Arquitecturas Híbridas

Juan Diego López Triana
Arquitectura de Computadores
Escuela de Ingeniería de Sistemas e Informática
Universidad Industrial de Santander
jd_@outlook.com

Andrés Felipe López Suárez
Arquitectura de Computadores
Escuela de Ingeniería de Sistemas e Informática
Universidad Industrial de Santander
pipe_lopez321@hotmail.com

Laura Vanessa López Serrano
Arquitectura de Computadores
Escuela de Ingeniería de Sistemas e Informática
Universidad Industrial de Santander
lauravlopez95@hotmail.com

Abstract - This document gives a short description of previous research, the concept of hybrid architecture, its operation, the types of architectures most used and the point of view of programming. You can also find, the definition of CUDA and how great a performance is demonstrating a computer level today, we also find a comparison with the XE parallel study for Xeon Phi reviewing each of the advantages that CUDA shows how both coprocessors provide high degrees of parallelism tan can deliver excellent application performance.

Resumen- Este documento realiza una breve descripción a partir de investigaciones previas, el concepto de arquitectura hibrida, su funcionamiento, los tipos de arquitecturas más usados y el punto de vista desde la programación. Se puede encontrar también, la definición de CUDA y cómo esta provee un gran rendimiento a nivel informático en la actualidad, asimismo se encontrará un comparación con parallel studio XE para Xeon Phi revisando cómo estos dos coprocesadores proporcionan un alto grado de paralelismo que puede ofrecen un excelente rendimiento en las aplicaciones.

I. INTRODUCCIÓN

Las cosas han cambiado de manera sorprendente desde que los computadores domésticos empezaron a implantarse en nuestros hogares, si bien, nos hemos dado cuenta su evolución de forma muy notable y cómo nuestros sistemas actuales son ahora mucho más complejos. En los primeros ordenadores, el procesador central (CPU), central processing unit, era el que se encargaba de gestionar y procesar todo tipo de información, comenzando desde los datos que el usuario quería operar hasta por supuesto el sistema operativo con su interfaz, sin embargo, se ha llevado a cabo la inclusión de mejores elementos de procesamiento debido a la necesidad de mejora respecto a la imagen y gráficos, los cuales han permitido disfrutar del ocio con videojuegos visualmente muy cercanos a la realidad. Para estas tareas el trabajo del procesador gráfico o GPU es fundamental. En la actualidad, diversas aplicaciones requieren un alto rendimiento, es por eso que se ha resuelto gracias a la computación paralela la cual permite resolver problemas que no caben en una CPU y resolverlos en un tiempo razonable, añadiéndole a esto también se pueden ejecutar mucho más rápido, esto es debido a que una GPU contiene cientos o miles de pequeños procesadores con lo que su potencial para procesar algoritmos paralelos es muy superior al de una CPU. Es por esto que NVIDIA realiza la invención del GPU y observo la necesidad de aprovechar su gran potencia para proporcionar un incremento extraordinario del rendimiento del sistema creando así una arquitectura de cálculo paralelo llamado CUDA. Este ha recibido un gran impacto a nivel de la comunidad científica y en el mercado de consumo ya que gracias a este, prácticamente todas las aplicaciones de video se han acelerado, o llegaran algún día a acelerarse. Por otro lado, INTEL de igual manera vio la necesidad de tener un software que corriera mucho más rápido que llevara a cabo las tareas que hay que hacer, creando el paquete de desarrollo de software Intel Parallel studio XE, que ayuda en el rendimiento de las aplicaciones mediante el aprovechamiento de los núcleo de procesador y el ancho de registro vectorial en los procesadores Intel Xeon Phi y coprocesadores y otros posibles procesadores, en total diseñando 90 núcleos Phi coprocesador Mientras que las aplicaciones CUDA pueden ejecutarse en hardware x86, es importante saber cómo las diferencias entre la arquitectura de GPU y los coprocesadores Intel Xeon Phi afecta al rendimiento y diseño de la aplicación. El desafío reside en lograr el mejor rendimiento posible.

II. ESTADO DEL ARTE

En los últimos años, los avances tecnológicos y el factor económico son criterios determinantes a la hora de poner a prueba las capacidades de cómputo de las arquitecturas de alto rendimiento, ya que los computadores cada día evolucionan para desarrollar tareas más grandes y más complejas, pero a su vez, las limitaciones físicas que impiden el aumento de la frecuencia debido al consumo energético y por consiguiente la generación de calor, hacen que la computación en paralelo sea una alternativa muy atractiva para cumplir dichas tareas. Hoy en día, la tendencia actual es que los supercomputadores basen su diseño en arquitecturas hibridas con distintos tipos de procesadores, donde varias tareas son realizadas simultánea e independientemente para maximizar el rendimiento y la eficiencia energética de los ordenadores. Al día de hoy, Nvida ofrece su última arquitectura denominada Pascal, la cual es la más avanzada del mundo y soluciono grandes problemas de su predecesora la arquitectura Maxwell cuando se ponían a trabajar paralelamente varios procesos gráficos. Aun así

Nvidia ya tiene lista la sucesora de Pascal, la Nvidia Volta que promete dar una revolución en el área de las tarjetas gráficas. Además de esto Nvidia cuenta con sus GPU's aceleradoras como la Nvida Tesla, esta plataforma hace un uso intensivo de los procesadores gráficos paralelos para ofrecer un rendimiento superior que agiliza el procesamiento de grandes cargas de trabajo y permite ahorrar costes importantes a los centros de datos. Por otro lado esta Intel con sus coprocesadores Xeon Phi, algo similar a lo que ofrece Nvidia Tesla pero desarrollado por Intel bajo su arquitectura Intel MIC (Many integrated Core), que ofrecen 1TFLOPS de poder computacional en doble precisión. Como podemos ver, los avances tecnológicos nunca se detienen, cada día evolucionan para darnos mejores prestaciones incluso cuando aún no se necesitan. [9][6][11]

III. MARCO TEORICO

A. Computación Paralela

La computación paralela es una forma de computa en la que muchas instrucciones se ejecutan simultáneamente, está basada en el principio de grandes tareas se pueden dividir en partes más pequeñas que pueden resolverse de forma concurrente. Las computadoras paralelas pueden clasificarse según el nivel de paralelismo que admite su hardware: equipos con procesadores multinúcleo y multi-procesador que tienen múltiples elementos de procesamiento dentro de una sola máquina y los clústeres, MPPS y grids que utilizan varios equipos para trabajar en la misma tarea. Muchas veces, para acelerar tareas específicas, se utilizan arquitecturas especializadas de computación en paralelo junto a procesadores tradicionales. Hay varias formas diferentes de computación paralela: paralelismo a nivel de bit, paralelismo a nivel de instrucción, paralelismo de datos y paralelismo de tareas. [10][7]

- Paralelismo a nivel de bit: Trata cuando se aumenta el tamaño de la palabra en la computadora, al hacer esto se reduce el número de instrucciones que son necesarias para ejecutar un programa en el cual sus operadores son más grandes que el tamaño de su palabra.

