COGNOMS:

NOM: DNI/NIE:

IMPORTANTE leer atentamente antes de empezar el examen: Escriba los apellidos, el nombre y el DNI/NIE antes de empezar el examen. Escriba un solo carácter por recuadro, en mayúsculas y lo más claramente posible. Es importante que no haya tachones ni borrones y que cada carácter quede enmarcado dentro de su recuadro sin llegar a tocar los bordes. Use un único cuadro en blanco para separar los apellidos y nombres compuestos si es el caso. No escriba fuera de los recuadros, todo lo que haya fuera de ellos es ignorado. La identificación del alumno se realiza de forma automática, no seguir correctamente estas instrucciones puede comportar no tener nota.

Problema 1. (5 puntos)

En un ordenador en el que tenemos instalado el entorno usado en el laboratorio de AC, hemos medido que un programa de 1000 instrucciones ensamblador se ha ejecutado en 3 segundos usando $9x10^9$ ciclos y ha ejecutado $4{,}8x10^9$ instrucciones enteras y además $2{,}4x10^8$ operaciones de coma flotante que, debido a la falta de hardware específico, ejecutan 10 instrucciones cada una.

a) **Calcula** el CPI del programa y la frecuencia de la CPU (usa el prefijo del sistema internacional más adecuado).

b) **Calcula** los MIPS y MFLOPS a los que se ejecuta el programa.

El tiempo de ejecución usado en el primer apartado se corresponde al tiempo de CPU (usuario + sistema). Usando el comando "time" de linux hemos obtenido que el tiempo de CPU representa solo el 20% del tiempo total del programa (wall time). El 80% restante es tiempo de entrada/salida (accesos al disco duro concretamente). Se lo comentamos a nuestro proveedor y nos ofrece cambiar el disco duro por un disco SSD que, asegura, nos ofrece una ganancia del 79900%. Hemos medido que cada acceso al disco duro actual del sistema tarda 8 milisegundos.

c) **Calcula** el tiempo medio de cada acceso a la SSD.

d) **Calcula** la ganancia total en el programa que se obtendría con el cambio de tipo de disco.

Hemos visto que el procesador tiene un tiempo de vida medio hasta fallos (MTTF) de 1000000 de horas, mientras que el disco duro tiene un tiempo de vida medio hasta fallos de 200000 horas y el de la SSD de 1000 horas debido a que es muy fácil que uno de sus sectores se estropee por las escrituras y suponemos que cualquier sector estropeado estropea toda la SSD.

e) **Calcula** el tiempo de vida medio hasta fallos (MTTF) del sistema con el disco duro y el tiempo de vida medio hasta fallos (MTTF) del sistema con la SSD en vez del disco duro.

A pleno rendimiento, la CPU funciona a una frecuencia de 3 GHz y está alimentada a 1,6 V. En modo bajo consumo la CPU funciona a una frecuencia de 0,8 GHz y está alimentada a 1 V. Hemos medido que el consumo de la CPU en alto rendimiento es de 120W y en modo bajo consumo es de 25 W. En estos datos solo se considera la potencia debida a conmutación y la debida a fugas. Tanto la corriente de fugas (I) como la carga capacitiva equivalente (C) son las mismas en ambos modos.

f) **Calcula** la corriente de fugas (I) y la carga capacitiva equivalente (C) de la CPU
(usar prefijo más adecuado del Sistema Internacional) .

g) **Calcula** la ganancia en energía que tendría el sistema si ejecutara el programa en el modo de bajo rendimiento en vez de en el modo de alto rendimiento suponiendo que el CPI medio no varía.

COGNOMS:

NOM: DNI/NIE:

Problema 2. (5 puntos)

Dado el siguiente código escrito en C, que compilamos para un sistema linux de 32 bits:

```
typedef struct {
  char a;
  short b;
  char c;
  int d;
  short e;
  char f;
  short g[3];
  } s1;
```

```
typedef struct {
  short u;
  s1 v[10];
  short w;
  } s2;
```

a) **Dibuja** cómo quedarían almacenadas en memoria las estructuras **s1** y **s2**, indicando claramente los desplazamientos respecto al inicio, el tamaño de todos los campos y el tamaño de los structs.

b) **Escribe** UNA ÚNICA INSTRUCCIÓN que permita mover **x.v[5].g[1]** al registro **%ax**, siendo **x** una variable de tipo **s2** cuya dirección está almacenada en el registro **%ecx**.
Indica claramente la expresión aritmética utilizada para el cálculo de la dirección.

Dado el siguiente código escrito en C, que compilamos para un sistema linux de 32 bits, y asumiendo que S1 es el struct declarado en el apartado a) de este ejercicio:

```
int examen(short a, short b, int c[2][2], S1 *d) {
  S1 *x;
  short y[3];
  char z, w;
  . . .
 return (x->d)
}
```

c) **Dibuja** el bloque de activación de la rutina examen, indicando claramente los desplazamientos respecto a **%ebp** y el tamaño de todos los campos.

d) **Traduce** a ensamblador x86 la instrucción `return(x->d)` hasta el final de la subrutina, **usando el mínimo número de instrucciones**, sabiendo que la rutina SOLO ha usado los registros **%eax**, **%ebx**, **%ecx** y **%edx**.