Ara ja hem fet els exercicis del mòdul del Sistema de Memòria, i passem a treballar el següent mòdul, el Sistema d'Entrada/Sortida. Així la propera setmana farem exercicis relatius a les tres tècniques bàsiques d'E/S que podem trobar.

És important que llegiu amb atenció el punt 1 (Aspectes bàsics de l'Entrada/Sortida), que dóna una visió general del mòdul i també cal que llegiu el punt 2 ("Entrada/Sortida programada"), que és la primera tècnica que estudiarem i els punts 3 i 4 corresponents a interrupcions i DMA.

Per a poder realitzar aquests exercicis cal que estudieu amb atenció també el punt 5 ("Comparació de les tècniques d'E/S").

Reviseu amb atenció els apartats següents del mòdul 5 sobre el sistema d'entrada/sortida:
Exercicis 4.1: reviseu del mòdul 5: apartat 1 (punt 1.2 operació d'entrada/sortida) , apartat 5 comparació de les tècniques.
Exercici 4.2: reviseu l'apartat 2 del mòdul 5 i també l'apartat 5.
Exercici 4.3: reviseu l'apartat 3 del mòdul 5 i també l'apartat 5.
Exercici 4.4: reviseu l'apartat 4 del mòdul 5 i també l'apartat 5.
Exercici 4.5: reviseu els apartats, 1, 2 i 3 del mòdul 5.

Els exercicis d'aquesta setmana tracten sobre la Entrada/Sortida fent servir un **disc**. Cal que llegiu amb atenció la part del problema on s'explica la organització d'un disc, per a poder inferir la velocitat de transferència del disc Aquest són els darrers exercicis de teoria que us proposem.

Exercici E-Teo.4

4.1. Perifèric

Primer heu de calcular algunes característiques quantitatives, a partir de certes dades sobre el disc. Després heu d'interpretar el codi CISCA que fa una transferència amb el disc mitjançant el mètode d'E/S programada. Donada la velocitat d'execució de les instruccions heu de calcular **límits quantitatius** de la transferència.

Es vol realitzar la transferència d'un sector de dades entre memòria i un disc.

El disc està organitzat en pistes, que són circumferències completes al voltant del centre del disc, i que contenen bits d'informació representats com orientacions magnètiques sobre la superfície del disc. El disc gira, i el capçal de lectura va llegint la informació d'una determinada pista. Per llegir-la sencera cal que el disc doni un gir sencer. Les pistes es divideixen en sectors més petits, i que constitueixen la unitat més petita d'informació que es pot transferir entre un disc i el computador. El capçal del disc es pot moure mecànicament per apropar-se o allunyar-se del centre del disc, per tal de posicionar-se sobre la pista desitjada. El temps que triga el capçal de lectura del disc en col·locar-se al començament d'un sector s'anomena temps de latència.

Les característiques del disc són les següents:

- Temps de latència: $t_{\text{latència}}$, = 6 ms
- Velocitat de gir: 12.000 rpm (revolucions per minut)
- Bytes per pista: 500.000 Bytes (totes les pistes contenen el mateix nombre de bytes)

Les característiques bàsiques de la transferència són:

- m_{dada}= 4 Bytes (en CISCA les dades són sempre de 4 Bytes)

a) Quina és la velocitat de transferència aproximada del disc (v_{transf}) sense considerar la latència per a posicionar el capçal?

12.000 rpm = 12000 revolucions / 1 minut = 12000 revolucions / 60 segons =>
1 revolució del disc cada 60 / 12000 = 0,005 segons = 5 ms,
En una revolució del disc llegim una pista, per tant, es poden transferir 500.000 Bytes cada 5 ms, es a dir,

v_{transf} = 500.000 bytes / 0,005 s = 100.000.000 Bytes/s = 100 Mbytes/s (1MByte = 10^6 Bytes)

b) Si les transferències al disc són d'una mida, m_{bloc}, de 256 KBytes consecutius, quin és el temps de transferència del bloc per part del perifèric ($N_{dades} * t_{dada}$) sense incloure el temps de latència del disc? Quin és el temps de transferència de tot el bloc, incloent la latència mitja del disc ($t_{bloc} = t_{latència} + N_{dades} * t_{dada}$)?
Per simplificar els càlculs suposarem que 1KByte = 1000 Bytes.

