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

3er Control Arquitectura de Computadors **Curs 2014-2015 Q1**

- **Temps: 8:00 a 11:00**
- **Poseu clarament amb LLETRES MAJÚSCULES a cada full els cognoms i el nom**

Problema 1. (3 puntos)

Dado el siguiente código escrito en C:

```
typedef struct {
  char a;
  short b;
  char c;
  short d[4]
  double *e;
  short f;
} s1;
```

```
typedef struct {
  double e;
  s1 v[100];
  int i;
  } s2;
```

```
int examen(s1 j, short n, s2 *k, char m[8]){
  short u;
  char v[4];
  double w;
    ...
}
```

a) **Dibuja** como quedarían almacenadas en memoria las estructuras s1 y s2, indicando claramente los desplazamientos de cada campo respecto al inicio de cada estructura, el tamaño de todos los campos y el tamaño de las dos estructuras.

b) **Dibuja** el bloque de activación de la función examen, indicando claramente los desplazamientos relativos al registro **%ebp** necesarios para acceder cada uno de los parámetros y de las variables locales.

c) Dado el número decimal -35,3, exprésalo EN BINARIO Y EN HEXADECIMAL en el formato IEEE 754 de simple precisión. Indica debajo los cálculos realizados.

Número en hexadecimal:

Número en binario:

Cálculos:

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

3er Control Arquitectura de Computadors **Curs 2014-2015 Q1**

Problema 2. (3 puntos)

Dada una memoria principal de 4 GB con 4 canales, 2 DIMMs SDRAM por canal, 8 chips (de un byte de ancho) por DIMM, 16 bancos por chip, 8K filas por banco, 512 columnas de un byte por fila y las siguientes funciones de mapeo de direcciones (bit de mas peso de la dirección a la izquierda):

A)	Canal	DIMM	Chip	Banco	Fila	Columna
B)	DIMM	Canal	Fila	Banco	Columna	Chip
C)	Canal	DIMM	Banco	Columna	Fila	Chip

a) ¿Cual de estas tres funciones de mapeo es más adecuada cuando las referencias a memoria tienen localidad espacial y por qué?.

El código hamming extendido es muy usado para detectar y corregir errores en memorias, donde se conoce bajo el nombre de SECDED (single error correction, double error detection).

b) **Cuantos** bits redundantes necesita el código SECDED para proteger un bloque de datos de 64 bits?

c) Y para un bloque de datos de 4 bits?

d) **Calcula** el código SECDED para el bloque de datos: 1011 (usando paridad par).

Dado el siguiente fragmento de código:

```
for (j=0; j<1024; j++)
   for (i=0; i<512; i++)
      A[i] = A[i]+ B[i] + C[i] - j;
```

donde A, B y C son vectores de 512 enteros (4 bytes) cuya dirección inicial se puede ser cualquiera, y las variables i y j están en registros.

e) **Calcula** las características de la memoria cache lo más pequeña posible que nos asegura que al ejecutar este código la cache de datos tiene sólo fallos de carga (compulsory).

Tamaño cache	
Tamaño de Linea (contesta todas las posibles)	
Asociatividad (contesta todas las posibles)	
Reemplazo (contesta todos los posibles entre LRU, FIFO y Random)	

f) **Transforma** el código de forma que en una cache de tamaño 12 bytes sólo hubiera fallos de carga.

g) **Especifica** las características que debería tener dicha cache.

Tamaño cache	**12 bytes**
Tamaño de Linea (contesta todas las posibles)	
Asociatividad (contesta todas las posibles)	
Reemplazo (contesta todos los posibles entre LRU, FIFO y Random)	

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

3er Control Arquitectura de Computadors **Curs 2014-2015 Q1**

Problema 3. (4 puntos)

Un servidor esta formado por los siguientes componentes principales:

Componente	Fuente alimentación	CPU	Placa base	DIMMs	Discos duros
Nº	1	1	1	4	1
MTTF (horas)	200.000	1.000.000	200.000	1.000.000	100.000

a) **Calcula** el MTTF del sistema suponiendo que este falla si falla alguno de los componentes.

