Cognoms: . . . . . Nom: . . . . . .

3er Control Arquitectura de Computadors **Curs 2011-2012 Q2**

Problema 1. (2 puntos)

Los dispositivos que no disponen de hardware específico de coma flotante deben realizar las operaciones de coma flotante por software. La función "sumamatrizescalar" escrita en C suma a su parámetro res todos los elementos del parámetro matfloats mediante llamadas a la rutina sumvecfloat que como su nombre indica realiza por software la suma en coma flotante de sus dos primeros parámetros (es decir, suma res y los N valores del vector vectorfloats), devolviendo un puntero al resultado. Recordad que los floats se representan mediante el formato IEEE 754 de simple precisión.

```
float *sumvecfloat(float *res, float vectorfloats[], int N)
void sumamatrizescalar(float *res, float matfloats[100][100])
{
    int i;
    for (i=0;i<100; i++)
       res=sumvecfloat(res, &matfloats[i][0], 100);
}
```

a) Dibujad el bloque de activación de la rutina sumamatrizescalar y traducidla a ensamblador x86.

```
sumamatrizescalar:
   pushl %ebp
   movl %esp, %ebp
   subl $4, %esp
   movl $0, -4(%ebp)
   for: cmpl $100, -4(%ebp)
        jge finfor
        pushl $100
        imull $400, -4(%ebp), %eax
        addl 12(%ebp), %eax
        pushl %eax
        pushl 8(%ebp)
        call sumvecfloat
        addl $8, %esp
        movl %eax, 8(%ebp)
        incl -4(%ebp)
        jmp for
   finfor:
   movl %ebp, %esp
   popl %ebp
   ret
```

El código anterior se ejecuta en un procesador que funciona a una frecuencia de 500 MHz y tarda 1 milisegundo. Sabemos que la rutina sumvecfloat contiene 400 instrucciones estáticas que en media generan 5500 instrucciones dinámicas en cada llamada a la misma.

b) Calculad los MIPS y MFLOPS del sistema.

Ins= (5500 + 12) * 100 = 551200 i

MIPS=551200 instr / 10^{-3} s / $1x10^{6}$=551 MIPS

MFLOPS=10000 ops / 10^{-3} s /$1x10^{6}$=10 MFLOPS

Si añadimos al procesador anterior una unidad aritmética de coma flotante podríamos substituir la llamada a la rutina sumvecfloat por las correspondientes operaciones en alto nivel. En este caso el bucle anterior tan solo tardaría 0,5 milisegundos en ejecutarse pero el procesador incrementaría su gasto en potencia media en un 10%.

c) Calculad cuál sería el porcentaje de ganancia en MFLOPS/W respecto al caso anterior.

MFLOPS/Wb=10 MFLOPS / X watios

MFLOPS/Wc=20 MFLOPS / (X watios * 1,1)

Ganancia = 20 / (10*1,1) = 1,82 -> 82%

Cognoms: . . . . . Nom: . . . . . .

3er Control Arquitectura de Computadors **Curs 2011-2012 Q2**

Problema 2. (4 puntos)

Tenemos una aplicación formada por dos fases. En una versión secuencial de la aplicación la Fase 1 representa el 20% del tiempo mientras que la Fase 2 representa el 80% del tiempo. Esta aplicación se ejecuta en un chip con un solo procesador (core). Cada core tiene una intensidad de fugas de 2,5 A, una carga capacitiva equivalente de 5 nF y funciona a una frecuencia de 2 GHz y a una tensión de alimentación de 1 V.

a) **Calcula** la potencia disipada por el chip de un solo core durante la ejecución de la versión secuencial.

P fugas = I*V = 2,5 A * 1 V = 2,5 W por core
P conmutación = $C*V^2*F = 5x10^{-9}$ F * $(1 V)^2$ * $2x10^9$ Hz = 10 W por core
P chip = 2,5 W + 10 W = 12,5 W

Hemos programado una versión paralela de la misma aplicación para un multiprocesador con 4 cores (idénticos al anterior) integrados en el chip. La Fase 1 no ha podido ser paralelizada, por lo que se ejecutara en un solo core, sin embargo la Fase 2 ha podido ser paralelizada en 4 partes iguales de forma que el trabajo se distribuye equitativamente entre los 4 cores (el overhead debido a sincronización es despreciable).