- Paralelismo a nivel de instrucciones: Cuando el grupo de instrucciones que componen un programa son ejecutadas simultáneamente sin afectar el resultado final.

- Paralelismo de datos: Cuando se distribuyen los datos a través de diferentes nodos de cómputo con tareas similares, para que los datos sean procesados en paralelo y el resultado sea uno solo. La paralelización de ciclos conduce a menudo a secuencias similares de operaciones (no necesariamente idénticas) o funciones que se realizan en los elementos de una gran estructura de datos. Muchas de las aplicaciones científicas y de ingeniería muestran paralelismo de datos.

- Paralelismo de tarea: El paralelismo de tareas es la característica de un programa paralelo en la que cálculos completamente diferentes se pueden realizar en cualquier conjunto igual o diferente de datos. Esto contrasta con el paralelismo de datos, donde se realiza el mismo cálculo en distintos o mismos grupos de datos. El paralelismo de tareas por lo general no escala con el tamaño de un problema.

B. Unidad Central de Procesamiento (CPU)

La CPU de un dispositivo, es el hardware programable que interpreta las instrucciones de un programa mediante operaciones básicas aritméticas, lógicas y de entrada/salida del sistema. Un ordenador puede tener más de una CPU, a esto se le conoce como multiprocesamiento. Todas las CPU modernas son microprocesadores, lo que significa que contienen un solo circuito integrado. Algunos circuitos integrados pueden contener varias CPU en un solo chip; estos son denominados procesadores multinúcleo. [13]

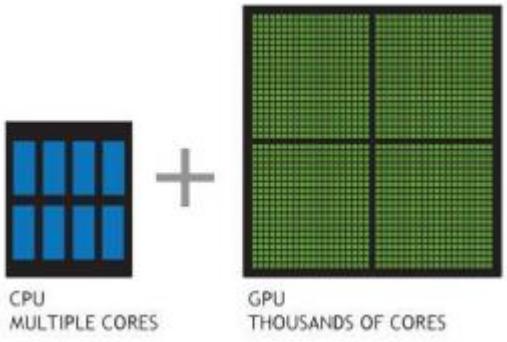

Figura 1: CPU y GPU

C. Unidad de Procesamiento Grafico (GPU)

Una GPU es un coprocesador dedicado al procesamiento de gráficos y operaciones de coma flotante, para aligerar la carga de trabajo del procesador central en aplicaciones como los videojuegos o aplicaciones 3D interactivas. De esta forma, mientras gran parte de lo relacionado con los gráficos se procesa en la GPU, la CPU puede dedicarse a otro tipo de cálculos. Una GPU se compone, entre otros, del procesador de vértices para hacer cálculos de sombreado, iluminación, etc. Y del procesador de pixeles, que se dedica exclusivamente a hacer cálculos sobre los pixeles, como efectos de agua, etc. [6]

D. Arquitecturas Híbridas:

Una arquitectura de computadores hibrida o heterogénea es la constitución de varios elementos de computo de distintas características como diferentes tipos de procesadores, los cuales no necesariamente comparten el espacio de direccionamiento de memoria ni el conjunto de instrucciones, en busca de maximizar notoriamente su rendimiento y su eficiencia energética, mediante instrucciones que permitan que la CPU y la GPU de un mismo ordenador se integren en el mismo bus con memoria y tareas compartidas. El ejemplo más común es aquel en el que se combinan CPU's multinúcleo de propósito general con aceleradores especializados. [8]

E. Arquitecturas Híbridas basadas en GPU

Estas son arquitecturas donde los procesadores multinúcleo son representados por procesadores de propósito general, y cada núcleo ejecuta las instrucciones de forma independiente

y los recursos son administrados por medio de un sistema operativo. Estos procesadores GPU han sido adaptados para el procesamiento de propósito general GPGPU (General-Purpose computing on Graphics Processing Unit), y por su composición de cientos de núcleos elementales y la competencia de las GPUs en el mercado dado su comparativo bajo costo y disponibilidad en las computadoras personales les convierten en una alternativa atractiva para el procesamiento de da-tos en paralelo, ya que cada núcleo permite la ejecución de cientos de hilos de procesamiento, y presentan un control de flujo menor que las plataformas multinúcleo enfocando más área de transistores en las unidades de procesamiento. [12]

F. Arquitecturas Híbridas basadas en coprocesadores tipo MIC's

La arquitectura Intel MIC se dirige a usuarios y aplicaciones capaces de sacar partido de un sistema procesamiento altamente paralelo, poniendo a disposición de los desarrolladores un gran número de núcleos de procesamiento de bajo consumo. Estos procesadores están fabricados con una tecnología de 22 nanómetros y un chip que puede contener hasta 61 núcleos de procesamiento, con soporte para unidades de procesamiento vectorial de ancho 512-bit. La VPU se puede utilizar para procesar 16 elementos de simple precisión u 8 de doble precisión por instrucción. Para mantener la disipación de energía por unidad de área bajo control, estos núcleos ejecutan las instrucciones en orden y funcionan a Baja frecuencia. La arquitectura está respaldada por grandes caches y un gran ancho de banda de memoria. La arquitectura MIC proporciona a los desarrolladores la ventaja de poder ser programada utilizando lenguajes de programación como C/C++ o Fortran, permitiendo mantener las herramientas y métodos de programación estándar, como son compiladores, librerías multi-thread y matemáticas para HPC, herramientas de depurado, etc. [17]

IV. CUDA

CUDA son las siglas de Compute Unified Device Architecture (Arquitectura Unificada de Dispositivos de Cómputo) que hace referencia tanto a un compilador como a un conjunto de herramientas de desarrollo creadas por NVIDIA que permiten a los programadores usar una variación del lenguaje de programación C para codificar algoritmos en GPU de NVIDA. [5]

