5. Resultats.

A continuació es mostraran els resultats del Simulador de l'arquitectura MIPS unicicle. Per a l'elaboració d'aquest capítol el que s'ha fet es introduir al camp d'instruccions del programa la instrucció **load word (lw)**. El fet d'utilitzar aquesta instrucció i no una altra, és degut a que és la que més passos utilitza, a més d'utilitzar les dues finestres de registres i memòria.

Ara el que es farà serà mostrar pas a pas tots el canvis sorgits al llarg de l'execució del simulador.

Aleshores, comencem introduïm el codi, escrit en binari, de la instrucció, per al nostre cas, la instrucció és:

lw $t0, 1200($t1)

la instrucció es correspon amb el tipus I, més concretament és una instrucció de transferència de dades, com el lector podrà saber de la lectura del capítol 2,. la qual cosa significa que a l'arquitectura el que s'ha de fer és el següent: primer calcular l'adreça d'on extraure la dada, açò és, sumar l'immediat (la constant 1200 multiplicada per quatre degut a que no ens menegem basant-se en *bytes* sinó per words, que són conjunts de 4 *bytes*) amb el contingut del registre $t1, aquesta suma ens dóna el valor de l'adreça on la dada que es vol extraure es troba. Per últim, tan sols farà falta extraure la dada i guardar-la dins del contingut del registre $t0.

Per tant el primer pas és executar el simulador, al fer-ho ens apareix el mostrat a la figura 5.1. Com es veu a la figura, ens apareixerà la finestra principal del programa amb, a la qual haurem d'introduir la instrucció a realitzar. Si la instrucció ha sigut introduïda incorrectament, la instrucció no ha estat definida o hem posat un text que res té a veure amb el simulador, ens apareixerà un missatge per pantalla indicant el problema i el seu origen. Com a exemple, la següent figura ens mostra el missatge que es produeix quan hem introduït una instrucció incorrectament:

Missatge d'error.

Una vegada introduïda la instrucció, polsarem el botó d'*Acceptar* i seguidament ens apareixerà el dibuix de l'arquitectura i el botó *Següent*, el qual com ja sabem del capítol anterior, ens servirà per anar incrementant els passos pels que passa la instrucció.

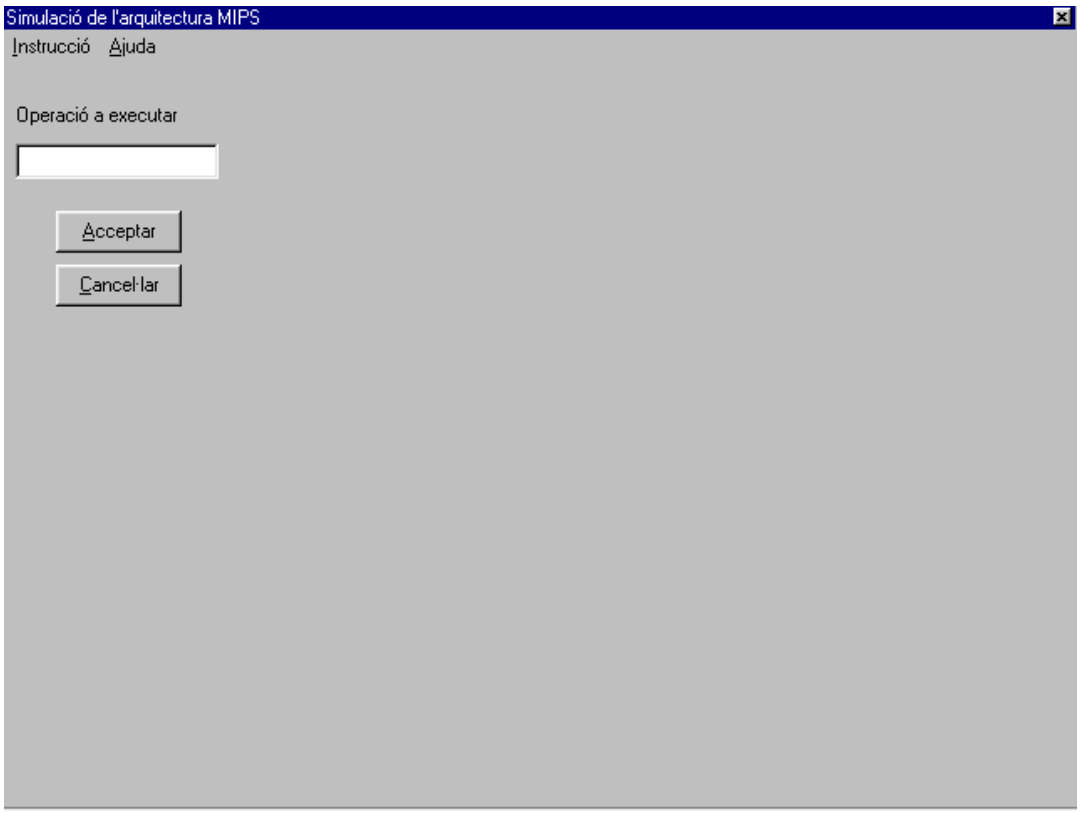


Figura 5.1.

