1.- Si el bit EXL val 1, les interrupcions seran ignorades. V

5.- Si el bit EXL (Exception Level) del registre Status val 1, el processador està en mode sistema i totes les excepcions resten inhibides. F

2.- En un sistema computador basat en el MIPS qualsevol programa pot inhibir les interrupcions, posant a 1 el bit ExL (bit 1) del registre Status (registre $12 del coprocessador 0) amb el codi: mfc0 $t0, $12 ori $t0, 2 mtc0 $t0, $12 F

10.- 4. Al processador MIPS hi ha certes adreces que si s'accedeixen en mode usuari (ExL=0) provoquen una excepció mentre que són accessibles normalment en mode sistema (ExL=1). V

3.- Si el bit EXL val 1, les excepcions seran ignorades. F

7.- La instrucció eret posa EXL a zero i copia al PC el contingut d'EPC. V

1.- Quan es produeix una excepció o interrupció, el bit EXL canvia per prohibir les interrupcions, indicant al mateix temps que estem en mode privilegiat (mode sistema). V

9.- Quan el processador es disposa a servir una excepció, posa el bit EXL=1, i copia l'adreça de retorn en el registre $ra. F

10.- Suposant que la rutina RSE no modifica el bit EXL, les peticions d'interrupció s'ignoren durant tota l'execució de la rutina. V

2.- Una excepció no pot ser atesa fins que la instrucció que l'ha causada hagi finalitzat. F

1.- Quan es produeix una excepció, el processador finalitza la instrucció en execució i a continuació avorta el programa. F

2.- En MIPS, la traducció d'adreces virtuals a físiques en una instrucció lw pot arribar a produir fins a quatre excepcions. V

8.- En MIPS, una instrucció lb mai pot produir una excepció per accés no alineat. V

5.- El processador MIPS genera una excepció quan es vol accedir a un registre segur que no s'ha guardat al bloc d'activació de la subrutina que s'està executant. El tractament de l'excepció permet guardar el valor del registre segur a la pila. F

8.- El tractament d'una excepció i el d'una interrupció requereixen accions similars, amb la diferència que el tractament d'una excepció el fa el mateix processador mentre que el tractament d'una interrupció el fa un dispositiu d'entrada/sortida. F

1.- La diferència entre les instruccions MIPS add/addu radica en què la instrucció add pot generar una excepció per sobreeiximent (overflow) a la suma d'enters mentre que la instrucció addu mai pot generar cap excepció. V

6.- Entre les accions software que es realitzen en cas d'excepció o interrupció, ens trobem amb que es salva el registre PC (a l'EPC) per tornar després al programa cas de ser necessari. F

2.- Quan es produeix una interrupció s'atura la instrucció en marxa, i quan s'ha acabat el servei a la interrupció es torna a llançar l'esmentada instrucció. F

8.- Una interrupció no pot ser atesa fins que la instrucció en curs hagi finalitzat. V

6.- L’excepció per accés no alineat a memòria pot ser inhibida a través del camp Interrupt Mask. F

10.- L’excepció de divisió per zero pot ser inhibida a través del camp Interrupt Mask. F

6.- L’excepció per accés no alineat a memòria pot ser inhibida a través del camp Interrupt Mask. F

9.- En el MIPS, el camp IM (Interrupt Mask) del registre Status usat en la gestió de les interrupcions serveix per indicar les peticions d’interrupció que no han de ser ateses quan s’acabi l’execució de la instrucció actual. V

4.- La divisió d’enters codificats en el format de Ca2 no pot produir overflow. F

4.- Si un cert número enter de 8 bits es representa en Ca2 per 0x8B, el mateix número es representa en Ca1 per 0x8C. F

4.- Si faig un producte d’enters amb operadors i resultat de 32 bits per mitjà de la instrucció mult, puc assegurar que si $hi és diferent de zero s’ha produït overflow. F

3.- Dividir per 2 un número enter en complement a 2 que estigui a $t0 és sempre equivalent a fer: sll $t0,$t0,1 . F