Figura 2: NVIDA CUDA

A. HISTORIA

Las primeras GPU fueron diseñados como aceleradores de gráficos, soporte de tuberías de función fija única específicos. A partir de finales de 1990, el hardware se hizo cada vez programable, que culminó en la primera GPU de NVIDIA en 1999. En 2003, un equipo de investigadores dirigido por Ian Buck dio a conocer Brook, el primer modelo de programación ampliamente adoptado para extender C con construcciones de datos en paralelo. El uso de conceptos tales como arroyos, granos y operadores de reducción, el sistema compilador y tiempo de ejecución Brook expone la GPU como un procesador de propósito general en un lenguaje de alto nivel. Lo más importante, los programas Brook no sólo eran fáciles de escribir el código GPU sintonizado a mano, que eran siete veces más rápido que el código existente similar. [2]

NVIDIA sabía que el hardware extraordinariamente rápido tenía que ser acoplado con herramientas de software y hardware intuitivas, e invitó a Ian Buck a unirse a la compañía y empezar a evolucionar una solución para funcionar sin problemas C en la GPU. Poner el software y el hardware juntos, NVIDIA CUDA dio a conocer en 2006, la primera solución del mundo para uso general en la computación en la GPU.[2]

B. ¿QUÉ ES CUDA?

Es un modelo de plataforma de computación y programación paralela inventado por NVIDIA. Permite un aumento espectacular en rendimiento informático aprovechando el poder de la unidad de procesamiento gráfico (GPU).
CUDA intenta explotar las ventajas de las GPU frente a la CPU de propósito general utilizando el paralelismo que ofrecen sus múltiples núcleos, que permiten el lanzamiento de un altísimo número de hilos simultáneos. Por ello, si una aplicación está diseñada utilizando numerosos hilos que realizan tareas independientes (que es lo que hacen las GPU al procesar gráficos, su tarea natural), una GPU podrá ofrecer un gran rendimiento en campos que podrían ir desde la biología computacional a la criptografía, por ejemplo. [2]

CUDA, puede enviar C, C ++ y Fortran directamente a la GPU, sin necesidad de ensamblaje de idiomas, proporciona unas cuantas extensiones de C y C++ que permiten implementar el paralelismo en el procesamiento de tareas y datos con diferentes niveles de granularidad. El programador puede expresar ese paralelismo mediante diferentes lenguajes de alto nivel como C, C++ y Fortran o mediante estándares abiertos como las directivas de OpenACC. En la actualidad, la plataforma CUDA se utiliza en miles de aplicaciones aceleradas en la GPU. [1]

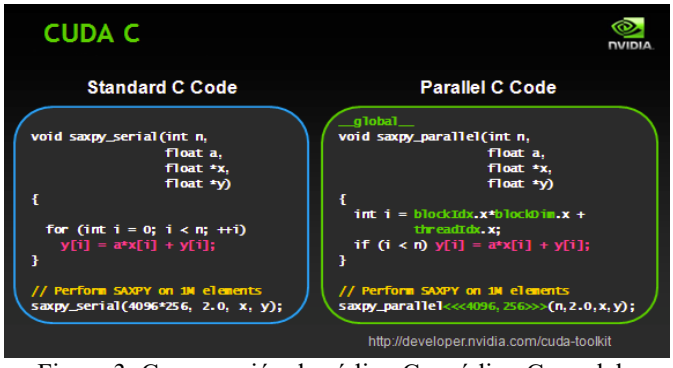

Figura 3: Comparación de código C y código C paralelo

Los desarrolladores en empresas como Adobe, ANSYS, Autodesk, MathWorks y Wolfram Research están despertando a ese gigante que duerme - la GPU - hacer de propósito general computación científica y de ingeniería a través de una variedad de plataformas. El uso de lenguajes de alto nivel, aplicaciones aceleradas por GPU ejecutan la parte secuencial de la carga de trabajo de la CPU - que está optimizado para un rendimiento de un solo subproceso - al tiempo que acelera el procesamiento paralelo de la GPU. Esto se llama "computación de la GPU." [1]

C. GPU COMPUTING

GPU computing es posible debido a que la GPU de hoy hace mucho más que representar gráficos: gracias a esto miles de desarrolladores, científicos e investigadores están encontrando innumerables aplicaciones prácticas para esta tecnología en campos como el procesamiento de vídeo e imágenes, la biología y la química computacional, la simulación de la dinámica de fluidos, la reconstrucción de imágenes de TC, el análisis sísmico o el trazado de rayos, entre otras, es ampliamente desplegado a través de miles de aplicaciones y publicados en trabajos de investigación y apoyado por una base instalada de más de 375 millones de GPUs habilitadas con ordenadores y superordenadores.

La plataforma de cómputo CUDA se extiende desde los miles de los procesadores informáticos de propósito general que aparecen en nuestra GPU de arquitectura de computación , extensiones de computación en paralelo a muchos lenguajes populares , de gran alcance de abandono en las bibliotecas acelerados para convertir las principales aplicaciones y dispositivos CUDA en notebooks, estaciones de trabajo, clusters de computacionales basados en la nube. [3]

D. ARQUITECTURA DE CUDA

CUDA se basa en una jerarquía de capas de abstracción; el hilo (thread) es la unidad de ejecución básica; Los hilos se agrupan en bloques (blocks), cada uno de los cuales se ejecuta en un sólo multiprocesador, donde pueden compartir datos a través de la memoria compartida, una memoria pequeña pero con un gran ancho de banda.

Un grid se compone de una serie de bloques, que se distribuyen y planifican por igual entre todos los multiprocesadores. Las secciones paralelas de una aplicación que se desean ejecutar en la GPU se definen como kernels. Estos kernels se ejecutan siguiendo un paradigma de programación conocido como

SPMD (Single Program Multiple Data), es decir, todos los hilos ejecutan el mismo código pero sobre distintos datos. Los hilos de un mismo bloque se dividen en grupos de 32 (en las arquitecturas Nvidia actuales), denominados Warps. El Warp es la unidad de planificación que se mapea a los cores de un multiprocesador, y se ejecuta en modo SIMD.[17]

El modelo de programación CUDA asume que el kernel CUDA se ejecuta en un dispositivo físicamente separado que opera como un coprocesador corriendo un programa C en el host, tal y como se muestra en la siguiente Figura 4.