$N_{dades} = m_{bloc} / m_{dada}$ = 256000 Bytes / 4 Bytes = 64000
$t_{dada} = m_{dada} / v_{transf}$ = 4 Bytes / 100 Mbytes/s = 4 / 100000000 s = $4 \cdot 10^{-8}$ s = 0,04 us (microsegons)
$N_{dades} * t_{dada}$ = 64000 * 0,04 us = 2560 us = 2,56 ms = 0,00256 s
$t_{bloc} = t_{latència} + (N_{dades} * t_{dada})$ = 6 ms + 2560 us = 6000 us + 2560 us = 8560 us = 8,56 ms = 0,00856 s

c) Fer el mateix càlcul que en l'apartat anterior suposant una mida de bloc, m_{bloc}, de 4 KBytes consecutius. Quina conclusió es pot extreure dels resultats?
Per simplificar els càlculs suposarem que 1KByte = 1000 Bytes.

$N_{dades} = m_{bloc} / m_{dada}$ = 4000 Bytes / 4 Bytes = 1000
t_{dada} = 0,04 us
$N_{dades} * t_{dada}$ = 1000 * 0,04 us = 40 us = 0,04 ms
$t_{bloc} = t_{latència} + (N_{dades} * t_{dada})$ = 6 ms + 0,04 ms = 6,04 ms

Es pot concloure que si llegim un bloc de mida petita, la major part del temps de resposta del disc es deu a la latència d'accés, i no al temps de transferència.

Per tant, amb un moderat percentatge d'increment de temps (de 6,04 ms a 8,56 ms l'increment és de 8,56/6,04 = 1,417, es a dir, de prop del 42%) podem llegir un bloc molt més gran (el bloc de 256KB és 64 vegades més gran que el bloc de 4KB)

4.2. Entrada/Sortida Programada

El següent codi realitzat amb el repertori CISCA transfereix un sector de dades entre memòria i el disc descrit en l'apartat anterior mitjançant la tècnica d'E/S programada.

Considerem les següents característiques del processador i del sistema d'E/S:

- Freqüència del rellotge del processador: 2,5 GHz
- Nombre d'instruccions per cicle: 4 instruccions

Es tracta d'un processador que es capaç d'executar diverses instruccions en un sol cicle de rellotge (un processador multinucli per exemple).

- $t_{prog} + t_{final}$ = 500 ns (nanosegons)

Les posicions 200 i 204 corresponen a ports d'Entrada/Sortida o registres d'E/S del controlador de disc (mapa independent d'E/S).

Suposar que just abans d'executar aquest codi s'acaba de programar el disc (donar l'ordre) per a posicionar-se en el sector a que es vol accedir, i que tota la informació accedida es troba contigua al disc. DATA és una etiqueta que indica l'adreça de començament d'un vector emmagatzemat a memòria.

```
1.          MOV   R1, 200
2.          MOV   R2, DATA
3.          MOV   R3, 50000
4.  Bucle:  IN    R0,[R1]
5.          AND   R0, 2
6.          JE    Bucle
7.          IN    R0,[R1+4]
8.          MOV   [R2],R0
9.          ADD   R2,4
10.         DEC   R3
11.         JNE   Bucle
```

a) Es tracta d'una transferència de sortida al disc o d'entrada des del disc? Quina és la mida de la dada bàsica que es transfereix en cada pas, expressada en bytes, m_{dada} ?