6.- En una memòria cache amb política d’escriptura immediata sense assignació, un accés a la memòria cache pot implicar dos accesos a memòria principal. F

1.- Si en una cache canviem la política d'escriptura immediata amb assignació a retardada amb assignació, sense cap més canvi, el nombre total de fallades no canvia. V

4.- Les escriptures del bit de Dirty del TLB segueixen una política d’escriptura immediata. V

5.- En les memòries cache que utilitzen escriptura immediata s’ha de posar el dirty bit a 1 només quan hi ha un encert d’escriptura. F

3.- En un sistema de cache amb escriptura retardada amb assignació (copy-back write allocate) una fallada en escriptura a la cache fa que ens portem el bloc de la memòria principal a la cache, i escrivim la dada modificada tant a la cache com a la memòria principal. F

7.- Quan s’executa una instrucció store en un sistema basat en el processador MIPS s’utilitza una política d’escriptura retardada per fer l’escriptura de la pàgina implicada al disc i s’utilitza una política d’escriptura immediata per gestionar el bit D dins la taula de pàgines. V

7.- En una subrutina, una variable local de tipus enter sempre es guardarà en un registre. F

3.- Suposem un sistema operatiu que funciona correctament, i modifiquem la RSE així: copiem tot el codi íntegre de la RSE en una altra subrutina que anomenem RSE2, i reduïm el codi de la RSE a una sola instrucció: "jal RSE2". Després del canvi, els programes seguiran funcionant igual de bé que amb la RSE original. F

10.- Si una subrutina f està declarada com void f(float *p) el paràmetre p s'ha de passar en el registre $f12. F

9.- Una rutina ha de salvar el registre $ra a la pila si aquesta rutina crida una altra rutina. V

3.- En un sistema amb memòria virtual, la mida total d'un programa i les seves dades poden excedir la capacitat de la memòria física. V

3.- En un sistema amb memòria virtual, la mida total d'un programa i les seves dades no pot excedir la capacitat de la memòria física. F

2.- En memòria virtual paginada, sempre que reemplacem de la memòria física una pàgina cal escriure-la en disc. F

10.- La codificació en excés de l'exponent en el format de coma flotant simplifica les operacions de comparació. V

5.- Si el resultat d'un càlcul en format normalitzat de coma flotant causa underflow, es pot reduir l'error absolut de precisió si el podem expressar en format denormal. V

10.- La codificació en excés dels exponents dels nombres en coma flotant permet que la comparació de les seves magnituds es pugui fer amb un comparador de nombres naturals. V

1.- El número real 1.2 no pot ser representat amb el format de coma flotant IEEE 754 sense pèrdua de precisió. V

1.- Multiplicar per 2 un número en coma flotant que estigui a $f0 és equivalent a fer: sll $f0,$f0,1 F

7.- Un programa en mode usuari pot copiar un registre qualsevol de la CPU al coprocessador CP0 per mitjà de la instrucció mtc0. F

3.- La instrucció mtc1 $t0, $f0 converteix el número enter que hi ha a $t0 al seu equivalent en real al registre $f0 (per exemple, si $t0=1 aleshores $f0=1.0). F

2.- Al format de coma flotant IEEE 754 de simple precisió, l'exponent 255 representa tant infinit, com menys infinit com el concepte NaN (not a number). V

4.- El número real en binari 1.00000000000000000011 pot ser representat sense pèrdua de precisió en coma flotant (IEEE754) però no en coma fixa de 1 bit de signe, 19 per la part entera i 12 per la fraccionària. V

5.- En format de simple precisió IEEE-754 (32 bits), la codificació 0x00F00000 representa un número normalitzat. V

4.- El número real en binari 1.00000000000000000011 pot ser representat sense pèrdua de precisió en coma flotant de simple precisió (IEEE-754) però no en coma fixa de 1 bit de signe, 19 de part entera i 12 de fraccionària. V

7.- Segons la representació de números en coma flotant IEEE-754 amb simple precisió el valor més gran representable just per sota del +infinit es representa 0x7f7fffff. V