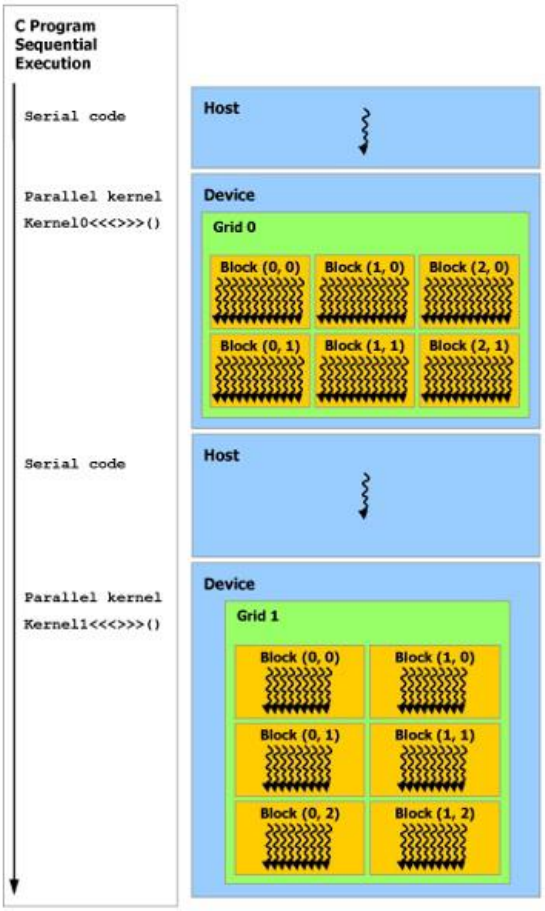

Figura 4: Modelo de ejecución de CUDA

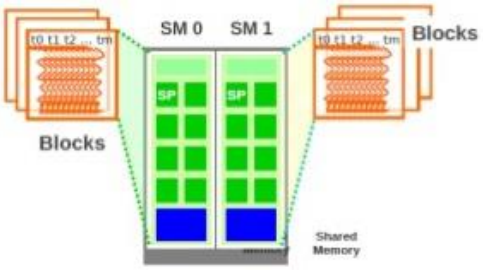

Figura 5: Hilos agrupados en bloques

El modelo de programación CUDA también asume que tanto el host como el device mantienen sus propios espacios de memoria por separado que además están implementados con diferentes tecnologías (DDR y GDDR), ofreciendo así diferentes anchos de banda. [17]

Kepler and Maxwell son las últimas generaciones de la arquitectura CUDA de Nvidia. En comparación con diseños anteriores (Fermi), ellas amplían el número de núcleos dentro de un multiprocesador de 32 a 192 (128 SMM en Maxwell), y las unidades de programación de 2 a un máximo de 8 warps a la vez. Además, la cache L2 duplica su tamaño. A pesar del aumento de los recursos (que resulta en una elevada cantidad de transistores por unidad de superficie), las GPU modernas son tres (seis) veces más eficientes energéticamente que las generaciones anteriores. Esto se consigue principalmente manteniendo la frecuencia por debajo de 1 GHz y utilizando un proceso de fabricación de 28 nm. [17]

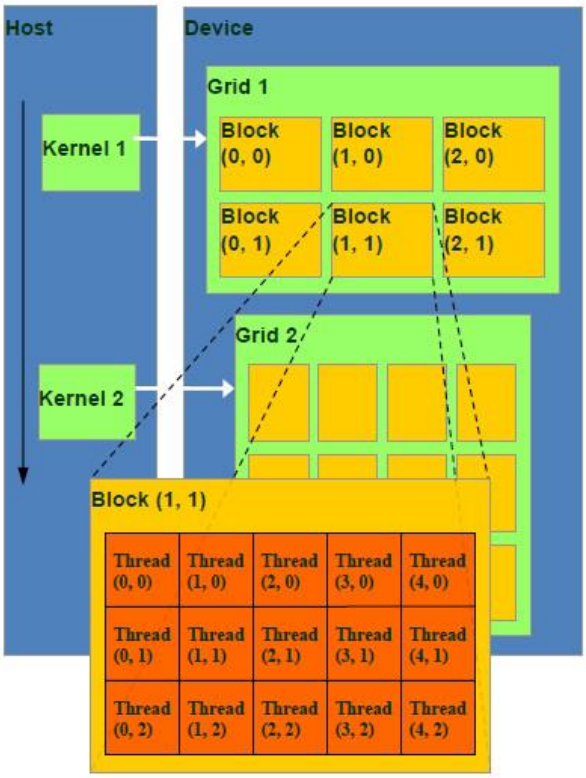

Figura 6: Bloques que componen un grid

CUDA funciona de una manera sencilla, esta puede ser resumida en 4 pasos:

1. Se copian los datos de la memoria principal a la memoria de la GPU.
2. La CPU encarga el proceso a la GPU.
3. La GPU lo ejecuta en paralelo en cada nucleo.
4. Se copia el resultado de la memoria de la GPU a la memoria principal. [18]

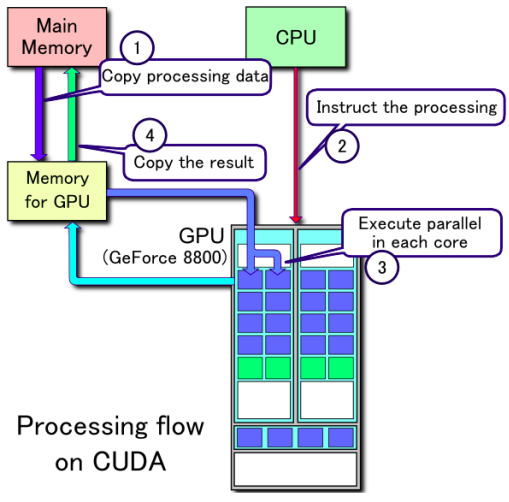

Figura 7: Funcionamiento sencillo de CUDA

E. AMPLIAMENTE UTILIZADO POR LOS INVESTIGADORES

En el mercado de consumo, prácticamente todas las aplicaciones de vídeo se han acelerado, o pronto se acelerarán, a través de CUDA, como demuestran diferentes productos de Elemental Technologies, MotionDSP y LoiLo, Inc. CUDA ha sido recibida con entusiasmo por la comunidad científica. Por ejemplo, se está utilizando para acelerar AMBER, un simulador de dinámica molecular empleado por más de 60.000 investigadores del ámbito académico y farmacéutico de todo el mundo para acelerar el descubrimiento de nuevos medicamentos.

En el mercado financiero, Numerix y CompatibL introdujeron soporte de CUDA para una nueva aplicación de cálculo de riesgo de contraparte y, como resultado, se ha multiplicado por 18 la velocidad de la aplicación. Cerca de 400 instituciones financieras utilizan Numerix en la actualidad.

Un buen indicador de la excelente acogida de CUDA es la rápida adopción de la GPU Tesla para aplicaciones de GPU Computing. En la actualidad existen más de 700 clusters de GPUs instalados en compañías Fortune 500 de todo el mundo, lo que incluye empresas como Schlumberger y Chevron en el sector energético o BNP Pariba en el sector bancario.

