Apellidos _____ **Nombre** _____ **DNI** _____ **A**

Examen de Fundamentos de los computadores. *Área de Arquitectura y Tecnología de Computadores* **Junio: 29-05-2004**

A

Instrucciones generales para la realización de este examen

La respuesta debe escribirse en el hueco existente a continuación de cada pregunta **con letra clara**.

Cada respuesta correcta suma un punto Cada respuesta incorrecta, ilegible o vacía no suma ni resta. El total de puntos se dividirá entre el total de preguntas y se multiplicará por 10 para obtener la nota del examen.

El siguiente programa para la CPU teórica dibuja un asterisco en pantalla y lo mueve hacia a la izquierda cuando se pulsa la tecla 'o' y hacia la derecha cuando se pulsa la 'p'. Para ello utiliza tres elementos:

- La variable **pos**: Indica la posición actual donde está el asterisco.
- El procedimiento **mueve_asterisco**: Recibe por la pila un valor que indica el incremento que hay sumarle a **pos** para volver a pintar el asterisco. Antes de pintarlo, lo borra de la posición anterior.
- La rutina de servicio **rut_teclado**: Comprueba qué teclas se han pulsado. Si se pulsa una 'o', llama a **mueve_asterisco** con parámetro –1; si se pulsa una 'p', lo llama con parámetro 1; si es otra tecla no hace nada.

```
ORIGEN 400h
INICIO main
.PILA XXXXh ; Eliminado intencionadamente
.DATOS
  pos VALOR 0F000h
.CODIGO

PROCEDIMIENTO mueve_asterisco
  PUSH R6
  MOV R6, R7
  PUSH R0
  PUSH R1
  PUSH R2

  ; Recuperar en R0 el parámetro
 ; -- 1 --

  MOVL R6, BYTEBAJO DIRECCION pos
  MOVH R6, BYTEALTO DIRECCION pos
  MOV R1, [R6] ; R1 = pos actual
  ; Escribir en blanco en la pos actual
  MOVL R2, ' '
  MOVH R2, 7
  MOV [R1], R2

  ; Modificar pos y escribir el asterisco
  ADD R1, R1, R0
  MOVL R2, '*'
  MOVH R2, 7
  MOV [R1], R2
  MOV [R6], R1 ; Actualizar pos

  POP R2
  POP R1
  POP R0
  POP R6
  RET
FINP

PROCEDIMIENTO rut_teclado
  PUSH R0
  PUSH R1
  PUSH R2
  PUSH R3
  PUSH R6

  MOVL R1, 1h    ; R1 = reg. control tecl.
  MOVH R1, 0FEh
  MOVL R2, 0     ; Hacer R2 = máscara
  MOVH R2, 1

bucle:  ; Hacerlo mientras queden teclas
  MOV R3, [R1]
  AND R3, R3, R2
  BRZ no_hay_tecla

  DEC R1       ; R1 = reg. de datos
  MOV R3, [R1] ; R3 = tecla pulsada
  INC R1       ; R1 = reg. de control

  ; Comprobar la tecla pulsada
  MOVL R6, 'o'
  MOVH R6, 12h ; scan de la 'o'
  COMP R6, R3
 ; --2-- Saltar si es necesario
  XOR R6, R6, R6
  DEC R6 ; Poner un -1 en R6
  PUSH R6
  CALL mueve_asterisco
  INC R7
  JMP bucle

no_es_o:
  MOVL R6, 'p'
  MOVH R6, 13h ; scan de la 'p'
  COMP R6, R3
  BRNZ bucle
  MOVL R6, 1
  MOVH R6, 0
  PUSH R6
  CALL mueve_asterisco
  INC R7
  JMP bucle

no_hay_tecla:
  POP R6
  POP R3
  POP R2
  POP R1
  POP R0
 ; --3--
FINP

main:
  XOR R0, R0, R0
  MOVL R1, BYTEBAJO DIRECCION rut_teclado
  MOVH R1, BYTEALTO DIRECCION rut_teclado
  MOV [R0], R1
  STI

  JMP -1
FIN
```

La figura de la página siguiente muestra un momento de la ejecución del programa.

– Después de haber generado una interrupción y de recibir de la CPU la señal INTA, ¿cuál es el primer número que la interfaz del teclado envía por el bus de datos?

0000h

– ¿Cuál es el código ASCII del asterisco? **Contestar en hexadecimal.**

2Ah

A

– ¿Qué instrucción o instrucciones falta/n en **-- 1 --**?

INC R6

INC R6

MOV R0, [R6]

– Sabiendo que no se ha pulsado ninguna vez la tecla ‘o’, ¿cuántas veces como mínimo se habrá pulsado la tecla ‘p’ antes de llegar al instante mostrado en la figura?