Tal com es veu a la figura 5.2., una vegada hem introduït la instrucció de manera correcta, la instrucció quedarà escrita també, a manera de recordatori, a la capçalera de la finestra principal. D'aquesta manera, com que l'*Edit* de la instrucció desapareix per a que no puga ser modificat en temps d'execució de la instrucció, sempre tindrem en compte la instrucció introduïda per a ser simulada.

Ara abans de polsar el botó *Següent*, canviarem els valors dels registres i de les posicions de memòria que després utilitzarem per poder executar la instrucció.

Per poder visualitzar ambdues finestres podem utilitzar el ratolí polsant primer damunt de *Instrucció*, i després fer el mateix damunt de Registres o Memòria, segons es desitge, o directament polsant les tecles abreujades *CTRL+R*

(per obrir la finestra dels registres) o *CTRL+M* per a accedir a la finestra de les dades emmagatzemades a la memòria.

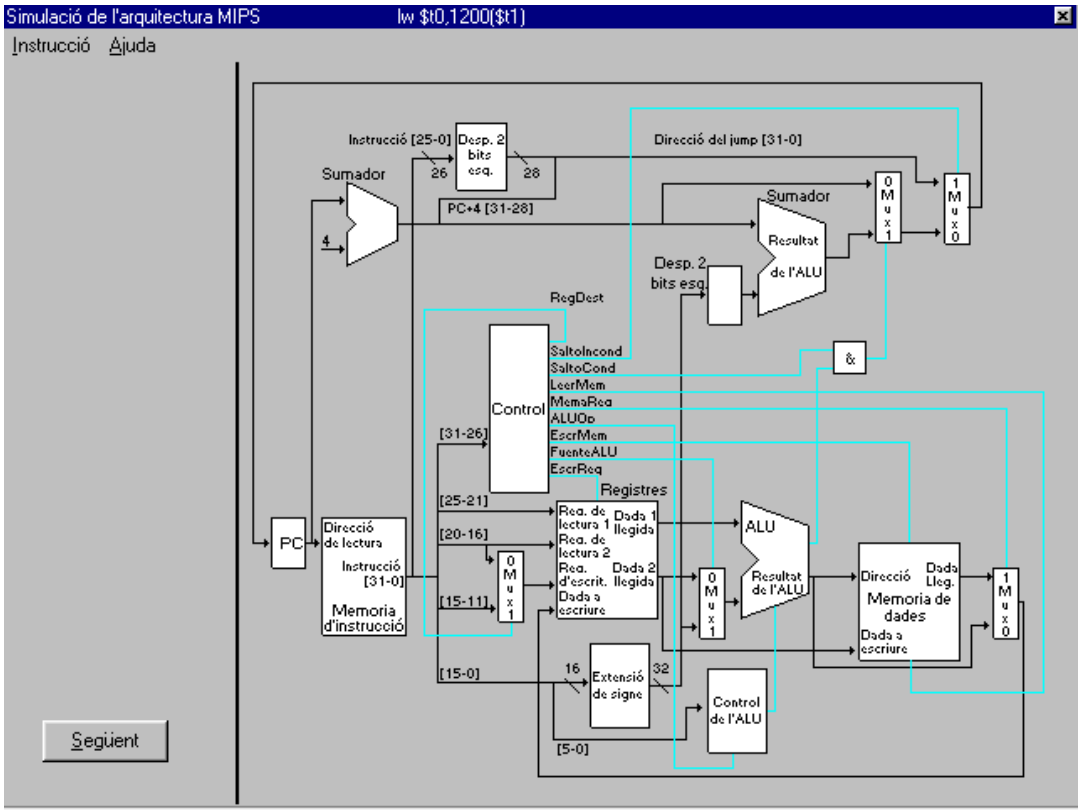
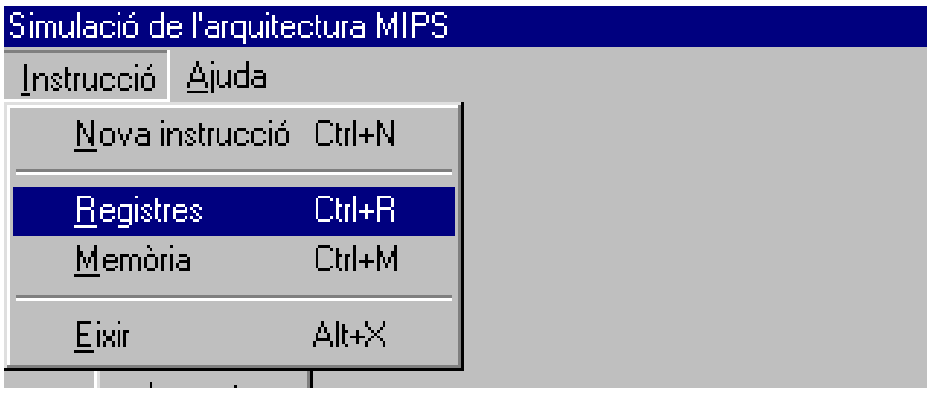


Figura 5.2.

