**Parte teoria [8 puntos]**

**Vectorizacion**

1. **Indicar la(s) afirmación(es) correcta(s): [1 punto]**

* A. La longitud de un vector SIMD se expresa en bits o Bytes
* B. Un set de instructions SIMD describe el proceso de fabricación de un procesador vectorial
* C. Un vector de 4 floats tiene un ancho de 16 Bytes
* D. Un procesador con soporte AVX512 puede procesar vectores de hasta 512 bits de ancho

1. **Indicar la(s) afirmación(es) correcta(s): [1 punto]**

* A. Añadiendo el atributo restrict (C/C++) a argumentos de funciones permite evitar el *aliasing*
* B. La ventaja de usar *instrinsics* para vectorizar es la alta portabilidad de código
* C. El uso de pragmas esta basado en compilar añadiendo opciones como por ejm. -fno-trapping-math
* D. La sgte. sentencia lee 256 bits de memoria y los asigna a una variable: \_\_m256d var\_v = \_mm256\_load\_pd(&var[i]);

1. **Indicar la(s) afirmación(es) correcta(s): [1 punto]**

* A. El componente *linker* de un compilador permite transformar el código fuente en lenguaje ensamblador
* B. La portabilidad del lenguaje ensamblador es limitada
* C. La auto-vectorización podría producir un resultado incorrecto
* D. Un programa escrito en lenguaje ensamblador es por lo general más extenso que su equivalente en C/C++

1. **Considerar la sgte sección de código: [1 punto]**

40 void examen\_cpar2 (float\* a, float\* b, float\* c) {

41 for(int i = 0; i < 100; i++) {

42 c[i] = a[i] + b[i] \* 2;

43 }

44 }

Cuando compilamos el programa completo, obtenemos el sgte. mensaje:

exam\_cpar2.c: 41:4: note: loop versioned for vectorization because of possible aliasing

* **Usando un máximo dos oraciones, explicar cual es el problema.**

Probablemente esté tratando de auto vectorizar usando una versión antigua de gcc.

* **Usando máximo dos oraciones, indicar como solucionar el problema.**

Una solución es actualizando la versión de gcc a una actual v9.4.0, y otro es agregar \_\_restrict\_\_ para decirle al compilador que no hay una overlap en la autovectorizacion..

**OpenMP**

1. Indicar la(s) afirmación(es) correcta(s): [1 punto]

* A. OpenMP se aplica a sistemas de memoria distribuida
* B. OpenMP permite paralelizar un programa usando múltiples hilos (threads)
* C. OpenMP requiere soporte hardware pero no del compilador
* D. El pragma OpenMP #pragma omp parallel distribuye iteraciones de un bucle for entre múltiples hilos

1. Indicar la(s) afirmación(es) correcta(s): [1 punto]

* A. En OpenMP, el hilo maestro (master thread) tiene un id variable cuyo valor es asignado durante la ejecución de un programa
* B. Respecto a la jerarquía de memoria en un sistema computacional: los niveles de memoria de mayor capacidad tienen accesos más veloces
* C. En OpenMP, las variables pueden tener los sgtes tipos de alcance: private, shared, and virtual
* D. En OpenMP, cada hilo tiene su propia región de memoria ("stack") donde almacena variables privadas

1. Indicar la(s) afirmación(es) correcta(s): [1 punto]

* A. Indique un método de OpenMP para sincronizar hilos

#pragma omp barrier

* B. Indique un método de OpenMP para ejecutar de manera serial

#pragma omp barrier

* C. Usando máximo dos oraciones, indique para que sirve export OMP\_NUM\_THREADS=8

Para especificar el numero de thread a usar.

* D. Usando máximo dos oraciones, explique en que consiste en una condición de carrera de hilos

Cuando los threads compiten por los mismos recursos, lo que implica un problema.

1. Considerar la sgte sección de código: [1 punto]

1 #include <stdio.h>

2 #include <omp.h>

3

4 int main(int argc, char \*argv[]){

5 #pragma omp parallel

6 {

7 int nthreads = omp\_get\_num\_threads();

8 int thread\_id = omp\_get\_thread\_num();

9 #pragma omp single

10 {

11 printf("Goodbye slow serial world and Hello OpenMP!\n");

12 printf(" I have %d thread(s) and my thread id is %d\n",nthreads,thread\_id);

13 }

14 }

15 }

* **A. Indicar el alcance de las variables nthreads y thread\_id**

nthreads: obtiene/consulta al sistema el número de threads existentes en el nodo

Thread\_id: Consulta al sistema el id.

* **B. Indicar el valor numérico impreso en el stdout de la variable thread\_id cuando se ejecuta la línea 12.**

Imprimirá solo un valor entre 0 y el número máximo de threads, debido al pragma: #pragma omp single