3

– ¿Qué instrucción o instrucciones falta/n en **-- 2 --**?

BRNZ no_es_o

– ¿Qué instrucción o instrucciones falta/n en **-- 3 --**?

IRET

– ¿Qué tamaño mínimo debe tener la pila? **Contestar en decimal.**

13

– Si en el paso 1 de la ejecución de la primera instrucción del programa se pulsase una tecla, ¿qué habría en IR en el primer paso de la primera instrucción que se ejecutase de **rut_teclado**? **Contestar en hexadecimal.**

A800h

Se pretende implementar un sumador elemental mediante un PLA.

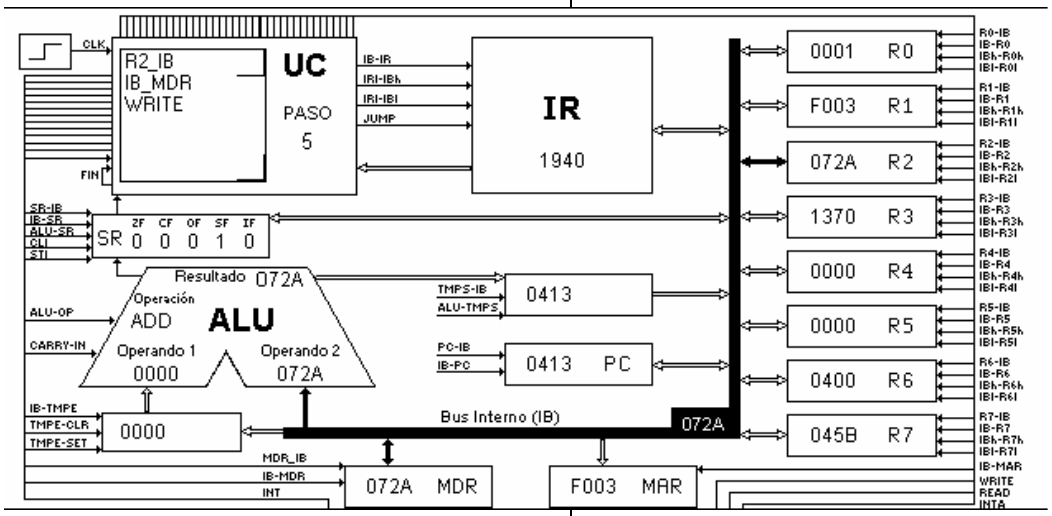
– ¿Cuál es el número **mínimo** de puertas AND y OR necesario para confeccionar el PLA? Nota: Las puertas pueden tener cuantas entradas necesiten.

AND: 7 OR: 2

Se desea diseñar un dispositivo de memoria que cubra la primera cuarta parte del espacio de direcciones de la CPU teórica. El dispositivo tiene un decodificador 2:4.

– ¿De cuántos bancos se compone el dispositivo?

4



– Sabiendo que el ancho de palabra de los chips es la mitad del ancho de palabra de la CPU, ¿cuál es la organización de los chips? (Ejemplo de formato de respuesta: 6Kx9).

4Kx8

Se dispone de una unidad de control microprogramada para la CPU teórica que se encarga de generar las palabras de control. Las palabras de control se interpretan como se indica en la figura (se muestran sólo los 14 bits menos significativos).

	LEER	JUMP	IB-MAR	IB-PC	IB-TMPE	IB-IR	MDR-IB	PC-IB	CarryIn	TMPS-IB	ADD	TMPE-CLR	ALU_TMPS	FIN
...														

IR = 1100 0000 1111 1001

– Para el valor del Registro de Instrucción que se muestra, ¿cuáles serian la palabras de control que generaría la unidad de control? Codificar las palabras de control en **HEXADECIMAL**.

4	0240
5	100A
6	0411
7	

En la CPU teórica se sabe que el siguiente fragmento de código ha tardado 3.2 microsegundos en ejecutarse y que la instrucción `INC R6` sólo ha producido acarreo la segunda vez que se ejecutó:

```
atrás: INC R6
       BRC alli
       JMP atras
alli:  XOR R6, R6, R6
```

— ¿Cuál es la frecuencia, **en MHz**, del reloj de la CPU?

