ITANIUM 25

25.1- INTRODUCCIÓN

IA-64 (Itanium) es la familia de procesadores con la que Intel se introduce al mercado de los procesadores de 64 bits. El primer producto de esta familia de procesadores es, precisamente, el Itanium que se introduce en el mercado en mayor del 2001.

Se ha diseñado para ser utilizado en servidores y workstations de alto rendimiento. Intel afirma que su diseño va más allá de conceptos como CISC o RISC mediante la incorporación de una cantidad masiva de recursos de procesamiento con compiladores inteligentes que permiten la producción de código objeto que hace explícito el paralelismo.

Una de las distinciones más importantes entre el Itanium y los microprocesadores de 32 bits es el uso de técnicas mejoradas de procesamiento, de las cuales estos últimos están pobremente equipados. La principal técnica se refiere al procesamiento de más de una instrucción al tiempo en una misma unidad. El término que usa Intel para esto es EPIC ("Explicitly Parallel Instruction Computing" o Cálculo de Instrucciones Estrictamente en Paralelo). Cómo de bien funcione esta técnica hará que ésta sea independiente de la calidad de los compiladores desarrolladas para ella, así como la optimización para el procesamiento paralelo implementada en el software.

Las diferencias entre 32 y 64 bits son muy grandes. Olvidando por el momento el proceso en paralelo y otras mejoras inteligentes, propias de la arquitectura del Itanium, la ventaja más directa es la cantidad de memoria que se puede direccionar. Hace catorce años, direccionar una memoria de 4 GB con las plataformas de 32 bits era más que suficiente. Hoy en día, las grandes bases de datos sobrepasan con creces este tamaño. El tiempo empleado en cargar de nuevo los datos en la memoria virtual, junto con el acceso a los dispositivos de almacenamiento, tiene un efecto negativo en el rendimiento. Las plataformas de 64 bits son capaces de direccionar 16 terabytes de memoria (cuatro mil millones de veces más que en una plataforma de 32 bits).

Otra ventaja de la CPU de 64 bits sobre la de 32 bits es que procesa el doble de instrucciones por ciclo. Si se trabaja con registros de 16 bits en paralelo para encriptado, por ejemplo, la CPU de 64 bits procesa cuatro registros por cada ciclo de reloj, mientras que la de 16 bits lo hará de dos en dos. Una instrucción de 64 bits puede procesarse en un único ciclo de reloj, mientras que en una plataforma de 32 bits necesita dos ciclos de reloj, más uno de limpieza. Por tanto, un sistema de 64 bits puede direccionar de forma directa mucha más memoria. Cada una de las plataformas de 64 bits y las arquitecturas futuras explotan estas ventajas, además de las mejoras específicas en la arquitectura de cada marca.

El problema surge debido a que Intel se encuentra con jugadores con mayor experiencia y más sólidos en el terreno de los 64 bits, como UltraSPARC de Sun, PowerPC de IBM y Alpha de Compaq, que actualmente se encuentran en el corazón de gran cantidad de servidores y estaciones de trabajo de alto rendimiento.

Ya terminó su fase final de producción y se comercializa en pequeñas cantidades. Cuenta con Sistemas Operativos como Windows Advanced Server, Windows XP 64-bit Edition, Linux Trillian, HP-UX y otros. Pero para realizar el cambio con mayor comodidad, el Itanium ejecuta código IA-32 en hardware.

También se tiene una batalla en el software usado en las plataformas de 64 bits. Microsoft, al igual que Intel, domina en el terreno de los sobremesa, pero en el terreno corporativo tenemos algo totalmente diferente. Su versión de Windows de 64 bits tiene que luchar contra una innumerable cantidad de versiones de Unix que también se renovarán. Un factor decisivo es el momento en el que las empresas tengan que elegir cuál de las plataformas de 64 bits van a adoptar.

El resto de los competidores en el campo de los 64 bits aseguran tener una superioridad técnica frente a Itanium, además de tener una mayor estabilidad y disponibilidad de software que Windows 2000. El lanzamiento de Itanium con la experiencia de Intel en el "mercado de consumo" y la versión de 64 bits de Windows 2000 con el dominio de software del que dispone Microsoft han hecho tambalearse los cimientos del mundo de 64 bits tal como lo conocemos, y tienen que haber algunas bajas. Sólo el tiempo puede decir quiénes sobreviven y de qué forma, pero sin duda, ahora que Intel y Microsoft van a involucrarse, el mundo de los 64 bits no es el mismo.

En cuanto al mercado al que se dirige, en las tareas informáticas, un aumento en el rendimiento siempre es bienvenido, pero en un primer momento, los chips de 64 bits dominarán únicamente en los entornos de alto rendimiento. Las aplicaciones de Internet de hoy día, ven cómo millones de usuarios en todo el mundo acceden, a través de ellas, a gigabytes o incluso terabytes de información en tiempo real. Debido a esta gran capacidad manejar de datos, los sistemas de 64 bits son bienvenidos por Internet, ISPs y ASPs, además de cualquiera que tenga servidores Web multimedia y OLTP ("On-Line Transaction Proceses"). La consolidación de las bases de datos de propiedad en un único "almacén" de datos, es también una de los principales atractivos de las aplicaciones de 64 bits.

Esta tecnología proporciona un gran empuje a cualquiera que desee manejar grandes cantidades de datos, como visualizaciones científicas, simulaciones y efectos especiales de renderizado.

Si finalmente llega a los PC de sobremesa algún procesador descendiente del Itanium ya no se podrá medir el rendimiento de un chip por los Mhz ya que su velocidad dependerá más del IPC (Instrucciones Por Ciclo) que de otros aspectos.

En la Figura 25.1 puede observarse la evolución de la arquitectura de ordenadores. En ella se percibe que al final de la evolución de las arquitecturas se sitúa el Itanium, el cuál desarrolla la tecnología EPIC a 0.13 μm. Se aprecia en la gráfica que a medida que se ha aumentado el número de transistores por milímetro cuadrado se han mejorado las características arquitectónicas.

