e mutató regiszter (LDTR) tartalmát is cserélni kell.	
Multiprocesszoros rendszereknél, dinamikus feladat hozzárendelésnél esetén egy fizikai processzor minden funkciót (feladatot, illetve az ezeket realizáló task-ok futtatását)) elláthatja.	

A 386/486 processzorban mely információkat tartalmazza a **szegmensleíró** bejegyzés az alábbiak közül?
A **helyes** állítás(oka)t jelölje **x**-szel, a **hibás**(aka)t - jellel!

a szegmens hossza	
a szegmens típusa	
az aktuális privilégiumszint(CPL)	

A szegmens leírótábla kezdőcíme	
a jelenlét (present) jelzőbit	
a globális/lokális leírótáblát jelző (T) bit	

a szegmens kezdőcíme	
a szegmens leírótábla kezdőcíme	
a szegmensleíró privilégiumszintje(DPL)	

a szegmens típusa	
a jelenlét (present) jelzőbit	
a globális/lokális leírótáblát jelző (T) bit	

A szegmenshez tartozó aktív lapok leíró táblájának kezdőcíme és száma	
a szegmens (lineáris-) kezdő címe	
a szegmens be van töltve az operatív tárolóba	

az aktív szegmensleíró tábla kezdőcíme	
a szegmensleíró privilégium szintje és a szegmens típusa	
a szegmens hossza	

a szegmensleíró hossza	
a szegmens leírótábla kezdőcíme	
az aktuális privilégiumszint(CPL)	

a szegmens típusa	
a jelenlét (present) jelzőbit	
a globális/lokális leírótáblát jelző (T) bit	

a szegmens (lineáris-)kezdőcíme	
a szegmens fut	
a szegmenst tartalmazó swap-file neve	

privilégiumszint	
a szegmens hossza	
aktív lapok száma	

a szegmens (lineáris-)kezdőcíme	
a szegmens be van töltve az op.tárba	
a szegmensleíró tábla kezdőcíme	

privilégiumszint	
a szegmens hossza	
aktív lapok száma	

A 386/486 processzorban mely információkat tartalmazza a ***lapleíró*** bejegyzés az alábbiak közül? A **helyes** állítás(oka)t jelölje **x**-szel, a **hibás**(aka)t - jellel!

a lap tartalma be van töltve az op.tárba (present)	
a lapra betöltés után történő írást jelző (D) bit	
a lap (lineáris-) kezdőcíme	

A lapra hivatkozó szegmens leírótábla kezdőcíme	
A lap hossza	
A lapkönyvtár tábla kezdőcíme (CR3)	

a lap (lineáris-)kezdőcíme	
privilégiumszint	
a lap tartalma be van töltve az operatív.tárba (present)	

A szegmens leírótábla kezdőcíme	
A lap hossza	
A laptábla-könyvtár tábla kezdőcíme	

a lap tartalma be van töltve az operatív tárolóba	
a lap (lineáris-) kezdő címe	
a lap privilégium szintje és attribútumai	

a lapkönyvtár tábla (directory) kezdőcíme	
az aktív lap hossza	
a globális szegmensleíró tábla kezdő címe	

a lap tartalma be van töltve az op.tárba (present)	
a lapra betöltés után történő írást jelző (D) bit	
a lap (lineáris-) kezdőcíme	

A lapra hivatkozó szegmens leírótábla kezdőcíme	
A lap hossza	
Az egy–illetve kétlépcsős működést jelző bit	

A **386/486** processzor védett üzemmódban, szegmentált lapszervezésű virtuális memóriakezelést alkalmaz. A **helyes** állítás(oka)t jelölje **x**-szel, a **hibás**(aka)t - jellel!

Minden task-hoz külön laptábla könyvtár kezdőcím tartozhat.	
Ha a kiépített fizikai memória méretét megduplázzuk, a laptábla könyvtár mérete is megduplázódik.	
Védett üzemmódban a szegmentálás nem kapcsolható ki.	
A kétlépcsős laptábla szervezés kétszer akkora memóriaterületet igényel, mint az egylépcsős laptábla.	
A szegmensleíró báziscíme bitjeinek száma független a lapszervezés ki- vagy bekapcsolásától.	
A virtuális és a fizikai címtartomány egyaránt 4GByte.	

A szegmentálás és a lapszervezés virtuális címtartománya 64 TeraByte, a fizikai címtartomány 4GByte.	
A laptábla könyvtár mérete függ a háttértárolón tárolt szegmensek számától és méretétől.	
A kétlépcsős laptábla szervezés hátránya, hogy a laptábla nagyobb memóriaterületet igényel, mint az egylépcsős laptábla.	
A virtuális és a fizikai processzor összerendelését az ütemezést megvalósító operációs rendszer a Task-hoz rendelt TSS tartalma alapján oldja meg.	
A lapleíróban a lap báziscíme bitjeinek száma nem függ a fizikai memória éppen kiépített (2GB vagy 4GB) méretétől.	
Minden új taszk-TSS tartalmának betöltése után a CR3 tartalmának megváltozása miatt a TLB-t realizáló cache tartalmát érvényteleníteni kell.	

Egy szegmens tetszőleges méretű (max. 4GB), tetszőleges címen kezdődhet, a fizikai címtartomány 4Gbyte, a virtuális címtartomány 64 Terabyte.	
A szegmensleíró tartalmazza a szegmens kezdőcímét, a szegmens hosszát és a leíró privilégium szintjét is.	
A kétlépcsős laptábla szervezés hátránya, hogy a laptábla nagyobb memóriaterületet igényel, mint az egylépcsős laptábla.	
A lapleíróban a lap báziscíme bitjeinek száma nem függ a fizikai memória éppen kiépített (2GB vagy 4GB) méretétől.	
Az I/O utasítások privilegizáltak, ezért nem minden privilégiumszinten futó program tudja végrehajtani azokat.	
Minden taszk-hoz külön lokális szegmensleíró tábla (LDT) és laptábla könyvtár (directory) kezdőcím tartozhat.	