La transferència és d'entrada: la instrucció 7 llegeix des del registre de dades del controlador de disc, port 204, i la instrucció 8 copia la dada llegida al vector DADES, en memòria.

La instrucció 7 copia una dada del port 204 i la guarda al registre R0. Com que CISCA utilitza una mida de paraula de memòria de 4 bytes, es transfereixen 4 bytes cada vegada. A més, la instrucció 9 suma 4 al registre R2 que es fa servir com a índex per a apuntar al següent element del vector on guardar la nova dada.
m_{dada}= 4 Bytes

b) Quantes dades es transfereixen en total entre la memòria i el disc, N_{dades} ?
Les instruccions 3, 10 i 11 controlen el bucle per tal que es repeteixi 50.000 vegades, i en cada iteració del bucle es transfereixen 4 bytes.

Es realitzen N_{dades} = 50.000 transferències, per tant es transfereixen 50.000 * 4 Bytes = 200.000 Bytes.

c) Quin bit del registre d'estat del controlador de disc s'utilitza per a verificar que es pot realitzar la transferència d'una dada? Quin valor d'aquest bit indica que la transferència es pot realitzar?

La instrucció 4 llegeix el contingut del registre d'estat, port amb adreça 200, i la instrucció 5 fa una màscara (instrucció AND) del valor llegit amb el bit 1, el segon bit menys significatiu (valor decimal 2), que ha de ser 1 per a permetre sortir del bucle de les instruccions 4, 5 i 6 i realitzar la transferència d'una dada de 4 bytes.

La operació AND bit a bit permet seleccionar els bits d'un cert valor V. Per exemple el resultat de V AND 1 serà diferent de zero si el bit menys significatiu de V (el bit zero) és 1. V AND 8 serà diferent de zero si el 4t bit menys significatiu (el bit 3) és 1.

d) Quin és el temps total que dedica la CPU a la tasca d'Entrada/Sortida, t_{cpu} ?
En E/S programada considerarem $t_{transf\_bloc} = t_{latència} + (N_{dades} * t_{transf\_dada})$

Amb la tècnica d'E/S programada la CPU està completament dedicada a la tasca d'E/S i no fa altra cosa durant aquest temps. El temps que dedica la CPU serà per tant el mateix temps que triga el perifèric:
$t_{transf\_dada} = t_{dada}$
Per tant:
$t_{transf\_bloc} = t_{latència} + (N_{dades} * t_{dada}) = t_{bloc}$
Per tant, podem calcular el temps de transferència del bloc, tal i com s'ha fet al exercici 4.1 b i c, per a calcular t_{bloc}:
N_{dades} = 50000
$t_{dada} = m_{dada} / v_{transf}$ = 4 Bytes / 100 Mbytes/s = 4 / 100000000 s = $4 \cdot 10^{-8}$ s = 0,04 us (microsegons)
$N_{dades} * t_{dada}$ = 50000 * 0,04 us = 2000 us = 2 ms = 0,002 s

$t_{transf\_bloc} = t_{bloc} = t_{latència} + (N_{dades} * t_{dada}) = 6$ ms + 2 ms = 8 ms
Durant els 6 ms d'espera fins que el disc es posicioni en el sector adequat s'executen les instruccions 4, 5 i 6. Després, durant els 2 ms que dura la transferència de dades, s'alterna entre la execució de les instruccions 4-6, del bucle d'espera activa de sincronització, i la execució de les instruccions 7-8, que fan el intercanvi de cada dada.

El temps final d'ocupació de la CPU cal que inclogui el temps de programació i finalització de la transferència:

$t_{cpu} = (t_{prog} + t_{final}) + t_{transf\_bloc} = 500$ ns + 8 ms = 8,0005 ms.

e) Si volguéssim fer servir el mateix processador i el mateix programa però amb un disc més ràpid, quina és la màxima taxa o velocitat de transferència del nou disc que es podria suportar sense que es perdessin dades?