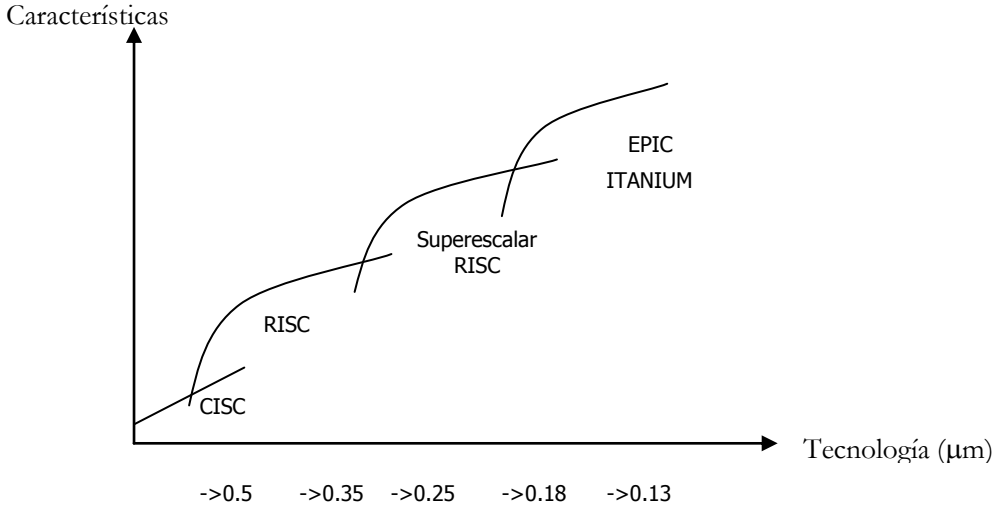

Figura 25.1. Evolución de la Arquitectura de los computadores

Tal como avanza la gráfica, Intel espera ir vendiendo más procesadores de 64 bits que con el tiempo coparán el mercado de los procesadores. Por esta razón Intel ya se ha puesto manos a la obra y ha realizado una segunda versión de este procesador, Itanium2.

25.2– CARACTERÍSTICAS

25.2.1- Características Generales

ITANIUM	CARACTERÍSTICAS GENERALES
AÑO	2001
TECNOLOGÍA	0,13 μm
MILLONES DE TRANSISTORES	25 (CPU) 300 (CACHE)
PROCESADOR	64 bits
FRECUENCIA	800 MHz
CACHE L1	32 KB
CACHE L2	96 KB
CACHE L3	2-4 MB
ALIMENTACIÓN	3.3V
RENDIMIENTO	370 SPECint_base2000
FRECUENCIA BUS SISTEMA	266MHz
ANCHO DE BANDA DE E/S	PCI-66MHz
TRANSFERENCIA	2,1GB/s

Figura 25.2 . Características generales.

25.2.2- Motivación para una arquitectura de 64-bit

La necesidad de procesadores de 64 bits es clara en aplicaciones que necesitan direccionar grandes cantidades de memoria ya sea esta virtual o física. Como ejemplos de dichas aplicaciones podemos citar:

- Bases de Datos: actualmente existen muchas bases de datos con más de 4Gb usados en registros y otra información, esta cantidad constituye el direccionamiento directo máximo posible con 32 bits. Aunque es posible y de hecho a menudo puesto en práctica, el direccionamiento de bases de datos muy grandes con procesadores de 32 bits recurriendo a distintas técnicas, es más fácil y más eficiente hacerlo si el procesador soporta operaciones y direcciones de 64 bits.
- Creación de contenido Digital: el DVD ("Digital Versatile Disk") y el HDTV ("High Definition TV") cada vez se vuelvan más comunes y el contenido de estos, por los grandes volúmenes de información que requieren, es el principal candidato para la edición en PCs avanzadas. TIVO (dispositivos semi inteligentes de grabación digital) y otros dispositivos similares podrían eventualmente volver obsoleto a las videograbadoras convencionales. En general toda la creación de contenido digital será beneficiada por el direccionamiento y procesamiento de 64 bits.
- CAD & Simulaciones: estos a menudo tratan con modelos enormes(edificios, aviones, explosiones nucleares, etc.) y como es de imaginar dichos modelos pueden fácilmente exceder los 4Gb de tamaño. Como ya se explicó antes cuando el rango direccionable natural del procesador es superado, se deben usar varios trucos para direccionar mas datos, esto implica a menudo una sobrecarga significativa al procesador.

- Criptografía: esta involucra grandes cantidades de cómputos con grandes enteros, y se beneficiaría inmensamente gracias a la disponibilidad de registros de 64 bit lo que le permitiría realizar cálculos mucho más rápidamente.
- Otros: las aplicaciones computacionalmente intensas también se verán favorecidas, por ejemplo:
    - Automatización de diseño electrónico
    - Servidores de alto rendimiento
    - Efectos gráficos como el Transform and Lighting

Todas estas aplicaciones se beneficiarán tanto de las direcciones de 64 bit como de un número mayor de registros.

25.2.3- Características de la arquitectura

- EPIC - Explicitly Parallel Instruction-Set Computing.
- Mejora el ILP ("Instruction Level Parallelism") maximizando la complementación de soluciones hardware-software. Provee mecanismos tales como "branch-hints", "cache-hints" (pistas de salto, de caché) para el compilador, para que este indique al procesador de tales eventos y permita que maneje el hardware más eficientemente. Estos mecanismos de aviso minimizan el costo computacional de los saltos y reducen fallos de consulta a la memoria caché. Además de eso, el conjunto de instrucciones en sí cuenta con "opcodes" que explícitamente permiten su ejecución en paralelo ("codeblock-hint").
- Otros conceptos que se manejan con EPIC son:
- Especulación: mejora el rendimiento permitiendo que el compilador adelante las instrucciones de carga debidas a saltos reduciendo el retardo debido a la memoria ("memory latency").
- Predicación: elimina muchos saltos y las penalizacioness debidas a las fallos en la predicción de los saltos ("branch prediction").
- Pila de registros: reduce el coste introducido por las secuencias de llamada y retorno mediante un modelo eficiente de registros enteros administrados por el RSE ("Register Stack Engine").
- Rotación de registros: renombra registros en hardware de manera automática para mejorar el rendimiento de los bucles, sin el coste debido a los métodos tradicionales ("loop unrolling").
- Instrucciones SIMD: mejoran la ejecución de aplicaciones multimedia al operar en varios datos enteros o de coma flotante en una sola instrucción.
- Cantidad masiva de registros: 128 registros de enteros, 128 registros de coma flotante, 8 registros de salto, 64 de instrucciones.
- Escalabilidad para 32 y más CPUs en paralelo.
- Uso optimizado de memoria:
    - tamaños de página de hasta 256 MB
    - tres niveles de caché
- Compatibilidad con IA-32 en hardware.

- Detección avanzada de errores (MCA – "Machine Check Architecture"), y ECC ("Error Correcting Code") en caché y bus de sistema.

Los principales objetivos que persigue Itanium son:

- Superar las limitaciones de las arquitecturas actuales.
- Aumentar la eficacia de las operaciones en coma flotante.
- Dar soporte a direccionamiento de 64 bits para memoria.
- Mantener la compatibilidad con la arquitectura IA-32.