Egy lap fix méretű (4KByte), tetszőleges címen kezdődhet, lineáris címtartománya 4GByt, a fizikai címtartomány 64 TeraByte e.	
A lapleíró tartalmazza a lap kezdőcímét, a lap hosszát a lapleíró privilégium szintjét, valamint a lap privilégium szintjét.	
A kétlépcsős laptábla szervezés előnye, hogy a laptábla kisebb memóriaterületet igényel, mint az egylépcsős laptábla.	
A szegmensleíróban a szegmens báziscíme bitjeinek a száma, valamint a szegmens hosszát megadó bitek száma függ a háttértárolón tárolt szegmensek számától és méretétől.	
Az I/O utasítások nem privilegizáltak, ezért minden privilégiumszinten futó program végre tudja hajtani azokat	
Minden új taszk-TSS tartalmának betöltése után a CR3 tartalmának megváltozása miatt a TLB-t realizáló cache tartalmát érvényteleníteni kell.	

A szegmentálás és a lapszervezés virtuális címtartománya 16 GByte, a fizikai címtartomány 4GByte.	
A laptábla könyvtár mérete felére csökkenhető, ha a kiépíthető fizikai memóriából csak 2Gbyte-ot építünk be az adott számítógépbe.	
A kétlépcsős laptábla szervezés kétszer akkora memóriaterületet igényel, mint az egylépcsős laptábla.	
Védett üzemmódban a szegmentálás nem kapcsolható ki.	
A szegmensleíró báziscíme bitjeinek száma nem függ a fizikai memória éppen kiépített (2GB vagy 4GB) méretétől.	
Minden task-hoz külön lokális szegmensleíró tábla és laptábla könyvtár kezdőcím tartozhat	

A virtuális és a fizikai címtartomány egyaránt 4GByte.	
Ha a kiépített fizikai memória méretét megduplázzuk, a laptábla könyvtár mérete is megduplázódik	
A kétlépcsős laptábla szervezés kétszer akkora memóriaterületet igényel, mint az egylépcsős laptábla.	
Védett üzemmódban a szegmentálás nem kapcsolható ki.	
A szegmensleíró báziscíme bitjeinek száma független a lapszervezés ki- vagy bekapcsolásától.	
Minden taszk-hoz külön laptábla könyvtár kezdőcím tartozhat	

A 386/486 processzoros számítógépen többtaszkos (multitasking) operációs rendszer működik, a védelmi lehetőségek teljes kihasználásával. A **helyes** állítás(oka)t jelölje **x**-szel, a **hibás**(aka)t **-** jellel!

Multitaskos rendszernél a fizikai és a virtuális processzor összerendelését megvalósító állapotleíró kezelését és a TASK váltás ütemezését az MMU végzi.	
Minden TASK-hoz külön szegmensleíró tábla tarozhat.	
A kétlépcsős laptábla szervezés előnye, hogy kevesebb memóriaterületet igényel, mint az egylépcsős.	
A szegmensleíró báziscím bejegyzés bitjeinek a száma függ az aktuálisan kiépített fizikai memória méretétől	
A GATE a különböző privilégium szintek közötti kommunikációt megvalósító hardver egység	
A szegmensleírónak is van privilégium szintje, és ennek ellenőrzése is a védelmi mechanizmus része	

Minden task-hoz külön laptábla könyvtár kezdőcím tartozhat	
Ha a kiépített fizikai memória méretét megduplázzuk, a laptábla könyvtár mérete is megduplázódik	
Védett üzemmódban a szegmentálás nem kapcsolható ki.	
A kétlépcsős laptábla szervezés kétszer akkora memóriaterületet igényel, mint az egylépcsős laptábla.	
A szegmensleíró báziscíme bitjeinek száma független a lapszervezés ki- vagy bekapcsolásától.	
A virtuális és a fizikai címtartomány egyaránt 4GByte.	

Multitaszkos rendszernél egy taszknak minden védelmi szinten van kód adat és veremszegmense.	
Az I/O utasításokat nem minden privilégiumszinten lehet végrehajtani.	
A tényleges privilégiumszint (EPL) értéke az aktuális (CPL) és a kérő privilégiumszint (RPL) közül a nagyobbik számértékűvel egyezik meg, azaz EPL=max(CPL,RPL).	
A taszkok állapotának nyilvántartására szolgáló különleges szegmens (TSS) címe a CR3 vezérlőregiszterben található.	
A CALL kapu (CALL gate) tartalmazza az új szelektort, az új eltolást és a paraméterhosszat.	
A megszakítási leíró tábla (IDT) is tartalmazhat taszk kapukat.	

A 386-os mikroprocesszor bekapcsolt lapszervezés esetén a 32bites lineáris címből kétlépcsős laptábla szervezéssel állítja elő a fizikai címet. A lapméret 4Kbyte. A laptábla könyvtár elejére a CR3regiszter tartalma mutat. Egy lapleíró bejegyzés 32bitet tartalmaz

Hány értékes bitet kell a CR3-nak tartalmaznia? | |
Minimálisan mekkora op. memória kell a kétlépcsős laptábláknak? | |
Egy TASK-hoz maximálisan hány lapleíró tábla tartozhat? | |

Milyen kivételes állapotot okoz az az eset, amikor a processzor "page not present" bejegyzést talál a lapleíróban?

A abort **B** fault **C** reset **D** trap [ ]

Mit jelent az eszköz-szintű I/O kezelés?
Sorolja fel és egy-egy mondatban ismertesse az eszközszintű I/O kezelés eseteit!
Mondjon példákat a direkt és a feltételes I/O kezelésre!
Magyarázza el mit jelent a perifériák szi
Rajzolja fel az SCSI interface általános blokkvázlatát! Sorolja fel a jellemző tulajdonságokat és az előnyöket!

Rajzolja fel az SCSI interface rendszer általános blokkvázlatát! Hány host és target egység alkalmazható?