El mínim nombre d'instruccions que ha d'executar el programa per a cada dada transferida és 8 instruccions (4-11). L'execució repetida de les instruccions 4, 5 i 6 només es realitza quan el dispositiu perifèric és més lent que el processador.
Freqüència de rellotge = 2,5 GHz, implica un temps de cicle de: $t_{cicle} = 1 / 2{,}5*10^9 = 0{,}4$ ns (nanosegons)
El processador pot executar 4 instruccions per cicle.
Per tant, el temps mínim per a transferir una dada (4 Bytes) serà el temps necessari per a executar 8 instruccions:
8 instruccions / 4 instr/cicle = 2 cicles, es a dir 2 * 0,4 ns = 0,8 ns
Per tant, es poden transferir com a molt 4 Bytes cada 0,8 ns, és a dir:
4 Bytes / $0{,}8*10^{-9}$ s = 5 GBytes/s, com a màxim (1 Gbyte = 10^9 Bytes = 1000000000 Bytes)

4.3. Entrada/Sortida per Interrupcions

Heu d'interpretar el codi CISCA que fa la transferència amb el disc fent servir el mètode d'E/S per interrupcions. També, donada la velocitat d'execució de les instruccions, heu de calcular **límits quantitatius** de la transferència.
Es vol realitzar la transferència d'un sector de dades entre la memòria d'un processador i el mateix disc que s'ha fet servir en l'exercici 4.1, considerant les mateixes característiques de l'apartat 4.2:

- Freqüència del rellotge del processador: 2,5 GHz
- Nombre d'instruccions per cicle: 4 instruccions
- $t_{prog} + t_{final} = 500$ ns (nanosegons)

A més a més considerem:

- t_{rec_ins} = temps d'un cicle = t_{cicle}

El següent codi realitzat amb el repertori CISCA és una rutina de servei a interrupció (RSI) que transfereix dades des del disc a la memòria del processador mitjançant la tècnica d'E/S per interrupcions. La RSI s'executa una vegada per a cada dada rebuda des del disc. Per simplificar el codi, ignorarem el control de la condició de finalització de la transferència.
ADDRA fa referència a una variable en memòria que conté adreces. El contingut d'aquesta variable ADDR és, per tant, una adreça que va apuntant successivament als elements d'un vector de dades emmagatzemat a memòria. En aquest vector es guarden les dades que es van llegint del disc.

```
1.    CLI
```

```
2.    PUSH  R0
3.    PUSH  R1
4.    IN    R0, [200]
5.    IN    R0, [204]
6.    MOV   R1, [ADDR]
7.    MOV   [R1],R0
8.    ADD   R1,4
9.    MOV   [ADDR],R1
10.   POP   R1
11.   POP   R0
12.   STI
13.   IRET
```

Suposar que s'acaba de programar el disc (donar l'ordre) per a posicionar-se en el sector a que es vol accedir.

a) Per quina raó cal executar les instruccions 1, 2, 3, 10, 11, 12 i 13?

Cal deshabilitar interrupcions (1) per tal que el mateix disc no torni a interrompre la pròpia RSI, i cal tornar-les a habilitar (12) per permetre noves interrupcions un cop acabi la RSI.
Cal salvar (2 i 3) el contingut dels registres que s'han de modificar a la RSI per tal de recuperar el seu valor (10 i 11) al final de la RSI, i així assegurar que el programa que s'ha interromput no veu modificat el seu estat.
La instrucció IRET (13) torna el control al programa interromput.

b) Les instruccions 4 i 5 llegeixen el contingut de dos registres (ports) d'E/S. El valor llegit del port 200 no es fa servir, però ÉS NECESSARI fer aquesta lectura per tal que la RSI funcioni! Reviseu l'ús que es feia d'aquest registre d'ESTAT en la versió d'E/S programada de l'exercici 4.2 i proposeu una hipòtesi respecte a perquè pot ser necessari executar la instrucció 4.

