UNIVERSIDAD DE CANTABRIA
Facultad de Ciencias

Proyecto Fin de Carrera

Superando las limitaciones del Apilado Vertical mediante nuevas arquitecturas de red en chip

(Overcoming 3D-Stacking Technology Limitations via new on-chip network architectures)

Para acceder al Título de

INGENIERO EN INFORMÁTICA

Autor: Alfonso de la Vega Ruiz

Director: Valentín Puente Varona

Codirector: Pablo Abad Fidalgo

Septiembre – 2014

Resumen

Las mejoras introducidas con cada nueva generación de procesadores se fundamentan en la continua disminución del tamaño de los transistores, permitiendo incluir cada vez más lógica y componentes en el mismo chip.

No obstante, esta reducción se ha visto mermada en los últimos años a causa de las dificultades tecnológicas encontradas en la producción de componentes de tan reducido tamaño.

Como respuesta a esta situación, una posible solución se encuentra en el apilado vertical de capas en un mismo chip conocido como “3D Stacking”, que permitiría mantener la relación de incremento de capacidad experimentada hasta este momento.

En cambio, esta tecnología trae consigo nuevos inconvenientes, siendo uno de ellos la escasa capacidad de comunicación vertical disponible con los métodos de fabricación existentes actualmente.

En este proyecto se ha realizado un planteamiento de red en chip 3D que tiene en cuenta esta limitación, y que mediante la utilización de microarquitecturas de router versátiles permite obtener un funcionamiento adecuado a los requerimientos presentes en las comunicaciones en el interior del chip.

Las contribuciones de esta arquitectura de red son las siguientes:

- Supera en rendimiento a las redes 2D ante situaciones de tráfico equivalentes, alcanzando hasta un 40% de mejora al utilizar cargas sintéticas y un 10% en la ejecución de aplicaciones reales.
- Ofrece una utilización uniforme y combinada de todas las capas de la red, ya que la inclusión de soluciones tradicionales en sistemas 3D provocaba desequilibrios de congestión.
- Realiza lo anterior con unos requerimientos de conexiones verticales reducidos, adaptándose a las limitaciones existentes en la tecnología.

Palabras clave: Redes dentro del chip, apilado vertical, micro-arquitectura del router.

Summary

The performance growth achieved by each new generation of processors is led by a continuous reduction of transistors size, which allows to build sophisticated systems with more capabilities and components into the same chip proportions.

Nevertheless, this size reduction has been slowed down during the last years, mainly due to the physical difficulties encountered when working at such a small circuit resolution.

As an alternative source of performance improvement, the 3D-Stacking of multiple layers to form a single chip aims to maintain the capacity increase experienced through the last technological generations.

However, new issues emerge with the utilization of this technology. Apart from physical constraints, one of the most concerning problems is the reduced vertical interconnection density which is feasible within the current status of the manufacturing process.

In this work, a 3D network approach has been developed. The utilization of novel router microarchitectures mitigates the negligent impact which these interconnection constraints could inflict to the system, and provides the required performance for an on-chip network.

The contributions achieved by this new 3D network architecture are as follows:

- It offers a better response when compared with 2D networks under the same workloads, obtaining up to 40% improvement against synthetic loads, and up to 10% increased performance when simulating real applications execution.
- There is an even usage of each layer of the network, which overcomes the traffic imbalance encountered when applying traditional mechanisms to 3D systems.
- It achieves this goals having in mind the current limitations of the technology, by requiring only a few vertical connections to operate.

Keywords: On-chip networks, 3D-stacking, router microarchitecture.

Contenido

1 Introducción 7

1.1 Antecedentes 7

1.2 Objetivos 8

1.3 Estructura del documento 8

2 Estado del arte 9

2.1 Redes en chip 9

2.2 "3D stacking" o Apilado vertical 11

3 Entorno de evaluación 16

3.1 TOPAZ 16

3.2 GEMS 17

4 Diseño 20

4.1 Arquitectura de red 3D 20

4.2 El problema del reensamblado de paquetes 22

4.3 Background: LIGERO 23

4.4 Inclusión de consumidores finitos 26

5 Modificaciones aplicadas sobre el diseño 28

5.1 Reducción de dispersión entre fragmentos 29

5.2 Reducción de la presión sobre la red 31

5.3 Adaptación para mensajes verticales 36

6 Evaluación 38

6.1 Configuraciones de red 38

6.2 Tráfico sintético 39

6.3 Aplicaciones 44

7 Conclusiones 46

8 Trabajo futuro 47

9 Bibliografía 48

Índice de Figuras

Figura 2.1: (a) Topología de red en anillo. (b) Red malla 2D. (c) Red Toro 2D. 9

Figura 2.2: Red en anillo de la arquitectura Intel Xeon PHI (Fuente: [4]). 9

Figura 2.3: Fragmento de red con topología malla del sistema TILEPro de Tilera.(Fuente: [5]). 10

Figura 2.4: Crossbar 5x5 con puertos de entrada/salida para las direcciones cardinales (norte, sur, este y oeste) y una pareja de puertos para la inyección y consumición de paquetes por parte del nodo asociado. 11

Figura 2.5: (a) Unión face-to-face de dos capas. (b) Unión face-to-back de dos capas. (c) Combinación de orientaciones face-to-face y face-to-back para conectar cuatro capas. 12

Figura 2.6: (a) Técnica de apilado wafer-on-wafer. (b) Técnica de apilado die-on-die. 12

Figura 2.7: Utilización de la red de un sistema de dos capas, que apila la LLC sobre los cores (Fuente:[7]). 13

Figura 2.8: (a) Distribución del paquete con una anchura de flit de 128 bits. (b) Troceado de los flits que realiza MIRA para su envío sincronizado a través de las múltiples capas de la red (32 bits de ancho de enlace). (Fuente: [8]) 14

Figura 2.9: (a) Área que posee el crossbar de un router del sistema base. (b) Área total de crossbar utilizada por el router MIRA en un sistema de 4 capas. (Fuente: [8]) 14

Figura 3.1: Fragmento del diagrama de clases de TOPAZ. 16

Figura 3.2: Estructura del simulador GEMS 17

Figura 4.1: Ilustración que refleja los pasos realizados por un paquetes en su tránsito por la red: 1 (distribución de porciones entre capas), 2 (envío bidimensional) y 3 (reensamblado). 21

Figura 4.2: Izquierda: Dos porciones, una por capa. Opción (i): Paquetes el doble de anchos. Opción (ii): Paquetes el doble de largos. Opción (iii): Envío de dos paquetes independientes. 21

Figura 4.3: Representación simplificada del bloqueo de un consumidor a causa del particionado. 22

Figura 4.4: (a) Representación del router LIGERO a nivel de bloque básico. (b) Contenido del bloque básico superior. (Fuente: [22]) 23

Figura 4.5: (a) Un paquete entra a la red en el nodo A con destino B. (b) En el trayecto, alcanza una zona congestionada e intenta sin éxito salir de ella rotando en el anillo interno de LIGERO. (c) Tras varios intentos, el paquete es redirigido hacia el primer puerto disponible. (d) Finalmente, el paquete alcanza el destino por otra ruta no congestionada. 24

Figura 4.6: Camino de escape embebido en un toro 4x4 creado con routers LIGERO (fuente: [22]). 25

Figura 4.7: (a) La porción del paquete 6 no puede ser consumida a causa de la saturación. (b) Esta porción es redirigida al anillo, permitiendo el paso de las que sí pueden acceder 27

Figura 5.1: Tamaños iniciales de las colas de consumo para tres patrones de tráfico. 28

Figura 5.2: (a) La inyección no se lleva a cabo porque el router 1 tiene tráfico en la salida del anillo. (b) En este caso, es el router 0 el que tiene tráfico en su camino de bypass. (c) Sin paquetes en tránsito, la inyección se lleva a cabo en ambas capas. 30

Figura 5.3: Nuevo tamaño de colas, incluyendo inyección simultánea. 30

Figura 5.4: Dos caminos posibles para ir desde A hasta B en una red toro 4x4. 31

Figura 5.5: Mapa de inyección normalizado del tráfico permutación para una red 8x8. 32

Figura 5.6: El router izquierdo inyecta paquetes (en verde) a través del router vecino. Si no hay ningún control, los paquetes avanzan girando por el anillo y saturando los puertos de salida, e impiden al router inyectar sus propios paquetes con normalidad. 33

Figura 5.7: Estructura de preinyección en el router LIGERO. 33

Figura 5.8: Comparación de inyección al aplicar los nuevos controles. 34

Figura 5.9: (a) El paquete alcanza su destino B, pero no puede ser consumido. (b) Se produce un retorno del paquete hacia su nodo origen. (c) Al llegar a origen, el paquete bloquea la inyección de A temporalmente. (d) Finalmente, es mandado de vuelta a destino donde esta vez sí logra consumirse. 35

Figura 5.10: Estado de las colas de consumo en saturación al aplicar todas las modificaciones al sistema. 36

Figura 5.11: Nuevo camino para los mensajes con tránsito únicamente vertical. 37

Figura 6.1: Organización de los nodos activos y pasivos (cores (C) y memoria (M) respectivamente) en las redes 3D y 2D. 40

Figura 6.2: Tiempos de ejecución normalizados para las pruebas reactivas con las tres redes. 41

Figura 6.3: Resultados del tráfico modal para las redes con dimensión 4. 42

Figura 6.4: Resultados del tráfico modal para redes de dimensión 6. 42

Figura 6.5: Resultados del tráfico modal para redes de dimensión 8. 43

Figura 6.6. Comparación reactiva de redes 3DP y 3D. 44

Figura 6.7: Comparación de redes 3D y 2D de dimensión 4 con aplicaciones reales. 44

Figura 8.1: Sistema de doble capa que ofrece 12 núcleos operativos. 47

Índice de Tablas

Tabla 3.1: Características del CMP utilizado en las pruebas con GEMS. 18
Tabla 6.1: Parámetros de cada una de las redes. 38
Tabla 6.2: Redes utilizadas en las comparaciones de los tres tipos de red. 39
Tabla 6.3: Topologías utilizadas en la comparación de las redes 3D. 39
Tabla 6.4: Número de peticiones pendientes máximas por nodo. 40
Tabla 6.5: Límite de solicitudes pendientes por inyector. 43

1 Introducción

1.1 Antecedentes

El nacimiento de los procesadores con múltiples núcleos o multicores hizo que fuera necesario un sistema de comunicación que poseyera una latencia mínima y un gran ancho de banda. El incremento sostenido del número de cores y memoria que se introducían en un chip obligó a que, además, la escalabilidad de este sistema fuera otro requisito indispensable, ya que los tradicionales buses de comunicación, aunque simples, no eran capaces de ofrecer una relación rendimiento-coste razonable. Fue así como comenzaron a fabricarse redes de interconexión en el interior del chip [1].

Estas redes establecen enlaces dedicados entre los componentes, de forma que un aumento en la cantidad de componentes supone también un incremento del número de enlaces disponibles. Por lo tanto, el ancho de banda disponible escala conforme crece el sistema. Adicionalmente, la incorporación de estas redes permite utilizar de forma eficiente mecanismos de comunicación muy versátiles, como la conmutación de paquetes [1].

Por otra parte, la inclusión de redes en el chip presenta una serie de consideraciones, que deben tenerse en cuenta: (i) el área disponible es reducida, y se comparte con el resto de elementos (núcleos, bancos de memoria cache, controladores, entre otros); (ii) las altas prestaciones requeridas en el interior del chip pueden hacer que el consumo energético de una red sofisticada llegue a ser significativo.

La reducción sistemática de la longitud del transistor se ha visto guiada hasta el momento por la conocida ley de Moore [2]. Esta ley promulga que el número de transistores en el interior de un chip se duplica cada 18 meses aproximadamente. En cambio, actualmente existen numerosos problemas para conseguir mantener este ritmo en las sucesivas generaciones. Si bien es posible que se solventen estos inconvenientes, el mantenimiento de la ley de escalado alcanzará un coste prohibitivo para su aplicación en producción.

La creación de chips de mayor tamaño no es una solución válida, por varios motivos:

- El yield o porcentaje de retorno por oblea disminuiría, ya que la probabilidad de error por chip es proporcional al área que ocupa.
- La distancia entre los componentes aumentaría, perjudicando seriamente al rendimiento del sistema y al consumo energético.

En este contexto, el apilado vertical o "*3D stacking*" solucionaría estos dos inconvenientes. Esta técnica consiste en la unión de varias capas para formar un mismo chip, encontrándose estas capas unidas mediante conexiones verticales, denominadas "*Through Silicon Vias (TSV)"*. Con su utilización aumentaría el número de componentes que es posible integrar, e implicaría una disminución en la distancia media que los separa al encontrarse las capas muy cerca unas de otras, lo que se traduciría en una mejora del rendimiento, coste y perfil energético.

Las posibilidades que presenta esta tecnología son muy amplias, incluyendo heterogeneidad en un mismo chip (capas con diferente densidad de transistores o, directamente, que incorporen tecnologías distintas, como sistemas fotónicos) o un control dinámico del consumo energético (encendido/apagado de capas bajo demanda).