Los procesadores actuales pueden pasar un core a modo bajo consumo cuando no está realizando ningún trabajo útil y devolverlo a modo activo cuando el core vuelve a realizar trabajo útil. En nuestro caso, en modo bajo consumo, el voltaje del core se reduce a 0,8 V y la frecuencia a 1 GHz. Además, los últimos procesadores aparecidos en el mercado son capaces de aumentar la frecuencia de funcionamiento, de modo que durante las fases en que algunos cores están en modo bajo consumo (no hacen trabajo útil), el resto puedan realizar su trabajo más rápido. Este modo lo denominaremos modo turbo (AMD lo denomina Turbo Core e Intel Turbo Boost). En nuestro caso, en modo turbo, un core aumenta el voltaje a 1,2 V y la frecuencia a 3 GHz. Obsérvese que, durante la Fase 1 el procesador activo estaría en modo turbo, mientras el resto estarían en modo bajo consumo, durante la Fase 2 los 4 cores estarían en modo normal.

b) **Calcula** el speed-up al ejecutar la versión paralela en el multiprocesador de 4 cores respecto a la versión secuencial ejecutada en un solo core.

Aumentar la frecuencia a 3 GHz ---> speedup fase 1 = 3 GHz / 2 GHz = 1,5
Paralelizar ----> speed-up fase 2 = 4

S = 1/(0,2/1,5+0,8/4) = 3

c) **Calcula** la potencia disipada por el chip de 4 cores durante la ejecución de la Fase 1 en la versión paralela

Procesadores bajo consumo:
P fugas = 2,5 A * 0,8 V = 2 W por core
P conmutación = $5x10^{-9}$ F * $(0,8 V)^2$ * $1x10^9$ Hz = 3,2 W por core

Procesador turbo:
P fugas = 2,5 A * 1,2 V= 3 W por core
P conmutación = $5x10^{-9}$ F * $(1,2 V)^2$ * $3x10^9$ Hz = 21,6 W por

P chip = 3W + 21,6 W + 3 * (2W+3,2W) = 40,2 W

d) **Calcula** la potencia media disipada por el chip de 4 cores durante la ejecución de la versión paralela.

La Fase 1 representa (0,2*t/1,5)/(t/3) = 40% del tiempo
La Fase 2 representa (0,8*t/4)/(t/3) = 60% del tiempo

P media = 0,4 * 40,2 W + 0,6 * (4*12,5 W) = 46,08 W

e) **Calcula** la ganancia en energía de la versión paralela ejecutada en el multiprocesador de 4 cores respecto a la versión secuencial ejecutada en un solo core

E = P*t ---> Es = Ps*ts ; Ep = Pp*ts/3

G = Ps*t/(Pp*t/3) = Ps/Pp*3 = 12,5 W / 46,08 W * 3 = 0,81

En este multiprocesador, cada core tiene una cache de datos (D1) y una de instrucciones (I1) local. Además, los 4 cores comparten una cache unificada de segundo nivel (L2). Las caches de datos de los 4 cores se comunican entre ellas y con L2 por medio de un bus compartido.

Con un juego de datos determinado, la versión secuencial de la aplicación, ejecutada en un solo core en modo normal (a 2GHz), ha tardado $15x10^9$ ciclos y se han realizado $5x10^9$ accesos a datos (el impacto de los accesos a instrucciones es despreciable). En la versión secuencial, la tasa de fallos en el primer nivel de cache de datos (D1) ha sido del 10% y se ha medido un tiempo de penalización medio por fallo de 20 ciclos (incluye los accesos a L2 y los fallos de L2 que se han servido desde memoria principal). La tasa de fallos y el tiempo de penalización es el mismo en la Fase 1 y la Fase 2, y el numero de accesos se distribuye proporcionalmente al tiempo de cada fase (20% y 80% respectivamente).

f) **Calcula** cuantos ciclos tardaría la versión secuencial si no hubiese fallos en la cache de datos.

Ciclos secuencial ideal = $15x10^9$ciclos - $5x10^9$ accesos * 0,10 fallos/acceso * 20 ciclos/fallo = $5x10^9$ciclos (a 2 GHz)

En la versión paralela la Fase 1 se ejecuta sobre un solo core a 3GHz (modo turbo). La tasa de fallos se ha mantenido en el 10% pero el tiempo medio de penalización por fallo es de 30 ciclos.

g) **Explica** porque la penalización por fallo ha pasado de 20 a 30 ciclos siendo las caches iguales en ambos procesadores. **Calcula** los ciclos de la Fase 1 de la versión paralela ejecutada en el multiprocesador de 4 cores teniendo en cuenta la jerarquía de memoria

El procesador funciona a 3 GHz por lo que la penalización se corresponde a más ciclos aunque tarde el mismo tiempo.