Molts dispositius perifèrics requereixen que es faci una lectura al registre d'Estat per tal d'esborrar el bit d'Estat que indica que hi ha una petició d'interrupció pendent. D'aquesta forma el programador indica al dispositiu perifèric que s'ha atès la interrupció i que la petició ja es pot esborrar.
Quan llegim o escrivim de/a un registre d'E/S es poden produir "efectes col·laterals". Per exemple, és possible (1) llegir un registre d'Estat dues vegades seguides i obtenir valors diferents, (2) escriure un valor i després llegir un valor diferent al que s'ha escrit, (3) escriure un valor i provocar un canvi en el valor llegit d'un altre registre ...
Per tant, no podem descartar la instrucció 4 pel fet que "sembli" que no té sentit. Quan accedim a registres d'E/S hem de comptar amb els "efectes col·laterals"

c) Quin és el temps total que dedica la CPU a la tasca d'Entrada/Sortida, t_{cpu} ? Quin percentatge d'ocupació representa **%**ocupació ?

Si la freqüència de rellotge és 2,5 GHz, llavors el temps d'un cicle és de 0,4 ns (calculat a l'apartat 4.2 e)
Com el processador és capaç d'executar 4 instr/cicle, el temps per a executar una instrucció serà:
$t_{instr} = t_{cicle} / 4 = 0{,}4 \text{ ns} / 4 = 0{,}1 \text{ ns}$
Temps per atendre la interrupció, t_{rec_int}: 1 cicle = $t_{cicle} = 0{,}4$ ns
Temps d'execució RSI, t_{rsi}: $N_{rsi} * t_{instr} = 13 \text{ instr.} * 0{,}1 \text{ ns} = 1{,}3$ ns
Temps consumit per CPU en cada interrupció:
$t_{transf\_dada} = t_{rec\_int} + t_{rsi} = 0{,}4 + 1{,}3 = 1{,}7$ ns
Nombre d'interrupcions produïdes (o nombre total de dades, N_{dades}):
200.000 Bytes / 4 Bytes per interrupció = 50.000 interrupcions.
Temps consumit en total en TOTES les interrupcions:
$t_{transf\_bloc} = t_{transf\_dada} * N_{dades} = 1{,}7 \text{ ns} * 50.000 \text{ interrupcions} = 85000 \text{ ns} = 85 \text{ us (microsegons)}$

El temps final d'ocupació de la CPU ha d'incloure el temps de programació i finalització de la transferència:
t_{cpu}= (t_{prog} + t_{final}) + t_{transf_bloc} = 500 ns + 85 us = 85,5 us.

Dels 8 ms de temps total que triga el perifèric per a realitzar la transferència (t_{bloc} = 8 ms = 8000 us), aquest temps es va calcular a l'exercici 4.2. d, la CPU està dedicada a la tasca d'E/S el percentatge següent:
% ocupació = t_{transf_bloc} * 100 / t_{bloc} = 85 * 100 / 8000 = 8500 / 8000 = 1,0625 % del temps.

d) Si volguéssim reduir la freqüència de rellotge del processador per tal de reduir el consum energètic, fins a quina freqüència podríem fer-ho sense que es perdessin dades en la transferència?
Cal considerar el temps que ha utilitzat el disc per transferir totes les dades, sense considerar el temps de latència.
En la fase de transferència de dades, el disc genera 50.000 interrupcions durant 2 ms (aquesta dada es va calcular en l'exercici 4.2.d):
N_{dades} * t_{dada} = 50000 * 0,04 us = 2000 us = 2 ms

És a dir, tenim una interrupció cada 2.000 us / 50.000 = 0,04 us = 40 ns. Aquest és el temps màxim que hauria de tardar la gestió de la interrupció, incloent el temps addicional per transferir el control a la RSI, el temps que pot consumir la CPU en una interrupció t_{transf_dada}