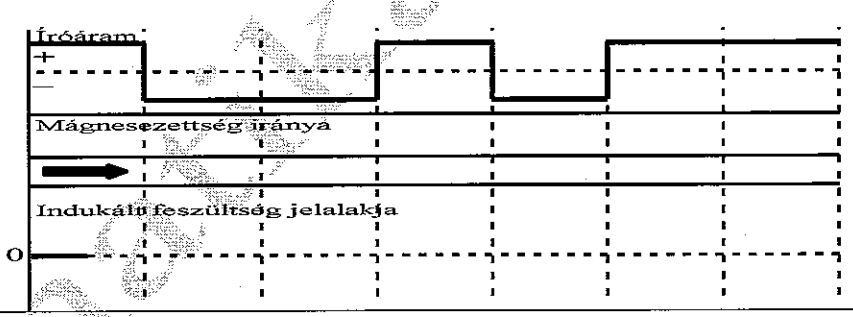
Rajzolja fel az ***I/O*** processzorra alapozott I/O, illetve perifériakezelés blokkvázlatát!

Sorolja fel milyen feladatokat lát el az I/O processzor

A nyolcbites SCSI interfészre max 8db egység (host és target) kapcsolódhat, az információ cserében résztvevő egységet a host jelöli ki egy szelekciós fázisban.	
A logikai I/O kezelés esetén a közvetlen input-output műveleteket az operációs rendszer végzi. A felhasználó csak op.-rendszer hívásokon keresztül érheti el ezeket.	
Az eszközszintű I/O kezelés esetén kell egy eszköz ami a KI/Be vonalakat az operációs rendszerhez kapcsolja	
A szinkron működésű periféria vezérlésére mindig UART-ot kell használnunk.	
Az aszinkron vezérlésű periféria működési idejét az I/O egység működési sebessége is befolyásolhatja	
Az I/O processzor az IN-és OUT utasításokat hajtja végre a CPU helyettl	

Mágneses háttértároló íróáram jelalakja látható az alábbi ábrán.

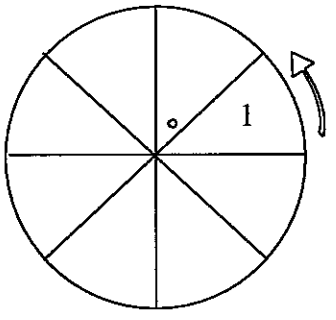
Adja meg a mágneses réteg ***mágnesezettségének irányát*** és a kiolvasáskor létrejövő ***indukált feszültség*** jelalakját.



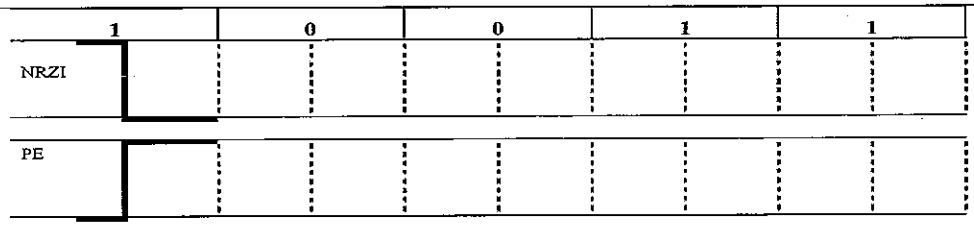
Rajzolja fel FM és MFM kódolás esetén a mágnesezettség vagy az íróáram jelalakját a következő bitsorozatra: 010001101

Mi a mark (jelző) szerepe, hogyan valósítják meg?
Magyarázza el hogyan szinkronózódik fel íráskor, illetve olvasáskor a mágneslemezes iró olvasó elektronika a formázáskor felvitt információra
Mi a GAP szerepe?

Egy floppy diszk egy ***sávja 8 szektort*** tartalmaz. (1p)
Rajzolja fel a szektorok ***fizikai*** elhelyezkedését a megadott forgásiránynál ***3:1*** szektor ***interleave*** esetén!
Milyen probléma kiküszöbölésére használható a megoldás?
Optimális esetben hány körülfordulás kell egy sáv összes szektorának írásához vagy olvasásához ***3:1*** szektorinterleave esetén?
.....
Feltételezve, hogy a ***fenti megoldás indokolt*** volt, ***mi történne 1:1 interleave létrehozása után***?



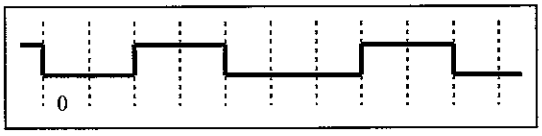
Egy floppy diszk egy sávja 9 szektort tartalmaz. Rajzolja fel a szektorok fizikai elhelyezkedését 2:1 szektor interleave esetén! Miért alkalmazzák ezt a megoldást.

Rajzolja be a mellékelt ábrába, egy mágneses tároló, íróáram jelalakját a megadott bitsorozatra NRZI és PE kódolás esetén	

A szokásos formátumú floppy-lemez egy szektora két részből áll, az azonosító mezőből (identification field) és az adatmezőből (data field). Az alábbi állítások közül jelölje x-szel az igazakat!

A szektornak csak az adatmezejében van hibaellenőrző kód (CRC)	
Adat írásakor a szektor azonosító mezejét is újraírják	
Nem csak a szektorok közt, hanem a szektoron belül is van egy "gap"	
Az adatmező hosszára vonatkozó információ az adatmező elején található	
a sáv és a szektor sorszámát	
Az azonosító mező tartalmazza a drive fizikai címét	

Egy mágneses elvű adatárolóban a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán egyforma időközönként vannak.
Az első bit értéke 0.



A felírásnál FM vagy MFM vagy PE kódolást alkalmazhattak. A jelalak alapján melyik kódolási eljárást használták a felírásnál?

Az alkalmazott kódolás: ☐

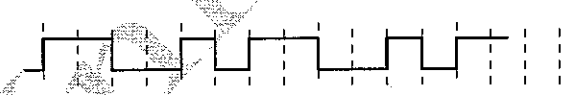
Írja be az üres kockákba a jelölt elsõ 0 értékű bitet követő további 4 bit értékét!