8.- La rutina RSE de tractament d'excepcions del MIPS segueix les regles de l'ABI que s'estableixen per programar les subrutines. F

6.- A l'inici de la rutina genèrica de servei d'excepcions de MIPS (RSE) aquesta sols ha de salvar a la pila aquells registres segurs que es modifiquin durant l'execució de la RSE. F

1.- Si la CPU rep una petició d'interrupció per part d'un dispositiu d'E/S, la instrucció en curs s'interromp sense arribar a produir cap resultat, i la CPU passa a executar la rutina RSE. F

6.- En MIPS, quan s'acaba d'executar la RSE, es torna a la instrucció següent a la que ha provocat l'excepció o la interrupció. F

8.- La RSE estudiada al tema 8 per al processador MIPS s'executa com a tractament de totes les excepcions. F

4.- La RSE salva els registres temporals (de tipus $t), però no els segurs (de tipus $s) X 5.- El processador MIPS no dóna suport a través del seu ISA a una gestió software de les fallades de TLB. F

6.- Si mentre s'està tractant una excepció es produeix una altra excepció sempre es passarà a atendre aquesta darrera excepció i es tornarà a començar a executar la RSE. V

1.- La rutina RSE coneix la causa (codificada) de l'excepció perquè aquesta se li passa com a paràmetre en $a0 (paràmetre ExcCode). F

5.- Un cop s'acaba d'executar la RSE, o bé s'avorta el programa o es torna a la instrucció següent a la que va provocar la RSE. F

1.- La rutina RSE de tractament d'excepcions del MIPS segueix les regles de l'ABI estudiades al Tema 3, és a dir que pot modificar lliurement els registres $t, $a i $v, però ha de preservar el valor original dels registres $s i $ra. F

2.- La rutina RSE coneix la causa (codificada) de l'excepció perquè aquesta se li passa com a paràmetre en $a0 (paràmetre ExcCode). F

7.- Un cop finalitzada la rutina RSE en ocasió d'una fallada de pàgina, es retorna al programa interromput executant la instrucció següent a la que ha causat el fallo. F

6.- Una rutina de servei a excepcions pot acabar amb jr $epc doncs al registre $epc es guarda l'adreça de la instrucció que cal seguir executant un cop s'ha atès l'excepció. F

5.- Les fallades de TLB en el MIPS provoquen una excepció i s'executa la RSE genèrica. F

9.- En un computador basat en el processador MIPS les fallades de TLB es gestionen a través d'una excepció que té un tractament optimitzat i específic diferent del tractament genèric que fa la RSE. V

8.- En MIPS les fallades de TLB causen una excepció i es tracten per software. No obstant, les fallades de TLB invoquen la rutina TLBmiss, molt més curta d'executar que la rutina d'excepcions genèrica RSE. V

9.- Al MIPS es detecta que un accés a memòria causa una fallada de pàgina consultant el bit V en el TLB. V

2.- Si l'accés a dades d'una instrucció produeix un encert al TLB, però el bit V val 0, llavors la instrucció causarà una excepció de fallada de pàgina. V

4.- La instrucció tlbwr escriu una entrada de la taula de pàgines a l'entrada del TLB que resulta d'aplicar un algorisme de reemplaçament RANDOM. V

2.- Una fallada al TLB no implica que hi hagi una fallada de página. V

8.- El tractament de les excepcions al MIPS es fa únicament en dues rutines de servei, una per a la fallada de TLB i una altra per a la resta d'excepcions. V

3.- Quan cerquem la traducció d'un número de pàgina (VPN) al TLB, perquè hi hagi un encert de TLB cal trobar una entrada amb el mateix VPN i que tingui el bit de presència V=1. F

9.- En un computador que disposa de memòria virtual amb TLB sempre cal fer dos accessos a memòria principal per cada referència: un a la Taula de Pàgines i l'altre a l'adreça de la referència. F

3.- En un sistema computador basat en el MIPS pot passar que al final de l'execució d'algunes instruccions el bit D del TLB per a alguna VPN no sigui coherent amb el bit D de la Taula de Pàgines associat a la mateixa VPN. F