El temps consumit per la CPU en cada interrupció és, com hem vist a l'apartat c), la suma del temps de transferir el control a la RSI + executar RSI:
A l'enunciat es defineix que t_{rec_int} = t_{cicle} i per tant:
t_{transf_dada} = t_{rec_int} + t_{rsi} = t_{rec_int} + (N_{rsi} * t_{instr}) = t_{cicle} + (13 * t_{instr})

Tal com hem vist a l'apartat c) el temps d'una instrucció és: t_{instr} = t_{cicle} / 4
Per tant: t_{transf_dada} = t_{cicle} + (13 * t_{cicle} / 4) = 4,25 * t_{cicle}

Volem trobar el temps de cicle tal que el temps de transferència d'una dada sigui 40 ns:
40 ns = 4,25 * t_{cicle} => t_{cicle} = 40 / 4,25 = 9,41 ns

La freqüència serà la inversa del temps de cicle:
1 / 9,41 * 10^{-9} = 0,106 GHz = 106 Mhz

4.4. Entrada/Sortida per DMA

Suposarem que el controlador de disc pot funcionar en mode DMA (Accés Directe a Memòria). Cada vegada que el controlador de disc amb DMA demana el bus a la CPU, aquesta li dona prioritat immediatament (mecanisme del robatori de cicle). Per a la transferència d'una dada entre la memòria i el controlador de disc, s'utilitza el bus de memòria durant 2 ns, i es torna a cedir l'ús del bus a la CPU. Aquests 2 ns inclouen el **temps de cessió** del bus, el **temps de la transferència** pel bus, i el **temps de recuperació** del bus ($t_{cessió}$ + t_{mem} + t_{recup}).
En cas de transferir una ràfega o bloc de dades entre la memòria i el controlador, s'hauria d'afegir 1 ns per cada dada addicional que es transfereixi (es a dir, t_{mem} = 1 ns).

a) Considerem que en la transferència per DMA, les dades s'envien entre el controlador de disc/DMA i la memòria d'una en una (4 Bytes cada cop), i per tant es generen tantes peticions del bus com dades es transfereixen. Calcular el temps total d'ocupació del bus per part del controlador de disc/DMA per a dur a terme la mateixa transferència que hem analitzat en els apartats 4.2 i 4.3.

Temps ocupació Bus, t_{transf_dada}: $t_{cessió}$ + t_{mem} + t_{recup} = 2 ns
Nombre de peticions del Bus, N_{dades}: 50.000
Temps total d'ocupació del Bus, t_{transf_bloc}: t_{transf_dada} * N_{dades} = 2 * 50.000 = 100 us

b) Contestar a la pregunta anterior suposant que el controlador de disc/DMA emmagatzema 4 dades temporalment i que la transferència d'aquestes 4 dades es fa en mode ràfega.
Temps ocupació Bus, t_{transf_dada}: $t_{cessió} + 4 * t_{mem} + t_{recup} = 2 + 3 * 1 = 5$ ns
Nombre de peticions del Bus, $N_{dades} / N_{ràfega}$: 50.000 / 4 = 12.500
Temps total d'ocupació del Bus t_{transf_bloc}: $t_{transf\_dada} * N_{dades}/N_{ràfega} = 5 * 12.500 = 62,5$ us

c) La CPU no pot fer cap tasca durant tot el temps en que el bus està ocupat per part del controlador de disc/DMA. Quin percentatge de temps té disponible la CPU per a executar codi efectiu d'altres programes durant la transferència (100 - **%**ocupació) ? Contestar en el cas de transferències amb i sense ràfegues.

