

SISTEMAS BASADOS EN MICROPROCESADORES

Grado de Ingeniería Informática

Parcial 2 – Grupo 226 – Curso 2010/11

NOMBRE : _____ DNI : _____
 APELLIDOS : _____

P1. Si **SP=000Ah** y **FLAGS=1234h** al inicio de la ejecución del código que se adjunta, indicar los valores contenidos en las **primeras dieciséis posiciones de la pila** en el momento de ejecutar la primera instrucción de la rutina de servicio de la interrupción 1Ch. Los valores desconocidos deben dejarse en blanco. (2 puntos)

```
549A:025E CD1C      int 1Ch
549A:0260 89161000  mov Datos[0], dx
```

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
				60h	02h	9Ah	54h	34h	12h						

P2. Al inicio de la ejecución de una función invocada desde lenguaje C, se tiene que **SP=4** y que las 16 primeras posiciones de la pila contienen los siguientes valores:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
11h	A0h	25h	00h	32h	00h	A2h	E9h	00h	C1h	24h	F1h	00h	63h	41h	12h

La signatura de dicha función es: `int fun (int *p, char c, int n);`

Indicar el valor de los tres parámetros con que esa función fue invocada desde C, tanto cuando todas las direcciones son cercanas (**NEAR**), como cuando son lejanas (**FAR**). (2,5 puntos)

Caso NEAR: p = E9A2h c = 00h n = F124h

Caso FAR: p = F124h:C100h c = 00h n = 1241h

P3. Escribir en ensamblador de 8086 un procedimiento cercano denominado `_Instalar_61h` que modifique el **vector de la interrupción 61h** con la dirección de otro procedimiento denominado `_RSI_61h`. El valor anterior de ese vector de interrupción debe almacenarse previamente en una única variable que deberá declararse dentro del propio procedimiento `_Instalar_61h`. Se valorará la eficiencia del código. (2,5 puntos)

```

_Instalar_61h PROC NEAR
    jmp inicio

    rsi61 dw ?, ?                ; Dirección larga anterior

inicio: push ax es
       mov ax, 0
       mov es, ax
       mov ax, es:[61h*4]        ; Guarda Offset de rsi anterior
       mov cs:rsi61, ax
       mov ax, es:[61h*4 + 2]    ; Guarda Segmento de rsi anterior
       mov cs:rsi61[2], ax

       ; Cambia rsi de 61h
       cli
       mov word ptr es:[61h*4], offset _RSI_61h
       mov word ptr es:[61h*4 + 2], seg _RSI_61h
       sti

       pop es ax
       ret

_Instalar_61h ENDP

```

P4. Escribir en ensamblador de 8086 utilizando instrucciones básicas (sin instrucciones de manipulación de cadenas ni de bucles) la función `strcmp` de C, cuya signatura se reproduce a continuación. Esta función retorna un entero que indica la relación entre las dos cadenas que recibe como argumentos: Un valor de cero indica que ambas cadenas son iguales. Un valor mayor que cero indica que el primer carácter que no coincide tiene un valor mayor en `str1` que en `str2`. Un valor menor que cero indica lo contrario. Se considera que el programa en C está compilado en **modelo pequeño** (*small*). Las cadenas de caracteres en C acaban con un cero. Se valorará la eficiencia del código. (3 puntos)

```
int strcmp (const char *str1, const char* str2);
```

```

_strcmp PROC NEAR
    push bp
    mov bp, sp
    push bx si di

    mov si, bp[4]    ; si <= str1
    mov di, bp[6]    ; di <= str2

    mov ax, 0        ; Por defecto son iguales

continuar: mov bl, [si]    ; bl <= str1[i]
           cmp bl, [di]    ; str1[i] - str2[i]
           je iguales
           ja str1_mayor
           mov ax, -1      ; str1 es menor que str2
           jmp final
str1_mayor: mov ax, 1      ; str1 es mayor que str2
           jmp final
iguales:   cmp bl, 0      ; str1[i] = 0?

```

```
        je final          ; Acaban ambas cadenas

        ; Continúa con siguiente carácter
        inc si
        inc di
        jmp continuar

final:   pop di si bx bp
        ret
_strcmp ENDP
```