Fase 1:
Accesos = $5x10^9$ accesos * 0,2 = $1x10^9$acccesos
Ciclos ideal = $5x10^9$ ciclos * 0,2 = $1x10^9$ ciclos

Ciclos Fase 1 Paralela = $1x10^9$ciclos + $1x10^9$accesos * 0,10 f/a * 30 c/f = $4x10^9$ ciclos (a 3 GHz)

Durante la Fase 2, en la versión paralela, los 4 cores están en modo normal (2 GHz), la tasa de fallos de las caches de datos (D1) ha aumentado al 15% por las invalidaciones debidas al mecanismo de coherencia entre los distintos cores. Respecto al tiempo de penalización por fallo, en la Fase 2, pueden darse 2 situaciones: 1) el bloque de memoria puede obtenerse de la cache D1 de otro core, en cuyo caso el tiempo de penalización es de 6 ciclos; 2) debe obtenerse de L2, en cuyo caso el tiempo de penalización medio es de 21 ciclos (ligeramente mayor que en la versión secuencial debido a la mayor saturación del bus de datos). Se ha observado que el 60% de los fallos son servidos desde otro core y el resto desde L2.

h) **Calcula** cuantos ciclos tardaría la Fase 2 de la versión paralela ejecutada en el multiprocesador de 4 cores teniendo en cuenta la jerarquía de memoria.

Fase 2 secuencial:
Accesos sequencial = $5x10^9$ accesos *0,8 = $4x10^9$ acccesos
Ciclos secuencial ideal = $5x10^9$ciclos *0,8 = $4x10^9$ciclos

Fase 2 paralela:
Ciclos ideal por core = $4x10^9$ciclos/4 = $1x10^9$ciclos
Accesos por core = $4x10^9$ accesos/4 = $1x10^9$accesos
Ciclos por core = $1x10^9$ciclos + $1x10^9$accesos * 0,15 f/a * (0,6 * 6 c/f + 0,4 * 21 c/f) = $2,8x10^9$ ciclos (a 2 GHz)

i) **Calcula** el speed-up de la versión paralela sobre la secuencial teniendo en cuenta la jerarquía de memoria.

Tiempo secuencial = $15x10^9$ ciclos / $2x10^9$ ciclos/s = 7,5 segundos
Tiempo paralelo = $4x10^9$ ciclos / $3x10^9$ ciclos/s + $2,8x10^9$ ciclos / $2x10^9$ ciclos/s = 2,73 segundos

speed-up = 7,5 segundos / 2,73 segundos = 2,74

Cognoms: . . . . . Nom: . . . . . .

Examen Final d'Arquitectura de Computadors **Curs 2011-2012 Q2**

Problema 3. (2 puntos)

En la figura adjunta se presenta el código de un sistema de control industrial. En este código se identifican 3 fases: **`RecolectarDatos`**, en esta fase básicamente se leen 4GB ($4 \cdot 10^9$ bytes) de datos, en bloques de 1MB (10^6 bytes) del sistema de almacenamiento; **`ProcesarDatos`**, en esta fase se procesan los datos leídos (esta fase es totalmente paralelizable); y **`MostrarResultados`**, en esta fase se procesan los resultados de la fase previa para presentar los resultados (esta fase es totalmente secuencial).

```
for (;;) {
  RecolectarDatos();
  ProcesarDatos();
  MostrarResultados();
}
```

Una iteración de esta aplicación tarda en ejecutarse 80s, distribuidos así: `RecolectarDatos` 20s, `ProcesarDatos` 55s, `MostrarResultados` 5s.

a) Suponiendo que disponemos de un infinito número de procesadores, calculad cual sería el speedup máximo que podríamos obtener si paralelizamos esta aplicación.

Speedup máximo = 80 s / 25 s = 3,2

En realidad, es muy difícil paralelizar una aplicación sin tener algún tipo de penalización. Se ha evaluado que paralelizar esta aplicación tiene una penalización fija de 5 s y una penalización variable de 1 s por cada procesador (por iteración).

b) Calculad el número de procesadores óptimo para esta aplicación, así como el tiempo de ejecución.