No obstante, también son numerosos los retos a los que se enfrenta, como la disipación del calor, la gran densidad de potencia, o el aumento de complejidad de las herramientas de diseño. Adicionalmente, como se describirá en el siguiente capítulo de este documento, las técnicas de fabricación actuales ofrecen una densidad de conexiones entre capas muy reducida, que no es equiparable a los enlaces existentes entre componentes de la misma capa.

Los sistemas de red tradicionales aplicados al 3D tratan a las conexiones verticales del mismo modo que a las horizontales, por lo que actualmente no es viable su aplicación práctica. Como añadido, estos sistemas dan pie a la aparición de un desequilibrio de carga en los planos de la red.

Existen soluciones más sofisticadas que palian estos desequilibrios en base a mecanismos avanzados y apropiados para una red tridimensional. En cambio, presentan unos requerimientos demasiados altos para su aplicación, principalmente a causa de problemas en los aspectos de fabricación de chips 3D existentes que se describirán en el capítulo 2.

1.2 Objetivos

La motivación para realizar este proyecto se enmarca en el contexto anterior. Se desea encontrar una arquitectura de red 3D capaz de cumplir los siguientes objetivos:

- Gestionar la utilización de los recursos de la red de forma uniforme, sea cual sea el tráfico de mensajes que circule por la misma.
- Conseguir lo anterior minimizando la colaboración entre planos lo máximo posible, de forma que el planteamiento se adecúe a las escasas capacidades de comunicación vertical que existen actualmente.
- Sobra añadir que, además, se busca que el rendimiento de este sistema sea adecuado para las características que se requieren para este tipo de redes.

1.3 Estructura del documento

Las secciones de este documento se organizan de este modo:

- En el **capítulo 2** se realiza una introducción a las redes en el interior del chip y a la técnica de apilado vertical, mostrando el estado del arte de estas tecnologías e incluyendo ejemplos existentes actualmente.
- El **capítulo 3** describe los diferentes entornos utilizados a la hora de evaluar el sistema creado.
- Los **capítulos 4 y 5** detallan las características de la arquitectura de red desarrollada.
- El **capítulo 6** recoge los resultados de las simulaciones realizadas sobre el sistema utilizando diferentes configuraciones de red y tráficos aplicados.
- El **capítulo 7** incluye las conclusiones obtenidas al término del proyecto.
- Como cierre, el **capítulo 8** menciona las posibles líneas de actuación como trabajo futuro al realizado en este proyecto.

2 Estado del arte

Como punto de partida para comprender mejor el interés y los objetivos del proyecto, en este capítulo se describen las características de las redes en el interior del chip (o redes en chip) y de la tecnología de apilado vertical.

2.1 Redes en chip

Estas redes interconectan a los componentes presentes en el chip, generalmente núcleos y bancos de memoria cache. A estos componentes se les denomina nodos, y se encuentran conectados a un router que les ofrece acceso a dicha red.

A continuación se describen las características principales que definen a una red en chip. Estas características pueden encontrarse explicadas con detalle en [3]:

- **Topología**: Determina qué enlaces existen entre los diferentes nodos de la red. Existen diferentes topologías que pueden encontrarse en redes en chip, siendo algunas de ellas las mostradas en la siguiente figura:

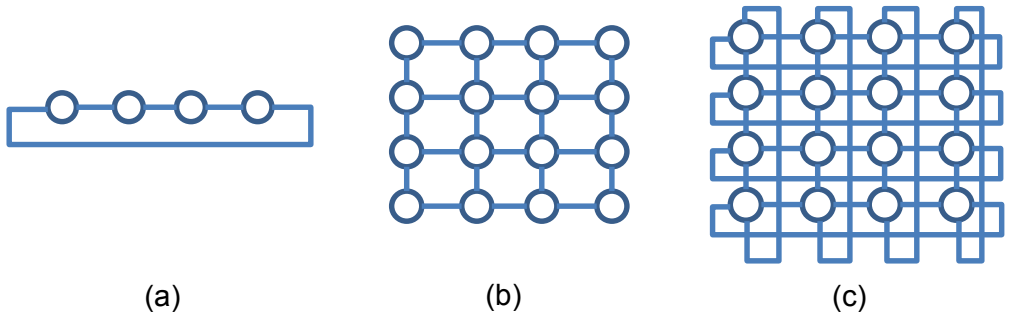

Figura 2.1: (a) Topología de red en anillo. (b) Red malla 2D. (c) Red Toro 2D.

Los sistemas reales han introducido estas topologías en sus productos de forma paulatina. Como ejemplo de uso, en la Figura 2.2 puede observarse una red de interconexión en anillo que se utiliza en la arquitectura Intel Xeon Phi [4], y en la Figura 2.3 un sistema de la compañía Tilera [5], que en este caso utiliza una red malla:

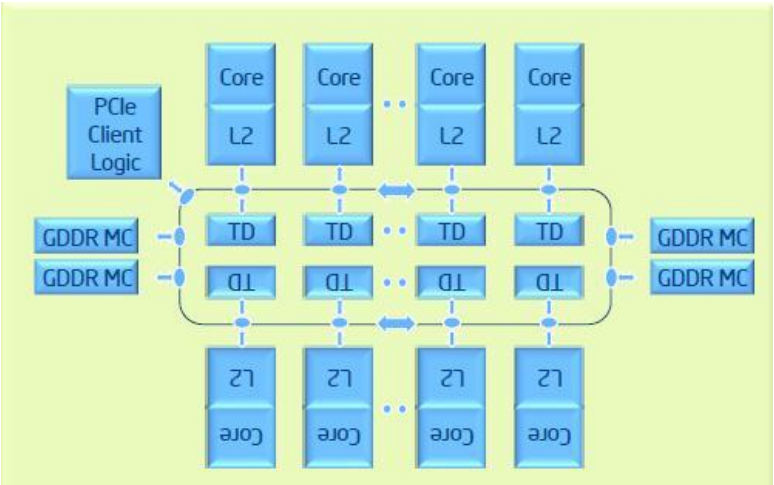

Figura 2.2: Red en anillo de la arquitectura Intel Xeon PHI (Fuente: [4]*).*

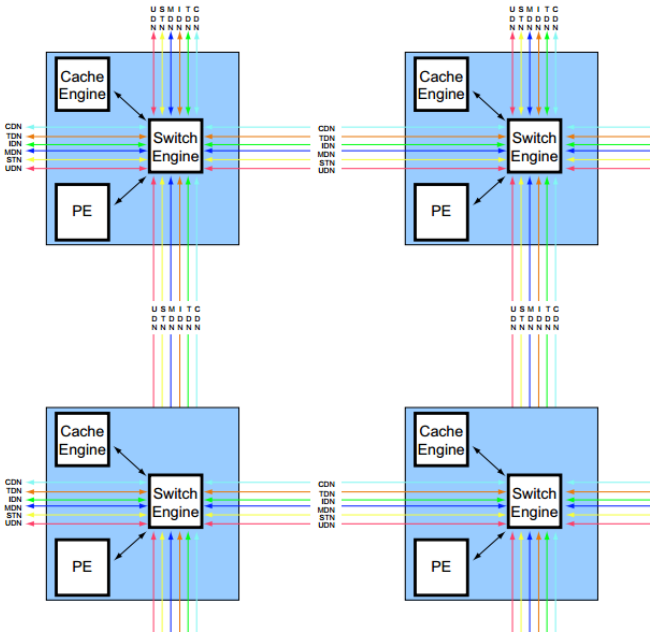

Figura 2.3: Fragmento de red con topología malla del sistema TILEPro de Tilera.(Fuente: [5]*).*

- **Enrutamiento**: Decide el camino que seguirán los mensajes entre un origen y un destino en el interior de la red. Por lo tanto, está estrechamente ligado a la topología seleccionada.
- **Control de Flujo**: Es el responsable de la administración de los recursos de la red, en términos de repartición de buffers y ancho de banda disponibles entre los paquetes que deben ser transmitidos. Esta gestión se realiza a bajo nivel, ya que los paquetes se fragmentan en unidades de información más pequeñas denominadas “flits”, sobre las que se aplican las decisiones de control de flujo. Esta subdivisión permite llevar a cabo una gestión más precisa de los recursos y evitar la necesidad de incluir buffers de mayor capacidad en los routers. La información que se transporta en cada flit depende de las características tecnológicas de la red (anchura del enlace, protocolo de comunicación, entre otras), existiendo flits de 16, 32, 64 o más bits de tamaño.
- **Micro-arquitectura del router**: Gestiona la entrada y salida de mensajes desde y hacia el nodo asociado, así como el tránsito de mensajes que atraviesan el router. Es un punto crítico de la red, ya que la mayor parte de la latencia asociada a un envío se produce en el router.

Cada mensaje atraviesa una serie de etapas en el interior del router desde su llegada por un puerto de entrada hasta su envío a destino por un puerto de salida. Uno de los primeros mecanismos que se aplicaron para crear los routers de una red en chip fue el uso de los denominados “crossbars”. Estos sistemas poseen una lógica centralizada para determinar el puerto de salida que utilizarán los flits que se encuentran en los puertos de entrada, estableciendo enlaces físicos entre ellos para que el tránsito se lleve a cabo.

En la siguiente imagen se muestra un crossbar para un router de una red directa bidimensional. Posee cuatro puertos de entrada y cuatro puertos de salida, que representan las direcciones cardinales que puede tomar un paquete en la red. Además,

al tratarse de una red directa es preciso incluir al menos la inyección y consumición como una pareja de puertos de entrada y salida extra:

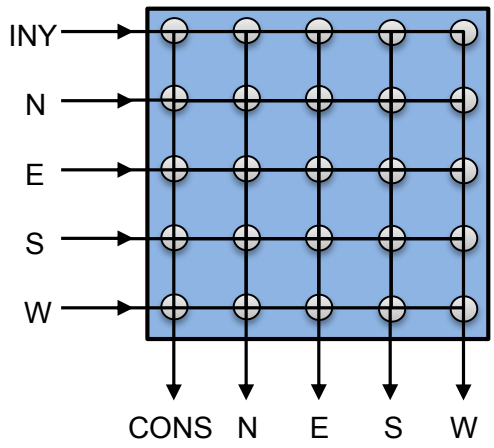

Figura 2.4: Crossbar 5x5 con puertos de entrada/salida para las direcciones cardinales (norte, sur, este y oeste) y una pareja de puertos para la inyección y consumición de paquetes por parte del nodo asociado.

Este sistema, aunque simple, posee ciertos inconvenientes, siendo uno de ellos el área de chip que ocupa (hay que tener en cuenta que, pese a que en la figura se haya representado cada enlace como un cable único por sencillez, en realidad posee un número de cables igual a la anchura del enlace, tal como se describió anteriormente). Por ello, con el tiempo se han desarrollado soluciones que utilizan crossbars más eficientes o que directamente los sustituyen por otros mecanismos de encaminamiento más eficientes. Se describirán algunas de estas microarquitecturas más adelante en este documento.

2.2 "3D stacking" o Apilado vertical

2.2.1 Introducción

El apilado vertical de capas en un mismo chip es un diseño emergente que se presenta como solución a dos problemas: (i) el reducido incremento de la densidad de transistores en un chip, y (ii) la elevada latencia de las comunicaciones a causa de la distancia entre los componentes en el plano y la velocidad de trabajo de los mismos. Mediante el apilado de capas se reduciría la distancia entre componentes relacionados, y sería posible incluir más lógica en un mismo chip.

El proceso de producción en masa de chips 3D en tecnologías nanométricas aún se encuentra en desarrollo. Existen dos aspectos principales a tener en cuenta en la fabricación: el primero de ellos se refiere a la orientación de las capas, existiendo la posibilidad de realizar uniones F2F (face-to-face, circuitería con circuitería) o F2B (face-to-back, la circuitería se conecta a través del sustrato). Las uniones F2B son más complejas, ya que requieren atravesar el semiconductor para formar el enlace vertical o TSV (through-silicon via). En cambio, son necesarias si se desean apilar más de dos capas. La siguiente imagen ejemplifica ambos tipos de uniones:

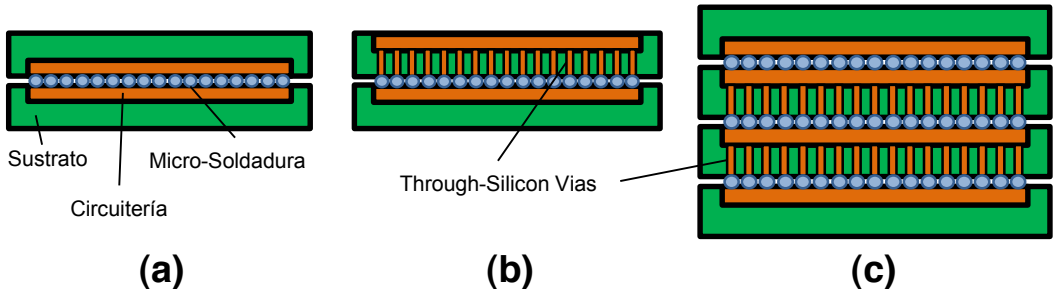

Figura 2.5: (a) Unión face-to-face de dos capas. (b) Unión face-to-back de dos capas. (c) Combinación de orientaciones face-to-face y face-to-back para conectar cuatro capas.