10 MHz

Apellidos _____ **Nombre** _____ **DNI** _____ **A**

Examen de Fundamentos de los computadores. *Área de Arquitectura y Tecnología de Computadores* **Junio: 29-05-2004**

Se ha creado un programa en lenguaje ensamblador para los procesadores compatibles con el Intel 80386 que encripta dos cadenas de texto terminadas por 0. La encriptación se lleva a cabo efectuando una operación lógica XOR del código ASCII de cada uno de los caracteres con un valor binario de 8 bits (que recibe el nombre genérico de máscara). El programa utiliza para encriptar las cadenas un procedimiento llamado **encripta_cadena** que recibe a través de la pila tres parámetros que son apilados antes de llamarlo en el siguiente orden:

- Un valor de 32 bits que es la dirección de inicio de la cadena a encriptar.
- Un valor de 32 bits que es la dirección de inicio de la cadena resultado.
- Un valor de 32 bits que es la dirección en la que se encuentra la máscara de 8 bits a aplicar (mediante la operación XOR) a los caracteres de la cadena a encriptar.

A continuación se muestra el código del programa, en el cual faltan una serie de instrucciones:

```
.386
.MODEL flat
EXTERN ExitProcess:PROC

.DATA
cad_entrada1 DB "12 jun 2004", 0
cad_salida1  DB 11 DUP (20h), 0
cad_entrada2 DB "patria querida", 0
cad_salida2  DB 14 DUP (20h), 0
mascara      DB 243

.CODE
inicio:
  ; Encriptar cad_entrada1 con la mascara
  ; y dejar el resultado en cad_salida1
  mov esi, OFFSET cad_entrada1
  push esi
  mov esi, OFFSET cad_salida1
  push esi
  mov esi, OFFSET mascara
  push esi
  call encripta_cadena

  ; Encriptar cad_entrada2 con la mascara
  ; y dejar el resultado en cad_salida2
  mov esi, OFFSET cad_entrada2
  push esi
  mov esi, OFFSET cad_salida2
  push esi
  call encripta_cadena

  ; Salir del programa
  push 0
  call ExitProcess

encripta_cadena PROC
  push ebp
  mov ebp, esp
  push edi
  push esi
  push eax
  push ebx
  push ecx

  ; Mover a edi el 1er. parámetro apilado
  ; (dir. ini. de la cadena a encriptar)

     --- HUECO 2 ---

  ; Mover a esi el 2do. parámetro apilado
  ; (dir. ini. de la cadena resultado)
  mov esi, [ebp+12]

  ; Mover a eax el tercer parámetro
apilado
  ; (dir. donde esta la mascara a aplicar)
  mov eax, [ebp+8]

  ; Traer a cl el valor de la mascara
  mov cl, [eax]

bucle:
  ; Traer a bl un carácter de la cadena
  ; a encriptar
  mov bl, [edi]

  ; Si es el valor cero, salir del bucle
     --- HUECO 3 ---
  ; Encriptar el carácter (XOR con
máscara)
  xor bl, cl

  ; Llevar el carácter encriptado
  ; a la cadena resultado
  mov [esi], bl

  ; Apuntar al sgte. carácter en las 2
cads
  inc esi
  inc edi
  jmp SHORT bucle

fin_cadena:
  pop ecx
  pop ebx
  pop eax
  pop esi
  pop edi
  pop ebp
     --- HUECO 1 ----
encripta_cadena ENDP
END inicio
```

– ¿Qué instrucción falta en el --- **HUECO 1** ---?

ret 12

– ¿Qué instrucción falta en el --- **HUECO 2** ---?

mov edi, [ebp+16]

– ¿Qué dos instrucciones faltan en el --- **HUECO 3** ---?

cmp bl, 0

jz/je SHORT fin_cadena

– ¿Qué valor toma el registro **esi** justo después de la primera ejecución de la instrucción mov esi, [ebp+12] (la que hay justo después del --- **HUECO 2** ---) a sabiendas de que la sección de datos del programa comienza en la dirección de memoria 00402000h? **Responder en hexadecimal**.

0040200Ch

Se sabe que las entradas de la ALU de 16 bits vista en clase tienen los siguientes valores:

- En el **operando A** se encuentra la combinación de bits que representa el uno en formato exceso a Z con 16 bits y exceso central.
- En el **operando B** se encuentran los 16 bits más altos de la codificación del número $2^{-127}(1+2^{-2})$ en formato IEEE-754.
- **CarryIn** = 0, **Resta** = 1, **Op1** = 1 y **Op0** = 1.

Se recuerda que las ALUs elementales que componen la ALU de 16 bits tienen conectadas las salidas de una puerta AND, de una puerta OR, de una puerta XOR y el resultado de un sumador elemental, a las entradas e_0, e_1, e_2 y e_3 del multiplexador, respectivamente.

– ¿Cuál es el resultado obtenido en la salida? Expresar el resultado con **cuatro dígitos hexadecimales**.