En este servidor (esta configuración la llamaremos servidor-base) se ejecuta una aplicación formada por tres fases:

1) **Lectura**: se realizan lecturas secuenciales del disco, y la CPU prácticamente no se utiliza.
2) **Cálculo**: se usa exclusivamente la CPU.
3) **Escritura**: se realizan escrituras a posiciones aleatorias del disco (tampoco se usa la CPU).

Sabemos que con la configuración descrita, la fase de **Lectura** representa el 64% del tiempo, la de **Cálculo** el 20% y la de **Escritura** el 16%. Durante la fase de **Cálculo** el conjunto CPU-placa-dimms consume 150 W y el disco duro 1W, mientras que durante las fases de **Lectura** y **Escritura** la CPU-placa-dimms consume 40 W y el disco duro 10 W. Para simplificar el problema supondremos que la fuente de alimentación tiene un rendimiento del 100%.

b) **Calcula** la potencia media consumida por el sistema.

El disco duro tiene una capacidad de 1 Tbyte, ofrece un ancho de banda efectivo de 250 MB/ s y el tiempo de cambiar un disco (MTTR) es de 10 horas. Dado que la aplicación esta limitada por los accesos al disco duro se ha decidido ampliar el número de discos a 8 discos idénticos al anterior. En esta nueva configuración, que llamaremos servidor-raid, se duda entre varias organizaciones RAID para configurar los 8 discos.

c) **Rellena** la siguiente tabla en relación a la organización de los discos.

	Capacidad útil	**Ancho banda fase Lectura**	**Ancho banda fase Escritura**	**MTTF de la configuración**
1 solo disco	1 TB	250 MB/s	250 MB/s	100.000 horas
RAID 0				
RAID 10				
RAID 5				

Finalmente el servidor-raid usará una configuración de discos en RAID 5.

d) **Calcula** el speed-up (ganancia en tiempo) del servidor-raid respecto el servidor-base al ejecutar la aplicación.

e) **Calcula** la ganancia en energía del servidor-raid respecto el servidor-base al ejecutar la aplicación, suponiendo que el consumo del RAID 5 se debe exclusivamente a los discos.

La productividad (throughput) se define como el trabajo realizado por unidad de tiempo. Si definimos como unidad de tiempo T el tiempo total de ejecución de la aplicación en el servidor-base, la productividad al ejecutar una única instancia de la aplicación en el servidor-base seria de 1 (1 aplicación ejecutada por unidad de tiempo T).

f) **Calcula** la productividad del servidor-raid al ejecutar una única instancia de la aplicación.

Dado que la aplicación tiene una parte en que se usa (casi) exclusivamente el disco (fases de **Lectura** y **Escritura**) y otra en que se usa exclusivamente la CPU (fase de **Cálculo**), se podrían ejecutar de forma solapada múltiples (tantas como hagan falta) instancias de la misma aplicación (suponemos que todas las instancias tardan lo mismo) de forma que mientras una instancia usa la CPU, otra puede usar el disco. Como ya debes saber de sistemas operativos, esto permite maximizar la productividad (a costa en ocasiones de penalizar el tiempo de ejecución). Para simplificar el problema considera despreciable el tiempo usado en cambios de contexto por el SO.

g) **Calcula** la productividad máxima del servidor-raid al ejecutar múltiples instancias solapadas de la aplicación.

La CPU del sistema soporta memoria virtual basada en paginación y tiene una jerarquía de memoria que incluye una cache de mapeo directo que se accede con direcciones físicas y un TLB. Un acceso a memoria puede encontrar tres tipos de fallo: fallo de TLB, fallo de página y fallo de cache.

h) **Considera** las posibles combinaciones de estos tres eventos e **indica** si puede darse dicha combinación. En caso negativo **explica** porque no puede producirse y en caso afirmativo **explica** bajo que circunstancias puede darse.

TLB	Tabla de páginas	Cache	¿Posible? Si/No	Explicación
Hit	Hit	Miss		
Hit	Miss	Miss		
Miss	Miss	Hit		
Miss	Hit	Miss		