El segundo aspecto es el procedimiento de fabricación utilizado, siendo posible elegir entre los dos siguientes:

- **Wafer-on-wafer**: En este proceso una o varias obleas de chips o wafers son apiladas directamente una encima de otra, para su posterior corte en los chips finales.
- **Die-on-die**: En este caso, cada oblea es previamente cortada y comprobada, y posteriormente se realiza la unión entre aquellas que se encuentren en buen estado.

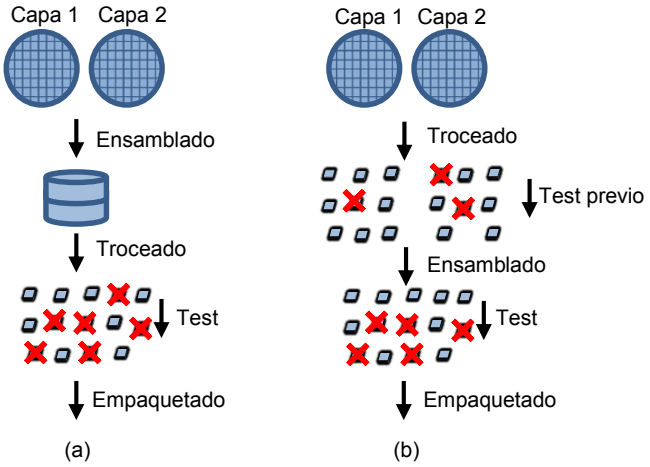

Figura 2.6: (a) Técnica de apilado wafer-on-wafer. (b) Técnica de apilado die-on-die.

De los dos procesos mencionados, en estos momentos solamente es rentable la realización del segundo a escala nanométrica. La unión wafer-on-wafer reduce el porcentaje de chips válidos por oblea de forma drástica, ya que los errores en una capa del chip afectan a las demás, sea cual sea su estado. En otras palabras, el porcentaje de retorno o yield de la oblea resultante es el producto de las obleas. Dado que el yield en tecnologías nanométricas está muy por debajo del 50%, es una aproximación inviable [6]. En die-on-die, al realizarse un test previo al ensamblado tal y como se muestra en la Figura 2.6, se evita perder capas que no poseen fallos de fabricación.

Por otra parte, el uso de la técnica die-on-die introduce otros inconvenientes, ya que las características mecánicas necesarias para realizar la unión chip a chip desembocan en un aumento del área necesaria para cada TSV, lo que se traduce en una reducción en la densidad de conexiones verticales.

2.2.2 Routers 3D

En cuanto al enrutado de paquetes en 3D, las redes y routers se dividen por una parte en aquellos que utilizan las técnicas conocidas de los sistemas bidimensionales, y por otra en los que tratan de aplicar un enfoque más avanzado. A continuación se describe un planteamiento de cada uno de estos grupos.

Una de las maneras más inmediatas de realizar esta conexión es dotando a cada nodo del sistema de un encaminador, que gestione los paquetes que lo atraviesan por medio de un crossbar ampliado. Por lo tanto, si se parte de un crossbar 5x5 como el mostrado anteriormente para topologías bidimensionales típicas como malla o toro, se pasaría a utilizar un crossbar 7x7, en el que se incluyan los tránsitos hacia arriba y hacia abajo en el eje Z de coordenadas. Se denominará a este sistema como la solución base.

La aplicación de esta solución base supone un aumento considerable del tamaño de cada router, al añadir dos parejas de puertos más a su crossbar con el consiguiente incremento en el área que ocupa. Adicionalmente, este sistema de encaminamiento da pie a que se generen irregularidades en la utilización de la red. En la red, los núcleos del procesador se comportan de forma más activa que los bancos de cache. Los condicionantes térmicos del apilado de procesadores hacen que sea más razonable colocarlos todos en la misma capa, utilizando los otros niveles para la LLC (“Last-Level Cache”, cache de último nivel). Estas dos premisas provocan que la aplicación de la solución base sobre esta disposición genere importantes diferencias en la utilización de los planos, existiendo una mayor congestión en el plano de los procesadores. Este hecho se observa en la siguiente figura:

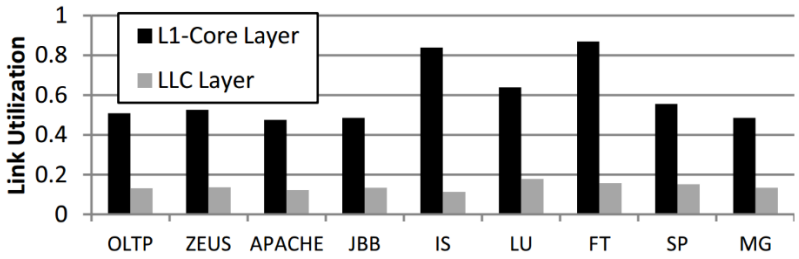

Figura 2.7: Utilización de la red de un sistema de dos capas, que apila la LLC sobre los cores (Fuente: [7]).

Un planteamiento más complejo y sofisticado se presenta con el router MIRA [8]. En este caso, en cada posición (x, y) existiría un único router, que se distribuye verticalmente entre todas las capas. De nuevo, se utiliza un crossbar para realizar la interconexión, existiendo uno por cada capa. En cambio, su tamaño se ve drásticamente reducido por los siguientes puntos:

- Cada uno de los flits originados en un nodo se trocea y distribuye por la vertical, y las porciones viajan en paralelo de forma sincronizada por cada una de las capas hasta llegar a destino, donde se procede al re-ensamblado. A causa de esto, la anchura necesaria para cada puerto del crossbar se reduce en función del número de capas (si la anchura del flit es W y se distribuye en 4 capas, los enlaces del crossbar solo necesitarían una anchura de W/4). Además, como los paquetes utilizan todas las capas al mismo tiempo, se consigue uniformidad en la utilización de la red. La siguiente figura compara la anchura del enlace necesaria en la solución base con el troceado que utiliza el router MIRA para una anchura de flit de 128 bits y una red con 4 capas:

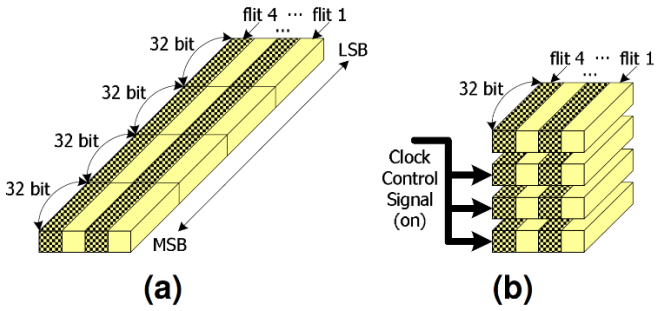

Figura 2.8: (a) Distribución del paquete con una anchura de flit de 128 bits. (b) Troceado de los flits que realiza MIRA para su envío sincronizado a través de las múltiples capas de la red (32 bits de ancho de enlace). (Fuente: [8]*)*

- A consecuencia del troceado previo al enrutado, no es necesario incluir dos nuevas parejas de puertos, con lo que las dimensiones de cada crossbar seguirían siendo de 5x5.

La comparación final del tamaño del crossbar utilizado en cada uno de los routers se aprecia de forma clara en la siguiente figura:

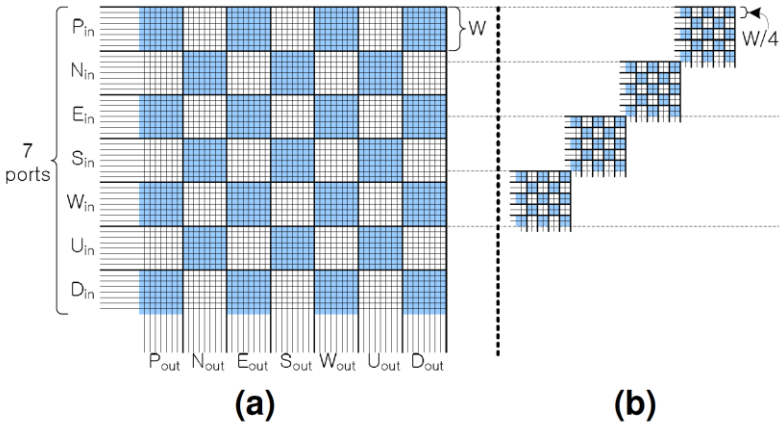

Figura 2.9: (a) Área que posee el crossbar de un router del sistema base. (b) Área total de crossbar utilizada por el router MIRA en un sistema de 4 capas. (Fuente: [8]*)*

En la imagen anterior, en (a) se observa el crossbar 7x7 que utiliza el router de un sistema base. En este caso, como en el sistema base existe un router por nodo, es necesario colocar uno de esos crossbars en cada nivel de la vertical. En cambio, en (b) se muestra el área total destinada a los crossbars en una red de cuatro capas que utilice el router MIRA. En cada una de las capas se situaría uno de los crossbars reducidos que aparecen en la imagen.

Pese al reducido uso de área y a solucionar los problemas de desbalanceo de red mencionados anteriormente, el establecimiento de un router multicapa que realiza envíos de flits troceados y sincronizados en el tiempo implica la necesidad de una gran densidad de conexiones verticales, algo que como se ha descrito no es razonable con las dificultades de fabricación existentes. Otro motivo, también determinante, es el siguiente: las señales que atraviesan los enlaces verticales poseen una baja latencia, pero no viajan de manera instantánea, lo que introduce bastantes problemas a la hora de querer utilizar una gestión completamente síncrona de la red.

2.2.3 3D-Stacking en la actualidad

Existen prototipos utilizados para analizar la viabilidad de las redes 3D, así como compañías que comienzan a utilizar el apilado de capas para mejorar el rendimiento de sus productos. A continuación se muestran algunos de ellos:

- El sistema **3DMaps** [9] se compone de dos capas apiladas mediante la técnica wafer-on-wafer, encontrándose los núcleos de procesamiento en la capa inferior y la memoria en la superior, ambas unidas cara con cara (F2F). La utilización de una tecnología antigua (130 nanómetros frente a los 22 nm actuales) hace que el yield por oblea sea muy elevado, por lo que es asumible la utilización de este procedimiento de fabricación. Este sistema se diseñó con el objetivo de demostrar el alto ancho de banda que una red 3D con una buena densidad de conexiones verticales era capaz de conseguir.
- En el prototipo **Centip3De** [10] se realiza de nuevo un apilado de dos capas (núcleos y memoria) con tecnología de 130 nm. En este caso, el objetivo del prototipo era medir las capacidades de funcionamiento del sistema utilizando tensiones umbrales (tensiones mínimas de funcionamiento de los componentes), para mejorar la eficiencia del sistema y poder de este modo combatir al aumento de densidad de potencia generado por el apilado de capas en un mismo chip.
- En un análisis hardware realizado sobre la consola **PsVita** [11], se comprobó que el procesador está formado por cinco capas: la capa inferior aglutina al procesador, y se une mediante una orientación F2F a la capa superior, que se trata de un primer módulo de memoria. Las siguientes tres capas se encuentran simplemente apiladas y no poseen conexiones verticales. Se conectan mediante enlaces convencionales.
- **Volta** [12], la siguiente generación de tarjetas gráficas de la compañía Nvidia, poseerá mejoras que aumentarán aún más el ya elevado ancho de banda de memoria que poseen estos sistemas. Esta mejora se ve apoyada por el apilado de varias capas de memoria DRAM sobre la GPU (Graphics Processing Unit, unidad de procesamiento gráfico), conformando un único chip que sería posteriormente ensamblado a la tarjeta periférica que sería conectada a los equipos de destino.
- La siguiente generación del coprocesador Xeon Phi de Intel incorporará la tecnología “Hybrid Memory Cube” de Micron [13]. Esta tecnología busca sustituir a las memorias DDR SDRAM convencionales, ofreciendo un mayor ancho de banda y un reducido consumo energético mediante el apilado vertical de hasta ocho capas de memoria.

3 Entorno de evaluación

En esta sección se describen las herramientas de simulación utilizadas durante el transcurso del proyecto.

3.1 TOPAZ

TOPAZ [14] es una herramienta de software libre desarrollada por el grupo de Arquitectura de Computadores de la Universidad de Cantabria. Está programado en C++, lo que le permite valerse de la orientación a objetos para conseguir un diseño modular y fácilmente configurable y ampliable. A continuación se describe brevemente su estructura.

En el simulador existen dos jerarquías de clases principales:

- **Jerarquía de componentes**: Representan los elementos de la red a nivel estructural, y las relaciones existentes en esta estructura. Son componentes las redes, los routers, los inyectores, consumidores, interfaces de conexión entre los puertos, etc.
- **Jerarquía de "*flows*" (flujos)**: Dentro del conjunto de componentes de la red, existen algunos cuyo trabajo es pasivo (como la propia red, o la representación de los puertos de un router) mientras que otros poseen un comportamiento activo, es decir, necesitan disponer de la lógica necesaria para tomar una serie de decisiones durante el transcurso de la simulación (denominados "runnable components". La definición de la lógica de estos componentes se realiza mediante los denominados flujos. Entre los elementos que poseen un flujo se localizan los inyectores, consumidores o el propio router, entre otros.

