

### Ejercicio 1

Si inicialmente los registros se encuentran en este estado:

```
a      = $fffffffffffffff
b      = $fffffffffffffff
x      = $fffffffffffff
```

y se ejecuta el siguiente código:

```
move $3d,x1
move $3d,a1
move $3d,b
```

Indicar el estado final de los registros

```
a      =
b      =
x      =
```

### Ejercicio 2

Si inicialmente los registros se encuentran en este estado:

```
a      = $000000000000000
b      = $000000000000000
x      = $000000000000000
```

y se ejecuta el siguiente código:

```
move $caba00,x1
move x1,a
move x1,b1
```

Indicar el estado final de los registros

```
a      =
b      =
x      =
```

### Ejercicio 3

Si inicialmente los registros se encuentran en este estado:

```
a      = $00a000000000000
y      = $000000000000000
ccr    = $00
```

y se ejecuta el siguiente código:

```
move a1,x1
move a,y1
move a,r7
move a1,x0
```

Indicar el estado final de los registros

```
x      =
y      =
r7     =
ccr    =
```

### Ejercicio 4

Si inicialmente los registros se encuentran en este estado:

```
a    = $00000123800000
b    = $ff000000ffffff
x    = $400000400000
```

y se ejecuta el siguiente código:

```
macr x0,x1,a
rnd  b
mpyr x0,x1,b
```

Indicar el estado final de los registros

```
a    =
b    =
```

### Ejercicio 5

Si inicialmente los registros se encuentran en este estado:

```
a    = $0000000000000000
sr    = $0300
```

y se ejecuta el siguiente código:

```
move $400000,x0
add  x0,a
add  x0,a
```


- 1) indicar el estado final de los registros y justificar este resultado  
sr =
- 2) repetir considerando que inicialmente sr = \$0700  
sr =

### Ejercicio 6

Si inicialmente los registros se encuentran en este estado:

```
a    = $0000000000000000
x    = $0c0000600000
r0   = $0000
```

y se ejecuta el siguiente código:

```
add  x1,a
rep  #$a 
norm r0,a
add  x0,a
```

Indicar el estado final de los registros

```
a    =
r0   =
```

Además indicar los cambios que se producen en el CCR a lo largo de la ejecución

### Ejercicio 7

Si inicialmente los registros se encuentran en este estado:

```
r0    = $0000    m0 = $ffff
r4    = $0000    m4 = $ffff
sr    = $0800
```

Se tiene el siguiente mapa de memoria:

```
X:$0000    $10fedc
X:$0001    $210fed
X:$0002    $4210fe
X:$0003    $84210f
X:$0004    $d84210
X:$0005    $fb8421
```

y se ejecuta el siguiente código:

```
    move x:(r0)+,a
    rep  #6
    move a,y:(r4)+  x:(r0)+,a
    jlc OK
    bset #0,y:$100
OK   bclr #6,sr
```

Indicar el estado final de la memoria Y.  
¿Qué significado tiene la memoria Y:\$100?

### Ejercicio 8

Escribir la subrutina vect\_max.

Compara elemento a elemento los vectores A y B, guarda el valor con modulo mayor en B. Recibe la dirección de inicio de los vectores en r0 y r4, y la cantidad de elementos en n0.

Utilizar la instrucción LOOP, y optimizar la cantidad de instrucciones dentro del bucle a la minima posible. Hint: considerar las instrucciones Tcc (transfer condicional).

Escribir un main de prueba y simular el resultado.