25.3- ARQUITECTURA

Para cumplir todos estos objetivos, la arquitectura Itanium se basa en el uso de mejoras sobre el código, en tiempo de compilación, para así simplificar el proceso de ejecución de las instrucciones. Entre estas técnicas se encuentran:

- Incrementar el ILP (paralelismo a nivel de instrucciones)
- Mejorar el tratamiento de los saltos
- Evitar los tiempos de latencia en las operaciones de acceso a memoria
- Soportar la modularidad en el código

25.3.1- Paralelismo a nivel de instrucciones (ILP)

La arquitectura Itanium soporta ILP mediante:

- Soporte para que el programador especifique directamente el paralelismo .
- Una estructura denominada paquete ("bundle"), que agrupa tres instrucciones y que facilita el procesado en paralelo de las mismas .
- Un gran número de registros, para facilitar que diferentes variables hagan uso de diferentes registros.

La predicación ("predication") mejora el paralelismo.

Los saltos son un problema importante de las arquitecturas RISC que ejecutan código fuera de orden. Estas arquitecturas tradicionales usan una técnica llamada "branch prediction" (predicción de saltos) para predecir el camino correcto.

Fallos en la predicción son de ocurrencia común, alrededor del 5-10% de las veces, resultando en grandes penalidades en la ejecución, de hasta 30-40%. IA-64 usa una técnica llamada Predicación para ejecutar ambos caminos en paralelo y evitar las limitaciones de las arquitecturas tradicionales.

Se utilizan 64 registros de 1 bit para poner en práctica esta técnica, descartando aquellas ramas en la ejecución que no son parte del camino correcto. IA-64 también provee soporte eficiente para saltos múltiples y comparaciones paralelas, ambas características permiten realizar el salto múltiple en 1 ciclo de máquina, acelerando la ejecución significativamente.

En todos los casos, el registro de predicación indica si una rama en particular y sus datos asociados están activos o no. Las ramas activas continúan su ejecución y si se desactivan su ejecución cesa.

Los predicados, que son parte de toda condicional de la IA-64, proveen un mecanismo muy eficiente para terminar la ejecución de ramas que no deben seguirse, para establecer bucles y terminarlos, para administrar la predicción dinámica y realizar comparaciones de n caminos.

Los predicados ayudan a manejar la compleja tarea del flujo de control, compleja debido a un agresivo paralelismo a nivel de instrucciones (ILP – "Instruction Level Paralelism") que se realiza en tiempo de compilación.

De esta manera, la predicación reduce los fallos de predicción en los saltos, incrementando el rendimiento. Es particularmente útil en aplicaciones donde es difícil predecir los saltos como en grandes bases de datos, "data mining", "warehousing", etc.

Como no cuentan con predicación, las arquitecturas RISC tradicionales han tenido poco éxito con ILP.

25.3.2- Evitar tiempos de latencia en accesos a memoria

La especulación ("speculation") minimiza el efecto del retardo de la memoria.

En el contexto actual, las CPUs van incrementando su velocidad a una tasa mucho mayor que la memoria. Se espera que esta tendencia continúe en el futuro, y posiblemente se acentúe. Entonces estamos ante un gran cuello de botella que aqueja a las arquitecturas de hoy.

Para reducir el impacto de esta limitación, las arquitecturas tradicionales permiten que el compilador y el procesador realicen la carga de datos tiempo antes de que estos sean necesarios, pero los saltos hacen el papel de barreras, de límites para el entorno en el que se aplica esta técnica.

IA-64 emplea una técnica llamada especulación para iniciar la carga desde memoria de forma anticipada, inclusive anticipando al uso en porciones de código más allá de un salto.

La especulación es una característica muy utilizada en IA-64. Las arquitecturas RISC tradicionales pueden utilizar una técnica llamada 'carga a prueba de fallos' ("non-faulting load") para evitar el costoso manejo de errores cuando la carga anticipada podría no ser válida.

IA-64 evita este problema ofreciendo una solución en la arquitectura para manejar la carga anticipada. La idea consiste en que cada dato va asociado a un bit que dice si un error se ha producido o no cuando se cargó dicho dato. Este bit de error (llamado bit "Nat") acompaña al dato a través de todas las operaciones aritméticas, moves y condicionales hasta que se realiza sobre el dato una instrucción de check, que detecta si hubo un error, y dispara la rutina manejadora del error.

Sin embargo, un 99% de las veces una carga anticipada que produce un error resulta pertenecer a una rama que nunca se debió ejecutar, por lo que simplemente se abandona su ejecución, sin saltar a la rutina manejadora de errores.

Esto significa que con IA-64, el compilador puede realizar carga anticipada en forma agresiva ("speculation") sin prestar atención a penalizaciones en caso de errores.

La especulación de datos permite al compilador realizar la carga aún antes de un store que podría escribir sobre el mismo registro para el cual se realiza la carga anticipada.

IA-64 mantiene un registro de todas las cargas anticipadas en una tabla llamada ALAT ("Advanced Load Address Table"). Cuando se encuentra una instrucción store que causa conflictos con una carga anticipada contenida en la tabla, la entrada correspondiente se elimina, y cuando la carga se chequea en el ALAT, la ausencia de su entrada indica que el valor debe ser recuperado nuevamente y se obtiene el resultado correcto.

- *Especulación de datos*: resuelve el problema de que el contenido de la posición de memoria referenciada por la instrucción de carga sea modificado entre el momento de lanzar la instrucción de carga adelantada y el momento de la verificación. En el momento de la verificación se realiza una comprobación sobre el contenido de la dirección de memoria referenciada por la instrucción de carga. Si el contenido ha cambiado, se realiza de nuevo la consulta a memoria, si no, se devuelve el resultado. Este tipo de load adelantado no retrasa el lanzamiento de las excepciones, sino que se atienden en el momento de producirse, por ejemplo, por fallo de página.
- *Especulación de control*: esta técnica intenta resolver el problema del orden correcto de ejecución de las instrucciones, en concreto, el problema del lanzamiento de excepciones y el problema de los saltos. Ambos problemas se resuelven del mismo modo. Mediante el mecanismo de load especulativo. Cuando se lanza un load especulativo las excepciones que se puedan producir de su ejecución quedan pospuestas hasta el momento de su verificación.
- *Especulación Combinada* : en algunos casos es necesario combinar las dos técnicas, por ejemplo cuando se quiere adelantar la ejecución de una instrucción de carga que está dentro de una subrutina fuera de ésta. En estos casos, se debe comprobar que los datos no varían, pero posponiendo el tratamiento de las excepciones hasta el punto donde se realiza la verificación. En caso de que los datos hayan variado, o se haya producido una excepción, se lanzará una excepción en el momento de la verificación.