Por otra parte, la reciente llegada de los últimos sistemas operativos de Microsoft y Apple (Windows 8 y Snow Leopard) está convirtiendo el GPU Computing en una tecnología de uso masivo. En estos nuevos sistemas, la GPU no actúa únicamente como procesador gráfico, sino como procesador paralelo de propósito general accesible para cualquier aplicación. [5]

F. VENTAJAS Y LIMITACIONES

1. VENTAJAS

- Lecturas dispersas: se puede consultar cualquier posición de memoria.
- Memoria compartida: CUDA pone a disposición del programador un área de memoria de 16KB (o 48KB en la serie Fermi) que se compartirá entre threads. Dado su tamaño y rapidez puede ser utilizada como caché.
- Lecturas más rápidas de y hacia la GPU.
- Soporte para enteros y operadores a nivel de bit. [4]

2. LIMITACIONES

- No se puede utilizar recursividad, punteros a funciones, variables estáticas dentro de funciones o funciones con número de parámetros variable
- No está soportado el renderizado de texturas
- En precisión simple no soporta números desnormalizados o NaNs
- Puede existir un Cuello de botella entre la CPU y la GPU por los anchos de banda de los buses y sus latencias.
- Los threads o Hilo de ejecución, por razones de eficiencia, deben lanzarse en grupos de al menos 32, con miles de hilos en total. [4]

V. PARALLEL STUDIO XE PARA XEON PHI

PARALLEL STUDIO XE

Las necesidades de tener software más rápido para llevar a cabo tareas que hay que hacer cuanto antes, incluyendo análisis de datos, análisis de imágenes, aplicaciones machine learning, análisis financiero de tiempo crítico y simulación entre otros, ha llevado a Intel a crear Intel Parallel Studio XE, un paquete de software que ayuda al rendimiento de las aplicaciones, impulsadas mediante el número de núcleos del procesador y el ancho de registro vectorial disponible en los procesadores Intel Xeon, Intel Xeon Phi y otros procesadores disponibles. [15]

Figura 8: Intel Parallel Studio XE

Intel Parallel Studio facilita el desarrollo de código nativo en Windows y Linux en C++ / C y Fortran para la computación paralela . La programación en paralelo permite que los programas de software puedan tomar ventaja de los procesadores multi-core de Intel y otros fabricantes de procesadores.

Parallel Studio se compone de varios elementos, para cada uno de los cuales es un conjunto de capacidades. [15]

- Intel C ++ Compiler con OpenMP
- Intel Fortran Compiler con OpenMP
- IDE plug-in de integración con Visual Studio y Eclipse
- Depuración a través del depurador de Visual Studio extensiones, GNU depurador extensiones
- Intel Asesor - especializada de perfiles para optimizar el rendimiento de vectorización y un sistema de prototipado hilo para añadir / mejora de roscado.
- Intel VTune amplificador (anteriormente VTune Performance Analyzer) es un generador de perfiles de rendimiento que analiza puntos de acceso y concurrencia .
- Intel Inspector mejora la fiabilidad mediante la identificación de errores de memoria con hilo errores. [15]

Actualmente la última versión usada es Intel Parallel Studio 2017.

VI. XEON PHI

El procesador Intel Xeon Phi es un procesador host de inicio automático que proporciona enorme paralelismo y vectorización para admitir las aplicaciones de informática de alto desempeño más exigentes. La arquitectura integrada y eficaz en el consumo de energía proporciona, significativamente, mayor cantidad de procesos por unidad de energía consumida, en comparación con plataformas semejantes, para ofrecerle un costo total de propiedad mejor. La integración de memoria y estructura está por encima del máximo de memoria y reduce costos para ayudarlo a resolver sus desafíos más grandes rápidamente. [14]

Figura 9: Intel Xeon Phi

A. ARQUITECTURA DEL INTEL XEON PHI

El procesador Intel Xeon Phi está basado en la arquitectura Intel MIC. La arquitectura Intel MIC se dirige a usuarios y aplicaciones capaces de sacar partido de un sistema procesamiento altamente paralelo, poniendo a disposición de los desarrolladores un gran número de núcleos de procesamiento de bajo consumo.

Fabricado con el proceso tecnológico 14 nm de Intel, el procesador Intel Xeon Phi proporciona hasta 72 núcleos fuera de orden, instrucciones para Intel Advanced Vector Extensions 512 y hasta 16 GB de memoria integrada de alto ancho de banda, junto con la capacidad de la plataforma de memoria de 384 GB DDR4. El resultado de esta arquitectura de vanguardia es más de 3 teraFLOPS (operaciones de punto flotante por segundo) de doble precisión a tan solo 215 W por procesador. [17]

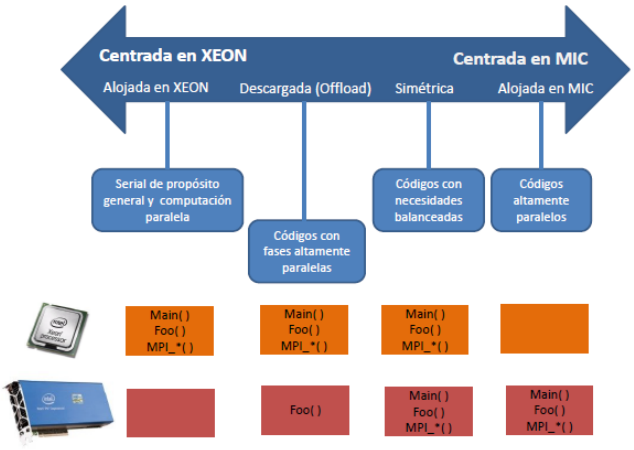

Figura 10: Esquema de un procesador y un coprocesador Xeon Phi

La arquitectura MIC proporciona a los desarrolladores la ventaja de poder ser programada utilizando lenguajes de programación como C/C++ o Fortran, permitiendo mantener las herramientas y métodos de programación estándar, como son compiladores, librerías multi-thread y matemáticas para HPC, herramientas de depurado, etc. Intel Xeon Phi utiliza una distribución completa del sistema operativo Linux, de tal manera que el mismo código fuente de programas escritos para Intel MIC pueden recompilarse y ejecutarse en un procesador Intel Xeon estándar. [17]

