

# Resultado del test



Universidad de Granada - Grado en Ingeniería Informática  
Estructura de Computadores (B,C)



Test nº 6 que realiza usted en esta asignatura

1

Elección  
única

[T1.1]

El conjunto de todos los atributos de un sistema que son visibles para el programador y son necesarios para programar en lenguaje máquina se denomina:

Usuario/a Correcta



- a) repertorio de instrucciones máquina
- b) arquitectura del computador
- c) organización del computador
- d) conjunto de componentes físicos del computador

Puntuación: **1,00**

[T1.1UniFun]

[E16SepTeo01]

2

Elección  
única

[T2.4.3]

Considerar las siguientes declaraciones de estructuras en una máquina Linux de 64-bit.

```
struct RECORD {  
    int value2;  
    short ref_count;  
    char tag[10];  
};
```

```
struct NODE {  
    long value;  
    struct RECORD record;  
    char string[8];  
};
```

También se declara una variable global "my\_node" como sigue:  
struct NODE my\_node;

¿Cuál es el tamaño de my\_node en bytes?

Usuario/a Correcta



- a) 28
- b) 32
- c) 40
- d) Ninguno de los anteriores

Puntuación: **-0,33**

[T2.4.3Struct]

[E12SepTeo05]

3

Elección  
única

[T6.2]

¿Cuántas patillas de dirección tiene una memoria DRAM de 1G palabra, siendo la longitud de palabra de 16 bits?

Usuario/a Correcta



- a) 15
- b) 20
- c) 30
- d) 16

Puntuación: **1,00**

[T6.2RAMROM]

[E13FebTeo23]

4

Elección  
única

[T6.2]

¿Cuántas líneas de dirección (patillas) son necesarias para direccionar un chip de memoria DRAM de 4096 x 4?

Usuario/a Correcta



- a) 12
- b) 6
- c) 11
- d) 10

Puntuación: **-0,33**

[T6.2RAMROM]

[E14SepTeo28]

5

Elección  
única

[T3.1]

Para realizar la microoperación  $MAR \leftarrow PC$ , habrá que activar:

Usuario/a Correcta



- a) EnPC y EnMAR
- b) LdPC y EnMAR
- c) LdPC y LdMAR
- d) EnPC y LdMAR

Puntuación: **1,00**

[T3.1CamDat]

[E14FebTeo19]

6

Elección  
única

[T3.1]

Alguna de las siguientes no es una operación básica de la unidad de control

Usuario/a Correcta



- a) realizar operación ALU y guardar resultado en registro
- b) transferir un registro a otro
- c) (guardar o recuperar) un registro (en / de) la pila
- d) (leer o escribir) un registro (de / a) memoria

Puntuación: **1,00**

[T3.1CamDat]

[E15SepTeo15]

7

Elección  
única

[T6.2]

¿En qué tipo de memorias coincide el tiempo de acceso y el tiempo de ciclo?

Usuario/a Correcta



- a) SRAM
- b) DRAM
- c) Tanto en a) como en b)
- d) Ninguna de las anteriores

Puntuación: **1,00**

8

Elección  
única

[T1.1]

En un procesador de la familia 80x86 una variable de 32 bits, entera con signo, almacenada a partir de la dirección  $n$  contiene: 0xFF en la dirección  $n$ , 0xFF en la dirección  $n+1$ , 0xFF en la dirección  $n+2$  y 0xF0 en la dirección  $n+3$ . ¿Cuánto vale dicha variable?

Usuario/a Correcta



- a) 4294967280
- b) -16
- c) 16
- d) -251658241

Como es little-endian, se trata de un número negativo de gran magnitud, y éste es el único con ese aspecto

Puntuación: **0,00**

[T1.1UniFun]  
[E13SepPra20]

9

Elección  
única

[T4.4]

Respecto a la segmentación, ¿cuál de las siguientes afirmaciones es falsa?

Usuario/a Correcta



- a) Cuantas más etapas tenga un cauce, más instrucciones se estarán ejecutando en distintas fases y más posibilidades se presentan de que existan riesgos entre ellas
- b) La técnica de register forwarding habilita una serie de caminos (buses) que se añaden al cauce para permitir que los resultados de una etapa pasen como entradas a la etapa donde son necesarias
- c) Retrasar la fase de decisión saltar/no saltar de las instrucciones de salto condicional contribuye a mejorar el rendimiento del procesador
- d) La reorganización del código y la introducción de instrucciones nop permite evitar dependencias de datos

Puntuación: **-0,33**

[T4.4Riesgs]  
[E15FebTeo11]

10

Elección  
única

[T2.1.4]

En un sistema Linux x86-64, ¿cuál de las siguientes variables ocupa más bytes en memoria?