Recordem de l'apartat 4.2 que $t_{bloc} = 8000$ us

Sense ràfegues: $t_{transf\_bloc} = 100$ us
Percentatge de temps disponible: 100 – % ocupació = 100 - (t_{transf_bloc} * 100) / t_{bloc} = 100 - (100 * 100) / 8000 = 100 – 1,25 = 98,75%

Amb ràfegues: $t_{transf\_bloc} = 62,5$ us
Percentatge de temps disponible: 100 – % ocupació = 100 - (t_{transf_bloc} * 100) / t_{bloc} = 100 - (62,5 * 100) / 8000 = 100 - 0,78 = 99, 22%

4.5. Qüestions sobre Entrada/Sortida

a) Quines implicacions i quines avantatges té tenir un mapa comú de memòria d'E/S o un mapa independent de memòria d'E/S?.
1) **Mapa comú de memòria i E/S**. No hi ha distinció entre adreces de memòria i registres d'E/S. Per a accedir als registres s'utilitza descodificadors que s'activen a partir de les línies del bus d'adreces i es fa servir els mateixos senyals de control (READ/WRITE) que s'utilitza per a seleccionar la memòria. Podem accedir als ports d'E/S amb les mateixes instruccions de transferència que utilitzem per a accedir a memòria (MOV i les variants que té).
Aquest sistema té l'avantatge que ens permet aprofitar l'ampli conjunt d'instruccions de què disposa el processador per a accedir a memòria i podem fer programes més eficients. El principal desavantatge és que hem de dedicar una part del valuós espai de memòria a les adreces d'E/S i cal ser curosos amb la quantitat d'espai assignat ja que aquest espai va en detriment de l'espai de memòria disponible, però cada cop aquest problema és menys important a causa de l'increment de l'espai de memòria adreçable.

2) **Mapa independent d'E/S.** Hi ha distinció entre adreces de memòria i registres d'E/S. Les línies d'adreces s'acostumen a compartir, però hi cal afegir algunes línies de control per a distingir si un accés és a memòria o a un port d'E/S. També són necessàries instruccions especifiques d'E/S. Les instruccions utilitzades habitualment són IN (per a llegir del port d'E/S) i OUT (per a escriure en el port d'E/S).
Aquest sistema té l'avantatge que la memòria disposa de tot el rang d'adreces i el clar desavantatge que disposa d'un reduït nombre d'instruccions especifiques d'E/S que només disposen dels modes d'adreçament més bàsics per a accedir als ports d'E/S.

b) Quines són les principals problemàtiques que cal atendre en un sistema d'E/S que hagi de gestionar múltiples dispositius d'E/S?
Tots els computadors han de gestionar més d'un perifèric i aquests poden treballar al mateix temps; per exemple, estem imprimint un document que tenim guardat al disc mentre escrivim un text amb el teclat que es mostra per pantalla. Per tant, hem de preveure que el nostre sistema d'E/S pugui gestionar transferències d'E/S amb dos perifèrics simultàniament o més. Això vol dir

que de manera simultània dos mòduls d'E/S o més han d'estar preparats per a fer la transferència de dades amb el processador. La transferència no la podem fer al mateix temps. Per aquest motiu hem de disposar d'un sistema que, primer, ens permeti determinar quins són els mòduls que hem d'atendre (identificar els mòduls d'E/S que estan preparats per a l'operació d'E/S) i, segon, ens permeti decidir qui atenem primer, tenint en compte que si ja atenem un altre perifèric o fem una altra tasca més prioritària no la podem interrompre (establir una política de prioritats).

Tant la identificació del mòdul d'E/S que està preparat per a transferir una dada com la manera d'establir una política de prioritats depenen de la tècnica d'E/S que utilitzem per a gestionar les operacions d'E/S i l'hem d'analitzar en detall en cada cas.

c) Quina és la principal desavantatge de la tècnica d'E/S programada, respecte de l'E/S per interrupcions i per DMA. Explica-ho breument.
Mentre es fa la sincronització, el processador està dedicat al cent per cent a aquesta tasca i, per tant, no pot atendre altres processos o aplicacions. Si aquesta espera és molt llarga pot degradar el nivell de prestacions de tot el sistema.
Per tant, és recomanable que les transferències fetes utilitzant aquesta tècnica siguin curtes i ràpides.

