

# Relacion-1.pdf



ferluque



**Fundamentos del Software** 



1º Doble Grado en Ingeniería Informática y Matemáticas



Facultad de Ciencias
Universidad de Granada



- Todos los apuntes que necesitas están aquí 0
- Al mejor precio del mercado, desde 2 cent. 0
- Recoge los apuntes en tu copistería más cercana o recíbelos en tu casa
- Todas las anteriores son correctas

# Relación 1

### Ejercicio 1

a) E/S Programada

#### **Ejercicio 2**

b) E/S Dirigida por Interrupciones

## **Ejercicio 3**

b) Un programa, para que se ejecute, debe estar cargado en la memoria principal

### Ejercicio 4

c) La dirección de memoria donde se ha producido el último salto

b) La dirección de memoria donde se encuentra la dirección donde debe saltar el programa después de ejecutarse la instrucción de retorno correspondiente

#### Ejercicio 5

PC	IR	PE	PD	R0	R1	R2	23	24
12	MOVER 22, R1	0	12	16	14			
13	SUMAR R0, R1, R2	0	12	16	14	30		
14	MOVER R2, 23	0	12	16	14	30	30	
15	IN PD, 24	0	12	16	14	30	30	12
16	MOVER 23, R0	0	12	30	14	30	30	12
17	MOVER 24, R1	0	12	30	12	30	30	12
18	SUMAR R0, R1, R2	0	12	30	12	42	30	12
19	OUT PD, R2	0	42	30	12	42	30	12
20	HALT	-	-	-	-	-	-	-

# Ejercicio 6

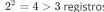
Tenemos que codificar 14 instrucciones en binario, para lo cual necesitamos 4 bits:

 $2^4 = 16 > 14$  instrucciones

Estos 4 bits son comunes a todas las instrucciones.

## Sumar R0, R1, R2

Para esta instrucción, además de los 4 bits de la instrucción necesitamos codificar 3 registros, para cada registro necesitaríamos 2 bits:







Luego necesitaríamos:

 $4\ bits + 3\ registros * 2\ bits = 10\ bits$ 

## MOVER 20, R0

Para esta instrucción, además de los 4 bits de la instrucción necesitamos codificar **1** registro (que ya hemos visto que necesita 2 bits), y **1** dirección de memoria que necesitará 8 bits:

 $2^8=256$  direcciones

Luego necesitaríamos:

 $4\ bits + 1\ registro*2\ bits + 1\ direcci\'on*8\ bits = 14\ bits$ 

## **Ejercicio 7**

- 1. El procesador detecta la interrupción
- 2. Guarda los contenidos del PC, PE, PD e IR en la pila
- 3. Se ejecuta el código de la interrupción (56-70)
- 4. Se finaliza la interrupción
- 5. Se restauran los valores de PC, PE, PD e IR que se habían almacenado en la pila y se continúa la ejecución del programa

## **Ejercicio 8**

Sí que hay diferencias pues la interrupción se produce con algún objetivo en concreto, por ejemplo el de atender a un dispositivo de entrada y salida, es decir, es un proceso deseado. Sin embargo, la excepción se produce cuando durante la ejecución del programa se produce un hecho inesperado tal como un desbordamiento aritmético o el acceso a una dirección inválida.

## **Ejercicio 9**

PC 12: IR: IN PD,M[26]

• M[26] <-- 15 = PD

PC 13: IR MOV M[26], R1

• R1 <-- 15 = M[26]

PC 14: IR COMP RO, R1

• No ocurre nada porque R0!=R1

PC 15/18: IR JNE 18

PC 19/20: IR CALL 20

• SP <-- 19

PC 21: IR ADD R0, R1, R2

• R2 <-- 20 = 5+15

PC 22: IR OUT R2,PD

• PD <-- 20

PC 23/19: IR RET

• SP <-- 30

PC 20: IR HALT