0 ☐☐☐☐

Azonos fluxusváltási sűrűséget feltételezve MFM kódolással kétszer annyi adat tárolható, mint FM kódolással

Igen	NEM

Egy mágneses elvű adattárolóban (pl. diszk) a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán egyforma időközönként vannak. Az első 2 bit értéke 0.

B 

Bit **0 0**

a/ Melyik kódolási eljárást használták a felírásnál **A** FM **B** MFM **C** NRZI ☐

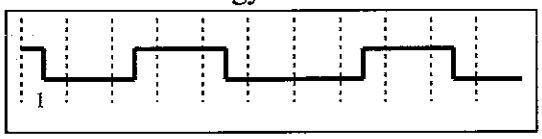
b/ Írja be az üres kockákba a jelölt első két 0 értékû bitet követő további 4 bit értékét!

0 0 ☐☐☐☐

c/

Szoft szektor szervezésnél speciális jelzések előállítására előre definiált módon megsértik a kódolási szabályt.	*Igen* *NEM*

Egy mágneses elvű adatárolóban a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán egyforma időközönként vannak.



Az első bit értéke **1.**

A felírásnál FM vagy NRZI vagy PE kódolást alkalmazhattak. A jelalak alapján melyik kódolási eljárást használták a felírásnál?

Az alkalmazott kódolás: ☐

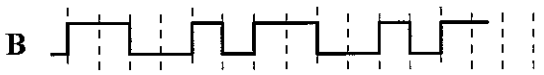
Írja be az üres kockákba a jelölt első 1 értékű bitet követő további 4 bit értékét!

1 ☐☐☐☐

A fenti három kódolási eljárás közül melyiknél a legnagyobb az egységnyi felületre eső fluxusváltozás? ☐

Egy mágneses elvű adatárolóban (pl. diszk) a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán egyforma időközönként vannak.

Az első 2 bit értéke 0.



Bit **0 0**

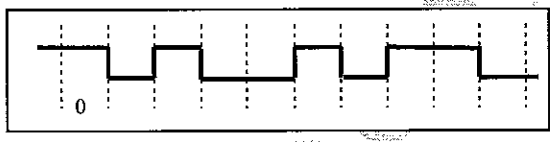
Melyik kódolási eljárást használták a felírásnál **A** FM **B** MFM **C** PE ☐

Írja be az üres kockákba a jelölt első két 0 értékű bitet követő további 4 bit értékét! **0 0** ☐☐☐☐

A fenti három kódolási eljárás közül melyikkel érhető el a legnagyobb adatbit sűrűség? ☐

Egy mágneses elvű adattárolóban a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán fél bit időközönként vannak.

Az első bit értéke 0.



A felírásnál FM vagy NRZI vagy PE kódolást alkalmazhattak. A jelalak alapján melyik kódolási eljárást használták a felírásnál?

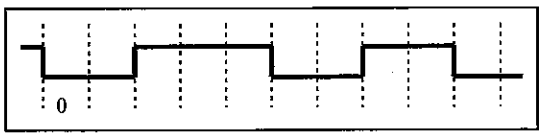
Az alkalmazott kódolás: ☐

Írja be az üres kockákba a jelölt első 0 értékű bitet követő további 4 bit értékét!

0 ☐☐☐☐

c/ Adja meg, hogy a fent említett kódolási módok közül melyiknél legkisebb a bitsűrűség! ☐

Egy mágneses elvű adattárolóban a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán fél bit időközönként vannak. Az első bit értéke 0.

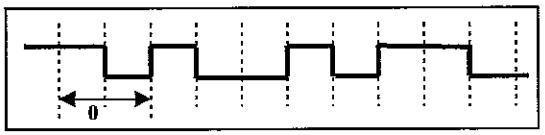


A felírásnál FM vagy MFM vagy PE kódolást alkalmazhattak. A jelalak alapján melyik kódolási eljárást használták a felírásnál? [ ]

A felírásnál FM vagy MFM vagy PE kódolást alkalmazhattak. [ ] A jelalak alapján melyik kódolási eljárást használták a felírásnál?

Adja meg, hogy a fent említett kódolási módok közül melyikkel érhető el a legnagyobb adat bitsűrűség! [ ]

Egy mágneses elvű adattárolóban a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán fél bit időközönként vannak. Az első bit értéke 0.



A felírásnál FM vagy NRZI vagy PE kódolást alkalmazhattak. A jelalak alapján melyik kódolási eljárást használták a felírásnál?

Az alkalmazott kódolás: [ ]

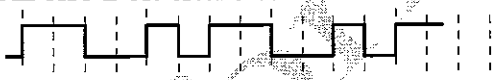
Írja be az üres kockákba a jelölt első 0 értékű bitet követő további 4 bit értékét!

0 [ ][ ][ ][ ]

Adja meg, hogy a fent említett kódolási módok közül melyiknél legkisebb a bitsűrűség! [ ]

Egy mágneses elvű adattárolóban (pl. diszk) a sáv mágnesezettsége (**B**) az ábrán látható módon változik. A szaggatott vonalak az ábrán egyforma időközönként vannak. Az első 2 bit értéke 0.

B



Bit **0 0**

Melyik kódolási eljárást használták a felírásnál **A** FM **B** MFM **C** PE? [ ]

Írja be az üres kockákba a jelölt első két 0 értékű bitet követő további 4 bit értékét!

0 0 [ ][ ][ ][ ]

A fenti három kódolási eljárás közül melyikkel érhető el a legnagyobb adatbit sűrűség? [ ]

Mi az alapvető különbség egy drive szintű (pl.:ST410) illetve egy IDE, vagy SCSI vezérlő között?

Ismertesse mit jelent, hogy két esemény konkurens!

Ismertesse mit jelent többprocesszoros rendszereknél a statikus, illetve a dinamikus feladat-hozzárendelés, sorolja fel előnyüket és hátrányukat!

Rajzolja fel egy lazán csatolt rendszernél a pont-pont közötti adatátvitel hardver jelzőbitre alapozott megoldásának blokkvázlatát! Mi a megoldás előnye, hátránya? Milyen probléma kiküszöbölésére használnak a továbbfejlesztett változatban FIFO elven működő átmeneti tárolót?