Una barrera de código es una instrucción por encima de la cual no se debería adelantar la ejecución de una instrucción de carga. Hay varias instrucciones que actúan de barrera de código:

- Instrucciones de almacenamiento (store): puede hacer referencia a la misma dirección de memoria que la instrucción de carga, que debería ejecutar después de dicha instrucción de almacenamiento.
- Instrucciones de salto (br): no se sabe si realmente la instrucción de carga se debe ejecutar .

La ejecución especulativa permite, evitar las barreras de código.

25.3.3- Desdoblamiento y rotación de bucles (“Loop unrolling and rotation”)

El manejo eficiente de bucles es uno de los factores clave para determinar el rendimiento de todas las arquitecturas, y IA-64 provee características especiales para ello.

Una técnica común utilizada es el desdoblamiento. Consiste en evitar los saltos al principio del bucle duplicando el código del bucle dos, tres o más veces para reducir el número de saltos. Esta técnica está disponible en IA-64 y en las arquitecturas RISC.

El problema de esta técnica es que conlleva un aumento del código. IA-64 sin embargo provee características especiales para explotar esta técnica. Cuenta con la rotación de registros que permite la ejecución de bucles sin el aumento de código.

Optimización en la Ejecución de Bucles

Para ello hace uso de:

- Instrucciones especiales de salto.
- Dos registros de propósito específico denominados LC ("Loop Count") y EC ("Epilogue Count").
- Mecanismo de rotación de registros: consiste en que el registro lógico X se convierte en la siguiente iteración en el X+1. Esto se puede aplicar a los registros de predicado, a los de enteros y a los de coma flotante.

En IA-64, los registros y los predicados pueden rotar bajo el control del compilador. Esto significa que con cada paso en la rotación, los valores que estaban en, por ejemplo, el registro 40, avanzarían al registro 41. El valor en el último registro se mueve al primer registro, o sea, los valores de los registros rotan. Lo mismo sucede con los registros que contienen predicados.

Con esta rotación, el compilador puede correr una copia del código dentro del bucle y los valores se asignan a distintos registros automáticamente a medida que el código itera. Cuando el bucle finaliza, con la ayuda de los registros "Loop count" y "Epilog count" se vuelve al estado anterior a la rotación y el bucle termina.

De esta manera se administran los bucles sin la necesidad de prólogos o epílogos al principio y al final de cada bucle para manejar la inicialización y la restauración, y sin usar predicción de saltos que podría llevar a penalizaciones en el rendimiento en caso de fallar.

Para realizar estos saltos, se utiliza la instrucción de salto(br) con el complemento cloop. Los "counted loops" hacen uso del registro de propósito específico LC ("Loop Count"). Este registro se chequea en cada salto, si es distinto de cero, se decrementa su contenido y se efectúa el salto. Cuando es igual a cero no se realiza el salto. La ventaja de esta técnica es que, mirando el contenido del registro LC, se puede saber de antemano si el salto se va a tomar o no.

25.3.4- Unidades funcionales

Las instrucciones de IA-64 están divididas en cuatro categorías:

- I: Operaciones con enteros.
- F: Operaciones con números en punto flotante.
- M: Operaciones de acceso a memoria.
- B: Saltos.

El Itanium maneja grupos de tres instrucciones, lo que Intel llama "bundle" (ternas de operaciones), que pueden ser despachadas al mismo tiempo por diferentes unidades funcionales.. Cada palabra de instrucción o "bundle" consiste de tres operaciones elementales de algunas de las categorías mencionadas.

Ejemplo de bundles:

- Memoria-entero-entero, memoria-salto-salto (MII/MBB)
- Memoria-punto flotante-entero, memoria-punto flotante-entero (MFI/MFI)

Como consecuencia del manejo de bundles a través de la ventana de ejecución el Itanium tiene un throughput máximo de 6 instrucciones por ciclo de reloj.

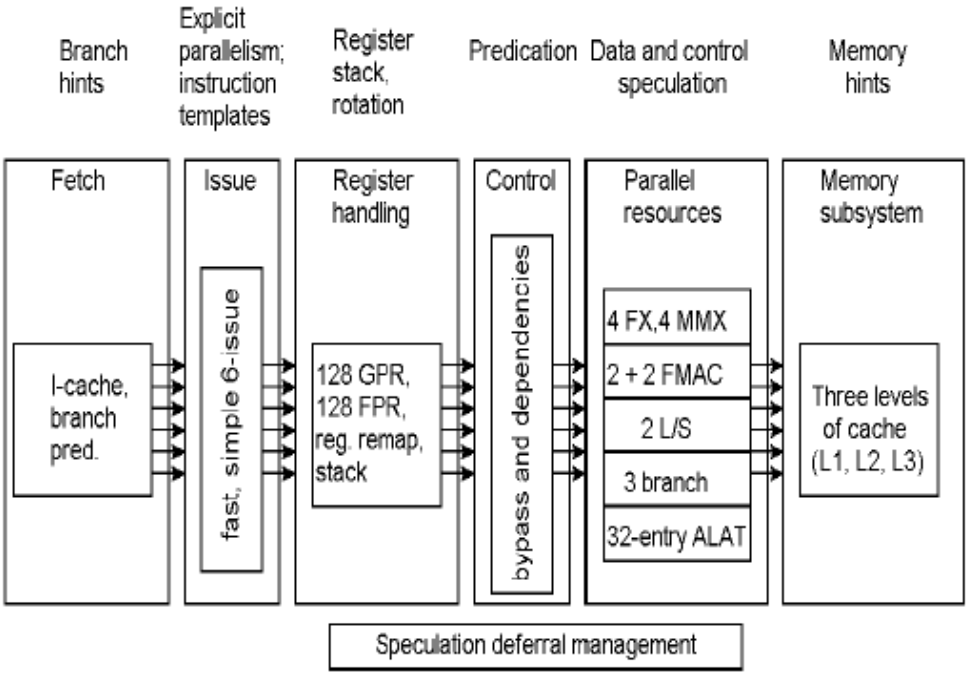

Figura 25.3 Flujo de instrucciones (Dispersión de bundles)

La principal característica que ofrece el Itanium es la de realizar hasta seis instrucciones a la vez, está particularidad esta representada a lo largo de toda la figura de su arquitectura (dispersión de bundles).

En el primer paso se realiza la búsqueda de instrucciones en la caché L1 de instrucciones y se aplican las técnicas de predicción de saltos, para ello se utilizan las tablas:

- BPT ("Branch Prediction Table") y MBPT ("Multiway Branch Prediction Table").