El siguiente diagrama es un pequeño fragmento de la herencia de clases de TOPAZ:

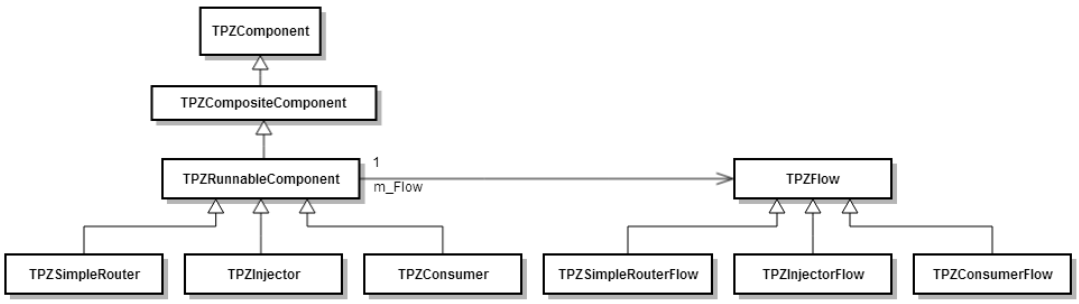

Figura 3.1: Fragmento del diagrama de clases de TOPAZ.

En la figura se observa como los componentes inyector, consumidor y router poseen un flujo asociado que almacena la lógica que aplicarán en la simulación. Introducir nuevos comportamientos para alguno de estos componentes se basa en definir un flujo derivado de los existentes.

Dentro de las características que ofrece el simulador, cuenta con diferentes tráficos sintéticos que pueden ser utilizados en las pruebas:

- "Random" o uniforme: El destino de los mensajes se selecciona de manera aleatoria.

- “Bit Reversal” o reverso de bits: El nodo con valor binario (a_{n-1}, a_{n-2},..., a_1, a_0) se comunica con el nodo (a_0, a_1,..., a_{n-2}, a_{n-1}).
- “Perfect Shuffle”, o barajado perfecto: Se coloca el bit más significativo del nodo origen como el menos significativo del nodo destino. El nodo con valor binario (a_{n-1}, a_{n-2},..., a_1, a_0) se comunica con el nodo (a_{n-2}, a_{n-3},..., a_0, a_{n-1}).
- Permutación o matriz traspuesta: La matriz de nodos se comunica con la matriz traspuesta de los mismos. Es decir, el nodo (1,0) se comunica con el (0,1), el (2,3) se comunica con el (3,2) y así sucesivamente.

En la descripción de tráficos anterior se ha supuesto como base una red 2D, aplicando las transformaciones sobre la representación lineal de los nodos. En el caso de los tráficos dirigidos (todos menos el uniforme), si se aplicaran las mismas transformaciones lineales sobre una red 3D los pares origen/destino variarían considerablemente, haciendo que las comparaciones entre simulaciones 2D y 3D no fueran del todo correctas. Por ello, se han modificado los tráficos dirigidos 3D para que el cálculo de las coordenadas (x, y) del nodo destino se haga sobre solamente un plano de la red, aplicando para ello las transformaciones definidas. El valor de la coordenada z se calcula de manera aleatoria en cada caso.

Cada una de las simulaciones se parametriza mediante unos ficheros especiales, escritos en formato SGM. Este formato es similar a XML, y en el mismo se recogen las características de la simulación, la red asociada y el router empleado en la misma. Estos ficheros hacen innecesario recompilar el programa cuando solamente se desee cambiar los parámetros de la simulación, ya que son accedidos en tiempo de ejecución.

3.2 GEMS

Esta herramienta [15] permite la simulación de un sistema completo y la ejecución de aplicaciones reales sobre el mismo. En el siguiente esquema se muestran los componentes que la conforman:

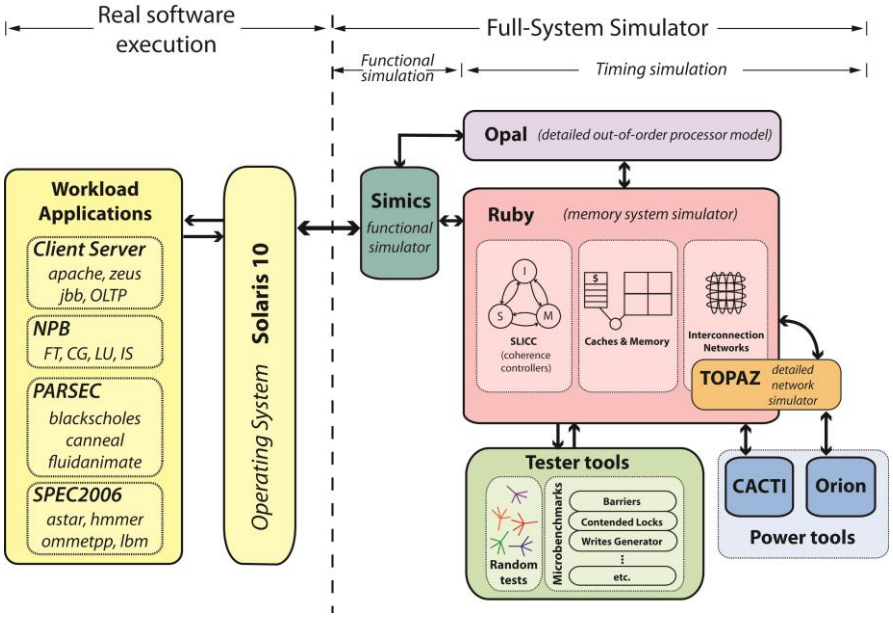

Figura 3.2: Estructura del simulador GEMS

En GEMS, ciertos componentes se encargan de la simulación funcional, que permite la ejecución de sistemas operativos y aplicaciones reales. El resto se encargan de generar la temporización detallada del sistema, para obtener unos resultados lo más cercanos posible a los obtenidos si se realizara la ejecución con la máquina real.
La modularidad que posee permite sustituir los componentes de acuerdo con las necesidades, así como realizar ejecuciones a diferente nivel de detalle. A continuación se describen los módulos principales que lo conforman:

- **SIMICS**: Es el simulador funcional del sistema completo. Permite la ejecución del sistema operativo y de los benchmark que se utilizan para evaluar el sistema.
- **OPAL**: Se encarga de modelar el procesador, determinando el avance de cada instrucción por las diferentes etapas del pipeline. Es capaz de simular una ejecución fuera de orden y es altamente configurable, permitiendo conseguir comportamientos similares a los encontrados en las últimas generaciones de procesadores reales.
- **RUBY**: Es un simulador temporal de la jerarquía de memoria para sistemas multiprocesador. Modela los diferentes niveles de cache, la memoria principal y los controladores de cada una de ellas, así como el protocolo de coherencia. Adicionalmente, ofrece un modelado simple de los sistemas de interconexión de la jerarquía de memoria. La estructura modular de GEMS permite sustituir esta red simple por el simulador TOPAZ para realizar pruebas contra la red 3D implementada.

Para las simulaciones realizadas, el sistema operativo sobre el que corren las aplicaciones es el Solaris 10, y la configuración de procesador y memoria utilizada es representativa de los procesadores del actual estado del arte, al estar basada en la serie “Haswell” de Intel. Estos son los parámetros del CMP (Chip Multi-Processor) más relevantes, tanto a nivel del procesador como de la jerarquía de memoria:

Arquitectura	Frecuencia	3 GHz
	Ventana de Instrucciones / Vías de ejecución	128 / superescalar de 4 vías
Caches Privadas	(L1) Tamaño / Asociatividad / Tamaño de bloque / Tiempo de acceso	2x32 KB (Instrucciones y datos) / 4 vías / 64 B / 1 ciclo
	(L2) Tamaño / Asociatividad / Tamaño de bloque / Tiempo de acceso	256 KB / 8 vías / 64 B / 4 ciclos
Cache L3 Compartida	Tamaño por banco de cache / Asociatividad / Tamaño de bloque / Tiempo de acceso	1 MB / 16 vías / 64 B / 6 ciclos
Memoria	Capacidad / Tiempo de acceso / Controladores de memoria / Throughput	4 GB / 240 ciclos / 4 / 32 GBs

Tabla 3.1: Características del CMP utilizado en las pruebas con GEMS.

3.2.1 Aplicaciones

Se han utilizado aplicaciones pertenecientes a tres conjuntos de pruebas:

- Transaccionales: Estas aplicaciones provienen de la suite *"Winsconsin Commercial Workload"* [16]. Permiten simular tests representando grandes sistemas transaccionales en computadores más modestos. Las aplicaciones utilizadas son las siguientes:
    - Apache: En esta prueba, se realizan transacciones sobre el conocido servidor web presente en multitud de sistemas reales.
    - OLTP: Evalúa el rendimiento de operaciones transaccionales online. Está basado en el benchmark *"TPC-C".*
    - JBB: Servidor de aplicaciones basado en el benchmark SpecJBB.
    - ZEUS: Servidor web estático, basado en SpecWeb.
- *"Nasa Parallel Benchmarks" (*NPB) [17]: Esta suite desarrollada por la NASA está formada por aplicaciones paralelas basadas en la computación del comportamiento de fluidos dinámicos. Se han usado las siguientes:
    - IS (*Integer Sort*): Ordenación de números enteros mediante el algoritmo "bucket-sort".
    - BT (*Block Tri-diagonal*): Resuelve ecuaciones de "*Navier-Stokes*" de dimensión 3.
    - CG (*Conjugate Gradient*): Resuelve un sistema lineal disperso mediante el método del gradiente conjugado.
    - FT (*Fourier Transform*): Resuelve un sistema de derivadas parciales de dimensión 3 aplicando la transformada de Fourier rápida.
    - MG (*MultiGrid*): Calcula la solución de una ecuación de Poisson de dimensión 3 mediante el método *"MultiGrid"*.
    - LU (*Lower-Uper Gauss-Seidel*): Algoritmo que diagonaliza sistemas de derivadas parciales mediante el método de la sobrerrelajación.
    - SP (*Scalar Penta-Diagonal*): Algoritmo que resuelve sistemas de ecuaciones dispersas pentadiagonales.
    - UA (*Unstructured Adaptative Mesh*): Resuelve un problema de transmisión de calor en un dominio cúbico representando mediante una malla desestructurada.
- SPEC: La *"Standard Performance Evaluation Corporation"* [18] se encarga de generar benchmarks para evaluar computadores de alto rendimiento. En concreto, se han utilizado aplicaciones pertenecientes al conjunto *"SPEC CPU 2006"*:
    - Astar: Algoritmo A* de búsqueda de caminos en mapas 2D.
    - Hmmer: Algoritmo de búsqueda de secuencias de proteínas.
    - Lbm: Simulación de fluidos 3D.
    - Omnetpp: Simulación de eventos discretos en una red Ethernet.

4 Diseño

En esta sección se describe la arquitectura de red 3D creada. Como recordatorio de la introducción, estos son los objetivos principales que busca conseguir:

- Ofrecer uniformidad de tráfico en las diferentes capas del sistema.
- Minimizar la necesidad de conexiones verticales para adaptarse a la situación actual de las técnicas de fabricación.

La estructura de la sección es la siguiente: en primer lugar, se dará una visión conceptual de la red 3D propuesta, para posteriormente describir aspectos más específicos relacionados con los inconvenientes encontrados durante el desarrollo de dicha arquitectura, y las soluciones aplicadas sobre los mismos.

4.1 Arquitectura de red 3D

Al igual que en el router MIRA descrito anteriormente en este documento, se realizará una división de los flits, viajando cada una de las porciones por una de las capas de la red. No obstante, en este caso no se utilizará un único router multicapa por cada una de las posiciones (x, y) de la red, sino que cada nodo dispondrá de un router completamente independiente (i.e. en principio no deberán estar sincronizados).

Respecto al lugar en que se situarán los enlaces verticales, con el objetivo de minimizar el número requerido de TSV los únicos componentes que se conectarán entre capas serán el inyector y el consumidor del nodo asociado al router. Para describir el sistema de forma clara, a continuación se detalla la secuencia que atraviesa cada uno de los paquetes en su viaje desde el nodo origen hasta el nodo destino. Un diagrama que muestra esta secuencia puede encontrarse en la Figura 4.1:

1. El inyector del nodo origen se encargará de dividir en porciones el paquete generado, y de distribuirlas entre los routers de la vertical que ese nodo ocupa. Cada una de esas porciones tiene que alcanzar el router situado en su nivel en la vertical del nodo destino, que se denominará en adelante como el destino temporal.
2. Una vez las porciones han sido distribuidas, tratarán de acceder a la red a través del router que les haya sido asignado. Estos routers son bidimensionales a todos los efectos, desconociendo en principio la existencia de los planos que se encuentran por encima o por debajo del suyo. El paquete se transmitirá hasta el destino temporal a través del plano utilizando la lógica de los routers de la red.
3. Cuando cada porción llegue a su destino temporal, será consumida como si se tratara de un paquete para el router de esa capa. Es en el consumidor donde se comprobará el verdadero destino de esa porción y donde, mediante las conexiones que existen entre los consumidores de la misma vertical, será entregará para su ensamblado con el resto de las porciones.

