Completa l'exercici 4.1 abans de continuar:

Exercici 4.1: Tradueix a assemblador MIPS la subrutina `descompon`.

```
descompon:
    slt  $t0, $a0 , $zero
    sw   $t0 , 0($a2)
    sll  $a0 , $a0 , 1

    bne  $a0, $zero, else
    li   $t0 , 0
    b    endif
else: li  $t0 , 18
while: blt  $a0, $zero, bi_w
    sll  $a0 , $a0 , 1
    addiu $t0 , $t0 , -1
    b while
bi_w: sra  $a0 , $a0 , 8
    li   $t1, 0x7FFFFF
    and  $a0 , $a0 , $t1
    addiu $t0 , $t0 , 127
endif:
    sw   $t0 , 0($a2)
    sw   $a0, 0($a3)

    jr $ra
```

Al fitxer **s4a.s** estan ja programats el programa principal (`main`) i les declaracions de variables globals en assemblador MIPS. Afegeix-hi el codi de l'exercici 4.1. A continuació, carrega'l al simulador, assembla'l i executa'l.

Comprova que al final de l'execució del programa `signe = 1`, que `exponent= 0x0000008D` (`141` en decimal), i que `matissa = 0x007A3140`.

Activitat 4.B: Funció `compon`

```
float compon(int signe, int exponent, int mantissa)
{
     return (signe << 31) | (exponent << 23) | mantissa;
}
```

Figura 4.3: Codi en alt nivell de la subrutina `compon`

La subrutina `compon` (figura 4.3) rep com a paràmetres els enters `signe`, `exponent` i `mantissa` passats per valor i retorna el número en coma flotant de simple precisió equivalent. El signe s'ha de desplaçar a la posició 31, i l'exponent a la posició 23. Els tres camps de bits resultants es poden combinar amb operacions `or` lògiques, ja que sabem que cada camp ocupa sols els bits que li són propis, essent la resta zeros.

Completa l'exercici 4.2 abans de continuar:

Exercici 4.2: Programa en assemblador la subrutina `compon`.

```
compon:
    sll  $a0 , $a0, 23
    sll  $a1, $a1, 23
    or   $v0, $a0, $a1
    or   $v0, $v0, $a2
    mtc1 $v0, $f0
    jr   $ra
```

Afegeix el codi de l'exercici 4.2 al fitxer **s4a.s**. Veuràs que el fitxer conté una subrutina `compon` provisional amb 1 sola instrucció, escrita per permetre fer proves a l'activitat 4.A: esborra-la. A continuació, assembla'l i executa'l.

Comprova que el número resultant, en coma flotant, val `cflotant=0xC6FA3140`. També pots comprovar el seu valor en decimal en el simulador. De quina manera ho consultaràs?

Al MARS es pot veure el contingut dels registres del CP1.
Aquí hem de consultar el registre $f1 que es troba a "Coproc1"

Quin és el valor final de `cflotant`, en decimal, que dóna Mars?

Valor decimal de `cflotant` = -32024'625

Completa el següent exercici abans de continuar:

Exercici 4.3: Codifica en coma fixa i en coma flotant (en hexadecimal) els següents números:

Decimal	cfixa (valor inicial)	cflotant (valor final)
0.0	0x00000000	0x00000000
- 0.0	0x80000000	0x80000000
12.75	0x0000CC00	0x414C0000

Verifica que el teu programa també converteix correctament els números de l'exercici 4.3, inicialitzant `cfixa` amb els valors de la segona columna i observant al simulador si el resultat en `cflotant` és el de la tercera columna.

Activitat 4.C: Errors de precisió en la conversió

Un cop llegida la pràctica i resolts els anteriors exercicis hauries de ser capaç de reflexionar sobre els errors de precisió que es poden cometre fent la conversió proposada, i respondre les preguntes del següent exercici:

Exercici 4.4: Contesta les següents preguntes

1) Quina condició ha de complir el valor inicial de `cfixa` perquè es produeixi pèrdua de precisió en la conversió que proposa aquesta pràctica?

Perquè es produeixi pèrdua de precisió en la conversió la mantissa tauria de tenir més de 23 xifres significatives.

2) Indica un valor de `cfixa` per al qual es produiria pèrdua de precisió al convertir-lo, i el corresponent valor en coma flotant:

cfixa	cflotant
0x00800001	0x45000001

3) En quina sentència concreta del programa en alt nivell es pot produir la pèrdua de precisió?

En la sentencia: cf = (cf >> 8) & 0x7FFFFF

4) Quin dels 4 modes d'arrodoniment que coneixes està portant a la pràctica aquest programa de conversió?

Aquest programa utilitza arrodoniment per truncament

5) El format de coma fixa explicat en aquesta pràctica permet codificar un rang de valors bastant limitat. Indica un número positiu que estigui DINS el rang del format de coma flotant de simple precisió (en decimal) però que estigui FORA del rang del format de coma fixa. Indica també quin és el MENOR número potència de 2 que compleixi aquesta condició.

El numero $1 \cdot 2^{20}$ es troba dins del rang del format de coma flotant però queda fora de rang del format de coma fixa. El numero més petit de potencia de 2 que queda dins del rang és $1 \cdot 2^{19} = 524288$.