Les dues finestres amb els canvis introduïts els podem observar a les figures 5.3. i 5.4., les quals mostren les finestres dels registres i de la memòria, respectivament. A la figura següent s'observa com podem triar qualsevol de les dues finestres des del menú despegable:



Ara ja podem realitzar l'execució de la instrucció pas a pas, com ja s'ha comentat abans polsant una vegada el botó *Següent* per pas.

Figura 5.3: Finestra dels registres.

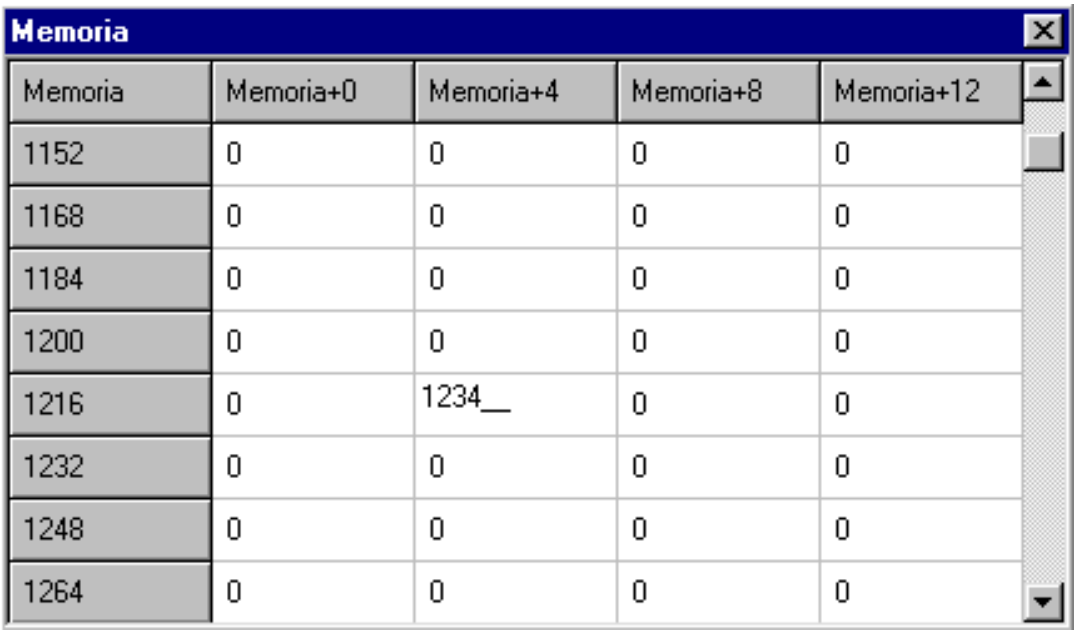


Figura 5.4: Finestra de la memòria de dades
introduint un valor per al exemple.

Seguint de nou a partir de la figura 5.2., al polsar ara el botó *Següent*, apareixerà, al cos de la finestra principal, el dibuix de l'arquitectura, l'activació de les línies corresponent al primer pas i a més les parts que formen part de la instrucció. El resultat de tot el que s'ha anat dient , es pot observar a la figura 5.5.