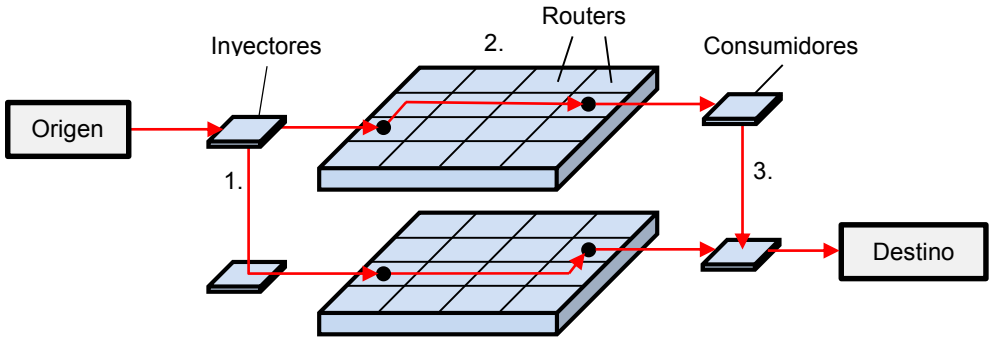

Figura 4.1: Ilustración que refleja los pasos realizados por un paquetes en su tránsito por la red: 1 (distribución de porciones entre capas), 2 (envío bidimensional) y 3 (reensamblado).

Las implicaciones de este diseño son numerosas:

- Al incluir el particionado de flits en porciones, la utilización de los distintos niveles de la red será uniforme. De esta forma se solucionan los problemas de desequilibrio en la congestión de planos presentados en la sección 0.
- La información transmitida en cada paquete se duplica sin necesidad de modificar los componentes de la red. Para transmitir la misma cantidad de información sin utilizar particionado, sería necesario realizar alguna de estas acciones: (i) duplicar el ancho de los flits, con el coste en área de enlaces y componentes que ello supone, (ii) duplicar el número de flits del paquete, siendo necesario en este caso duplicar el tamaño de los buffers de la red, o (iii) enviar dos paquetes por separado, asumiendo el incremento de latencia que ello conllevaría. La siguiente imagen ejemplifica estas situaciones para una red de dos capas, flits de W bits de ancho y 5 flits de longitud de paquete:

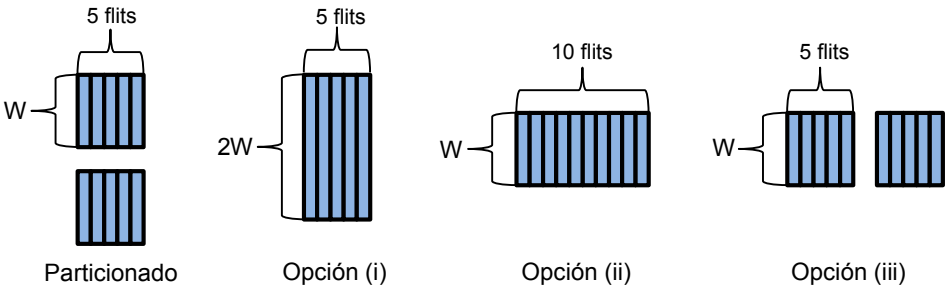

Figura 4.2: Izquierda: Dos porciones, una por capa. Opción (i): Paquetes el doble de anchos. Opción (ii): Paquetes el doble de largos. Opción (iii): Envío de dos paquetes independientes.

- Los routers intermedios en una comunicación tienen total libertad a la hora de dirigir las porciones involucradas. Esto permite que la trayectoria de las porciones pueda adaptarse de acuerdo con la situación presente en cada uno de los planos.

En suma, con este sistema se consigue desacoplar los planos en el tránsito de paquetes por la red, al realizarse el encaminamiento como si de una red bidimensional se tratara y no existieran más capas en el sistema. Además, el particionado aporta uniformidad en el uso de las distintas capas de la red.

No obstante, no todo son ventajas en el uso de esta arquitectura de red. El siguiente punto describe el mayor inconveniente encontrado a la hora de utilizar esta configuración: el reensamblado de las porciones en el nodo destino.

4.2 El problema del reensamblado de paquetes

En la anterior sección se mencionó que las porciones del mismo paquete podían desplazarse por medio de rutas personalizadas para cada una de ellas y, por lo tanto, adaptarse mejor a la situación de tráfico en cada plano. Sin embargo, este dinamismo introduce problemas. Las porciones llegarán a destino en diferentes instantes temporales, y en un orden que no es posible determinar a priori. Esta situación puede traducirse en un fenómeno conocido como “deadlock” o bloqueo [19].

En los consumidores, existe un buffer dedicado al reensamblado de paquetes. Este buffer es finito, siendo únicamente posible ensamblar un número determinado de paquetes al mismo tiempo. Cuando una porción de un paquete nuevo llega al consumidor, ocupará una de estas posiciones hasta que el resto de porciones del mismo paquete la alcancen en destino. Estos ingredientes son perfectos para que suceda un bloqueo en nuestro sistema.

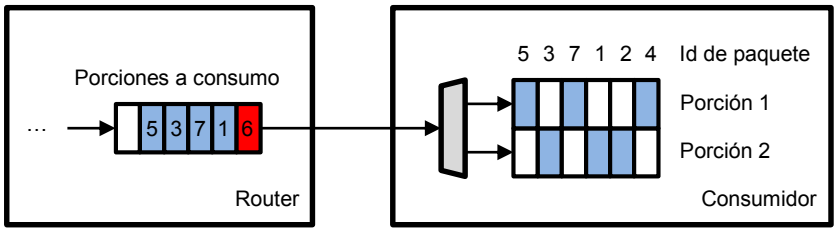

Figura 4.3: Representación simplificada del bloqueo de un consumidor a causa del particionado.

Esta situación es una variante más del conocido “Head-of-Line blocking (HoLB)”, o problema del bloqueo de la cabeza [19]. En la Figura 4.3 se observa cómo la primera porción que se encuentra en la cola del router para acceder a consumo (porción con id 6, resaltada en rojo) no puede hacerlo, al no disponer de ninguna posición libre en la cola de consumo y no existir porciones previas de su mismo paquete ocupando una posición. A continuación se almacenan varias porciones que sí podrían acceder al consumidor y completar los paquetes que se encuentran esperando. En cambio, éstas tampoco pueden pasar a causa del bloqueo que existe en la cabeza del buffer.

Existen dos soluciones básicas a este problema, planteadas en sistemas donde también se realizaba un reensamblado de los paquetes en consumo:

- Dimensionar las colas de consumo con un tamaño suficientemente grande como para poder recibir todo el tráfico existente en la red [20]. No es aplicable, ya que dicho tamaño sería enorme y tendría un gran impacto en el coste del sistema.
- Desechar los paquetes en destino cuando no puedan ser consumidos [21]. Aunque simple, esta solución requiere de un aumento de la complejidad del protocolo de coherencia, ya que debería encargarse de retransmitir los fragmentos desechados, con las correspondientes desventajas asociadas de complejidad y coste de implementación.

Está claro que la solución a este problema radica en la capacidad de quitar esta porción que se encuentra molestando en la cabeza del buffer, para que las siguientes puedan ser consumidas y de esta forma se libere espacio de la cola del consumidor.

Si no es aconsejable que se desechen porciones a causa de la complejidad que ello conllevaría, la alternativa restante consiste en redirigir el paquete a otro punto de la red de forma temporal y que, pasado cierto intervalo de tiempo, pueda tratar de acceder de nuevo al consumidor. De esta forma, las porciones que sí pueden acceder a consumirse tendrían el camino libre, y se liberarían posiciones para permitir la entrada de nuevos paquetes al reensamblado.

Los routers basados en estructuras centralizadas tipo crossbar, como los que se han descrito hasta ahora en este documento, no parecen la mejor opción para este fin. Por ello, lo que en esta arquitectura de red se propone es la utilización del router modular LIGERO [22], con el que será más sencillo afrontar los problemas existentes en el reensamblado de paquetes. Una explicación inicial del funcionamiento de este router se realiza en el siguiente apartado.

4.3 Background: LIGERO

Este router [22] presenta una serie de características que le permiten lidiar con las situaciones habituales que se dan en una red en chip de forma sencilla y eficiente. A continuación se describen las que son relevantes de cara a facilitar la explicación del diseño realizado en este proyecto.

Como se menciona en la sección 2 de este documento, en un router sencillo es muy común la existencia de un control centralizado, que gestiona los enlaces entre sus puertos de entrada y sus puertos de salida, generalmente conectados mediante un crossbar. En cambio, en el router LIGERO la lógica se encuentra distribuida, ya que posee una estructura modular donde cada uno de los componentes realiza su función de forma autónoma. A continuación se muestra esta estructura:

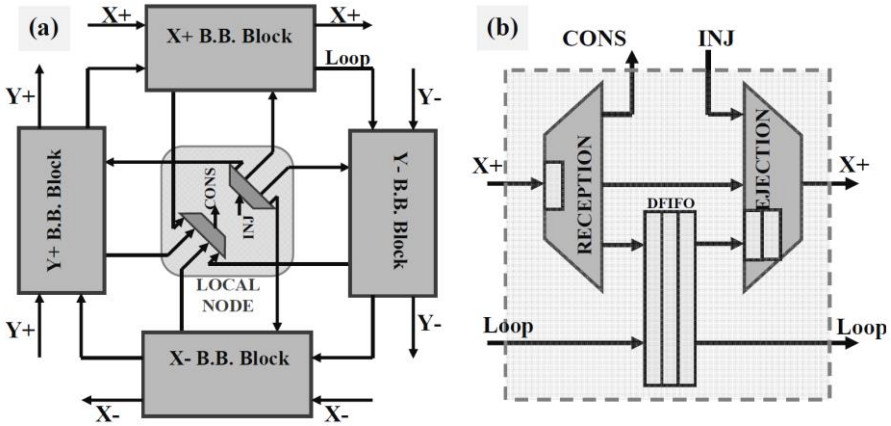

Figura 4.4: (a) Representación del router LIGERO a nivel de bloque básico. (b) Contenido del bloque básico superior. (Fuente: [22]*)*

En la figura anterior se encuentra la composición del router. Está formado por cuatro componentes principales, denominados Bloques Básicos. Cada uno de estos bloques gestiona el puerto de entrada y salida en una de las direcciones, y posee conexiones de inyección y consumición con el nodo que se encuentra asociado con el router para permitir la entrada y salida de paquetes en la red. Además, los bloques básicos se encuentran conectados entre sí formando un anillo interno. Este anillo es utilizado por

los paquetes para cambiar de bloque básico dentro del router, con el objetivo de alcanzar el puerto de salida que dirige el paquete hacia su destino.

El hecho de que los paquetes dispongan de un anillo para moverse a través de los bloques básicos del router evita que se produzcan contenciones del tipo HoLB descritas anteriormente.

Como añadido a esto, el router posee lógica adaptativa para responder a congestiones en la red, de forma que si un paquete está teniendo serias dificultades para avanzar a destino utilizando su ruta actual, puede ser redirigido para que pruebe suerte por otro itinerario. Esto se realiza mediante una operación conocida como “missrouting”.

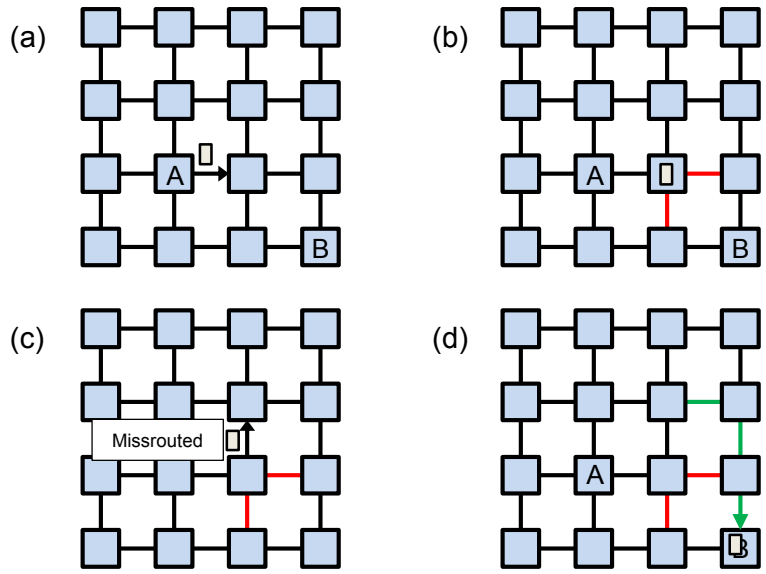

Figura 4.5: (a) Un paquete entra a la red en el nodo A con destino B. (b) En el trayecto, alcanza una zona congestionada e intenta sin éxito salir de ella rotando en el anillo interno de LIGERO. (c) Tras varios intentos, el paquete es redirigido hacia el primer puerto disponible. (d) Finalmente, el paquete alcanza el destino por otra ruta no congestionada.