Rajzolja fel egy szorosan csatolt rendszer vázlatát!

Magyarázza el a szemaforok használatának elvét a szorosan csatolt rendszereknél?

Multiprocesszoros szorosan csatolt rendszereknél ***milyen hardver*** támogatás szükséges a szemaforkezelésnél szükséges ***kölcsönös kizárás*** biztosításához? ***Írjon*** példákat a megoldásra!

Sorolja fel a rendszersín funkcióit!

Ismertesse a rendszervezérlő, a master és a slave szerepét!

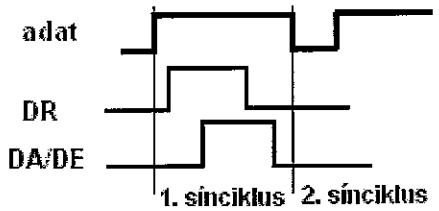
Ismertesse mit értünk erőforrás-particionált rendszeren? ***Sorolja fel*** az alapvető tulajdonságait!

Ismertesse mit értünk feladat-particionált rendszeren? ***Sorolja fel*** az alapvető tulajdonságait!

Sínrendszereknél mit jelent a geografikus címzés? Mondjon példát az alkalmazásra!

Ismertesse egy üzenetkapcsolt rendszer működésének elvét (fő lépéseit) pl.: háttértárolóról történő adatolvasás esetén!

Teljesen reteszelt protokollt alkalmazó sínrendszer jelei láthatók az alábbi ábrán.
Rajzolja be a jelátmenetek közötti, ok-okozati viszonyt mutató nyilakat!



Adja meg, hogy az egyes jelek a forrástól (F) vagy a nyelőtől (N) származnak?

adat:	DR:	DA/DE:

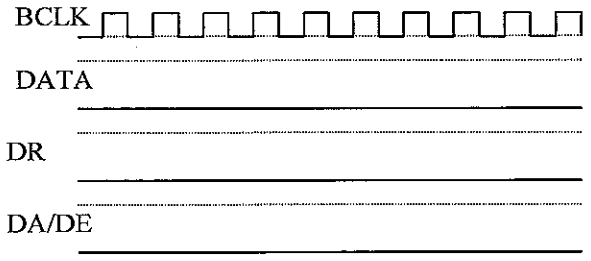
A sín átviteli sebessége megegyezik a sínen lévő leglassúbb moduléval (1), a master moduléval (2), az éppen aktív modulokéval ?

Rajzoljon fel egy teljesen reteszelt (kapcsolt) szemiszinkron adatátviteli protokolt!
Milyen értékűek lehetnek egy ilyen protokol időzítései?

8. Rajzolja be a mellékelt ábrába egy szemiszinkron teljesen kapcsolt adatátvitel jeleit.
Hogyan védekeznek egy ilyen protokollal rendelkező sínnél a nemlétező címre kiadott írás/olvasás esetén bekövetkező hiba ellen?

.....

.....

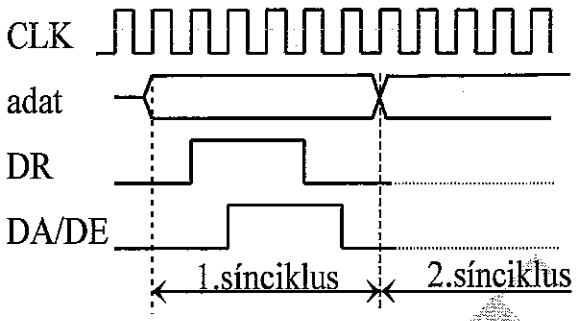


Teljesen ***reteszelt*** protokollt alkalmazó ***szemiszinkron*** sínrendszer jelei láthatók az alábbi ábrán (az órajel **10MHz**).

Miben különbözik egy teljesen kapcsolt aszinkron protokoll az ábrán adott protokolltól?
Mekkora lehetne egy írási sínciklus ***minimális*** ideje, egy ilyen rendszerben?

T_{WCmin}=.....

Rajzolja be a jelátmenetek közötti, ok-okozati viszonyt mutató nyilakat írás esetén!



Milyen hiba léphet fel ***teljesen kapcsolt aszinkron*** protokollt használó rendszerben egy ***nemlétező*** címre történő íráskor illetve olvasáskor? ***Hogyan kezelik*** ezt a problémát?

Ismertesse a buszmegszerzési stratégiákat, mondjon példát melyiket milyen esetben alkalmazzák!

Sorolja fel, és ***röviden ismertesse*** a buszelengedési stratégiákat!

Multiprocesszoros rendszereknél a processzorokat ***statikusan***, vagy ***dinamikusan*** rendelhetik a feladatokhoz. Milyen buszmegszerzési stratégiát alkalmaznak az egyik, illetve a másik esetben? Röviden indokolja a választ!

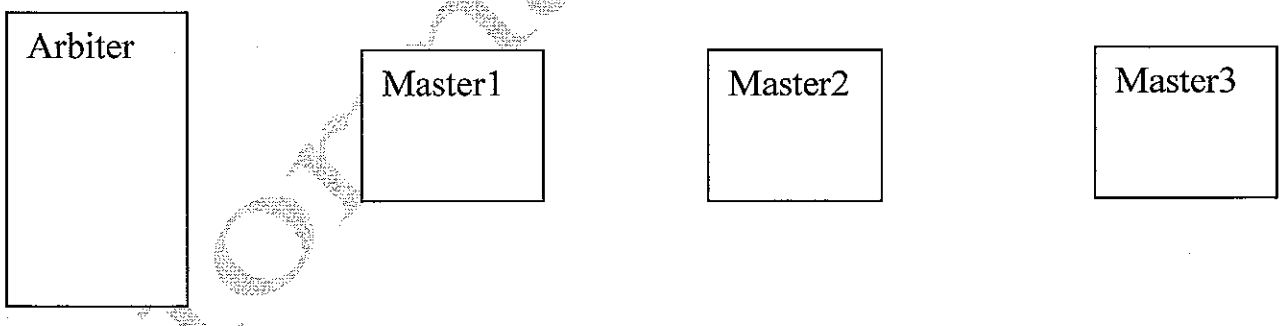
Statikus hozzárendelésnél ..... ***Indoklás:***.....