Usuario/a Correcta



- a) float d
- b) char a[7]
- c) int \*c
- d) short b[3]

Puntuación: **1,00**

[T2.1.4x86-64]  
[E15FebPra04]  
[E16FebPra05]

**11**Elección  
única

[T2.2.1]

Respecto a direccionamiento a memoria en ensamblador IA32 (sintaxis AT&T), de la forma D(Rb, Ri, S), sólo una de las siguientes afirmaciones es FALSA.

¿Cuál?

Usuario/a Correcta



- a) El factor de escala S puede ser 1, 2, 4, 8
- b) El desplazamiento D puede ser una constante literal (1, 2 ó 4 bytes)
- c) ESP no se puede usar como registro índice
- d) EBP no se puede usar como registro base

Puntuación: **-0,33**

[T2.2.1ModDir]

[E12SepTeo07]

**12**Elección  
única

[T1.1]

Una memoria que está estructurada en palabras de 8 bits tiene una capacidad de 64 Kbits. ¿Cuántas líneas de dirección tiene dicha memoria?

Usuario/a Correcta

- a) 24
- b) 8
- c) 13
- d) 12

Puntuación: **0,00**

[T1.1UniFun]

[T1.3EstBus]

[E15FebTeo22]

**13**Elección  
única

[T2.4.3]

Considerar las siguientes declaraciones de estructuras en una máquina Linux de 64-bit.

```
struct RECORD {  
    long value2;  
    int ref_count;  
    char tag[4];  
};  
  
struct NODE {  
    double value;  
    struct RECORD record;  
    char string[8];  
};
```

También se declara una variable global "my\_node" como sigue:  
struct NODE my\_node;

Si la dirección de my\_node es 0x601040, ¿cuál es el valor de &my\_node.record.tag[1]?

Usuario/a Correcta

- a) 0x601050
- b) 0x601054
- c) 0x601055
- d) Ninguna de las anteriores

Puntuación: **0,00**

14

Elección  
única

[T1.1]

Según la clasificación m/n, las máquinas de acumulador son de tipo

Usuario/a Correcta



- a) 2/2 ó 2/3
- b) 0/0
- c) 1/2
- d) 1/1

Puntuación: -0,33

[T1.1UniFun]

[E15SepTeo02]

15

Elección  
única

[T2.4.3]

Al traducir la sentencia C `r->i = val;` gcc genera el código ASM `movl %edx, 12(%eax)`. Se deduce que

Usuario/a Correcta



- a) `val` es un entero que vale 12  
`val` está en `%edx`
- b) `r` es un puntero que apunta a la posición de memoria 12  
`r` es puntero a struct y seguramente valdrá `0x0804XXXX` en ejecutables x86
- c) `i` es un entero que vale 12  
después de la asignación, `i` vale lo que había en `%edx` (que es `val`)
- d) el desplazamiento de `i` en `*r` es 12  
y en concreto `r` está en `%eax`

Puntuación: 1,00

[T2.4.3Struct]

[E15SepPra13]

16

Elección  
única

[T2.4.2]

En el siguiente código, ¿qué reordenamiento de los bucles muestra mejor localidad?

// X, Y, Z ctes #define previo

int a[X][Y][Z]

int i, j, k, sum = 0;

for (i = 0; i < Y; i++)

for (j = 0; j < Z; j++)

for (k = 0; k < X; k++)

sum += a[k][i][j];

Usuario/a Correcta



- a) El orden de los bucles no afecta a la localidad
- b) `i` externo, `j` central, `k` interno (el orden en que están ahora)
- c) `k` externo, `i` central, `j` interno
- d) `j` externo, `k` central, `i` interno

Puntuación: 1,00

[T2.4.2Arrays]

[T6.1ConLoc]

[E14FebTeo04]

17

Elección  
única

[T1.1]

En una CPU de 32 bits con memoria de bytes, el problema es que...

Usuario/a Correcta



- a) Hay que respetar el ordenamiento de bytes y reglas de alineamiento con que se diseñó la CPU
- b) No tiene sentido, un registro no cabría en memoria
- c) Hay que usar 4 instrucciones de lectura (o escritura) para leer (o escribir) un registro completo
- d) No hay problema, cuando se salva un registro a memoria se escribe en la posición deseada

Puntuación: 0,00

[T1.1UniFun]

[E13FebTeo01]

18

Elección  
única

[T2.2.5]

De entre las siguientes construcciones de flujo de control en lenguaje C, la que se traduce más directamente a lenguaje ensamblador es...