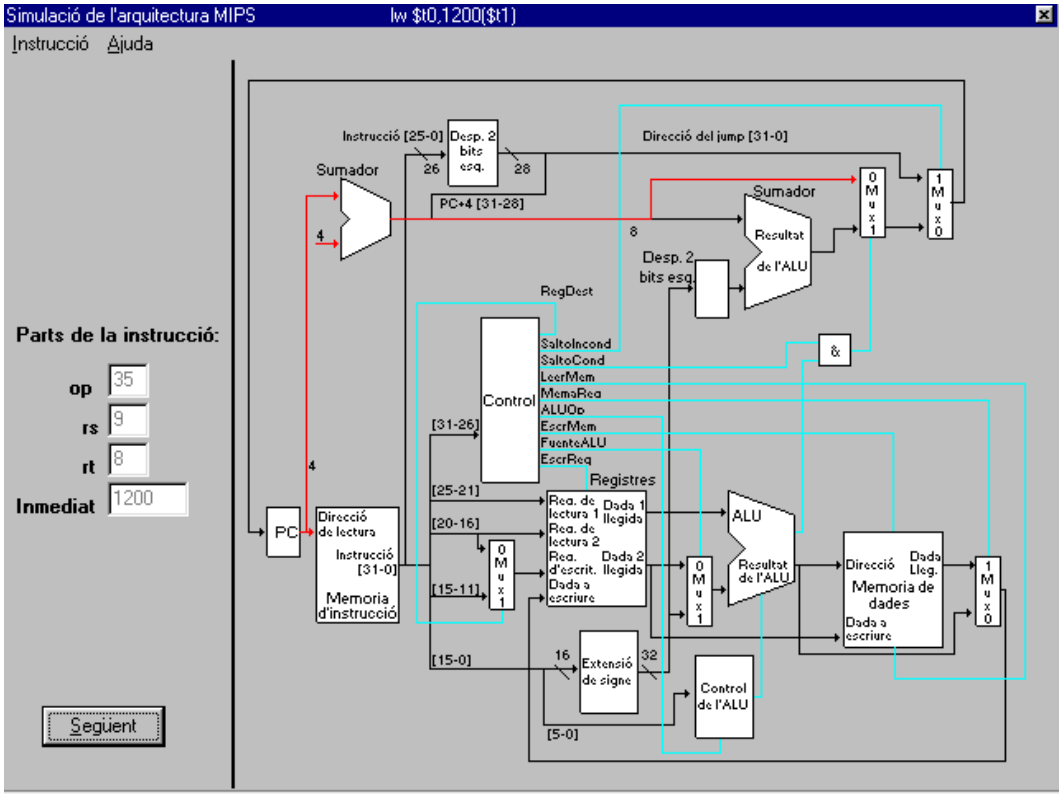


Figura 5.5: Primer pas de la instrucció *lw*.

Respecte a la descomposició en parts de la instrucció, com ja es sap per la lectura del capítol 2, la instrucció constarà de 4 camps:

- Camp d'operació, que per a la instrucció load word es correspon amb el nombre decimal 35.

- El registre rs, o siga el registre amb el que es va a fer la suma amb l'immediat és el registre $t1, que com es pot observar al capítol 2, el nombre de registre MIPS és 9.

- El registre rt, és on es guardarà la dada llegida des de la memòria, en aquest cas es correspon amb el registre $t0, que a l'arquitectura MIPS és el mateix que 8.

- El camp immediat, la constant 1200, la qual es sumarà amb el registre $t1 per trobar l'adreça, com ja hem comentat anteriorment. Aquest camp, com s'observa i com ja s'ha dit abans al capítol 2, consta de 16 bits.

Doncs amb totes aquestes dades, el que s'observa és el següent:

35	9	8	1200

mentre que en codi binari el resultat seria el següent;

100011	01001	01000	0000 0100 1011 0000

Com que al registre $t1 el qual s'ha de sumar a la constant 1200, s'ha introduït un 20, l'adreça d'on s'intentarà extraure la dada és 1220, per tant si introduïm una dada a aquesta posició de memòria, tal com mostra la figura 5.4, després d'haver-se desplaçat mitjançant la barra desplaçadora situada a la dreta de la finestra:

Una vegada feta aquesta aclaració, es passa al següent pas, per realitzar-ho, polsarem damunt del botó *Següent*. Apareixerà per pantalla aquesta nova finestra de la figura 5.6.