Como se muestra en la Figura 4.5, si un paquete ha dado un número determinado de vueltas en el anillo de un router sin tener la posibilidad de salir por un puerto que le convenga, el paquete entra en el estado de “missrouted”, y abandonará el router por el primer puerto de salida que encuentre disponible (tanto si le acerca a destino como si no). Cuando un router recibe un paquete en este estado, recalcula la ruta desde su posición para enviarlo de nuevo a destino.

No obstante, para evitar que un paquete viaje indefinidamente por el interior de la red, existe un límite en el número de veces que puede sufrir un “missrouting”. Cuando este límite se alcanza, el paquete es introducido en un camino virtual especial que atraviesa todos los nodos de la red, denominado camino de escape. En la siguiente imagen puede observarse dicho camino:

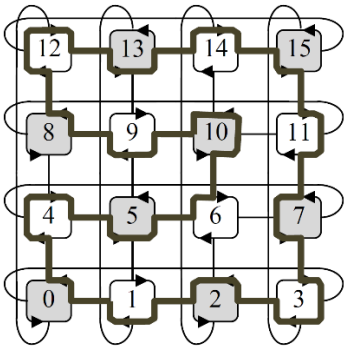

Figura 4.6: Camino de escape embebido en un toro 4x4 creado con routers LIGERO (fuente: [22]*).*

Existe un número máximo de paquetes que pueden viajar por el interior del camino, posibilitando el avance continuo de los paquetes en el mismo. Por ello, este mecanismo es utilizado por LIGERO para afrontar aquellas situaciones que en otro caso supondrían un bloqueo del sistema.

La Figura 4.4 (b) muestra la estructura de uno de los bloques básicos. Posee una cola FIFO (First In First Out) con dos puertos de entrada y de salida. Una pareja de los mismos se utiliza para formar el anillo interno del router, y la otra para gestionar el acceso y salida del mismo, además de conectar las dos etapas que posee el router, que a continuación se describen:

- **Recepción o entrada**: En esta etapa es necesario decidir el camino que seguirá un paquete que se encuentre en el puerto de entrada asociado al bloque básico, existiendo tres posibilidades, de la superior a la inferior:
    - **Consumo**: El destino del paquete es el nodo asociado al router, por lo que debe ser extraído de la red.
    - **Camino de bypass o cortocircuito**: El paquete necesita continuar en la dirección en la que se encuentra viajando. Para utilizar este camino es necesario que haya espacio en la etapa de salida para encolar este paquete.
    - **Entrada al anillo**: Si las condiciones para tomar las opciones anteriores no se cumplen, el paquete es introducido al anillo interno de bloques básicos, si el anillo no está saturado. Si tampoco fuera posible el acceso al anillo, el paquete esperaría en el puerto de entrada hasta el siguiente ciclo, donde realizaría de nuevo las comprobaciones.
- **Expulsión o salida**: Decide cuál de los paquetes que se encuentran esperando para abandonar el router por el puerto asociado al bloque lo hacen, eligiendo de nuevo entre tres orígenes:
    - **Inyección**: El nodo intenta introducir un nuevo paquete en la red.
    - **Salida de bypass**: Utilizada por los paquetes que atraviesan el router por el camino de cortocircuito**.**
    - **Salida del anillo**: Los paquetes que se encuentran en el anillo lo abandonan a través del puerto superior de la cola FIFO.

Mientras que la prioridad de los enlaces en la etapa de entrada va del superior al inferior (consumo, bypass y entrada en anillo), en el caso de la salida no ocurre así. En primer

lugar se gestionarían los caminos de bypass y de expulsión desde el anillo (existiendo la posibilidad de colocarlos en cualquier orden o, si se desea, de alternarlos), mientras que la inyección de nuevos paquetes sería la menos prioritaria en todas las configuraciones. Éste es un punto clave de esta arquitectura de router, ya que prioriza a los paquetes que se encuentran en tránsito sobre los que desean entrar a la red. El uso de esta prioridad mantiene una densidad de paquetes en la red baja, lo que hace que el movimiento de los mismos sea muy fluido. De este modo, se consigue disminuir la latencia y asegurar un rendimiento sostenido en el tiempo. Por ejemplo, el camino de bypass está disponible en la mayoría de ocasiones, salvo cuando los paquetes necesiten girar o exista una congestión en algún punto de la red.

En este momento se dispone de una idea inicial de cómo funciona este router. Cuando sea necesario, se realizarán explicaciones más profundas acerca de aspectos del router LIGERO, que servirán para comprender mejor algunos de los problemas encontrados durante el desarrollo del proyecto y las decisiones tomadas al respecto.

Es preciso realizar una serie de cambios iniciales sobre el mismo para hacer posible su funcionamiento en 3D, principalmente asociados a la comunicación con los consumidores finitos.

4.4 Inclusión de consumidores finitos

En un control de flujo básico “stop & wait”, el receptor activa una señal de parada o stop cuando su buffer de consumo alcanza un máximo establecido, impidiendo que se reciban más paquetes. Cuando el tamaño en ese buffer vuelve a ser adecuado, el receptor retira la señal de stop para que prosiga la comunicación.

En nuestro caso, no se puede utilizar esta señal de stop de forma global, ya que la cola de consumo no se vaciará por sí misma, sino cuando reciba las porciones restantes para completar los paquetes. Por lo tanto, el mecanismo de control de flujo debe variar ligeramente:

- Cuando una porción es almacenada en la cola de consumo de su destino verdadero, se notifica a los consumidores de la vertical, que colocarán el identificador del paquete en una lista de paquetes admitidos.
- Si en algún momento la cola de un consumidor alcanza el límite, éste activará su señal de stop.
- Cuando una porción llega a destino a través de su capa, comprueba la señal de stop de su destino verdadero. Si esta señal no está activa, la porción es consumida. Si por el contrario se encuentra activa, comprobará en la lista si se trata de uno de los paquetes permitidos, en cuyo caso es consumido.
- El identificador será borrado de las listas de admitidos de la vertical cuando el paquete se haya recibido por completo.

Aquellas porciones que no pueden ser consumidos porque la señal de stop del consumidor está activa y no se encuentran admitidos entrarán automáticamente al anillo interno en estado “missrouted”. Como se explicó anteriormente, esto provocará que abandonen el router por el primer puerto que se encuentre disponible, y permitirá tanto que las porciones que vienen por detrás y están admitidas lleguen como que las

porciones redirigidas puedan tratar de consumirse más tarde. En la siguiente figura se observa cómo con este sistema se evita la aparición de bloqueos:

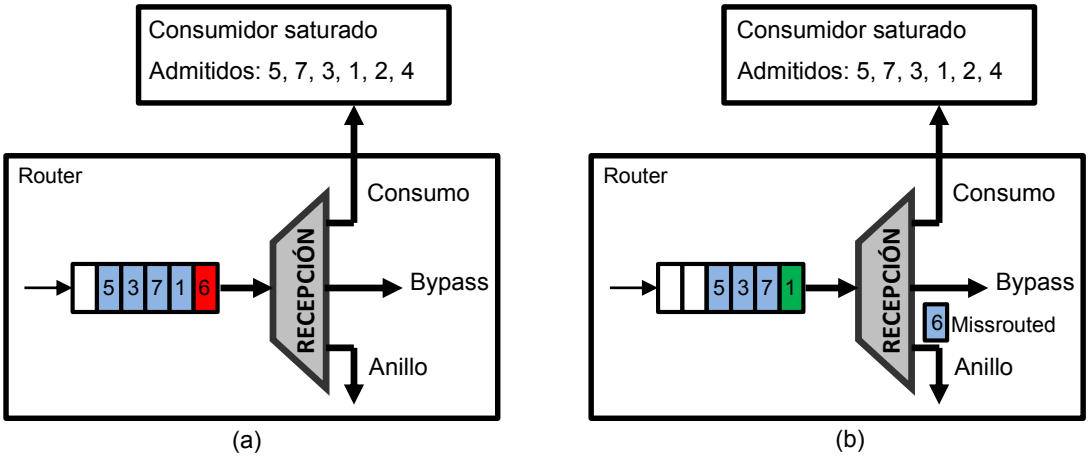

Figura 4.7: (a) La porción del paquete 6 no puede ser consumida a causa de la saturación. (b) Esta porción es redirigida al anillo, permitiendo el paso de las que sí pueden acceder

5 Modificaciones aplicadas sobre el diseño

Al evaluar el rendimiento del sistema inicial, se realizaron pruebas en las que no se estableció un límite en el tamaño de las colas de consumo. Estas pruebas buscaban averiguar cuántos paquetes coincidían en promedio y en el peor caso en la fase de re-ensamblado. Un ejemplo de los resultados obtenidos para tres de los tráficos utilizados se muestra en la siguiente figura:

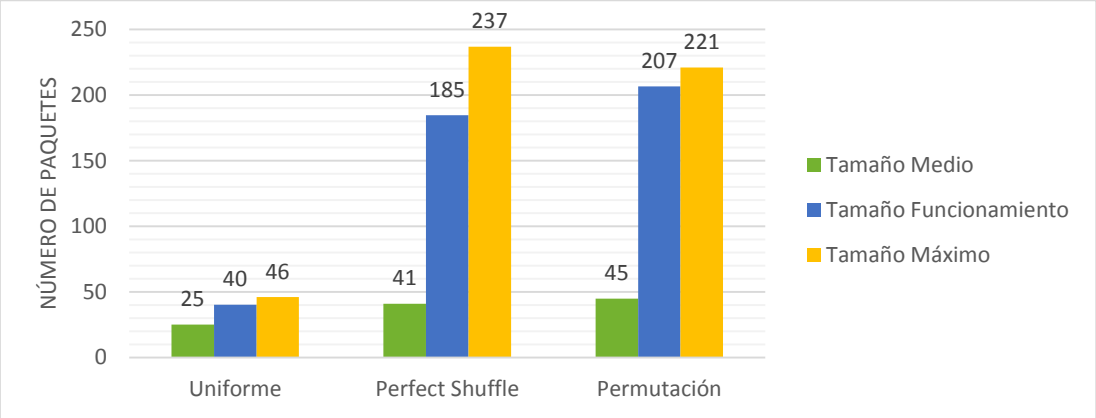

Figura 5.1: Tamaños iniciales de las colas de consumo para tres patrones de tráfico.

La figura representa el tamaño medio, de funcionamiento y de peor caso de las colas de consumo al aplicar los patrones uniforme, perfect shuffle y permutación con carga máxima de inyección. El tamaño de funcionamiento se ha obtenido mediante la suma del valor medio y de tres veces el valor de la desviación estándar o sigma.

Simplemente observando el tamaño de funcionamiento del caso uniforme, con 128 bits por flit y 5 flits por paquete, serían necesarias unas colas con capacidad para reensamblar 200 flits, que equivalen a aproximadamente 3,2 kilobytes de memoria, siendo una cantidad demasiado alta y que tendría un gran impacto en el coste de nuestra red.

El principal causante de este tamaño de colas de reensamblado es la diferencia temporal que existe entre la llegada de las porciones de un mismo paquete. Los siguientes dos puntos recogen los motivos de esta divergencia:

- La autonomía que poseen los planos es beneficiosa desde el punto de vista de simplificar la señalización de control vertical. En cambio, hay algunas situaciones en las que la falta de sincronización perjudica gravemente al sistema, y en las que una coordinación simple entre capas sería suficiente para obtener una red más estable e impedir comportamientos anómalos.
- Una carga elevada por parte de los inyectores favorece la aparición de estas diferencias de tiempo de forma indirecta, ya que cuando existe contención se toman más decisiones adaptativas dependientes del estado de la red. Adicionalmente, si estas decisiones no son deterministas, esto es, pueden ofrecer respuestas diferentes frente a una misma situación, es posible que porciones del mismo paquete tomen distintas decisiones durante su tránsito, lo que favorecerá la aparición de una mayor divergencia en el tiempo de llegada.

Como agravante, el tamaño de las colas de consumo presentaba cierta tendencia a crecer conforme aumentaba la duración de las simulaciones realizadas.

Por ello, en esta sección se recogen aquellas propuestas que han sido añadidas al diseño inicial para tratar de mejorar la respuesta de la red, y los resultados obtenidos por su inclusión.

5.1 Reducción de dispersión entre fragmentos

5.1.1 Inyección simultánea de porciones

Para comprender esta modificación, es necesario explicar más en detalle en qué consiste la fase de inyección de un paquete. Esta fase abarca desde el momento en que el inyector recibe un paquete por parte del nodo asociado para su envío hasta la puesta de ese paquete en tránsito en el interior de la red.

Generalmente, esto sucede en dos etapas: en primer lugar, el inyector introduce el paquete en el router a través del puerto de entrada dedicado a tal fin, y éste es almacenado en el buffer de entrada de ese puerto. En adelante se denominará **preinyección** a esta etapa. En segundo lugar, cuando se dan las condiciones que permiten el avance, el paquete abandonará el buffer de entrada y se dirigirá hacia una de las salidas del router, realizándose la **inyección** propiamente dicha.

En nuestro sistema particionado, se mantienen las dos etapas mencionadas. La preinyección se produce cuando el inyector utiliza los enlaces verticales para colocar las porciones del paquete en los routers de la vertical, y la inyección tiene lugar cuando en la etapa de salida del router LIGERO se permite el avance de los paquetes que se encuentran en el buffer de inyección hacia el puerto de salida.