  Estas se usan para predecir dinámicamente si serán o no tomados los saltos, cada tabla se encarga de diferentes tipos de "bundles". Por ejemplo la BPT se ocupa de los saltos contenidos en ternas del tipo MMB mientras que la MBPT se encarga de los MBB o BBB (ternas con más de un salto en general).

- TAR ("Target Address Register"): tabla de cuatro entradas manipulada directamente por el compilador para especificar la dirección de saltos predichos estáticamente.

- TAC ("Target Address Cache") tabla de 64 entradas responsable de proporcionar la dirección de salto para los predichos dinámicamente.

- Una pila de direcciones de retorno.

La zona registros consta de 128 registros para números enteros (de 64 bits cada uno), 128 registros de números en coma flotante (de 82 bits cada uno), 8 registros para saltos, y varios registros más con diversas funciones, además hay otra serie de registros para la compatibilidad x86.

Así mismo en esta zona se encuentra la "Stack Engine", sistema de pila mediante registros que permite reducir el número de accesos a memoria en las llamadas y retornos de subrutinas (optimización en las llamadas a procedimientos). Desde el punto de vista de la pila el banco de registros de propósito general queda dividido de la siguiente manera:

- Registros r0 - r31: registros de propósito general.
- Registros r32 - r127: registros de pila.

Después las instrucciones se dirigen a una zona donde hay una serie de recursos cuya principal misión es ejecutar las instrucciones.

El Itanium tiene 17 unidades de ejecución paralelas (superescalar), todas ellas segmentadas ("fully pipelinet") con 10 etapas. Por tanto, cada unidad funcional puede aceptar una nueva instrucción por ciclo de reloj o, en su caso, un parón. Estas unidades son:

- 4 unidades para ejecutar operaciones con números enteros y accesos a memoria .
- 1 unidad de punto flotante que contiene 2 unidades FMAC ("Floating-point Multiply Acumulate" u operaciones acumuladas en punto flotante) que operan con operandos de 82 bits.
- 4 unidades para las instrucciones multimedia (MMX).
- 2 unidades punto flotante de precisión simple
- 2 unidades de carga /almacenamiento
- La Tabla de carga de direcciones adelantadas ALAT("Advanced Load Address Table"), que proporciona apoyo para especulación y minimización de latencia (burbujas de espera).
- 3 unidades de saltos

25.3.5- Microarquitectura

- Búsqueda /Emisión y ejecución desacopladas
- Prebúsqueda
- L1 Instrucciones, 16 KB
- Jerarquía de predictores de saltos
- Estática
- Dinámica en dos niveles (2,2)
- Multivía
- Corrección en la predicción asociada lazos
- Número de iteraciones conocido en tiempo de compilación
- Renombre de registros
- 9 puertas de emisión a unidades de ejecución
- 2 Memoria
- 2 enteros
- 2 punto flotante
- 3 saltos
- 6 instrucciones por ciclo
- Instrucciones independientes (template)
- Disponibilidad de recursos (template)
- 17 unidades de ejecución

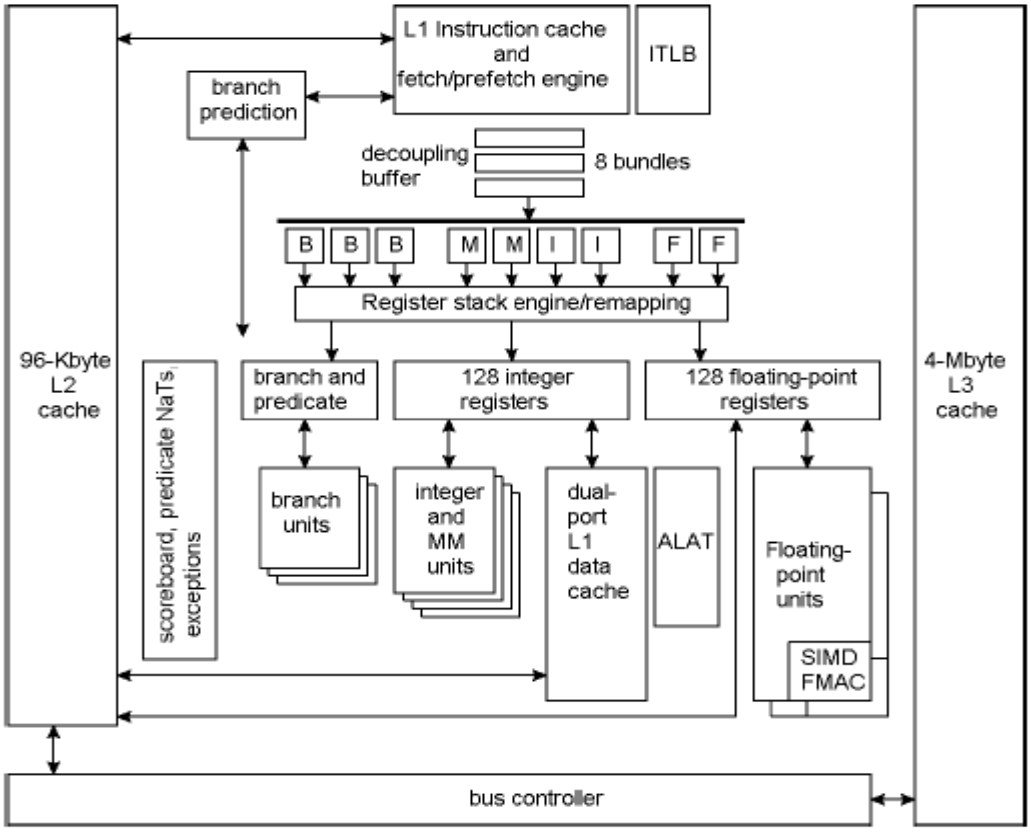

Figura 25.4 Microarquitectura del Itanium.

25.3.6- Segmentación

Las 10 etapas de las que dispone Itanium están dispuestas según la figura 25.3, éstas se han reducido de las 20 que contiene el Pentium 4 para evitar conflictos por dependencia de datos.