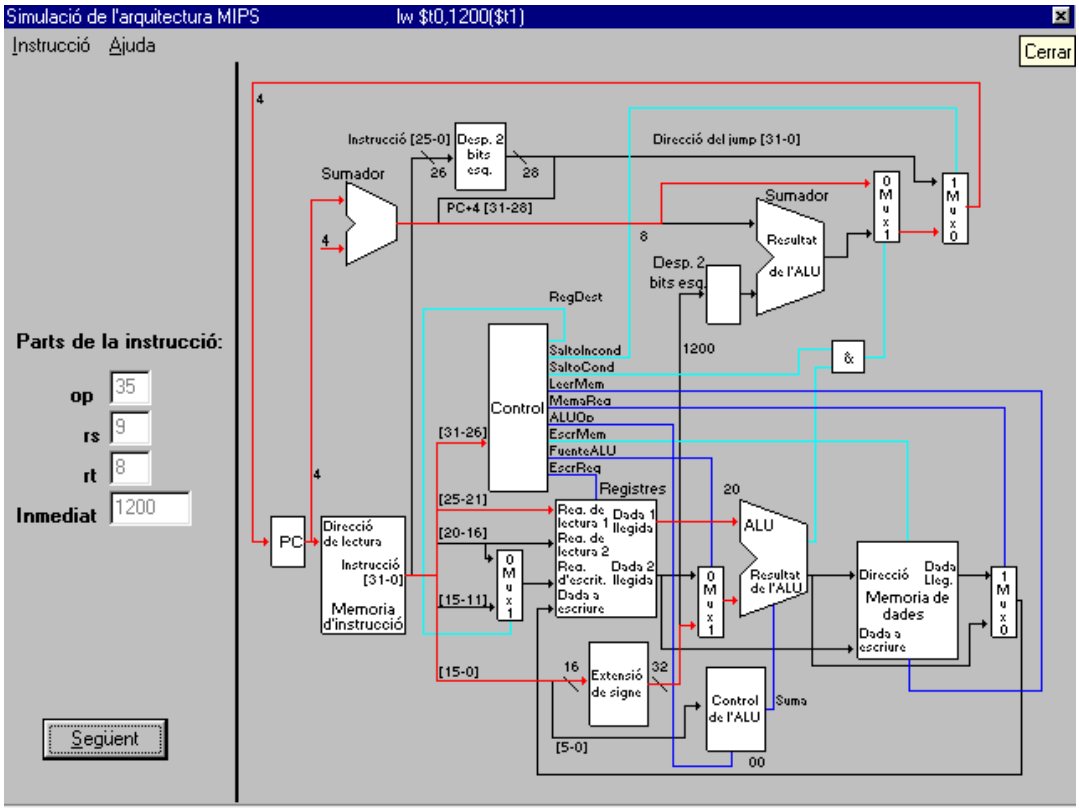


Figura 5.6.

Com s'observa a la vista de la figura anterior en aquest pas s'activen les senyals de control necessàries per fer funcionar els blocs que es necessiten per a l'execució de la instrucció. L'estat d'aquestes línies el podem saber veient l'última part del capítol 2, corresponent a l'activació de les línies de control. Per a una ressaltació i una menor confusió per part de l'usuari, les línies de control canvien a un blau més intens, indicant així la seua activació.

Respecte a les línies que uneixen els blocs del diagrama, com podem observar s'activen les següents:

- El PC ja s'incrementa amb el valor PC+4.

- Al control s'introdueixen els 6 bits del codi d'operació, en aquest cas 100011_2 i que és realment el que activa les senyals de control.

- Al banc de registres s'introdueix el nombre del registre a llegir, com sabem va a ser el $t1, el qual té com a valor decimal 9, que també l'anomenem rs, el contingut aquest registre com ja s'ha dit abans va ser sumat al camp immediat. Com es veu a la entrada de l'ALU es troba el nombre 20, que es el contingut del registre $t1 i que va a ser sumat amb el camp immediat.

- El camp immediat esmentat entra al bloc d'extensió de signe, explicat al capítol 2, en el que de 16 bits es passa a 32 per a poder sumar amb el contingut del registre rs. Com vegem ambdós s'introdueixen a l'ALU principal.

- Dita ALU principal el que realitzarà és l'operació que l'indica el seu propi control, que com ja sabem del capítol 2, consisteix a sumar els dos valors anteriors, o siga rs i l'immediat. Com també s'observa el codi aplicat pel control de l'ALU principal per sumar ambdós valors és 010_2.

Al polsar de nou el botó *Següent*, s'efectuarà un nou canvi de pas, tal com s'observa a la figura 5.7.

Tal com es pot observar a aquesta figura aquest tercer pas tan sols suposa el canvi, o millor dit, l'activació d'una línia, que com es veu a la figura 5.7.es tracta de l'eixida de l'ALU principal, que s'ha encarregat de, com s'ha dit al pas anterior, sumar el contingut del registre $t1 amb la constant immediata introduïda al codi de la instrucció.

Realment aquest pas el que produeix és l'accés a la posició de memòria 1220 de la memòria de dades de l'arquitectura, que com ja s'ha comentat amb anterioritat és el resultat de l'operació suma efectuada per l'ALU. Com s'ha vist

a la figura 5.4. en aquesta posició es troba el valor que volem emmagatzemar al registre $t0, que no és altre que el valor 1234.