Como se mencionó en la sección 4.3, la inyección de nuevos paquetes es la opción menos prioritaria de las tres disponibles en la etapa de salida (junto con el camino de bypass y la salida del anillo). Los paquetes que se encuentren en tránsito por el router determinan si dicha inyección se lleva a cabo o no.

A causa de esto, tiene lugar el siguiente fenómeno: si cada uno de los routers de la vertical inyecta las porciones de forma autónoma y dependiendo solamente de su estado, sucederá que la inyección en la red de porciones del mismo paquete tendrá lugar en momentos potencialmente muy separados (en especial si la red está saturada). En el caso extremo, incluso puede darse una situación en la que una de las porciones llegue a destino mientras que el resto aún se encuentren esperando en origen a que tenga lugar su inyección, lo que puede desembocar en un bloqueo de la red.

Para solucionar este problema, es muy importante que la inyección de porciones del mismo paquete en la red suceda de forma simultánea. Por lo tanto, es necesaria una sincronización entre capas en la fase de inyección. Para simplificar al máximo el control necesario, se ha realizado lo siguiente:

- Cuando una porción accede a un router en la etapa de preinyección, se le asignará un puerto de salida fijo. Las porciones del mismo paquete repartidas en la vertical utilizarán el mismo puerto de salida en cada router.
- Para que un mensaje pueda ser inyectado en la red desde un puerto de salida, es necesario que la inyección esté disponible para todos los routers de la vertical.

De esta forma, cuando en un ciclo se compruebe que todos los routers pueden inyectar en una dirección, y solamente entonces, se realizará dicha inyección de forma simultánea. Al haber colocado en la misma dirección todas las porciones, lo que

sucederá es que se introducirán todas las porciones del paquete de una vez. La siguiente imagen ejemplifica este mecanismo para una red con dos capas:

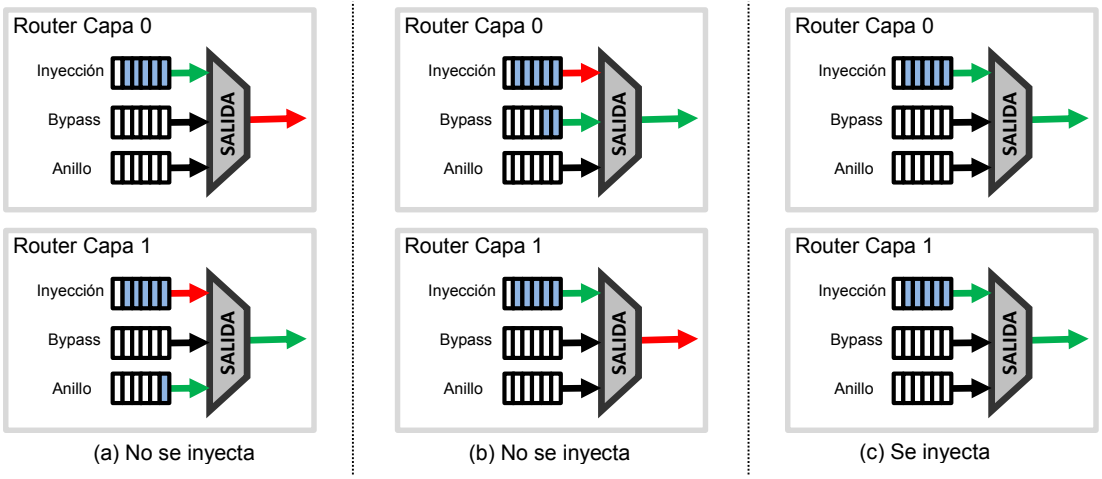

Figura 5.2: (a) La inyección no se lleva a cabo porque el router 1 tiene tráfico en la salida del anillo. (b) En este caso, es el router 0 el que tiene tráfico en su camino de bypass. (c) Sin paquetes en tránsito, la inyección se lleva a cabo en ambas capas.

Para simplificar el ejemplo de la figura anterior, se ha omitido el caso en el que es el router vecino quien bloquea la inyección al no tener espacio para albergar el nuevo paquete, siendo esta otra de las condiciones a tener en cuenta en la inyección.

Con este nuevo mecanismo, el tiempo entre llegadas de porciones del mismo paquete se ve reducido considerablemente, reflejándose esto en el tamaño de las colas de consumo (Figura 5.3). Además, el comportamiento de estas colas se muestra estable independientemente del número de ciclos simulado.

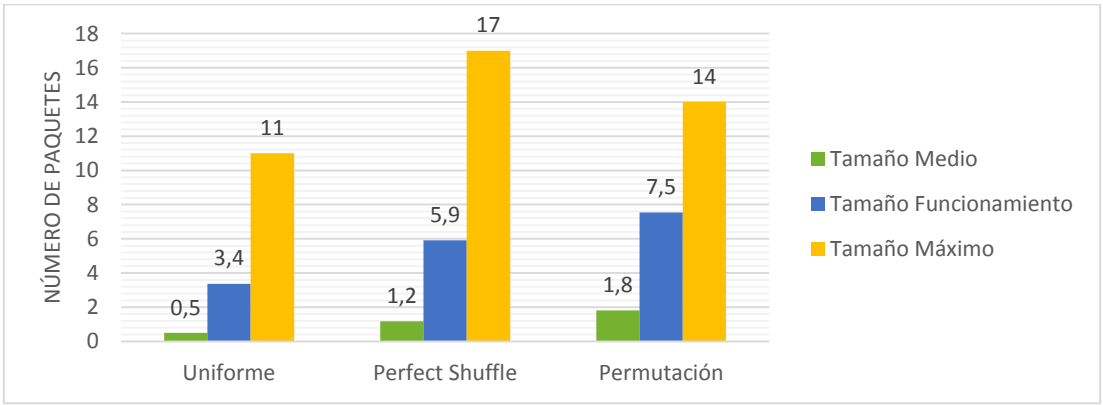

Figura 5.3: Nuevo tamaño de colas, incluyendo inyección simultánea.

5.1.2 Toma de decisiones alternativas determinista

Cuando un router recibe un paquete que se encuentra en estado “missrouted”, debe calcular de nuevo su ruta hacia destino, al ser posible que la previa ya no sea válida. En el simulador, el enrutado que se utiliza es aritmético [19], indicándose el número de saltos que el paquete debe tomar en cada dirección y sentido para llegar a destino.

En las redes toro utilizadas en las simulaciones, existen pares origen/destino donde es posible completar el trayecto con el mismo número de saltos en un sentido o el otro de una de las direcciones, tal como se muestra en la siguiente figura:

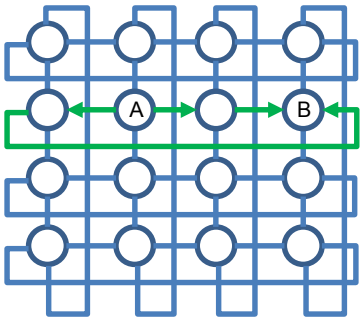

Figura 5.4: Dos caminos posibles para ir desde A hasta B en una red toro 4x4.

En estos casos, se tiende a uniformizar el número de paquetes enviados hacia cada dirección, por lo que la decisión es aleatoria hasta que tiene lugar. En las pruebas realizadas, lo que se observa es que porciones del mismo mensaje tienden a realizar el mismo itinerario hasta que se descubre una congestión en la red, a causa de la cual pueden tener lugar cambios al estado “missrouted”. Si en este cambio cada una de las porciones toma una decisión diferente, desde ese punto llegarán a destino por caminos opuestos, lo que introduce un desequilibrio en los planos (siguiendo con el ejemplo, una porción podría tomar el camino de la izquierda para ir desde A hasta B, mientras que otra podría tomar el camino de la derecha).

La solución a esta divergencia pasa por hacer que las porciones coincidan en su decisión al recalcular su camino a destino, seleccionando siempre la misma de las dos direcciones en caso de empate. Si bien no se utilizarán los dos posibles caminos de una manera uniforme, no es muy frecuente que un paquete llegue al estado “missrouted” a causa de los pocos paquetes que circulan por la red en un mismo instante, por lo tanto esta decisión no afectará notablemente al rendimiento del sistema.

En cambio, aunque sean pocas las porciones que sufran esta situación, el hecho de que cada una de las porciones tome caminos distintos también afecta al resto de porciones de la red, incentivándose la aparición de una reacción en cadena de divergencias que se traducen en una pérdida de rendimiento.

5.2 Reducción de la presión sobre la red

5.2.1 Inclusión de equidad en la inyección

Aunque las restricciones a inyectar paquetes nuevos en la red favorecen a su correcto funcionamiento, en ocasiones pueden dar lugar a injusticias en lo que respecta al número de paquetes existente en la red por nodo origen. En la Figura 5.5 se muestra el mapa de inyección normalizado en flits por ciclo y router de una de las capas de una red 8x8x2, sobre la que se ha aplicado un tráfico de matriz transpuesta:

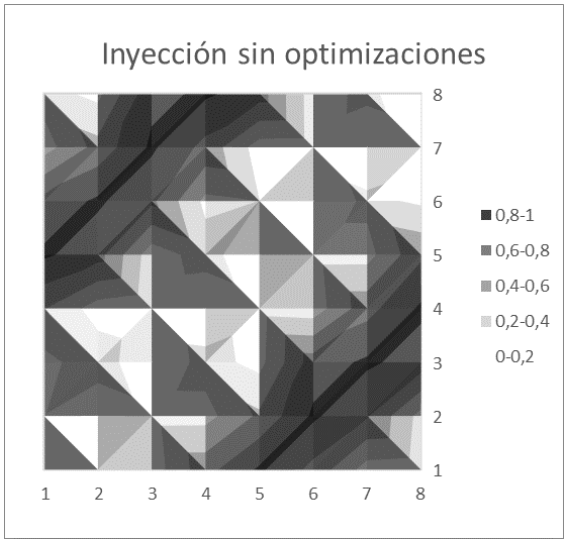

Figura 5.5: Mapa de inyección normalizado del tráfico permutación para una red 8x8.

En la imagen anterior, cada una de las posiciones (x, y) se corresponde con un router de la red. Se observa cómo ciertos nodos han adquirido el control frente a los que tienen alrededor (las dos diagonales oscuras), inyectando más paquetes que ellos. Además, lo que parece más problemático es que ciertos nodos no son apenas capaces de realizar inyecciones en la red.

Principalmente, esto se debe a que la inyección de paquetes se realiza hacia los routers vecinos a través de la etapa de salida, y las decisiones de inyección dependen únicamente del estado propio y de que el vecino en cuestión tenga sitio para nuevos paquetes. Si un router toma el control e inyecta continuamente paquetes en sus vecinos, impedirá que éstos puedan inyectar y equilibrar la situación.

Este desequilibrio afecta a las colas de consumo en el momento en que los paquetes que estos nodos generan se dirigen al mismo destino, ya que se introducen más paquetes en la red de los que el nodo destino es capaz de gestionar.

Para tratar de evitar esta situación lo máximo posible, se han realizado dos ajustes sobre el router:

- **Gestión del anillo**: Cuando el router dominante está saturando de paquetes a uno de sus vecinos, primero rellena el buffer de los paquetes que pueden utilizar el camino de bypass, para posteriormente introducir paquetes en el anillo interno. Como los paquetes en el anillo se encuentran en constante movimiento, si no existe ningún control es posible que un solo router complete el anillo de un vecino, perjudicando al tráfico que atraviesa ese router en las otras direcciones e impidiendo que pueda inyectar al tener preferencia los paquetes que abandonan el anillo, tal como se muestra en la siguiente imagen:

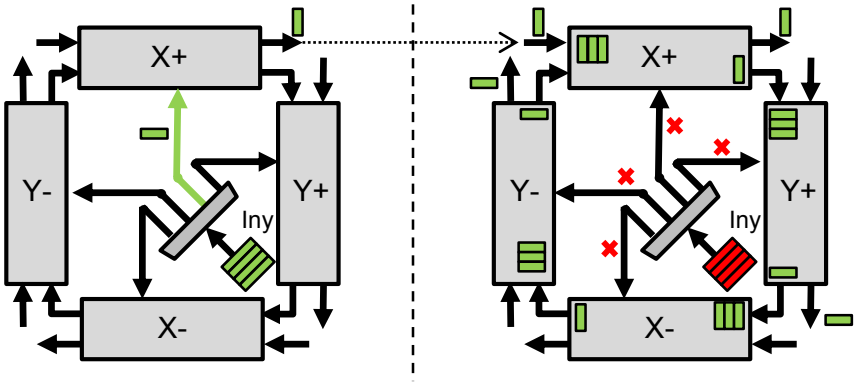

Figura 5.6: El router izquierdo inyecta paquetes (en verde) a través del router vecino. Si no hay ningún control, los paquetes avanzan girando por el anillo y saturando los puertos de salida, e impiden al router inyectar sus propios paquetes con normalidad.

Para evitar esto, la capacidad total del anillo de un router se dividirá de forma equitativa entre todos los puertos de entrada. Cuando se ha completado esa porción de anillo que es posible utilizar desde uno de los puertos de entrada, no se permitirá a los siguientes paquetes acceder al anillo hasta que los actuales lo abandonen.

