

Todas las preguntas tienen la misma puntuación. Cada respuesta correcta suma un punto. Cada respuesta incorrecta, ilegible o vacía no suma ni resta. La nota del examen se obtiene multiplicando el número de preguntas correctas por 10 y dividiendo por el número total de preguntas del examen

2P2006-07-prog-ensa A ☐ A continuación se muestra el código de un programa que realiza la división entera entre los números naturales de dos listas dadas. El programa va dividiendo los números de la lista de dividendos entre los correspondientes de la lista de divisores. El cociente obtenido lo va almacenando en la lista de cocientes y el resto de la división lo guarda en la lista de restos.

En la posición  $n$  de la lista de cocientes se encuentra el cociente entero de dividir la posición  $n$  de la lista de dividendos entre la posición  $n$  de la lista de divisores. En la posición  $n$  de la lista de restos se encontraría el resto de esa misma división.

Para realizar su trabajo, el programa utiliza el procedimiento `divide`. Este procedimiento recibe 4 parámetros:

- El valor del dividendo.
- El valor del divisor.
- La dirección en la que hay que dejar el cociente.
- La dirección en la que se guardará el resto.

Para implementar el algoritmo, al dividendo se le va restando el divisor mientras sea más pequeño. Cada vez que se puede restar el divisor se incrementa en uno el cociente. Cuando ya no se le pueda restar al dividendo el divisor, lo que nos quede será el resto de la división entera.

Sabemos que, al empezar la ejecución del programa, los registros de la CPU tenían los siguientes valores:

R0=0000	R3=0000	R6=0000
R1=0000	R4=0000	R7=036E
R2=0000	R5=0000	PC=0339

Nota: El programa presenta algunos huecos sobre los que no se hacen preguntas.

```

1      ORIGIN 300h
2      INICIO empieza
3      .PILA 20
4      .DATOS
5      Nelem VALOR 5
6      Dividen VALOR 10, 28, 36, 43, 55
7      Divisor VALOR 10, 11, 12, 13, 14
8      Cocient VALOR 5 VECES 0
9      Resto VALOR 5 VECES 0

```

```

10      .CODIGO
11      PROCEDIMIENTO Divide
12
13
14
15
16
17
18
19
20      ;Se desapila la dirección del resto
21      INC R6
22      INC R6
23      MOV R5, [R6]
24      ;Se desapila la dirección del cociente
25      INC R6
26      MOV R4, [R6]
27      ;Se desapila el divisor
28      INC R6
29      MOV R1, [R6]
30      ;Se desapila el dividendo
31      INC R6
32      MOV R2, [R6]
33
34      XOR R3, R3, R3
35      XOR R0, R0, R0
36      ;Si el divisor es cero salimos
37      COMP R0, R1
38      BRZ Salir
39
40      repite:
41      ;Se compara dividendo con divisor
42      COMP R2, R1
43      ;Si es menor, salimos
44
45      ;Si es mayor, restamos
46      SUB R2, R2, R1
47      ;Incrementamos el cociente
48      INC R3
49      JMP repite
50      Salir:
51      ;Guardamos los valores del cociente y del resto
52
53
54      POP R5
55      POP R4
56      POP R3
57      POP R2
58      POP R1
59      POP R0
60      POP R6
61      RET
62      FINP
63
64      empieza:
65      ;Se accede a las variables del programa
66      MOVL R0, [ ] Nelem
67      MOVH R0, [ ] Nelem
68      MOV R1, [R0]
69      MOVL R2, [ ] Dividen
70      MOVH R2, [ ] Dividen
71      MOVL R3, [ ] Divisor
72      MOVH R3, [ ] Divisor

```

```

73      MOVL R4, [ ] Cocient
74      MOVH R4, [ ] Cocient
75      MOVL R5, [ ] Resto
76      MOVH R5, [ ] Resto
77      ;Si no hay elementos en las listas, acabamos
78      XOR R6, R6, R6
79      COMP R6, R1
80      BRZ Sacabo
81      bucle:
82      MOV R0, [R2] ;Se apila el dividendo
83      PUSH R0
84      MOV R0, [R3] ;Se apila el divisor
85      PUSH R0
86      ;Se apila la dirección del cociente
87      PUSH R4
88      ;Se apila la dirección del resto
89      PUSH R5
90      CALL Divide
91      ;Se limpia la pila
92      [ ]
93
94
95
96      ;Se pasa al siguiente elemento de cada lista
97      INC R2
98      INC R3
99      INC R4
100     INC R5
101     ;Se decreuenta el número de elementos
102     DEC R1
103     ;Si nos quedan elementos, seguimos
104     COMP R1, R6
105     BRNZ bucle
106
107     Sacabo:
108     JMP -1
109     FIN

```

— ¿Qué instrucción/es falta/n en el hueco de la línea 92 a la 95?

```

INC R7
INC R7
INC R7
INC R7

```

— Codifica la instrucción que se encuentra en la línea 49 del programa y expresa el resultado en **hexadecimal**

C0FB

— ¿Qué instrucción falta en la línea 44?

BRC Salir

— Reescribe la línea 3 usando el valor más pequeño que se pueda utilizar para este programa.

.Pila 12

— ¿Qué falta en los huecos de las líneas 66 y 67?

BYTEALTO DIRECCION  
YTEBAJO DIRECCION

— Si sabemos que la instrucción CALL se encuentra en la posición de memoria 034D ¿Cuál será su codificación? Responder en **hexadecimal**.

