***Pràctica 2***

**Objectius**

L’objectiu d’aquesta pràctica és familiaritzar-nos amb el funcionament dels registres que hi ha a la CPU.

A més de veure estructures condicionals i analitzar la operació de multiplicar per veure de quina manera es pot fer més eficient.

**Explicació de la practica**

Aquesta segona practica consisteix en fer 3 exercicis i respondre a les preguntes plantejades a més dels 2 exercicis del pdf  [Teoricopràctic 2 on s’havien de fer dos programes en el Ripes.](https://campusvirtual.ub.edu/mod/resource/view.php?id=3407262&redirect=1" \t "_blank)

**Informe:**

Exercici 1:

**Un cop acabat el programa quant val el contingut de A5?**

El contingut de A5 un cop acabat el programa és 12.

**Un cop acabat el programa quant val el contingut de A6?**

El contingut de A6 un cop acabat el programa és 4.

**Un cop acabat el programa quant val el contingut de A3?**

El contingut de A3 un cop acabat el programa és -1.

**Quines instruccions reals s’utilitzen per a les pseudoinstruccions nop i bgez? (mireu el codi màquina que es genera i consulteu la taula d’instruccions, utilitzant els camps opcode i func3/7 si cal)**

La instrucció real per bgez és: si n >= 0 el flux del programa saltarà a la etiqueta que hi hagi associada.

La instrucció real per nop és: No operació, es a dir, que el programa no realitzarà cap instrucció.

**En quina posició de memòria es troben els valors de Resultat? En quina posició de memòria es guarda el contingut del registre A6?**

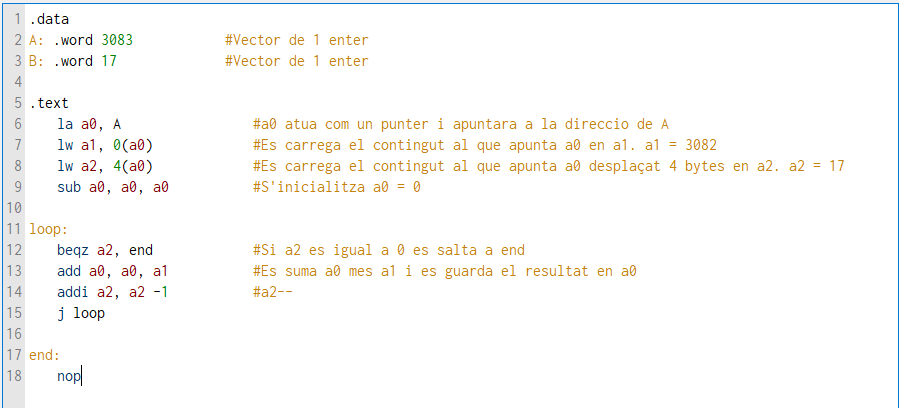
Els valors de Resultat es troben en les posicions de memòria 0x10000010 i 0x10000014.

El contingut del registre A6 es guarda en la posició de memòria 0x10000014.

**Quan executem bgez, quin registre es veu afectat?**

El registre que es veu afectat quan executem bgez és el PC ja que aquest ha de apuntar a la següent instrucció, per tant depenent de si es compleix o no la condició del bgez el PC haurà d’apuntar a la següent línia on s’executarà sw a5, 16(a0), o tres línies cap endarrere per executar add a5,a1,a2.

Exercici 2:



**Quina operació matemàtica està realitzant aquest codi?**

La operació matemàtica que s’està realitzant es una multiplicació, en el loop es fa 3083 x 17.