7F B1h

En la misma ALU del ejercicio anterior se introducen ahora los siguientes valores:

- En el **operando A** el número positivo más grande que se puede codificar en complemento a 2.
- En el **operando B** el número negativo más grande que se puede codificar en exceso a Z central.
- **CarryIn** = 0, **Resta** = 0, **Op1** = 1 y **Op0** = 1.

– ¿Cuál es el resultado obtenido en la salida y cuál será el valor de los bits de estado? Expresar el resultado en **DECIMAL, interpretado en complemento a 2**.

Resultado (decimal)= -2 ZF= 0 OF= 1 SF= 1 CF= 0

Se ha creado una nueva instrucción para la CPU teórica que permite realizar un salto si el valor del registro R0 es cero. El mnemónico de la nueva instrucción es:

BR_R0Z Inm_8

Durante las señales de control de la instrucción, para saber si el registro R0 contiene el valor cero, se realiza una operación de suma en la ALU y se comprueba si el bit de cero está activo o no.

Sabiendo que la CPU tiene un reloj de 5 MHz y que la ejecución de esta instrucción dentro de un programa duró 1,2 microsegundos, completar la siguiente tabla con las señales de control que se generaron durante la ejecución de la instrucción (el paso 5 ya se suministra completo y se supone que la CPU ha sido modificada para realizarlo):

4	R0-IB; TMPE_CLR; ADD; ALU-SR
5	(comprobar si ZF=0)
6	FIN
7	
8	

— ¿Cuáles de las siguientes afirmaciones son **CIERTAS**? (Puedes responder "ninguna" si así lo consideras.)

a) El sistema UNICODE de representación de caracteres, al utilizar conjuntos distintos para distintas lenguas, permite la asignación del mismo código a distintos caracteres, siempre y cuando pertenezcan a conjuntos diferentes.

b) Las CPUs con unidades de control microprogramadas reservan una parte de su espacio de direccionamiento en la memoria del sistema para almacenar la memoria de microprograma.

c) Una CPU con unidad de control cableada ejecuta un programa a la misma velocidad que otra CPU con unidad de control microprogramada siempre y cuando ambas tengan la misma frecuencia de reloj y ejecuten los pasos de control en el mismo tiempo.

d) La entrada/salida basada en muestreo periódico obliga a la CPU a derrochar tiempo en la consulta del estado de los periféricos.

C, D

— ¿Cuáles de las siguientes afirmaciones son **CIERTAS**? (Puedes responder "ninguna" si así lo consideras.)

a) El registro PC de la CPU de ejemplo y el registro EIP de Intel tienen funciones similares, excepto que EIP se puede usar en las rutinas de interrupción para acceder al registro de estado almacenado en la pila.

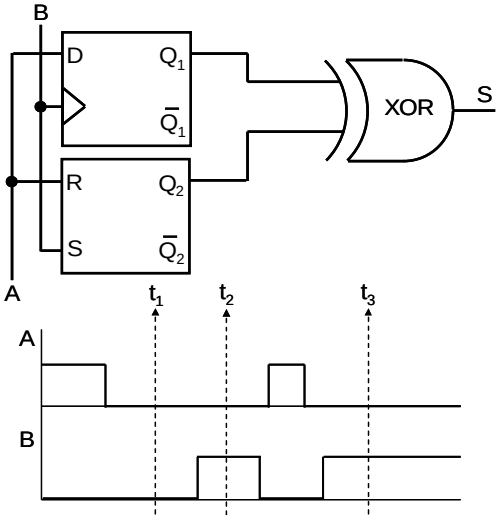
b) En la CPU de ejemplo, el usuario, a través de un programa, NO puede cargar en el registro PC el valor que desee.

c) En Intel, se puede usar cualquier registro de propósito general (EAX, EBX, ECX, EDX, EDI, ESI) para acceder, dentro de la pila, a los parámetros de un procedimiento.

d) La relación capacidad de almacenamiento/velocidad de acceso es mayor en los chips de RAM estática que en los de RAM dinámica.

C

Dado el circuito y el cronograma mostrados a continuación:



– ¿Cuál es el valor de la salida S en los instantes t_1, t_2 y t_3 teniendo en cuenta de que el valor inicial de Q_1 es 1?

t_1: 1 t_2: 1 t_3: 1

– Se ha diseñado un formato de representación de números reales en coma flotante con las siguientes características: 3 bits para el exponente que se representa en exceso a Z central, y 5 bits para la mantisa que se codifica en signo-magnitud y se normaliza a todo fracción con bit implícito a la derecha de la coma decimal. ¿Cuál sería en la representación en IEEE-754 del número más grande que se puede codificar en este formato diseñado? **Contestar en hexadecimal.**

40 F8 00 00