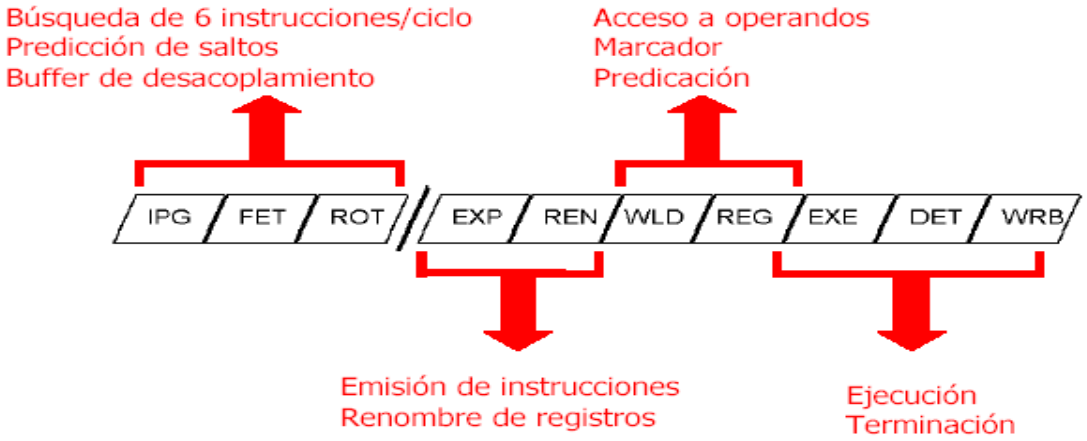

Figura 25.5. Segmentación de la unidad de enteros.

- IPG: generación de instrucciones.
- FET: búsqueda de instrucciones.
- ROT (rotación): rotación de registros para generar la burbujas necesarias para evitar las dependencias de datos .

- EXP (expansión): tiene como misión repartir las seis instrucciones a los diferentes puertos, en esta etapa además existe la posibilidad de parar la instrucción a través de los bits de parada.
- REN (renombramiento) :acomoda los registros en los lugares adecuados de la pila, para optimizar su tamaño.
- WLD: decodificación.
- REG: lectura de operandos.
- EXE: ejecución.
- DET: detección de excepciones.
- WRB: post-escritura.

25.4- APORTACIONES Y NUEVOS RECURSOS ARQUITECTONICOS

Las arquitecturas tradicionales, aparte de tener muchas restricciones para llevar a cabo ILP, también cuentan con pocos recursos en hardware para permitir la ejecución en paralelo. En contraste, los procesadores basados en IA-64 cuentan con 128 registros enteros de propósito general, 128 registros de punto flotante, 64 registros de predicación y muchas unidades de ejecución para asegurar el buen funcionamiento con altas cargas de trabajo. La arquitectura se diseñó para ser escalable, permitiendo la fácil adición de unidades de ejecución.

25.4.1- Procesador de 64 bits

Hay dos implicaciones en el término de procesador de 64 bits. Uno es la habilidad de acceder a 64 bits (8 bytes) de información a la vez, esta es una función de la estructura del bus. La otra es la habilidad de usar 64 bits para definir una dirección de memoria. Un procesador de 32 bits puede acceder a 2^{32} bits de datos, aproximadamente 4,3 billones de bits. Comparando esto con 2^{64} bits, aproximadamente 18.4 quintillones de bits (alrededor de 2,1 billones de gigabytes), a los que puede acceder un procesador de 64 bits

25.4.2- EPIC (Explicitly Parallel Instruction Computing)

EPIC como una nueva forma de concebir el código. La implementación de la filosofía de diseño EPIC permite alcanzar nuevos niveles de paralelismo y rompe el paradigma de ejecución secuencial que existe en las arquitecturas tradicionales. El uso de mecanismos como la predicación y la especulación combinadas con el paralelismo explícito permiten que IA-64 logre ganancias de rendimiento donde otras arquitecturas no han podido hacerlo (retardo de memoria, fallos en la predicción de ramas).

Es una nueva arquitectura que dota al Itanium de 17 unidades de ejecución. Utilizando este paralelismo de forma eficaz, estos microprocesadores son capaces de ejecutar hasta 20 instrucciones por ciclo de reloj.

EPIC tiene la singularidad de poder indicar al microprocesador, qué partes integrantes del programa pueden ser ejecutas con total paralelismo, algo que las herramientas de generaciones anteriores no podían.

Intel y HP acentúan las bondades de la predicación y la especulación más allá de simples características que mejoran la ejecución, y las sitúan al nivel de adelantos revolucionarios que permiten diseñar todo un nuevo paradigma para lograr verdaderamente la ejecución paralela más allá de su simple aplicación a casos especiales.

25.4.3- Un gran número de registros

El Itanium ofrecen un enorme número de estos registros. Para un buen uso de ellos, es necesario que el compilador aproveche las nuevas características de procesado en paralelo.

La arquitectura Itanium pone a disposición del programador gran cantidad de registros:

- 128 registros de enteros (64 bits)
- 128 registro de coma flotante (82 bits)
- 64 registros de predicado (1 bit), para guardar predicciones (formalmente hablando las evaluaciones de predicados) .
- 8 registros para direccionamiento en saltos (64 bits)
- 128 registros de propósito específico para aplicaciones (64 bits)
- 1 registro de puntero a la instrucción en curso (IP): no puede ser accedido ni modificado directamente, apunta a la dirección del paquete de la instrucción en curso y cambia cuando entra a ejecutar la siguiente instrucción.

Adicionalmente, el Itanium contiene varios registros dedicados a la monitorización de las prestaciones de la CPU. Información tal como el número de instrucciones ejecutadas, que pueden ser seguidas por software, haciendo posible al software de administración de sistemas la escritura en tiempo real, teniendo un impacto mínimo en las prestaciones del sistema y registros de identificación de la versión de IA-64 implementada que identifican al procesador (CPUID).

Validación de registros

La ejecución especulativa, para el reordenamiento de instrucciones, obliga a comprobar que el contenido de un registro es un dato válido en el momento de la consulta. Para ello, todos los registros de enteros y de coma flotante llevan asociado un bit NaT ("Not a Thing"), para indicar si el contenido del registro es válido en ese momento o no. En el caso de los registro de coma flotante, se llama NaTVal.

25.4.4- Organización de Memoria

Se define un solo espacio de direcciones lineal y uniforme, con un tamaño de 264 bytes. Esto quiere decir que, tanto datos como código, comparten el mismo espacio de memoria. Este espacio de memoria está sin segmentar, y no hay definidas zonas de uso específico.

El Itanium tiene 3 niveles de caché, L1 y L2 van integradas, es decir, residen en la cpu y trabajan a la misma velocidad que el núcleo. La memoria caché L3 a pesar de estar fuera del encapsulado del procesador también trabaja a su misma velocidad.