d) Que fa el processador en un transferència d'E/S programada, segons si és un operació d'entrada o una operació de sortida.
En el cas del perifèric d'entrada, cada vegada que el processador comprova que el controlador de perifèric disposa d'una nova dada, la recull i queda disponible per a rebre una nova dada o processar l'anterior. En el cas d'un perifèric de sortida, cada vegada que el processador disposa d'una dada per a enviar al perifèric i detecta que el seu controlador la pot rebre, efectua l'enviament.

e) Com afecta al cicle d'execució de les instruccions que el processador gestioni les transferències d'E/S per interrupcions.
Els processadors que han de gestionar interrupcions han de tenir en el cicle d'execució d'instrucció una fase de ***comprovació d'interrupcions***.

Cicle d'execució d'instrucció
Recordeu que les quatre fases principals per a executar una instrucció són:
1) Lectura de la instrucció.
2) Lectura dels operands font.
3) Execució de la instrucció i emmagatzematge de l'operand destinació.
4) Comprovació d'interrupcions.

f) Explica quines tasques realitza la CPU des del moment que el perifèric sol·licita atenció fins que comença a executar-se la Rutina de Servei d'Interrupció.
En el moment en què el processador reconeix que ha arribat una petició d'interrupció, comença un cicle de reconeixement d'interrupció per a aturar l'execució del programa actual i transferir el control a la rutina de servei de la interrupció (RSI), rutina que accedeix al mòdul d'E/S corresponent per dur a terme la transferència de dades i un cop s'acabi l'execució de l'RSI continuar l'execució del programa que havíem aturat fent el retorn d'interrupció.

El cicle de reconeixement de la interrupció és segurament la part més complexa de la gestió d'interrupcions perquè es produeixen molts esdeveniments en poc temps i com veurem més endavant alguns d'aquests esdeveniments es poden produir encavalcadament en el temps i cal estar atent a qui genera l'acció, qui la rep i quina resposta hi dóna.
Reconeixement de la interrupció. Si les interrupcions estan habilitades, el processador accepta la petició. Aleshores cal que el processador avisi el mòdul d'E/S, activant la INTA, perquè sàpiga que l'està atenent i perquè desactivi la INT, i també cal que el processador inhibeixi les interrupcions per evitar noves peticions.

Si les interrupcions estan inhibides el processador no atén la petició i continua executant el programa. El perifèric s'ha d'esperar a ser atès i ha de deixar la INT activa fins que el processador les torni a habilitar.

Cicle de reconeixement de la interrupció.

a) Reconeixement de la interrupció.
b) Salvaguarda de l'estat del processador.
c) Crida a l'RSI.

g) En un sistema d'E/S gestionat per DMA, quines tasques realitza el mòdul d'E/S.

En aquesta tècnica el processador programa la transferència d'un bloc de dades entre el perifèric i la memòria encarregant a un nou element connectat al bus del sistema fer tota la transferència. Un cop acabada, aquest nou element avisa el processador.

D'aquesta manera el processador pot dedicar tot el temps que dura la transferència del bloc a altres tasques. Aquest nou element que gestiona tota la transferència de dades entre el perifèric i la memòria principal l'anomenem ***mòdul o controlador de DMA*** o també en versions més evolucionades canal o processador d'E/S.

h) Si s'està executant una subrutina en la CPU i un controlador d'E/S que actua per DMA demana el bus de dades. Es pot inhibir la cessió del bus? Es necessari que el processador guardi el valor dels registres un cop s'hagi cedit el bus? Raona la resposta.

No es pot inhibir la cessió del bus davant d'una petició de BUSREQ. El processador no cal que guardi el valor dels registres ja que el processador no passa a executar altres instruccions quan cedeix el bus: simplement congela l'execució d'instruccions durant el temps que té cedit el bus.