Evaluación Taller 3. Extensión vectorial SIMD: SSE

Micro-investigación

1. ¿Qué es el set SSE, cual es su utilidad y qué aplicaciones tiene?

SSE (Streaming SIMD Extensions) es una extensión al grupo de instrucciones MMX para procesadores Pentium III, introducida por Intel en febrero de 1999. Las instrucciones SSE son especialmente adecuadas para decodificación de MPEG2, que es el códec utilizado normalmente en los DVD, procesamiento de gráficos tridimensionales y software de reconocimiento de voz. Estas instrucciones operan con paquetes de operandos en coma flotante de precisión simple (FP).

2. ¿Cómo realiza la compilación de un código c (.c) que utilice el set SSEx de Intel?

Los compiladores de C/C++ vienen con los llamados encabezados intrínsecos SSE. Estos proporcionan un conjunto de funciones C, que brindan acceso casi directo a las instrucciones vectorizadas y los tipos de datos empaquetados necesarios.

3. ¿Que importancia tienen la definición de variables y el alineamiento de memoria al trabajar con un set SIMD vectorial, como SSE?

Los procesadores de Intel y AMD transferirán los datos desde y hacia la memoria a los registros más rápido si los datos se alinean con los límites de 16 bytes. Si bien el compilador se ocupará de esta alineación cuando use el tipo básico de 128 bits, significa que los datos deben almacenarse de manera óptima en conjuntos de cuatro valores de punto flotante de 32 bits en la memoria. Este es un obstáculo más que hay que enfrentar al usar SSE. Si los datos no se almacenan de esta manera, se necesitan movimientos de memoria escalar no alineados más costosos en lugar de movimientos alineados de 128 bits empaquetados.

Ejercicios prácticos

1. hello_simd.c:

    1.1. ¿Qué operación realiza el mismo? ¿Qué instrucciones SIMD se están utilizando y con qué fin? ¿Qué versión de SSE utiliza el código?

El código del ejemplo hello_simd.c realiza una operación de suma de vectores en la cual se suma cada elemento del vector 1, contra su correspondiente elemento por posición en el vector 2; vector 1 + vector 2 = result. Se utilizan las instrucciones de SIMD:

- _mm_set_epi32: Establece enteros de 32 bits empaquetados en el vector dst con los valores proporcionados.
- _mm_add_epi32: Sumar enteros de 32 bits empaquetados en vectores, a y b, y almacene los resultados en el vector dst.
- _mm_extract_epi32: Extrae un entero de 32 bits del vector a, seleccionado con imm8, y almacene el resultado en vector dst.

El código de ejemplo usa la versión 4 de SSE

    1.2. Realice una compilación del código y ejecutelo. Escriba la salida de consola de la aplicación.

```
vidarr@esteban-Ubuntu-PC:~/Projects/TEC/Arqui 2/Talleres_Arqui_II/Taller 3/Taller1_Esteban_Sanabria$ gcc hello_simd.c -o hello
vidarr@esteban-Ubuntu-PC:~/Projects/TEC/Arqui 2/Talleres_Arqui_II/Taller 3/Taller1_Esteban_Sanabria$ ls -a
.  ..  cmake-build-debug  CMakeLists.txt  hello  hello_simd.c  .idea
vidarr@esteban-Ubuntu-PC:~/Projects/TEC/Arqui 2/Talleres_Arqui_II/Taller 3/Taller1_Esteban_Sanabria$ ./hello
Hola Mundo desde SSE
Result **********
6       8       10      12
vidarr@esteban-Ubuntu-PC:~/Projects/TEC/Arqui 2/Talleres_Arqui_II/Taller 3/Taller1_Esteban_Sanabria$
```

Referencias:

[1] "SSE", Es.wikipedia.org, 2019. [Online]. Available: https://es.wikipedia.org/wiki/SSE. [Accessed: 10- Apr- 2019].

[2] "A practical guide to SSE SIMD with C++", Sci.tuomastonteri.fi, 2019. [Online]. Available: http://sci.tuomastonteri.fi/programming/sse. [Accessed: 10- Apr- 2019].

[3] "Getting started with SSE programming", The Supercomputing Blog, 2019. [Online]. Available: http://supercomputingblog.com/optimization/getting-started-with-sse-programming/. [Accessed: 10- Apr- 2019].

[4] "Intel Intrinsics Guide", Software.intel.com, 2019. [Online]. Available: https://software.intel.com/sites/landingpage/IntrinsicsGuide/#. [Accessed: 10- Apr- 2019].