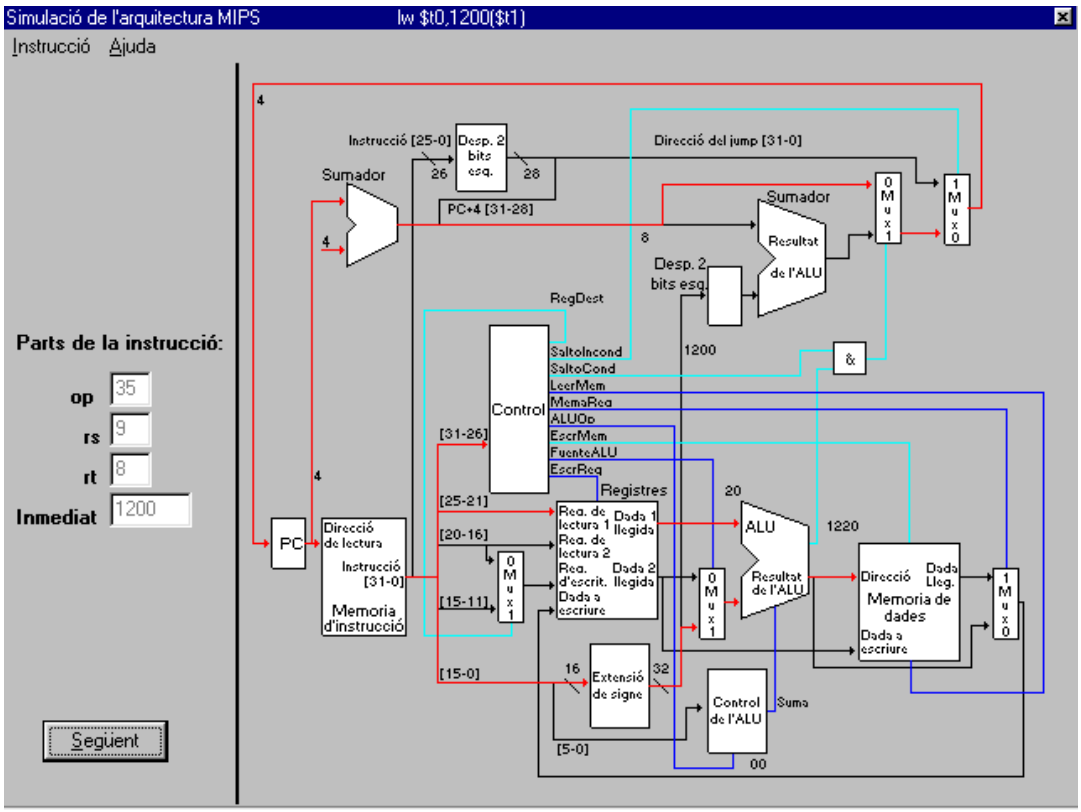


Figura 5.7: Tercer pas de la instrucció *lw*.

Polsant de nou el botó *Següent* ens trobem al pas 4, la imatge gràfica del qual la trobem a la figura 5.8. de la pàgina següent.

Com es pot observar, el que es pretén a aquest quart pas és extraure de la memòria de dades de l'arquitectura la dada que necessitem, la qual es troba a la posició 1220. Doncs en aquest pas s'accedeix a aquesta posició de memòria i s'extrau la dada tal com es veu a la figura 5.8. i aquest valor es situa a l'eixida, on a l'últim pas s'introduirà com ja es conegut al registre $t0 del banc de registres.

Com s'observa amb l'activació de la línia també s'activa el text amb el valor extret de la memòria (valor 1234).

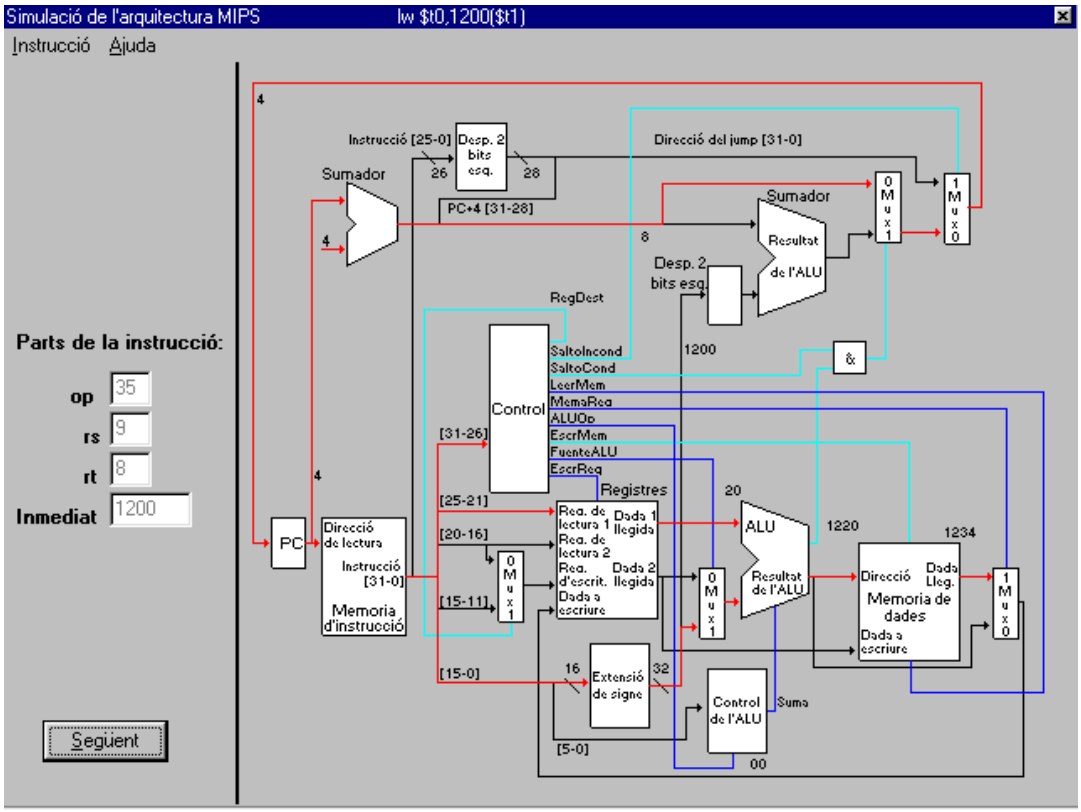


Figura 5.8.