Usuario/a Correcta



- a) La selección switch-case
- b) El bucle for
- c) El bucle do-while
- d) El bucle while

Puntuación: 1,00

[T2.2.5BWhile]

[E13FebTeo05]

19

Elección  
única

[P3.2]

La práctica "parity" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. La siguiente función contiene un único error (en realidad se han editado 2 líneas de código sobre la versión correcta) pero produce resultado correcto cuando se usa como test el array...

```
int parity4(unsigned* array, int len){
    int i;
    unsigned x;
    int val=0, result=0;
```

```
    for (i=0; i<len; i++){
        x = array[i];
        asm("\n"
"ini:                \n\t"
        "xor %[x], %[v] \n\t"
        "shr %[x]      \n\t"
        "jnz ini       \n\t"
        : [v]"r" (val)
        : [x] "r" (x)
        );
        result += val & 0x1;
    }
    return result;
}
```

Usuario/a Correcta



- a) array={5, 4, 3, 2}

debería salir 2 (impares el 4 y el 2)  
sale 2 (0,1,1,0, el 3 cuenta como impar, el 2 deshace el impar)

- b) array={0, 1, 2, 3}  
debería salir 2 (0,1,1,0, impares el 1 y el 2)  
sale 1 (0,1,0,0, el 2 deshace el impar del 1)
- c) array={1, 16, 256, 1024}  
debería salir 4 (todos impares)  
sale 2 (1,0,1,0, cada otro impar deshace el anterior)
- d) array={1, 2, 4, 8}  
debería salir 4 (1,1,1,1, todos impares)  
sale 2 (1,0,1,0, cada otro impar deshace el anterior)

Puntuación: **0,00**

[P3.2Parity]

[E15SepPra17]

El error consiste en que val=0 se inicializa al principio, en lugar de tras cada x=array[i], de manera que si el LSB de val se queda activado la siguiente paridad se calcula mal. En concreto, tras calcular un impar, todos los pares que sigan cuentan como impar hasta que llegue un impar que deshaga el impar (y entonces el nuevo impar cuenta como par).

**20**

Elección  
única

[P3.2]

La práctica "parity" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity4:

```
int parity4(unsigned* array, int len){
    int val,i,res=0;
    unsigned x;
    for (i=0; i<len; i++){
        x=array[i];
        val=0;
        asm("\n"
"ini3:      \n\t"
"xor %[x],[v] \n\t"
"shr %[x]   \n\t"
"test %[x], %[x]\n\t"
"jne ini3   \n\t"
":[v]"+r" (val)
:[x]"r" (x)
);
        val = val & 0x1;
        res+=val;
    }
    return res;
}
```

La sentencia asm() del listado anterior tiene las siguientes restricciones

Usuario/a Correcta



- a) un registro y dos sobrescritos (clobber)  
b) ninguna  
c) dos entradas y una salida  
[v] cuenta como salida y entrada, [x] como entrada  
d) arquitectura de 32 bits

Puntuación: **0,00**

21

Elección  
única

[P5.1]

Suponer una memoria cache con las siguientes propiedades: Tamaño: 512 bytes. Política de reemplazo: LRU. Estado inicial: vacía (todas las líneas inválidas). Suponer que para la siguiente secuencia de direcciones enviadas a la cache: 0, 2, 4, 8, 16, 32, la tasa de acierto es 0.33. ¿Cuál es el tamaño de bloque de la cache?

Usuario/a Correcta



- a) 4 bytes
- b) 8 bytes
- c) 16 bytes
- d) Ninguno de los anteriores

Puntuación: -0,33

[P5.1Line]

[E14SepPra08]

[E15FebTeo28]

22

Elección  
única

[P2.2]

En la práctica "media" se programa la suma de una lista de 32 enteros de 4 B para producir un resultado de 8 B, primero sin signo y luego con signo. Si la lista se rellena con el valor que se indica a continuación, ¿en qué caso ambos programas producen el mismo resultado?

Usuario/a Correcta



- a) 0xFFFF FFFF  
0x0000 001f ffff ffe0 != 0xffff ffff ffff ffe0
- b) 0x1111 1111  
resultado 0x0000 0002 2222 2220  
porque es positivo incluso en complemento a 2  
todos los demás valores se interpretan como negativos,  
lo primero que hace la suma con signo es extenderlos a  
64bit de manera que se activan los 32 bits superiores...  
resultado radicalmente distinto
- c) 0xAAAA AAAA  
0x0000 0015 5555 5540 != 0xffff fff5 5555 5540
- d) 0x9999 9999  
0x0000 0013 3333 3320 != 0xffff fff3 3333 3320

Puntuación: 1,00

[P2.2SumSgn]

[E16SepPra06]

Recordar que multiplicar por 32 es desplazar 5 posiciones a la izquierda

23

Elección  
única

[T6.2]

Cada celda de un chip de memoria DRAM de 1M x 1, organizada en una matriz de 512 filas x 2048 columnas, necesita ser refrescada cada 16 ms. ¿Cada cuánto tiempo ha de realizarse una operación de refresco en el chip?