**Quant val el contingut d’A0, A1 i A2 quan acaba el programa?**

A0 = 52411

A1 = 3083

A2 = 0

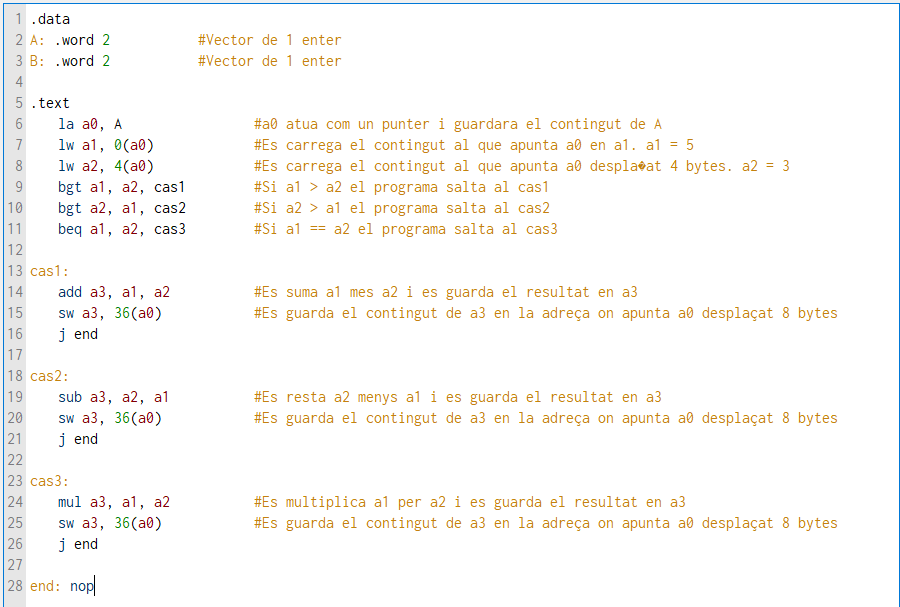
**Quantes iteracions del bucle executa el codi?**

Com que a2 = 17 i cada vegada que es realitza el bucle decrementa una unitat, s’executaran 17 iteracions.

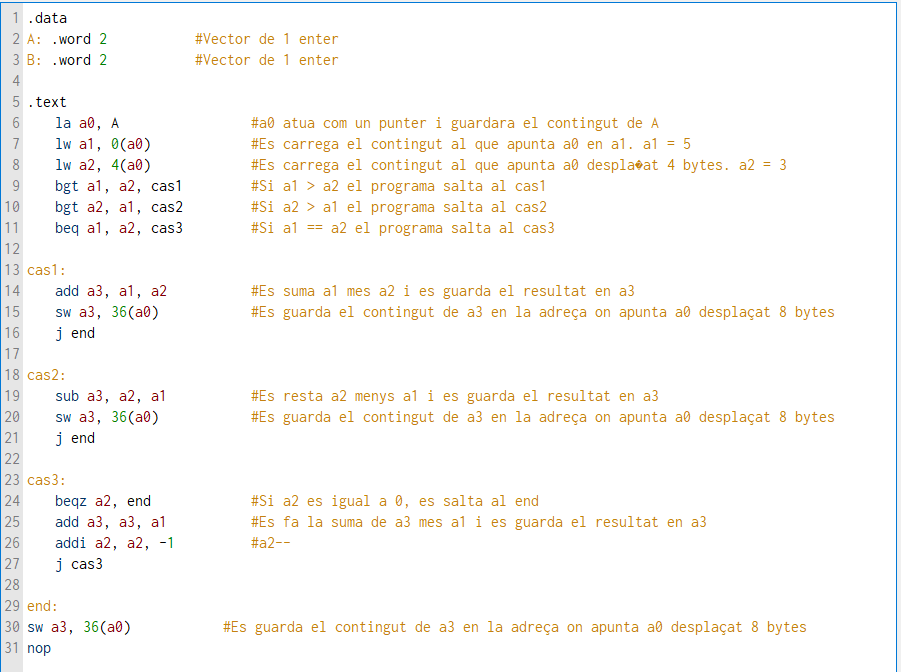
Exercici 3:

**Ens demanen calcular un algoritme que ens faci el següent: Donades dues entrades emmagatzemades en les posicions de memòria A i B fer la comparació. Si A>B calculem la suma. Si B>A fem la diferència (B-A) i si són iguals que multipliqui el seu valor. Carregueu diferents valors en memòria per tal de comprovar el funcionament del programa. Quina instrucció feu servir per fer la multiplicació? Què és més eficient, una instrucció directament que faci aquesta operació o el càlcul iteratiu? Raoneu la resposta.**

Per la multiplicació és més eficient fer la instrucció mul ja que només fa un cicle per executar-la. En canvi fer el càlcul iteratiu trigarà molt més ja que ha de realitzar les instruccions que hi han dins del bucle tantes vegades com aquest es repeteixi.