.....

Dinamikus hozzárendelésnél ..... ***Indoklás:***.....

Rajzolja fel a közös kérés –felfűzött válasz elvű arbitrációs mechanizmus vázlatát! Mi az előnye és mi a hátránya?

Rajzolja be egy közös kérés-felfűzött válasz(daisy chain) elvű buszarbitrációs rendszerben használt jeleket és a jel funkciójára utaló elnevezésüket!



Milyen prioritási stratégia alakítható ki a fenti masterek között.....

Rajzolja fel a VME bus kombinált arbitrációs rendszerének blokkvázlat

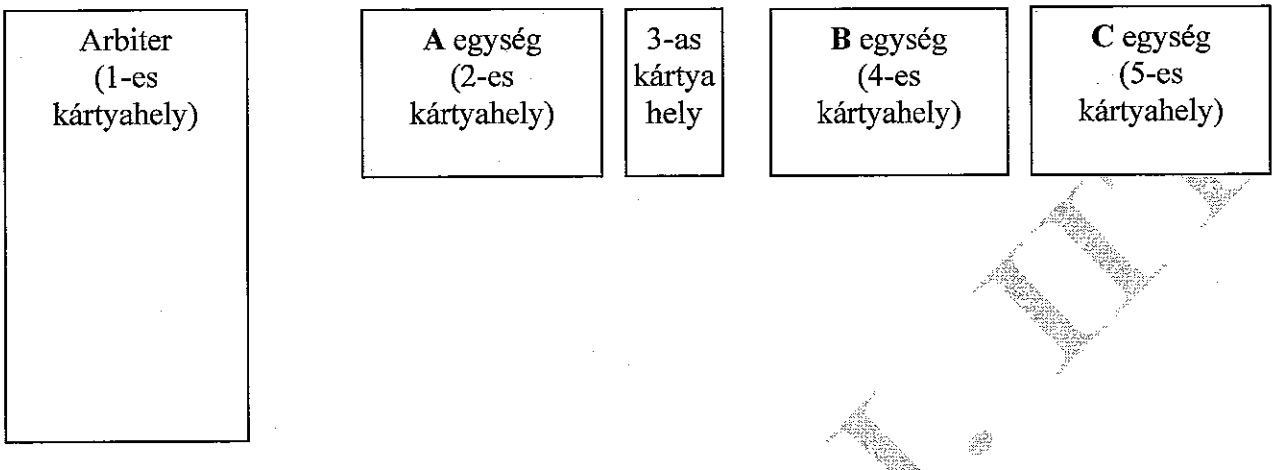
Hány **arbitrációs modul** (arbiter) és hány **master modul** lehet egy VME rendszerben**? Indokolja a választ!**
Arbiter:.....***Indoklás:***.....

.....

Master:.....***Indoklás:***.....
.....

Milyen buszmegszerzési stratégiák valósíthatók meg ***VME*** rendszerben? (Indokolja a választ)
Rajzolja fel a decentralizál, párhuzamos versenyeztetés elvű hardver mechanizmust használó arbitter blokkvázlatát!

VME rendszernél az **A** egység a **BR2***-őn, a **B** a **BR2***-őn, a **C** egység a **BR0***-án kér buszvezérlési jogot. Az arbiter a BR3-BR0 szintek között fix prioritásos elven működik. Prioritás, csökkenő prioritási sorrendben, BR3-BR2-BR1-BR0
a .Rajzolja be az arbitrációs rendszerhez tartozó fontos jeleket az alábbi ábrán, ha az **A** a 2-es, a **B** a 4-es, és a **C** az 5-ös kártyahelyen található!



Milyen sorrendben kapják meg a buszvezérlési jogot?

1:	2:	3:

VME rendszernél az **A** egység a **BR0***-án, a **B** a **BR2***-őn, a **C** egység a **BR2***-őn kér **egyidőben** buszvezérlési jogot. Az arbiter a BR3-BR0 szintek között **Round-Robin** elven működik. Az **A** master ROR (elengedés kérésre) a **B**- és **C** master RWD (elengedés, ha kész) buszelengedési stratégiát használ.

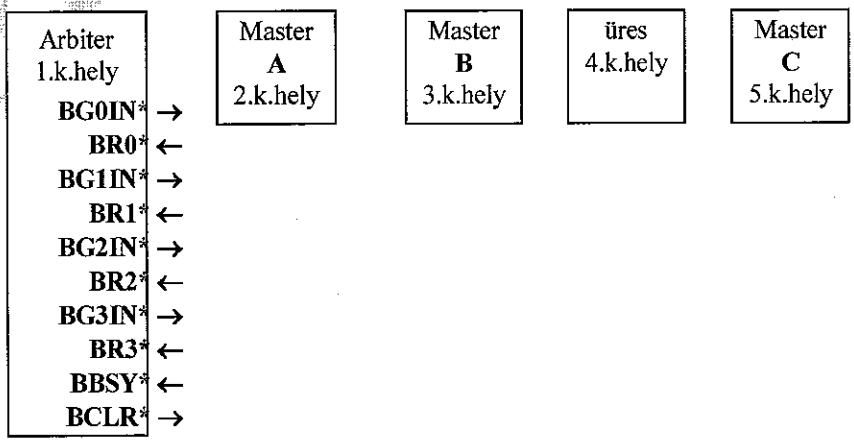
Milyen sorrendben kapják meg a buszvezérlési jogot ha a pillanatnyi prioritás, csökkenő prioritási sorrendben, BR2-BR1-BR0-BR3?