Adicionalmente el Itanium posee buffers de traducción de direcciones lineales a direcciones reales TLB's, basándose en el principio de localidad. El bus del sistema presenta un ancho de banda de 2.1 GB/s. La jerarquía de memoria está organizada en los siguientes niveles:

- Cache de datos de nivel 1 (L1D), de 16 Kbytes.
- Cache de instrucciones de nivel 1 (L1I), de 16 Kbytes.
- Cache de nivel 2 (L2), de 96 Kbytes.
- Cache de nivel 3 (L3), el tamaño puede variar de 2 a 4 Mbytes.
- TLB de datos de nivel 1 (L1-DTLB), con 32 entradas.
- TLB de datos de nivel 2 (L2-DTLB), con 96 entradas.
- TLB de instrucciones (ITLB), con 64 entradas.

25.4.5- Compatibilidad con las instrucciones de 32 bits

Uno de los requisitos de Itanium es conservar la compatibilidad con el juego de instrucciones de la arquitectura IA-32 (procesadores actuales Pentium). Itanium puede ejecutar aplicaciones de la arquitectura IA-32, así como aplicaciones que tengan mezcladas instrucciones de la arquitectura IA-32 e instrucciones de la arquitectura Itanium. Esto significa que los programas escritos para las máquinas actuales deberían funcionar sin modificaciones. Objetivamente parece que dichas aplicaciones funcionarán a una velocidad más lenta de lo normal debido a la compatibilidad.

25.4.6- Optimización operaciones de coma flotante y multimedia

Un gran porcentaje de la cpu del Itanium, aproximadamente el 10%, está dedicado a la unidad de coma flotante (FPU). La arquitectura implementa las siguientes optimizaciones para las operaciones en coma flotante:

- Banco de registros de coma flotante de 128 registros de 82 bits.
- Instrucciones especiales como multiplica y acumula (fma).
- Instrucciones especiales de load y store para números en coma flotante.
- Se permite la transferencia de datos entre el banco de registros de enteros y el de coma flotante.
- Fácil conversión entre enteros y números en coma flotante.
- Soporte para técnicas como la especulación o la rotación de registros.

25.4.7- Optimización en la Ejecución de Saltos

Hoy en día, la optimización de los saltos es uno de los grandes problemas con los que se enfrentan los diseñadores de hardware. La técnica más usada actualmente es la predicción de salto. Si la predicción es correcta, no se producen parones en el pipeline. Sin embargo, si es errónea, el coste es muy alto.

La arquitectura define dos tipos de saltos:

- Saltos relativos al registro IP
- Saltos indirectos, que hacen uso de los registros de salto.

Se permite la ejecución en paralelo de varios saltos a la vez. La forma de decidir qué salto se toma primero es mediante los registros de predicado. El salto cuyo registro de predicado se haga cierto antes, es el que primero salta.

Itanium optimiza el manejo de los saltos mediante dos Técnicas:

- Instrucciones con predicados

Se puede condicionar la ejecución de operaciones al contenido de los registros de predicado. Esta técnica nos permite convertir muchas de las dependencias de control (saltos) del código en dependencias de datos. Para ello, se lanzan a la vez las dos ramas del salto, condicionando la ejecución de las instrucciones de cada rama al contenido de un registro de predicado que se ha evaluado previamente. Así, la rama cuyo registro de predicado esté a uno, completará su ejecución, mientras que las instrucciones de la otra rama serán tratadas como instrucciones NOP.

- Instrucciones de salto especiales

Un ejemplo son los "counted loops", utilizados para realizar bucles de tipo for (cuya condición de parada es que un contador alcance un determinado valor). Este tipo de saltos no se puede eliminar con la ejecución con predicados.

25.5.- MODELO DE PROGRAMACIÓN

Como ya se dijo antes, las prestaciones del Itanium dependerán de la correcta relación entre el software y el hardware. Los compiladores, a través de instrucciones específicas para ello, tendrán control total, si así lo desean, sobre la predicción de saltos, el orden de las operaciones, la predicación y la especulación. En particular el compilador puede decidir, si un salto debe ser predicho estáticamente o dinámicamente con base en su historial. También puede decidir si un salto debe ser predicho tomado o no tomado, en caso de ser estática la predicción. El Itanium posee el hardware necesario para poder ofrecer estas facilidades al compilador.

El Itanium, tiene un nuevo conjunto de instrucciones, no es ya una máquina CISC, para ejecutar el código que podrían ejecutar sus ancestros tendrá que traducir las instrucciones mediante hardware específico para tal propósito. Esto hará, en principio, la ejecución de programas viejos más lenta de lo que será su ejecución en computadores anteriores de la arquitectura de Intel. Tampoco es RISC, más bien es del tipo VLIW ("Very Large Instruction Word") o EPIC ("Explicitly Parallel Instruction Computing").

VLIW o EPIC, es el paradigma idóneo para incrementar el paralelismo a nivel de instrucciones (ILP). Hacer muchas instrucciones o trozos de ellas al mismo tiempo.

La IA-64 representa el nuevo modelo ISA ("Instruction Set Architecture" o Conjunto de Instrucciones de Arquitectura) basado en la tecnología EPIC, siendo totalmente compatible por hardware con las instrucciones de su predecesora, la IA-32.

Las instrucciones se organizan en paquetes o grupos de 3 instrucciones máximo("bundle") que no poseen dependencias entre ellas y que, por lo tanto, se pueden ejecutar en paralelo. En un momento determinado, el procesador intentará procesar en paralelo tantas instrucciones de un grupo de instrucciones como le sea posible, dependiendo de los recursos disponibles.

Los paquetes de instrucciones definidos para este concepto están formados por 128 bits, a diferencia de lo que ocurría en la arquitectura IA-32, cuya longitud de instrucción no era fija en un principio.

Las tres instrucciones EPIC del paquete consumen 123 bits (41 para cada una), dejando los cinco restantes (bits de template) para ser utilizados como almacén extra de información, de cara a asistir al procesador en la utilización más eficiente de recursos.

El mecanismo template, deja al compilador la decisión de qué unidad funcional será la que ejecute la instrucción cada instrucción del paquete. En los templates se puede indicar qué paquete termina el grupo. Un paquete puede indicar que un grupo se acaba en la primera, segunda o tercera instrucción de dicho paquete. Para terminar un grupo, se indica añadiendo un stop. Además, también permite especificar si todas las instrucciones pertenecen al mismo grupo o no.

Los distintos tipos de templates disponibles son los siguientes:

MII, MIIs, MIsI, MIsIs, MLX*, MLXs*, MMI, MMIs,

MsMI, MsMIs, MFI, MFIs, MMF, MMFs, MIB, MIBs,

MBB, MBBs, BBB, BBBs, MMB, MMBs, MFB, MFBs

Donde:

M : Instrucción de memoria (ld o st)
I : Instrucción de enteros
F : Instrucción de coma flotante
B : Instrucción de salto
L : Instrucción que trabaja con datos inmediatos largos
s : Indica que se realice un stop
(*) LX : Tipo especial, que se procesa en la unidad funcional de enteros

25.5.1.- Tipos de datos

Los tipos de datos que soporta Itanium son los siguientes:

- Enteros : de 1, 2, 4 y 8 bytes.
- Coma flotante : formatos simple, doble y doble-extendido.
- Punteros a memoria: 8 bytes .