Per últim passem al quint i darrer pas, que gràficament podem observar a la figura 5.9. de la següent pàgina.

Tal com s'observa, en aquest pas es finalitza la instrucció, s'activa la línia que va des del multiplexor que tria entre les senyal de l'eixida de l'ALU i de la memòria de dades fins al banc de registres, aquesta línia com és fàcil suposar, contindrà el valor 1234 que serà emmagatzemat al registre $t0, complint així la tasca que li s'havia demanat a la instrucció *load word*.

Un detall que té aquest simulador és que la línia amb el registre on es va a emmagatzemar la dada 1234, en aquest exemple el registre $t0, no s'activa fins a aquest darrer pas. Açò realment no ocorre així si realitzàrem de manera física l'arquitectura, igual que també les dues línies d'eixida de l'ALU deurien activar-se, ja que realment contenen el mateix valor (i és després el multiplexor esmentat al paràgraf anterior el que s'encarrega de decidir entre ambdues dades). La raó fonamental de perquè s'ha seguit aquesta filosofia es per a la comprensibilitat visual ràpida del funcionament de l'arquitectura.

Si tornem a polsar el botó *Següent*, ens eixirà per pantalla un missatge indicant que la simulació ha arribat a la seua finalització. Una vegada dins d'aquesta finestra del missatge polsem acceptar, desapareixerà el botó *Següent* al

mateix temps que podrem visualitzar de nou el contingut dels registres i de la memòria de dades.

L'aspecte de la finestra del missatge és el mostrat a la figura 5.10.

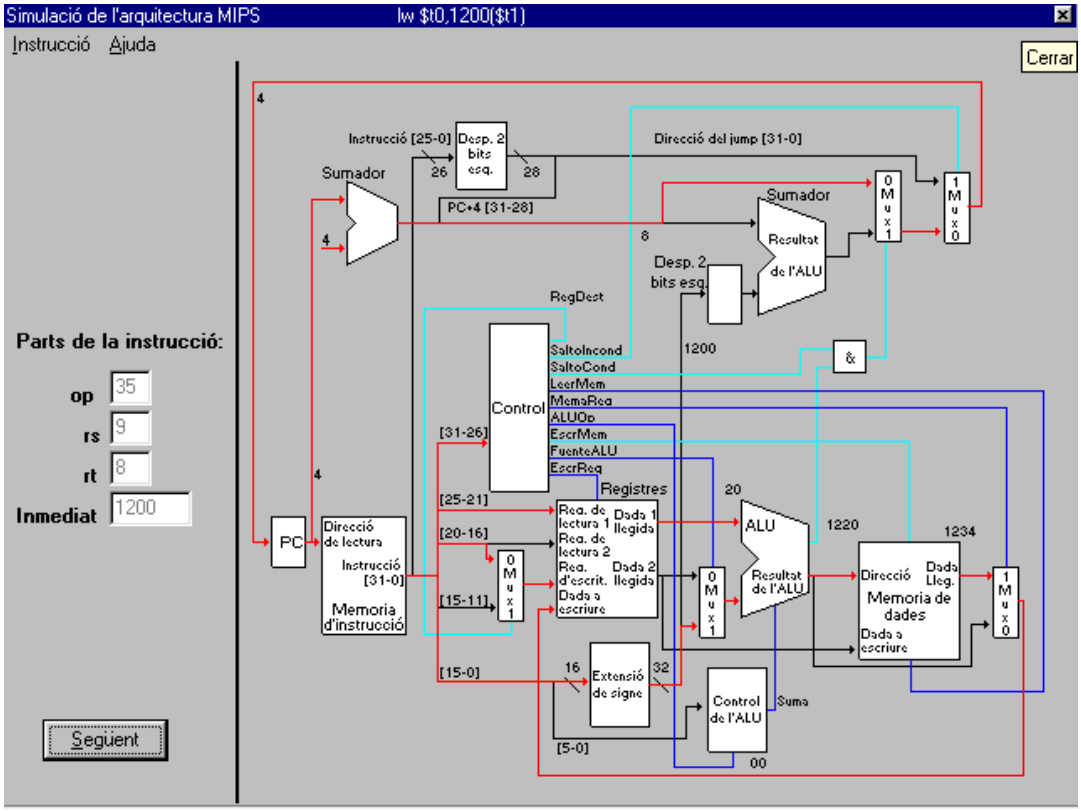


Figura 5.9: Darrer pas de la simulació amb la instrucció *lw*.