Codi amb la instrucció mul per fer la multiplicació 

Codi amb el bucle per fer la multiplicació



**Expliqueu el funcionament del programa que heu implementat.**

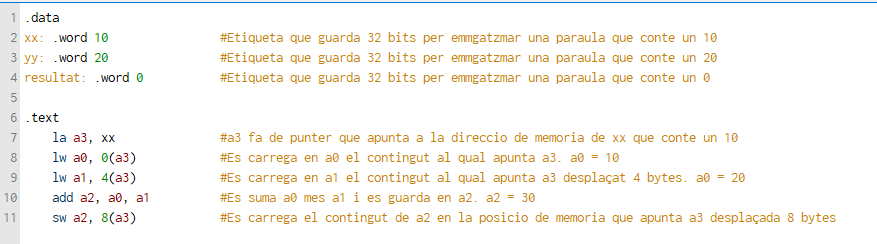
El programa comença amb 2 valors a memòria que es carregaran en a1 i a2, després es farà una estructura condicional per comparar els 2 valors. Si a1 > a2, es farà la seva suma, si a2 > a1, es farà la seva resta, i si a2 = a1, es farà la seva multiplicació. El resultat de l’operació realitzada es guardarà en a3 i també en memòria.

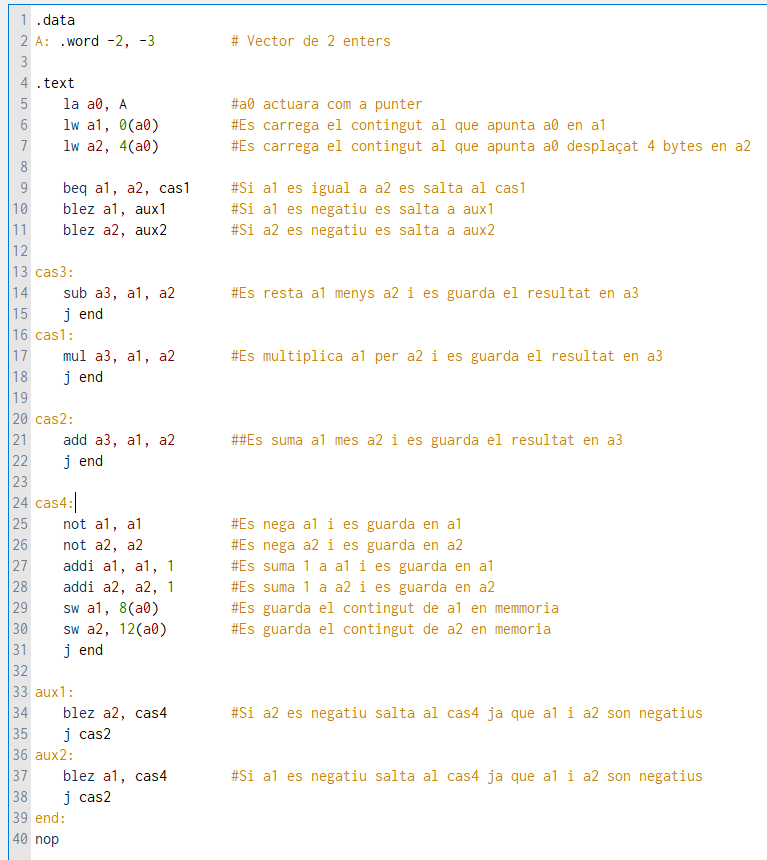
**Feu que el resultat es guardi a la posició de memòria 24h bytes després de l’inici de la secció .data.**

Veure en les captures anteriors.

**Exercicis addicionals**

Teòrico-Pràctica 2:

**1. Feu un programa en assemblador que carregui dos valors de una determinada posició de memòria en dos registres: a1 i a0, faci la suma i ho guardi en a2. Finalment guarda el resultat en memòria.**

**2. Feu un programa en assemblador que carregui dos valors de la memòria en dos registres. Si els valors són iguals, que faci el producte, si un és negatiu que faci la suma, si els dos són positius que faci la resta i si tots dos són negatius, que els passi a positiu i els guardi en memòria.**

**Conclusions**

Amb aquesta practica m’ha quedat més clar com funciona el Ripes ja que havíem de realitzar diversos programes.

Encara que m’ha costat una mica, he acabat entenent les estructures condicionals amb els salts a diferents etiquetes per fer les operacions corresponents en funció de la condició imposada.

Nacho Rivera

Grup B