D0C7

— A partir de la línea 12 se han ocultado ocho instrucciones del programa. ¿Cuáles han sido las **cuatro** primeras?

PUSH R6  
MOV R6, R7  
PUSH R0  
PUSH R1

— ¿Cuál es la primera posición de la memoria que se modifica y con qué valor? Constar en **hexadecimal**.

Dirección: 036D      Contenido: 000A

— ¿Qué instrucciones faltan en el hueco de la línea 52?

MOV [R4], R3  
MOV [R5], R2

```
1      ADD R0, R0, R0
2      COMP R2, R0
3      BRZ final
4 bucle: MOV R4, [R3]
5      ADD R0, R0, R4
6      INC R3
7      DEC R2
8      BRNZ bucle
9 final: MOV [R1], R0
```

4, 6 microsegundos

❑ En una unidad de control cableada para la CPU teórica ¿Cuántas puertas AND se activan en el primer paso de la ejecución de las instrucciones aritméticas?

0

❑ Si se ponen todas las entradas de la ALU de la CPU teórica a cero y después se activa la señal ALU-SR, ¿Cuál será el valor de los bits del registro de estado ZCOS? Responder con cuatro dígitos binarios.

Nota: Op0=0 y Op1=0 *seleccionan la operación lógica AND*.

Z= 1    C= 0    O= 0    S= 0

❑ A continuación se muestra el estado de algunos registros de la CPU teórica justo en el instante en que se acaba de ejecutar el paso 3 de cierta instrucción. También se muestra el contenido de ciertas posiciones del mapa de memoria:

Registros		Memoria	
Registro	Contenido	Dirección	Contenido
R2	227B	227A	47CF
R3	227A	227B	78AC
R4	227D	227C	223D
R5	227C	227D	36AD
PC	336B		
IR	C004		

Además se sabe que después de la instrucción en curso se ejecuta la secuencia de instrucciones:

```
1      MOV R2, [R3]
2      MOV R5, [R4]
3      SUB R2, R2, R5
```

Contestar a las siguientes preguntas:

— ¿Qué instrucción se está ejecutando? (Contestar con el mnemónico)

JUMP +4

— ¿Qué valor tendrá R2 al finalizar la ejecución de dicha secuencia de instrucciones? Contestar en **hexadecimal**.

1122h

❑ Se ha programado una nueva instrucción en la CPU teórica, cuyas palabras de control se muestran a continuación (solo se consideran las señales de control que intervienen en la instrucción y las palabras de control no comunes).

Señal	Paso		
	4	5	6
ADD	1	0	0
ALU-TMPS	1	0	0
CARRYIN	1	0	0
FIN	0	0	1
IB-MAR	1	0	0
IB-R7	0	1	0
IB-SR	0	0	1
LEER	1	0	0
MDR-IB	0	0	1
R7-IB	1	0	0
TMPE-CLR	1	0	0
TMPS-IB	0	1	0
...			

¿Cuál sería el mnemónico de esta instrucción?

POP SR

❑ Se tiene un ordenador cuya CPU es igual a la CPU teórica vista en clase. Esta CPU tiene un reloj de frecuencia 5 MHz. Justo antes de ejecutarse la primera instrucción del fragmento de código mostrado los registros contienen: R2=0002h, R0=0001h, R3=1000h, R4=0000h, ¿cuánto tiempo tardará en ejecutarse el fragmento de código? Indicar las unidades de la respuesta.

2P2006-07-respuesta-multiple

- ☐ Indica cuál o cuáles de las siguientes afirmaciones son ciertas. Puedes contestar *todas* o *ninguna* en el caso de que todas o ninguna de ellas sean ciertas.

1. El registro R7 de la CPU teórica apunta a la dirección de memoria en la que se introducirá el siguiente dato a apilar.
2. En el Bus de Datos la memoria escribe el dato solicitado en una petición de lectura y en el Bus de Direcciones escribe la dirección de dicho dato.
3. Para no mezclar los datos con el código, la Arquitectura Von Neumann utiliza dos tipos de memoria.
4. Todas las instrucciones modifican el Registro de Estado (SR).

ninguna

2P2006-07-CPU-seniales

- ☐ Se quiere definir una nueva instrucción en la CPU teórica que introduzca un dato inmediato de 8 bits a la pila:

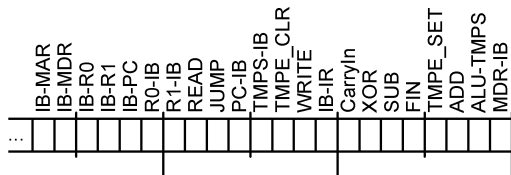
PUSH Im\_8

El dato inmediato irá codificado en el byte de menor peso, aunque el valor que se apila se obtiene extendiendo el signo a 16 bits. Indicar la secuencia de señales de control necesarias para su ejecución.

Paso	Señales
4	R7-IB, TMPE-SET, ADD, ALU-TMPS
5	JUMP, IB-MDR
6	TMPS-IB, IB-R7, IB-MAR, ESCRIBIR
7	FIN

2P2006-07-UC-microprograma

- ☐ Una unidad de control microprogramada genera señales de control que se interpretan como se muestra en la figura adjunta (sólo se muestran los 22 bits inferiores de la palabra de control).



En un instante dado la palabra de control que se genera es:

...000100 00001000 00010000

¿Cuál será la próxima palabra de control que se generará? (Responder en **hexadecimal**)

205486h (100000 01010100 10000110)