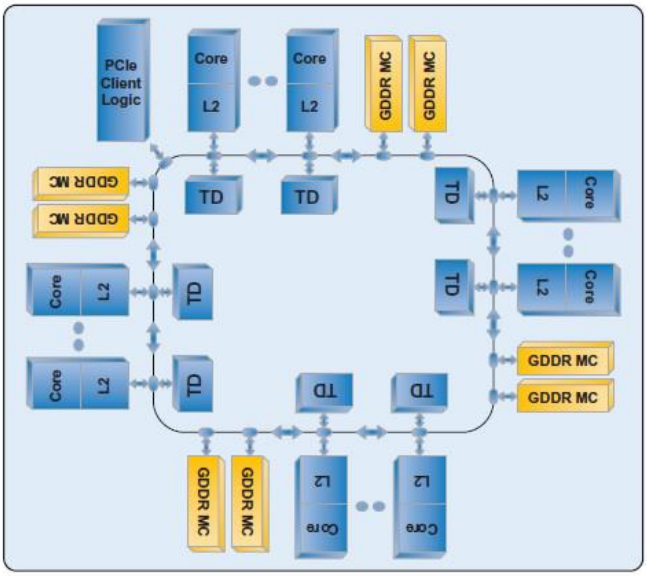

Figura 11: Esquema general de la arquitectura del Xeon Phi

La arquitectura está diseñada para poder ejecutar hasta cuatro hilos por núcleo de forma independiente, donde cada núcleo (en orden) puede decodificar hasta dos instrucciones por ciclo. Esto permite una velocidad de ciclo más alta y facilita la gestión de los hilos. A diferencia de arquitecturas orientadas a latencia, la arquitectura MIC asume que las aplicaciones que se ejecutan en el sistema serán altamente paralelas y escalables. Con el fin de ocultar la latencia cache-memoria, la política de planificación intercambia hilos en cada ciclo. Cuando una aplicación se ejecuta en un solo hilo por núcleo, el planificador cambia a un hilo nulo especial antes de volver al hilo de la aplicación. Por tanto, Intel recomienda tener en ejecución al menos dos hilos por núcleo, mientras el óptimo puede variar entre 2 y 4. La ejecución de un solo hilo por núcleo reducirá la capacidad máxima del sistema a la mitad. Existen dos unidades de procesamiento, una unidad escalar que ejecuta código x86 y x87, y una unidad de procesamiento vectorial (VPU - Vector Processing Unit). [17]

B. APLICACIONES CON PROCESADORES XEON PHI

Dependiendo de las necesidades de computo de la aplicación, la ejecución puede ser iniciada en un procesador central o en uno o más coprocesadores. En cualquier caso, el rendimiento óptimo de la aplicación se puede alcanzar en muchos casos utilizando una combinación de cálculo entre el procesador y el coprocesador. Para ello, Intel Xeon Phi proporciona diversos modos de programación extremadamente flexibles, pudiendo ser utilizado como un coprocesador o como un nodo independiente. Podemos distinguir los siguientes cuatro modos de uso. [16]

1. Ejecución centrada en CPU: Este modo se usa en dos circunstancias: i) computación CPU pura, que comúnmente se aplica a los programas secuenciales y/o con bajo grado de paralelismo, y ii) computación centrada en CPU, con cooperación de MIC, que se aplica a los programas de cómputo secuencial que contienen segmentos de computación altamente paralela. La Programación en Intel Xeon Phi CPU inicia la función principal del programa. Cuando se llega a la parte de computación altamente paralela, la computación paralela se ejecuta en la MIC. [16]

2. Ejecución descargada (Offload) en MIC. Aplicado a programas de computación altamente paralelos con un segmento parcial de computación secuencial (habitualmente al inicio y fin de la aplicación para las operaciones de E/S). Dichos programas son iniciados por la CPU, descargados a la MIC para que ejecute la computación paralela, y finalmente terminados en la CPU. La CPU se encarga principalmente de las operaciones de control y tareas de transmisión de datos (raramente realiza tareas de computación). [16]

3. Ejecución CPU y MIC de igual a igual. Se aplica a múltiples programas de computación en paralelo. Dichos programas son iniciados por CPU y MIC al mismo tiempo. [16]

4. Ejecución en MIC (nativo). Este modo se aplica a programas de computación altamente paralelos que se ejecutan nativamente en MIC.[16]

C. EVOLUCIÓN DE XEON PHI: KNIGHTS LANDING

Knights Landing utiliza núcleos de CPU x86 compatibles plenamente con extensiones AVX512 para ofrecer un rendimiento vectorial similar a los obtenidos por las GPUs: 3 TFLOPs trabajando con datos en doble precisión, y 6 TFLOPs en simple precisión, operando a una frecuencia de reloj de 1.31 GHz. Knights Landing utiliza una microarquitectura de CPU basada en el diseño Silvermont de Intel optimizada para HPC, eliminando la necesidad de un procesador anfitrión separado (host). Se espera que obtengan un rendimiento aproximado de un 80% del de un servidor dual-socket consumiendo sólo una cuarta parte de la energía, o lo que es lo mismo, una mejora de la eficiencia energética de 3.5x FLOPs/Watt. Knights Landing estará disponible en tres familias de productos, los cuales tendrán hasta 16 GB de MCDRAM. La diferencia entre estas familias será cómo interactúa el procesador con el resto del sistema. Dos de ellas podrán acceder a memoria (DDR4), mientras que la tercera es una tarjeta coprocesadora concebida como una actualización para los clientes Knights Corner que carecerá de soporte DDR4 y sólo utilizará MCDRAM. [14]

Su arquitectura interna está organizada en una malla de 54 bloques (tiles), con ocho controladores MCDRAM colocados a lo largo de los bordes norte y sur de la malla, y dos controladores de memoria DDR4-2400, cada uno de ellos con tres canales, tal y como se muestra en la Figura 12. [14]