Tiempo (N) = 25 + 5 + 55/N + N

Tiempo(N)' = $-55/N^2 + 1$ ---> N = $55^{1/2}$ = 7,41 ~ 7 CPUs

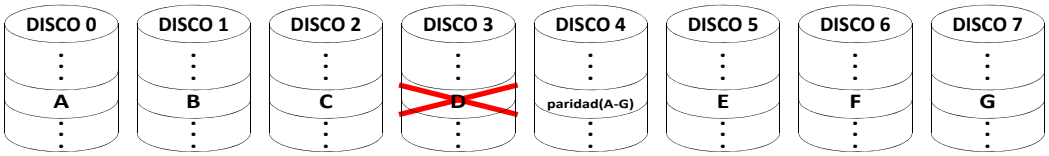
Tiempo (7) = 25 + 5 + 55/7 + 7 = 44,85 s

Para optimizar esta aplicación necesitamos mejorar las fases no paralelas. Hemos decidido utilizar un sistema de almacenamiento RAID5 con 8 discos de 1TB, y un ancho de banda de 200 MB/s por disco y bloques de 1MB.

c) **Calculad** el tiempo de ejecución de RecolectarDatos, contando exclusivamente el tiempo de transferencia de información y suponiendo que los 4GB son accesos aleatorios uniformemente distribuidos.

Tiempo = 4 GB / (8·200 MB/s) = 2.5 s

Esta aplicación se ejecuta en un entorno industrial en condiciones extremas. Esto provoca que algunos accesos al sistema de almacenamiento no se realicen por un error de lectura. Cada vez que se produce un error en lectura, el controlador de disco ha de obtener el bloque que no ha podido leer utilizando el resto del RAID5.En la siguiente figura mostramos de forma parcial el RAID5, con el contenido de los bloques A, B, ... G, así como su paridad, que en este caso estaría almacenado en el disco4.



d) Suponiendo que no hemos podido leer el bloque D, qué hay que hacer para que el RAID5 pueda obtener el bloque D.

Hay que leer A, B, C, paridad, E, F y G

D = A xor B xor C xor paridad xor E xor F xor G

e) Si el 5% de los accesos a un bloque en el RAID producen un error de lectura, ¿cual sería el tiempo de penalización por esta causa?

Bloques que producen error: 4000 · 0,05 = 200.

Cada error obliga a leer 1 bloque de cada disco -> Tiempo penalización = 200 · 1 MB / 200 MB/s = 1s

Problema 4. (2 puntos) (problema 3.8 de la colección)

Tenemos un procesador con memoria virtual basado en paginación. El sistema de memoria virtual tiene 16 bits de dirección lógica, 15 bits de dirección física y páginas de 8Kbytes. El contenido inicial de la tabla de páginas se muestra en la Figura 1, donde: VPN = virtual page number, P = bit de presencia, M = bit de página modificada y PPN = physical page number. La Figura 2 muestra el contenido de la memoria física.

Figura 1 Contenido **inicial** de la **Tabla de Páginas**

VPN	P	M	PPN
0	0	0	-
1	0	0	-
2	1	0	0
3	1	0	1
4	1	0	2
5	0	0	-
6	0	0	-
7	0	0	-

Figura 2 Contingut **inicial** de **Memoria**

página física	página lógica
0	2
1	3
2	4
3	-

En caso que la memoria física se llene y haga falta reemplazar una página se sigue un algoritmo de reemplazo LRU. De las 3 páginas inicialmente presentes en memoria:

- la página física 1 (lógica 3) es la más recientemente accedida
- la siguiente es la página física 0 (lógica 2)
- la que hace más tiempo que ha sido accedida es la física 2 (lógica 4).

a) **Rellena** la siguiente tabla indicando, para cada referencia, la página lógica (VPN), el desplazamiento, la dirección física resultante de la traducción, indicando con una cruz (X) cuando se produce fallo de página, cuando se lee de disco duro y cuando se escribe en disco duro, y en caso de reemplazar una página indicad el VPN y el PPN de la página reemplazada.

dirección lógica (hexa)		**VPN (hexa)**	**desplazamiento (hexa)**	**dirección física (hexa)**	**fallo de página**	**lectura disco**	**escritura disco**	**Página reemplazada**	
								VPN	**PPN**
escritura	F458	7	1458	7458	X	X			
escritura	8666	4	0666	4666					
lectura	1BBF	0	1BBF	1BBF	X	X		2	0
escritura	5C44	2	1C44	2C44	X	X		3	1
lectura	6600	3	0600	6600	X	X	X	7	3
lectura	4000	2	000	2000					

b) **Indica** el contenido final de la tabla de páginas y de la memoria física

Contenido **final** de la **Tabla de Páginas**

VPN	P	M	PPN
0	1	0	0
1	0		
2	1	1	1
3	1	0	3
4	1	1	2
5	0		
6	0		
7	0		

Contenido **final** de **Memoria**

página física	página lógica
0	0
1	2
2	4
3	3