1	2	3

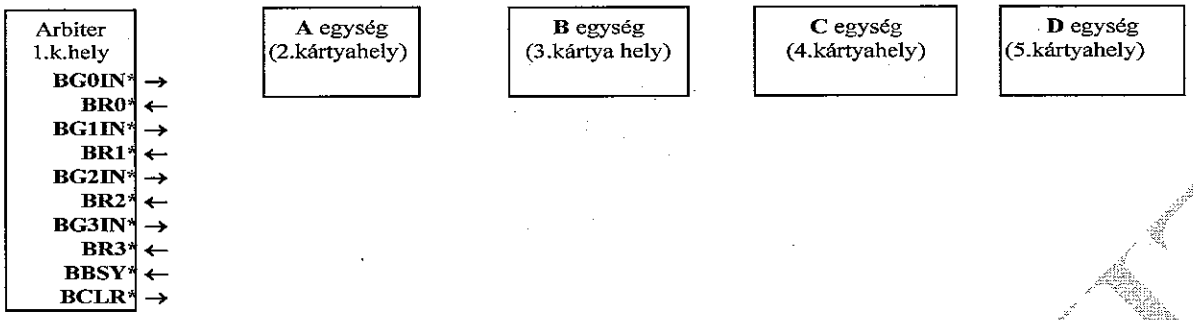
Mi a sorrend, ha a pillanatnyi prioritásnál BR0 a legmagasabb?

1	2	3

Rajzolja be az arbitrációs rendszerhez tartozó jeleket, a fentieknek megfelelően, az ábrába!



VME rendszernél az **A és D** egység a **BR0***-án, az **B és C** egység a **BR1***-en kér buszvezérlési jogot. Az arbitter a BR3-BR0 szintek között Round-Robin elven működik. Pillanatnyi prioritás, csökkenő prioritási sorrendben, BR3-BR2-BR1-BR0 .
A és C egység valamennyi, míg a **B és D** csak a **RWD** és a **Pre emption** buszelengedési stratégiát képes megvalósítani.



b) Milyen sorrendben kapják meg a buszvezérlési jogot?

1:	2:	3:	4:

c)Mi a sorrend ha az első egység adatátvitele alatta 6.kártya-helyen lévő **E** egység **BR3*-on** is kér buszvezérlési jogot?

1:	2:	3:	4:	5:

Rajzolja fel a egy párhuzamos versenyeztetés elvű decentralizált arbitrációs rendszer blokkvázlatát, röviden ismertesse a működés elvét

Multibus II rendszernél az **A, B, C** master az alábbi arbitrációs azonosító kódot adja a buszra:

	ARB5*			***ARB0****	
A	**1**	**0**	**1**	**0**	**0**
B	**1**	**0**	**0**	**0**	**1**	**1**
C	**1**	**0**	**0**	**0**	**1**	**0**

---------- ----------

Mutassa be, hogyan dől el, ki kapja meg elsőnek a buszvezérlési jogot!
Mi a további sorrend?

1:	2:	3:

Hogyan alakul a fenti sorrend, ha az elsőként kiszolgálásra kerülő masternél újabb buszvezérlési igény lép fel a második egység kiszolgálása alatt?

1:	2:	3:

Milyen buszmegszerzési stratégiát valósít meg a ***MBII*** a fenti megoldással?
Buszmegszerzési stratégia:.....
Hogyan biztosítja ezt a ***MBII*** ?

Multibus II rendszernél az **A, B, C** master az alábbi arbitrációs azonosító kódot adja a buszra:
Mutassa be, hogyan dől el, melyik master kapja meg elsőnek a buszvezérlési jogot.
Hogyan alakul a teljes sorrend ,ha az ***elsőként kiszolgálásra*** kerülő masternél ***újabb*** igény lép fel a ***második egység kiszolgálása alatt?*** Milyen buszmegszerzési stratégiát valósít meg a fenti mechanizmus?

	ARB5*			***ARB0****	
A	**1**	**0**	**1**	**1**	**1**	**1**
B	**1**	**0**	**0**	**1**	**1**	**0**
C	**1**	**0**	**0**	**1**	**1**	**1**

---------- ---------- ----------

-

Első Master:.....

Teljes sorrend : 1:.....2:.....3:.....4.....

Buszmegszerzési stratégia.....

.....

Multibus II rendszernél az **A, B, C** master az alábbi arbitrációs azonosító kódot adja a buszra: **Mutassa be,** hogyan dől el, ki kapja meg elsőnek a buszvezérlési jogot	***ARB5**** ***ARB0**** *A 1 0 0 1 1 1* **B 1 0 1 1 1 1** **C 1 0 0 1 1 0** ---------- ---------- ----------
Az **ABC** bejelentkezése után három órajellel később a mellékelt arbitrációs kódú **D** és **E** egységnél is buszvezérlési igény keletkezik. Adja meg hogy az **ABCDE** egységek milyen sorrendben kapnak buszvezérlési jogot! (1p)	**D 1 0 0 0 0 1** ***E 0 0 1 1 1 1*** 1: … 2:… 3:… 4:… 5:…

Rajzolja fel a láncolásos (dasy chain) bővítésű bus-vektoros megszakítási rendszer blokkvázlatát, magyarázza el a működését, sorolja fel az előnyeit és hátrányai!

Rajzolja fel a VME láncolásos (dasy chain) bővítésű bus-vektoros megszakítási rendszerének blokkvázlatát! Magyarázza el a működését, sorolja fel az előnyeit és hátrányai!

Hány ***Megszakítás kezelő*** lehet egy ***VME*** rendszerben (indokolja a választ)?

Mit értünk virtuális megszakítás-kezelő rendszeren?

VME rendszernél az **A** egység az **IRQ7***-en, a **B** az **IRQ2***-n, a **C** egység az **IRQ2***-en kér megszakítást. A megszakítás kezelő egység fix prioritású, az 1-es a legmagasabb, 7-es a legalacsonyabb szint.Rajzolja be a megszakításkérő egységekhez tartozó fontos jeleket az alábbi ábrán, ha az **A** a 2-es, a **B** a 4-es, a **C** a 5-ös kártyahelyen található!