4.- La instrucció tlbwr del MIPS escriu el contingut d'una entrada de la Taula de Pàgines al TLB, en aquella entrada del TLB que fa més temps que no s'accedeix en aplicació de l'algorisme LRU. F

5.- La rutina TLBmiss estudiada al tema 8 s'executa quan hi ha una fallada de pàgina. F

1.- Si l'accés a dades d'un store produeix un encert al TLB, però el bit D val 0, llavors es produeix una excepción. V

2.- Si al traduir una adreça amb el TLB es troba una entrada amb el mateix VPN però amb el bit V que val 0, llavors es produeix una fallada de página. V

4.- Si s'executa la instrucció tlbwr en mode usuari, es produeix una excepción. V

5.- Després d'una fallada de TLB la instrucció causant de l'excepció s'ha de reexecutar. F

8.- El bit D del TLB en el processador MIPS estudiat al tema 8 és sempre coherent amb el bit D de la TP. X 9.- Una excepció no pot ser atesa fins que la instrucció en curs hagi finalitzat. F

4.- Es pot produir més d'una excepció per fallada de TLB en l'execució d'una instrucció. V

9.- Un processador sense TLB pot suportar memòria virtual. V

2.- En l'execució d'una instrucció es poden produir com a màxim una fallada de TLB i una fallada de pàgina. F

4.- La rutina TLBmiss copia l'entrada de la taula de pàgines al TLB sense examinar si el bit de presència P (a la taula de pàgines) val 1 o 0. V

5.- Es pot produir més d'una excepció per fallada de TLB en l'execució d'una instrucció. V

4.- La MMU del MIPS detecta que un accés a memòria causa una fallada de pàgina consultant el bit V en el TLB. V

5.- La rutina TLBmiss del processador MIPS estudiada al Tema 8 copia tots els camps de la TP al TLB excepte el bit de presència, que es posa sempre a 1. F

6.- El TLB és com una cache de la TP, les etiquetes de la qual consisteixen en el número de pàgina virtual (VPN). V

7.- En el MIPS, una mateixa instrucció pot causar durant la seva execució 2 fallades de pàgina. V

10.- En el MIPS, una mateixa instrucció pot causar durant la seva execució 2 fallades de página. V

3.- Una mateixa instrucció pot causar durant la seva execució 2 fallades de pàgina. V

3.- En el MIPS, una mateixa instrucció pot causar durant la seva execució 2 fallades de pàgina. V

3.- Quan hi ha una fallada de pàgina hi haurà un o dos accessos a disc depenent del valor del Dirty Bit de la pàgina a substituir V

8.- Un sistema que tingués una memòria cache més gran que la memòria principal mai tindria fallades. F

9.- El temps mitjà d'accés a memòria de les instruccions load i store en un computador que disposa de memòria virtual amb TLB i memòria cache, és inferior al temps d'accés de la memòria principal. V

7.- Si mantenim la mida de la cache constant, el temps mitjà d'accés a memòria sempre és inversament proporcional al grau d'associativitat de la memòria cache. F

6.- El temps de penalització en cas de fallada d'escriptura d'una memòria cache d'escriptura immediata sense assignació és zero segons el model estudiat. V

3.- En un processador amb adreces de 32 bits, una cache associativa de 4 vies, de 32KB i blocs de 32 bytes, s'han de dedicar 20 bits a etiqueta (TAG), 7 a número d'entrada (conjunt)i5a desplaçament. F

1.- En un processador amb adreces de 32 bits, una cache associativa de 4 vies, de 32KB i blocs de 32 bytes, s'han de dedicar 19 bits a etiqueta (TAG), 8 al número d'entrada (conjunt)i 5 al desplaçament. V

2.- En un processador amb adreces de 32 bits, una cache associativa de 2 vies, de 256KB i blocs de 16 bytes, s'han de dedicar 15 bits a etiqueta (TAG), 13 a número d'entrada (conjunt)i4a desplaçament. V