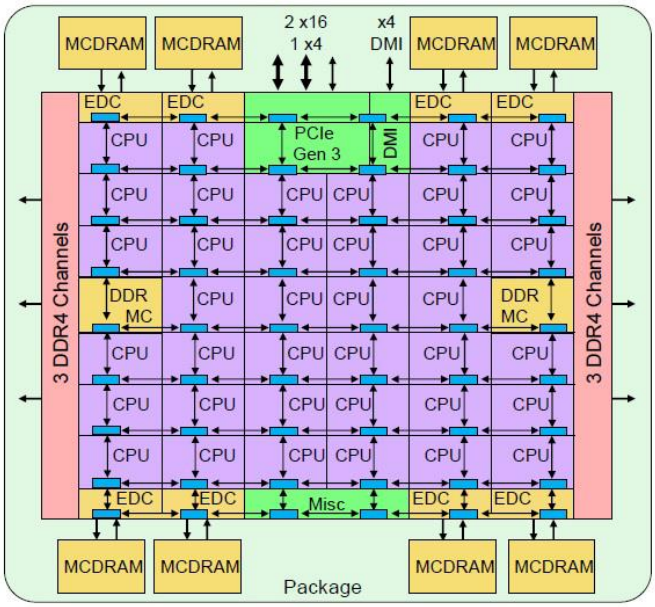

Figura 12: Arquitectura del procesador Knights Landing

Cada núcleo Knights Landing puede emitir dos instrucciones por ciclo de reloj, tiene cuatro vías multithreading simultáneas, y posee dos unidades vectoriales de 512 bits. También puede ejecutar seis uops por ciclo con capacidad fuera de orden, ofreciendo hasta 32 FLOP de doble precisión por ciclo de reloj usando las dos unidades vectoriales AVX512.
La conexión de datos entre agentes de malla es de 32 bytes de ancho. Estimaciones realizadas con el benchmark de memoria stream, sitúan el ancho de banda de cada EDC en más de 50 GB/s, lo que significa que la malla debe operar a más de 1.05 GHz. Dentro de una columna vertical, cada salto en la malla tarda un único ciclo de reloj, sin embargo, moverse a través de una fila horizontal tiene dos ciclos de latencia. Debido a que la mayoría de los agentes que sientan el tráfico de malla de inyección en la parte superior e inferior, la comunicación es enrutada y-x moviéndose primero verticalmente y luego horizontalmente. [17]

VII. COMPARACION ENTRE CUDA Y XEON PHI

Los dos coprocesadores CUDA y Phi proporcionan un alto grado de paralelismo que puede ofrecer un excelente rendimiento de la aplicación.

Para poder comprar estas dos tecnologías de alto rendimiento, en primer lugar, hay que aclarar que el Xeon phi ejecuta código de ensamblaje de Intel igual que la CPU de un ordenador cualquiera. Por otro lado, las GPU CUDA ejecutan su propio código de ensamble, esto significa que se compila código especialmente para ellos y que solo se puede ejecutar en estas GPU's. Esto no significa necesariamente que sea malo, pero si tiene algunos inconvenientes.

En particular, como primera medida se debe saber si el compilador puede dirigirse a la GPU y a su código de ensamblaje. Así que si se está comenzando un nuevo proyecto, esto no será problema ya que se pueden elegir las herramientas disponibles para trabajar sobre la GPU. Pero si se tiene un código ya existente, no se podrá simplemente descargarlo a la GPU como sí puede hacerlo con el coprocesador Xeon Phi. Además, este código se podrá ejecutar en un equipo que tenga una GPU Nvidia.

Los dos coprocesadores Intel Xeon Phi y dispositivos GPU pueden acelerar MPI (Message Passing Interface) aplicaciones en modo de descarga, donde partes de la aplicación son acelerados por el dispositivo remoto. Sin embargo, los programadores CUDA necesitan recordar que ellos también pueden ejecutar código MPI y OpenMP nativa en el coprocesador Phi. De hecho, cada Phi ejecuta Linux internamente. Aunque no es el foco de este artículo, los programadores MPI ha encontrado que aquellos que sólo puede volver a compilar las aplicaciones existentes para su ejecución en el Phi sin ningún esfuerzo portar. Ejecución de MPI nativa, cada Phi puede actuar como un nodo SMP separado en una aplicación MPI distribuida. Alternativamente, el gran número de núcleos (actualmente 61 en los dispositivos de preproducción, de los cuales 60 están disponibles para el cálculo) también significa que cada coprocesador puede actuar como un dispositivo que contiene un clúster de nodos MPI.

Intel diseñó el 60 núcleos Phi coprocesador (anteriormente denominado "MIC" en la literatura) por lo que se puede programar como un núcleo de procesador x86 convencional al tiempo que incorpora extensiones tales como interconexión anillo bidireccional para el paralelismo masivo y una amplia 512 bits por vector de núcleo unidad para ofrecer un rendimiento alto de punto flotante. Mientras que las aplicaciones CUDA pueden ejecutarse en hardware x86, es importante saber cómo las diferencias entre la arquitectura de GPU y los coprocesadores Intel Xeon Phi afecta al rendimiento y diseño de la aplicación. La buena noticia es que las aplicaciones CUDA cartografiarán fácilmente sobre la arquitectura del vector-paralelo del coprocesador Phi y correr con un alto rendimiento.

Software y Herramientas de código puerto CUDA para Phi coprocesadores A pesar de que están en las etapas finales de la pre-producción, coprocesadores Phi ya disfrutan de un ecosistema de compiladores y depuradores, perfiladores, como resultado de su compatibilidad con el hardware x86. Para ejecutar el coprocesadores Intel Xeon Phi, núcleos CUDA necesitan ser modificados. Por el momento, esto tiene que hacerse a mano. Si bien es técnicamente posible ejecutar CUDA en coprocesadores Phi, productos tales como CUDA-86 actualmente no generan código para estos dispositivos. Un compilador OpenCL para el coprocesador Intel Xeon Phi está por venir. Esto significa que los programadores pueden considerar CUDA de Wu Feng CU2CL Traductor fuente CUDA-a-OpenCL al puerto de su código. En el futuro, un LLVM proyecto de traducción podría ser capaz de crear un código ejecutable para el Phi.

Tanto las aplicaciones CUDA coprocesador Intel Xeon Phi y utilizan un gran número de hilos de ejecución concurrentes para explotar el paralelismo de hardware y lograr un alto rendimiento. Es importante entender las diferencias entre CUDA y las roscas del coprocesador Intel Xeon Phi. Desde un punto de vista programático, tanto CUDA e Intel Xeon Phi hilos coprocesador son regiones de código de serie que han sido identificados por el programador o compilador como candidatos para la ejecución en paralelo. Corresponde al

tiempo de ejecución del dispositivo para decidir cómo los hilos paralelos se ejecutarán.

Los programadores CUDA necesitan recordar que el Phi está diseñado como un coprocesador que se ejecuta en Linux. A diferencia de las GPU que actúan sólo como aceleradores, el coprocesador Phi tiene la capacidad de ser utilizado como un procesador de apoyo muy capaz simplemente mediante la compilación de las aplicaciones existentes que se ejecuten de forma nativa en él.