Figura 5.10: Indicació del final de la simulació.

Ara per confirmar que la instrucció s'ha efectuat de manera correcta, obrirem de nou la finestra que mostra el contingut dels registres. Per a realitzar açò, obrirem el menú desplegable amb l'opció *Instrucció* i obrirem el submenú

Registres. A la figura 5.11., s'observa l'aspecte d'aquesta finestra una vegada finalitzada la instrucció.

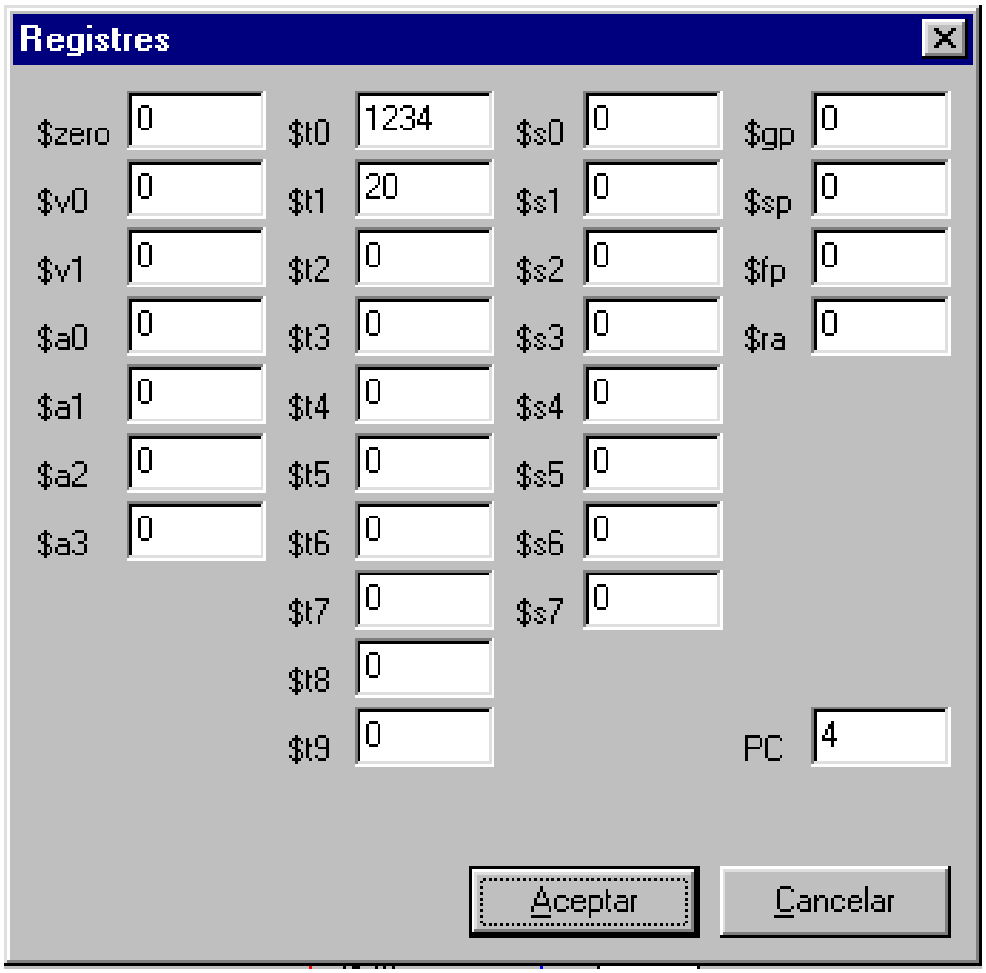


Figura 5.11.

També s'observa, a la vista de la figura 5.11., que el PC ha augmentat en 4, indicant així que es passaria a una hipotètica següent instrucció.

En aquest capítol s'ha observat el correcte funcionament del simulador a més d'una explicació detallada pas a pas de la instrucció *lw*. Cal dir en aquest punt que les instruccions acceptades al simulador són les següents:

mult Multiplica dos registres.
multu Multiplica dos registres de tipus unsigned.
div Divideix dos registres.
divu Divideix dos registres de tipus unsigned.

add	Suma dos registres i ho guarda a un tercer.
addu	Igual que l'anterior però de tipus unsigned.
sub	Resta dos registres i ho guarda a un tercer.
subu	Igual que l'anterior però de tipus unsigned.
slt	Compara dos registres.
sltu	Igual que l'anterior però de tipus unsigned.
j	Bot incondicional.
beq	Bot si dos registres són iguals.
bne	Bot si dos registres contenen diferents valors.
addi	Suma amb immediat,
addiu	Igual que l'anterior però de tipus unsigned.
lw	La instrucció exposada a aquest capítol.
sw	Guarda un valor a la memòria de dades

El funcionament d'aquestes instruccions s'explica al capítol 2.