Usuario/a Correcta



- a) 7,8125 microsegundos
- b) 31,25 microsegundos
- c) 8192 milisegundos
- d) 61 nanosegundos

Puntuación: 0,00



24

Elección  
única

[T2.2.2]

¿Cuál de las siguientes secuencias de instrucciones multiplica el (contenido del) registro EAX por 18?

Usuario/a Correcta



- a) sarl \$1, %eax  
imul \$9, %eax
- b) imull \$0x18, %eax
- c) shll \$18, %eax
- d) leal (%eax,%eax,8), %eax  
leal (%eax,%eax), %eax

Puntuación: -0,33

[T2.2.2OpArit]  
[E13SepTeo03]

25

Elección  
única

[P2A2]

En x86\_64 se pueden referenciar los registros

Usuario/a Correcta



- a) %r12q, %r12d, %r12w, %r12l  
sería %r12 y %r12b, no %r12q ni %r12l
- b) %rax, %eax, %ax, %ah, %al  
Ver libro Hallaron Figura 3.35.  
%ah no sería compatible con los nuevos nombres x86-64 como %sib, %r8b.
- c) %r8, %r8d, %r8w, %r8l  
sería %r8b, no %r8l
- d) %rsi, %esi, %si, %sih, %sil  
no existe %sih

Puntuación: -0,33

[P2Apendice2]  
[T2.4.1x86-64]  
[E15SepPra10]

26

Elección  
única

[T2.2.1]

¿Cuál de las siguientes instrucciones convierte  $\%eax = 5 * \%eax$ ?

- 1) mov (%eax, %eax, 4), %eax
- 2) lea (%eax, %eax, 4), %eax

Usuario/a Correcta



- a) Ambas 1 y 2
- b) Ninguna
- c) Sólo 2
- d) Sólo 1

Puntuación: 1,00

[T2.1.3ConASM]  
[T2.2.1ModDir]  
[E14FebTeo02]

27

[P5.1]

En la práctica de la cache, el código de line.cc incluye la sentencia

Elección  
única

```
for (unsigned long long line=1;  
     line<=LINE; line<=1) { ... }
```

¿Qué objetivo tiene la expresión `line<=1`?

Usuario/a Correcta



- a) sacar un uno (1) por el stream line
- b) salir del bucle si el tamaño de línea se volviera menor o igual que 1 para algún elemento del vector
- c) volver al principio del vector cuando el índice exceda la longitud del vector
- d) duplicar el tamaño del salto en los accesos al vector respecto a la iteración anterior

Puntuación: **1,00**

[P5.1Line]

[E15FebPra18]

**28**

Elección  
única

[T2.1.4]

La arquitectura x86-64 tiene:

Usuario/a Correcta



- a) 64 registros RPG de 64 bits
- b) 8 registros de propósito general (RPG) de 64 bits (%rax, %rbx, ... %rsp, %rbp)
- c) 32 registros RPG de 64 bits
- d) 16 registros RPG de 64 bits

Puntuación: **-0,33**

[T2.1.4x86-64]

[E13SepTeo08]

**29**

Elección  
única

[T2.2.3]

La instrucción IA32 test sirve para...

Usuario/a Correcta



- a) Realizar la operación and lógico bit-a-bit (`a&b`) pero no guardar el resultado, sino simplemente ajustar los flags
- b) Testear el código de condición indicado, y poner un byte a 1 si se cumple
- c) Realizar la operación resta (`a-b`) pero no guardar el resultado, sino simplemente ajustar los flags
- d) Mover el operando fuente al destino, pero sólo si se cumple la condición indicada

Puntuación: **-0,33**

[T2.2.3CodCon]

[E13SepTeo04]

**30**

Elección  
única

[T1.2]

¿En qué pareja de registros están el dato/instrucción que se leerá o escribirá en memoria, y la dirección de memoria?

Usuario/a Correcta



- a) MBR y PC
- b) MAR y ACUMULADOR
- c) MBR y MAR

d) IR y ACUMULADOR

Puntuación: **1,00**

[T1.2ConBas]

[E12FebTeo18]

[E12SepTeo13]

Puntuación: **9,67 (3,22 sobre 10)**