1-es kártya hely	**A** egység (2-es kártyahely)	3-as kártya hely	**B** egység (4-es kártyahely)	**C** egység (5-es kártyahely)	Megszakítás kezelő egység (6-os kártyahely)

Milyen sorrendben adhatják fel a megszakítási vektoraikat, ha az éppen elfogadott szint a 2-es?

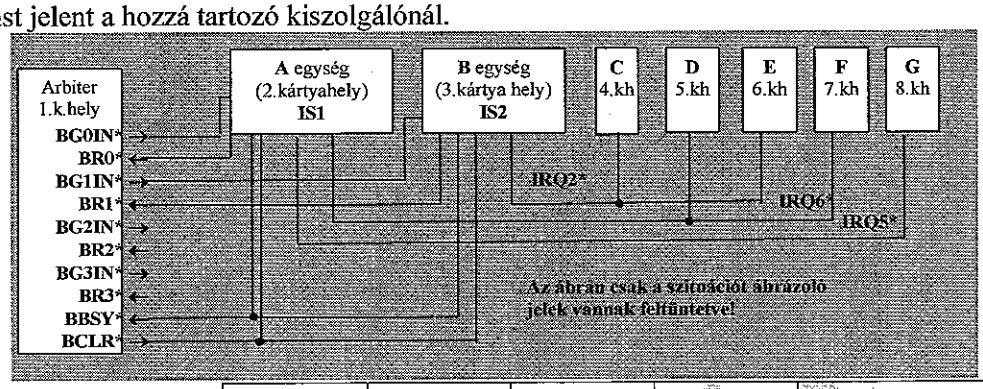
1:	2:	3:

Mi a sorrend, ha az IRQ7*-es a legmagasabb prioritású szint és az elfogadott szint a 2-es?

1:	2:	3:

VME rendszernél a 2.kártyahelyen (2.kh) ***A*** egység egy ***IS1*** jelű a 3kh.-en ***B*** egység egy ***IS2*** jelű megszakításkezelőt tartalmazó master . Az ***A*** a ***BR0**** a ***B*** a ***BR1****jelen kapcsolódik az arbiterhez. Az arbiter fix prioritású a magasab számú vonal magasabb prioritást jelent. ***IS1*** fogadja az ***IRQ7*, IRQ6*,IRQ5**** jeleket, míg ***IS2*** fogadja az összes többi megszakításkérő jelet. ***IS1*** és ***IS2*** fix prioritással működik a magasabb számú vonalon érkező kérés magasabb prioritást jelent a hozzá tartozó kiszolgálónál.

C egység (4.kh) és ***E*** egység (6.kh) az ***IRQ2****vezetékre a ***D*** (5.kh) és ***F*** (7kh) egység az ***IRQ6**** vezetékre G egység (8.kh) ***IRQ5**** vezetékre kapcsolódik. ***C,D,E,F,G*** egységeknél egyszerre keletkezik megszakításkérési esemény.



a) Adja meg milyen sorrendben kerülnek kiszolgálásra.

1:	*2:*	*3:*	*4:*	*5:*

Indoklás:

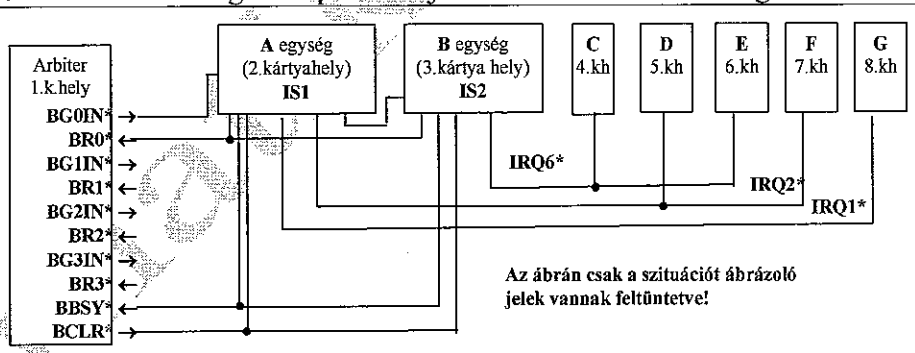
.....

b) Mi a sorrend, ha csak egy IS1 kezelő van a rendszerben valamennyi IRQ vonal számára?

1:	*2:*	*3:*	*4:*	*5:*

VME rendszernél a 2.kártyahelyen (2.kh) ***A*** egység egy ***IS1*** jelű a 3kh.-en ***B*** egység egy ***IS2*** jelű megszakításkezelőt tartalmazó master . Az ***A*** és a ***B*** egység a ***BR0****jelen kapcsolódik az arbiterhez. ***IS1*** fogadja az ***IRQ1*, IRQ2*,IRQ3**** jeleket, míg ***IS2*** fogadja az összes többi megszakításkérő jelet. ***IS1*** és ***IS2*** fix prioritással működik a magasabb számú vonalon érkező kérés magasabb prioritást jelent a hozzá tartozó kiszolgálónál.

C egység (4.kh) és ***E*** egység (6.kh) az ***IRQ6****vezetékre a ***D*** (5.kh) és ***F*** (7kh) egység az ***IRQ2**** vezetékre ***G*** egység (8.kh) ***IRQ1**** vezetékre kapcsolódik. ***C,D,E,F,G*** egységeknél egyszerre keletkezik megszakításkérési esemény.



a) Adja meg milyen sorrendben kerülnek kiszolgálásra.

1:	*2:*	*3:*	*4*	*5*

Indoklás:

.....

b) Mi a sorrend, ha csak egy ***IS*** kezelő van a rendszerben valamennyi IRQ vonal számára?

1:	*2:*	*3:*	*4*	*5*

c) ***Hogyan*** oldható meg, hogy az ***A*** és ***B*** egység egyenlő esélyű buszmegszerzési stratégia szerint kerüljön kiszolgálásra?

.....

Magyarázza el mit értünk virtuális megszakítási rendszeren! Mondjon egy alkalmazási példát!

Hány darab megszakítás-kérő és hány megszakítás-kezelő lehet a ***Multibus II*** rendszerben? ***Indokolja*** a választ!