El formato estándar para los enteros es 8 bytes. Los registros también poseen 8 bytes (64 bits) de longitud. Cuando se trabaja con operandos enteros de 1, 2 ó 4 bytes, se rellenan con ceros hasta alcanzar la longitud de 8 bytes.

25.5.2.- Formato de instrucciones

El formato de instrucción es el siguiente:

[(preg)]	cop	[.comp1]	[.comp2]	destino =	fuente	[,fuente]
6 bits	14 bits			7 bits	7bits	7bits

Donde:

- (preg) : registro de predicado.
- cop (codigo de operación) : identifica a la instrucción.
- .comp1, .comp2: algunas instrucciones pueden llevar complementos, que indican una variación sobre la instrucción de base.
- destino, fuente : casi todas las operaciones tienen al menos dos operandos fuente y un destino.

El modelo de ejecución es registro-registro.

Las únicas instrucciones de acceso a memoria son ld (carga) y st (almacenamiento). Ejemplos de formato de instrucción son:

Instrucción simple: add r1 = r2, r3

Instrucción con predicado: (p4)add r1 = r2, r3

Instrucción con dato inmediato: add r1 = r2, r3, 1

Instrucción con complemento: cmp.eq p3 = r2, r4

25.6- ANÁLISIS DEL RENDIMIENTO

El Itanium representa el principal proyecto de desarrollo de Intel. Este procesador representa un cambio significativo en un mercado basado en instrucciones RISC. Intel se arriesga con su nueva arquitectura EPIC, la cual no puede ser efectiva fácilmente dentro de la arquitectura RISC que predomina en estos momentos.

Las implicaciones del Itanium en el mundo del PC son muchas:

- La velocidad de un ordenador, actualmente medida en megahercios en el mundo del PC, puede tener menos importancia con la llegada del Itanium.

- Para el desarrollo efectivo del Itanium se necesita una cooperación sin precedentes con las empresas de compiladores para que desarrollen software específico de 64 bits.

El Itanium es capaz ejecutar billones de operaciones de coma flotante por segundo, esto es debido a su habilidad para realizar cálculos de 6 instrucciones por cada ciclo de reloj y eso le da al Itanium un potencial impresionante. Considerando que el software aprovecha éstas cualidades vamos a analizar su rendimiento.

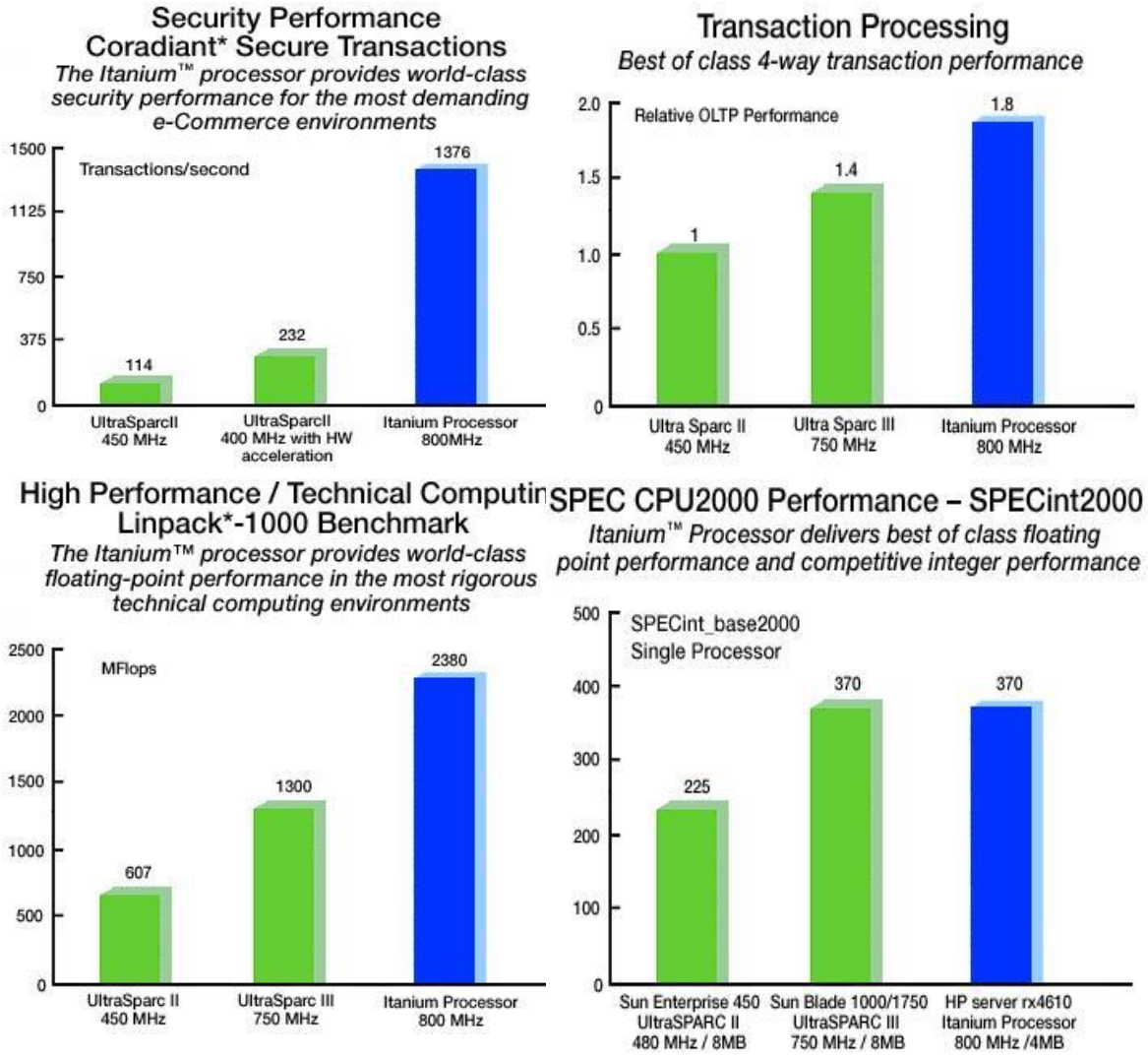

Figura 25.6. Benchmarks del Itanium