7.- La sincronització dels dispositius d'entrada/sortida en un computador, es gestiona exclusivament per interrupcions. F

10.- Les crides al sistema són excepcions que es produeixen quan el sistema operatiu requereix l'atenció del processador. F

1.- En un programa escrit en assemblador MIPS, que consta de 2 mòduls A i B que es compilen per separat, i on el mòdul A invoca una funció func que está declarada al mòdul B, l'assemblatge del mòdul A fallarà sense generar cap fitxer objecte. F

8.- L'execució d'un bucle traduït literalment amb una estructura de control FOR és menys eficient que si el mateix bucle s'executa havent-lo traduït amb una estructura de control DO_WHILE. V

9.- Una de les funcions de la instrucció OR és la de poder posar un bit o grup de bits d'un registre a valor 0. F

10.- La tècnica d'accés seqüencial a elements d'una matriu només es pot aplicar si entre dos elements qualsevols consecutius de la seqüència d'accessos hi ha una distància constant. V

7.- Si s'executa un mateix programa en dues memòries cache diferents, però les dues de correspondència directa, sempre tindrà una major taxa d'encerts la que tingui els blocs de mida més gran. F

10.- Si a un procediment o funció en C se li passa una matriu com a paràmetre, s'ha de definir la segona dimensió de la mateixa per poder calcular correctament l'accés a la matriu, però la primera dimensió no és necessària (per exemple, la capçalera void funcio(int matriu [][10]) és correcta). V

2.- Considerant un computador amb CPU MIPS, memòria virtual paginada amb pàgines de 4KB i un TLB que conté una entrada amb els valors V=0, D=0, VPN=0x00403, PPN=0x17, si fem un accés a memòria a l'adreça 0x00403A50 es produirà una fallada de TLB. F

6.- Treballem a una empresa que produeix un compilador de C a MIPS i hem decidit afegir una opció que permet escollir quins registres s'utilitzen com registres segurs de manera que l'usuari pot demanar que es facin servir els registres $t0,$t1,... enlloc dels registres $s0,$s1,... Tothom que recompila els seus programes amb aquesta opció utilitzant registres "t" és incapaç de tornar a executar molts d'aquests programes. V

1. La RSE estudiada al tema 8 per al processador MIPS s'executa .

d) Totes les anteriors són correctes

2. Una interrupció

d) pot ser ignorada fent ús del bit corresponent del camp Interrupt Mask

3. La rutina TLBmiss estudiada al tema 8

d) retorna sempre a la instrucció que ha provocat l'excepció

4. El camí de dades del processador MIPS unicicle estudiat al tema 8

c) inclou un camí que connecta la sortida del registre PC amb l'entrada del registre EPC

5. L'excepció per accés no alineat a memoria

a) no es pot produir en l'execució d'una instrucció sb

b) és tractada de forma que s'avorta l'execució del programa

c) no pot ser inhibida a través del camp Interrupt Mask

6. El bit D del TLB en el processador MIPS estudiat al tema 8

a) és sempre coherent amb el bit D de la TP

a) Quines condicions s'han de complir en el processador perquè accepti la petició d'interrupció del dispositiu número 3 en un moment donat, i inicïi les accions pertinents per servir-la?

Que en el registre Status del CP0, el bit EXL valgui 0, i el bit 3 del camp Interrupts Mask (IM) valgui 1

b) Quins registres ha de salvar obligatòriament la Rutina de Servei d'Excepcions?

Tots, excepte $k0, $k1 (que estan reservats al SO), i excepte els registres de coma flotant $f del CP1 (que sols se salvarien en el cas que es facin servir)

c) Com ho fa la Rutina de Servei d'Excepcions per identificar la causa de l'excepció?

Consultant el codi contingut al camp ExcCode del registre Cause

d) Quina instrucció executa la Rutina de Servei d'Excepcions per retornar al punt on s'ha interromput el programa?

eret

Què fa aquesta instrucció?

Posa a zero el bit EXL del registre Status, i copia el valor del registre EPC en el PC