VIII. CONCLUSIONES

- Los coprocesadores CUDA y Intel Xeon Phi están destinados para dar altos grados de paralelismo que pueden transportar fabulosos rendimientos en las aplicaciones. Sin embargo, el reto consiste en lograr el rendimiento más ideal con menos esfuerzo de programación

- En este artículo analizamos dos de las últimas tendencias en tecnología de arquitecturas para computo de alto rendimiento, una arquitectura basada en GPU como lo es Nvidia CUDA y otra basada en coprocesadores tipo MIC como el Intel Xeon Phi, donde ambas han revolucionado la computación de alto rendimiento, gracias a sus prestaciones en el área de la computación paralela que aprovecha la unificación computacional, la eficiencia multihilo y la potencialidad de hardware.

- El rendimiento de la arquitectura basada en coprocesadores tipo MIC se compara bien con el de la GPU cuando se utilizan operaciones regulares y patrones de cálculo. La GPU es más eficiente para aquellas operaciones que realizan el acceso a datos irregulares y utilizan intensamente operaciones atómicas.

- La computación con GPU actualmente es una tendencia muy usada en campos de procesamiento de video e imágenes, biología y química computacional, simulación de fluidos entre otros, ya que, a diferencia de épocas pasadas, la GPU de hoy hace mucho más que representar gráficos y renderizar polígonos, además gracias a la nueva tecnología CUDA ofrecida por Nvida cálculos intensivos que antes tomaban meses o años, se desarrollan en mucho menos tiempo.

- Al momento de decidir cuál arquitectura elegir al desarrollar un nuevo proyecto, como primera medida hay que tener en cuenta en que maquinas va a hacer ejecutado. Ya que si al principio se tiene pensado correr el programa en un solo computador de laboratorio, y el equipo podría tener una GPU de NVIDIA y ningún coprocesador Xeon Phi, en este caso tendría sentido elegir la arquitectura basada en GPU, pero debe tenerse en cuenta que cualquier otra máquina que no tenga una GPU NVIDIA no podrá ejecutar el código. O por otro lado podría estar desarrollando un software para diferentes ordenadores, y muy seguramente muchos de estos no tendrán una GPU NVIDIA, pero tendrán al menos un procesador de cuatro núcleos, la mejor opción sería irse por la tecnología de Intel.

- Una ventaja para aquellos programadores CUDA es que los coprocesadores Phi proporcionan una plataforma de hardware con capacidad de teraflops que se debe ejecutar aplicaciones CUDA con un alto rendimiento.

IX. REFERENCIAS

[1]"Parallel Programming and Computing Platform | CUDA | NVIDIA | NVIDIA", *Nvidia.com*, 2017. [Online]. Available: http://www.nvidia.com/object/cuda_home_new.html. [Accessed: 01- Feb- 2017].

[2]"Procesamiento paralelo CUDA | Qué es CUDA | NVIDIA", *Nvidia.es*, 2017. [Online]. Available: http://www.nvidia.es/object/cuda-parallel-computing-es.html. [Accessed: 01- Feb- 2017].

[3]"Nvidia's David Kirk on CUDA, CPUs and GPUs", *bit-tech*, 2017. [Online]. Available: http://www.bit-tech.net/hardware/cpus/2008/04/30/david-kirk-interview/1. [Accessed: 01- Feb- 2017].

[4]"CUDA", *Es.wikipedia.org*, 2017. [Online]. Available: https://es.wikipedia.org/wiki/CUDA. [Accessed: 01- Feb- 2017].

[5]"About CUDA", *NVIDIA Developer*, 2017. [Online]. Available: https://developer.nvidia.com/about-cuda. [Accessed: 01- Feb- 2017].

[6]"¿Por qué elegir una GPU Tesla para alta computación? | NVIDIA", *Nvidia.es*, 2017. [Online]. Available: http://www.nvidia.es/object/why-choose-tesla-es.html. [Accessed: 01- Feb- 2017].

[7]"Alta computación para servidores | GPUs Tesla | NVIDIA", *Nvidia.es*, 2017. [Online]. Available: http://www.nvidia.es/object/tesla-server-gpus-es.html. [Accessed: 01- Feb- 2017].

[8]"Arqui-G1-DVas-Aldo-Ric-DSan - Paralelismo", *Arqui-g1.wikispaces.com*, 2017. [Online]. Available: http://arqui-g1.wikispaces.com/Paralelismo. [Accessed: 01- Feb- 2017].

[9]"Arquitectura NVIDIA Pascal: un salto cuántico en gaming", *NVIDIA*, 2017. [Online]. Available: http://www.nvidia.es/graphics-cards/geforce/pascal/. [Accessed: 01- Feb- 2017].

[10]"Computación paralela", *Es.wikipedia.org*, 2017. [Online]. Available: https://es.wikipedia.org/wiki/Computación_paralela. [Accessed: 01- Feb- 2017].

[11]"Familia de productos Intel® Xeon Phi™", *Intel*, 2017. [Online]. Available: http://www.intel.la/content/www/xl/es/processors/xeon/xeon-phi-detail.html. [Accessed: 01- Feb- 2017].

[12]"GPU", *Sabia.tic.udc.es*, 2017. [Online]. Available: http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/Hardware/tarjetas%20graficas/gpu.html. [Accessed: 01- Feb- 2017].

[13]"Unidad central de procesamiento", *Es.wikipedia.org*, 2017. [Online]. Available: https://es.wikipedia.org/wiki/Unidad_central_de_procesamiento. [Accessed: 01- Feb- 2017].

[14]"Developer Access Program (DAP) for Intel® Xeon Xeon Phi™ Processor Codenamed Knights Landing", *Dap.xeonphi.com*, 2017. [Online]. Available: http://dap.xeonphi.com/. [Accessed: 01- Feb- 2017].

[15]"Intel Parallel Studio", *En.wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Intel_Parallel_Studio. [Accessed: 01- Feb- 2017].

[16]"Procesador Intel® Xeon Phi™", *Intel*, 2017. [Online]. Available: http://www.intel.la/content/www/xl/es/processors/xeon/xeon-phi-processor-overview.html. [Accessed: 01- Feb- 2017].

[17]M. Hernández Hernández, "Análisis y evaluación de arquitecturas heterogéneas basadas en Intel Xeon Phi para problemas científicos basados en patrones de cómputo Stencil", Doctorado, Universidad de Murcia, 2016. [Online]. Available: http://www.tesisenred.net/handle/10803/396230?locale-attribute=es [Accessed: 01- Feb- 2017].

[18]C. Balaguera Peña and D. Vega Camelo, *Arquitecturas Híbridas o Heterogéneas*, 1st ed. Bucaramanga, 2016, pp. 1-9. [Online]. Available: http://wiki.sc3.uis.edu.co/images/5/59/ArtG2.pdf. [Accessed: 01- Feb- 2017].