COGNOMS: . . . . . NOM: . . . . . .

1er Control Arquitectura de Computadors **Curs 2014-2015 Q1**

- **Temps: 13:30 a 14:00**
- **Poseu clarament amb LLETRES MAJÚSCULES a cada full els cognoms i el nom**

Problema 1. (4 puntos)

Ejecutamos un aplicación (P) en un computador (C). En dicha ejecución, la rutina (R) representa el 80% del tiempo de ejecución de (P). Esta rutina (R) realiza 300 millones de operaciones de punto flotante y ejecuta 1200 millones de instrucciones a un promedio de 800 MIPS. La siguiente tabla muestra la distribución de instruccines para la rutina (R) junto con el CPI medio de cada tipo de instrucción.

	punto flotante	**enteras**	**memoria**
Nunero de instrucciones	500 millones	300 millones	400 millones
CPI	4	2	7

a) **Calcula** el tiempo de ejecución de la rutina (R).

b) **Calcula** el rendimiento en MFLOPS de la rutina (R).

c) **Calcula** los ciclos que tarda la rutina (R).

d) **Calcula** la frecuencia del procesador (C).

e) **Calcula** el CPI del programa (P).

Un programador experimentado esta intentando optimizar la rutina (R) usando instrucciones SIMD.

f) **Calcula** la ganancia en tiempo de ejecución de la rutina (R) que deberia conseguir para que el programa (P) se ejecute 4 veces más rápido.

Esta CPU, tiene una capacidad efectiva equivalente de 8 nF (nanofaradios), y una corriente de fugas de 12 A y funciona a un voltaje de 1,25 V.

g) **Calcula** la potencia media debida a fugas, la debida a conmutación y la potencia total disipada por la CPU (C).

Este computador está formado por los componentes mostrados en la tabla siguiente. La tabla también muestra el numero de componentes de cada tipo y el tiempo medio hasta fallo (MTTF) de cada componente.

Componente	Fuente alimentación	CPU	Ventilador CPU	Placa base	DIMMs	Discos duros	Tarjetas graficas
Nº	1	1	1	1	4	2	2
MTTF (horas)	100.000	1.000.000	100.000	200.000	1.000.000	125.000	500.000

El tiempo medio para reemplazar un componente que ha fallado (*mean time to repair*) es de 5 horas y la probabilidad de fallo sigue una distribución exponencial.

h) **Calcula** el tiempo medio hasta fallos del hardware (MTTF), el tiempo medio entre fallos (MTBF) y la disponibilidad del sistema.

COGNOMS: . . . . . NOM: . . . . . .

1er Control Arquitectura de Computadors **Curs 2014-2015 Q1**

Problema 2. (3 puntos)

Dado el siguiente código escrito en C:

```
typedef struct {
  int c1, c2;
} Estru;
int subr2(int p1, int *p2, Estru p3)
{
  int a, b;
  …
}
int subr1()
{
  Estru e;
  int res;
…
  res = subr2(e.c1, &e.c2, e);
…
}
```

a) **Dibuja** los bloques de activación de las subrutinas **subr1** y **subr2** indicando claramente el tamaño y desplazamiento de cada campo respecto al registro ebp.

b) **Traduce** a ensamblador la sentencia "`res = subr2(e.c1, &e.c2, e);`".

c) Si ebx, ecx, edx y esi tienen valores utiles antes de llamar a **subr2** que hay que utilizar después del retorno de **subr2**, cuales de ellos debería salvar **subr1** y cuales debería salvar **subr2**.

COGNOMS: . . . . . NOM: . . . . . .

1er Control Arquitectura de Computadors **Curs 2014-2015 Q1**

Problema 3. (3 puntos)

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

```
typedef struct {
  char a;
  short b[2];
  char c;
  char d[4];
  short e[2];
  float *f;
  int g;
  short h;
 } s1;
```

```
typedef struct {
  s1 v[100];
  char a;
  } s2;
```

a) **Dibuja** como quedarían almacenadas en memoria las estructuras **s1** y **s2**, indicando claramente los deplazamientos 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[10].a al registro %al , siendo x una variable de tipo s2 cuya dirección está en el registro %ecx. **Indica** la expresión aritmética para calcular la dirección de x.v[10].a.

c) **Escribe** UN CONJUNTO DE 2 INSTRUCCIONES que permita mover x.v[y.g].d[3] al registro %al , siendo x una variable de tipo s2 cuya dirección está almacenada en el registro %ecx e y una variable de tipo s1 cuya dirección está almacenada en el registro %ebx. **Indica** la expresión aritmética para calcular la dirección de x.v[y.g].d[3]