- **Missrouting en inyección**: Como se mencionó al describir la inyección, cuando un paquete es particionado e introducido en el router en la fase de preinyección todas sus porciones se sitúan en la misma dirección de salida, para poder llevar a cabo la inyección simultánea de una manera sencilla. En la siguiente imagen se visualiza la estructura de la preinyección en un router:

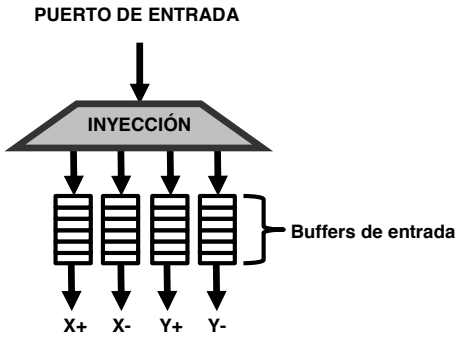

Figura 5.7: Estructura de preinyección en el router LIGERO.

Al seleccionar una dirección de salida, se almacena cada porción en el buffer de entrada que la enviará en ese sentido. El estado de estos buffers es el mismo en todos los routers de la vertical, ya que solo existe un puerto de entrada que genera las porciones al mismo tiempo y la inyección de paquetes en la red (y, por tanto, el vaciado de estos buffers) es simultánea. Dependiendo del destino, al paquete le vendrá bien situarse en una o dos direcciones de salida de las cuatro que se disponen. Si los buffers de las direcciones de salida que les convienen están completos, las porciones del paquete esperarán en el puerto de entrada hasta que exista un hueco para salir.

Si existen routers vecinos que acaparan la red enviando mensajes en la misma dirección en la que un router desea inyectar, es posible que el paquete tenga que esperar a que sus vecinos terminen la ráfaga completa para acceder a la red

en esa dirección, mientras que en el resto de direcciones sí que podría realizar una inyección.

Lo que se propone con esta modificación es lo siguiente: cuando las porciones de un paquete no han podido acceder a los buffers de entrada que les convienen transcurrido un número de intentos determinado, éstas comenzarán a probar con todos los buffers, saliendo por el primero que se encuentre disponible. Como puede que la salida seleccionada no sea en principio conveniente para el paquete, se establecerá como “missrouted”, para que se lleve a cabo un recálculo de su ruta si fuera necesario.

Con este sistema se permite que los routers más perjudicados en inyección por sus vecinos puedan inyectar paquetes, y de este modo traten de igualar la situación en la red. El sobrecoste asociado a inyectar el paquete en una dirección errónea o no adecuada es aceptable, ya que de otro modo el paquete seguiría esperando para su entrada en la red.

Con la aplicación de las modificaciones anteriores se consigue la siguiente mejora con respecto al mapa de inyección inicial:

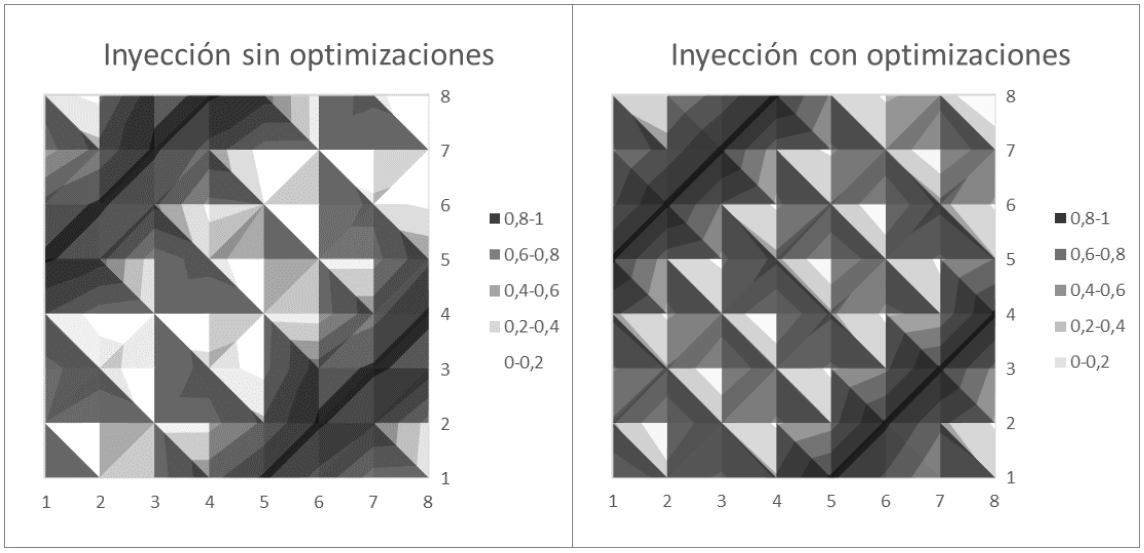

Figura 5.8: Comparación de inyección al aplicar los nuevos controles.

De nuevo, el mapa no es del todo uniforme, existiendo todavía ciertos nodos que, aunque en menor medida, acaparan los recursos de ciertas zonas de la red. En cambio, lo que sí es apreciable es el incremento en la capacidad de inyección de los nodos que anteriormente sufrían de graves problemas a la hora de enviar nuevos paquetes, tal como se aprecia en la mayor uniformidad de los colores claros del gráfico.

5.2.2 Respuesta mejorada ante congestión en las colas

Con las modificaciones anteriores, es posible comenzar con las pruebas que establecen un límite finito en las colas de consumo. Este límite se ha fijado en 14 paquetes, para ofrecer cierta holgura sobre los tamaños de funcionamiento que la red presentaba en las pruebas anteriores.

Pese a los controles de inyectores dominantes realizados, sigue siendo posible que varios nodos envíen mensajes hacia el mismo destino en ciertos momentos, por lo que se necesita un mecanismo más avanzado para controlar la inyección.

En el diseño inicial, si una porción no podía ser consumida era redirigida como “missrouted”, para que momentáneamente liberara la entrada a consumo. En las pruebas, esta práctica se ha encontrado insuficiente. Dicha porción era recibida por el router vecino, y entonces utilizaba el anillo interno para dar la vuelta y así poder volver por donde fue rebotada para intentar ser consumida de nuevo. Esto se traduce en que en las inmediaciones de un consumidor saturado se encontrarán paquetes continuamente saliendo y entrando en el router asociado, generando una zona de congestión. Además, los nodos que quieran mandar mensajes a este destino en principio no son conscientes de lo que ocurre y continuarán haciéndolo, empeorando aún más la situación.

Para evitar que la llegada de las porciones necesarias se vea dificultada por este tráfico alrededor del router, las porciones no admitidas serán reenviadas de nuevo a su nodo origen, como se muestra en la figura:

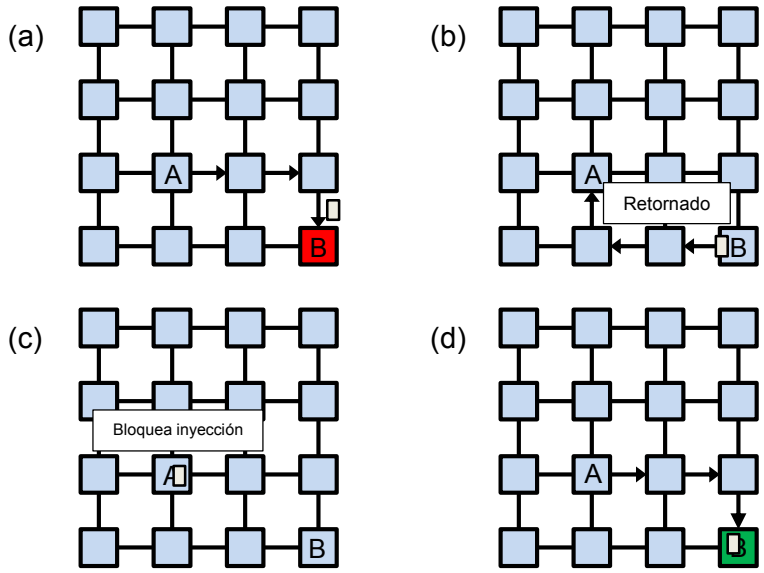

Figura 5.9: (a) El paquete alcanza su destino B, pero no puede ser consumido. (b) Se produce un retorno del paquete hacia su nodo origen. (c) Al llegar a origen, el paquete bloquea la inyección de A temporalmente. (d) Finalmente, es mandado de vuelta a destino donde esta vez sí logra consumirse.

Este mecanismo permite que el consumidor congestionado pueda recibir las porciones que completarán los paquetes que almacena y hacer que la situación vuelva a la normalidad. Cuando los paquetes retornados alcanzan el router origen, son enviados de nuevo al nodo destino.

En la Figura 5.9 (c) se observa un añadido a este sistema de retorno. Cuando un router recibe un paquete retornado, no podrá realizar nuevas inyecciones en la red hasta que dicho paquete haya abandonado el router de vuelta a destino. Con esto se evita que continúe con la ráfaga de paquetes hacia el destino saturado, y se favorece a la eliminación del punto de congestión.

La incorporación de estas modificaciones ha mejorado la equidad de inyección en la red, manteniendo los valores de colas de consumo obtenidos al aplicar la inyección simultánea. Aún existen situaciones que saturan las colas de reensamblado, pero su aparición es remota y únicamente ante cargas realmente exigentes. Los resultados se muestran en la siguiente imagen:

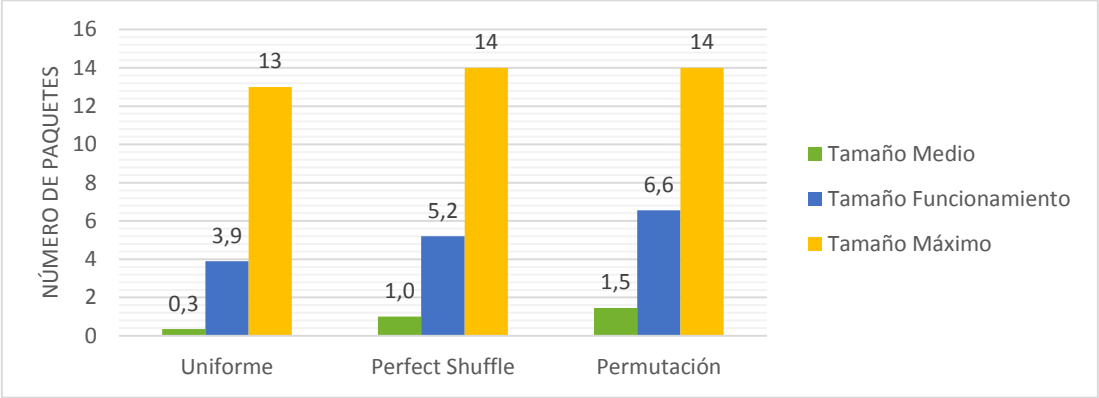

Figura 5.10: Estado de las colas de consumo en saturación al aplicar todas las modificaciones al sistema.

5.3 Adaptación para mensajes verticales

Con la utilización de sistemas con más de una capa, tiene lugar una situación especial que en principio no se había tenido en cuenta en el desarrollo del router LIGERO, al no ser posible su aparición en una red bidimensional. El hecho de que haya nodos apilados unos encima de otros permite la existencia de mensajes que en principio solamente necesiten viajar en vertical, esto es, atravesando los planos de la red sin desplazarse por los mismos.

En el router LIGERO, todos los paquetes realizan su inyección en uno de sus nodos vecinos. Lo que se descubrió en las pruebas fue que, cuando un paquete sólo necesitaba moverse entre capas, le sucedía lo siguiente:

- Las porciones del paquete se distribuían entre capas del mismo modo que se hacía en el resto de los casos.
- Cuando las porciones accedían a la red, lo hacían a través de uno de los puertos de salida, saliendo del router en el que se encontraban.
- Al recibir la porción el router vecino, comprobaba que no era para él, y que existía algún problema con la ruta de esa porción. Asumía que se encontraba en estado “missrouted”, y entonces recalculaba dicha ruta.
- Con la nueva ruta, las porciones volvían de nuevo a su nodo origen, donde eran consumidas de forma natural.

Por lo tanto, se estaban añadiendo dos saltos más de los necesarios a todos aquellos paquetes cuyo origen y destino se encontraba en la misma vertical. A causa de las conexiones verticales disponibles, existen dos caminos que se pueden utilizar para enviar estos paquetes: a través de los inyectores o a través de los consumidores.

El camino a través de los inyectores parece el más rápido, ya que el mensaje generado es transmitido cuanto antes al nodo destino sin tener que pasar por la red. No obstante, esto introduce complejidad en la lógica de los inyectores, haciendo que su comunicación con el nodo pase de ser de un solo sentido a ser bidireccional. Por ello, se ha realizado el envío mediante las conexiones de los consumidores, ya que se trata a estos paquetes de una manera más acorde con el funcionamiento general del sistema.

Para evitar la salida a los routers vecinos, se ha añadido un camino que conecta las zonas de inyección y consumo, tal y como se muestra en la siguiente figura:

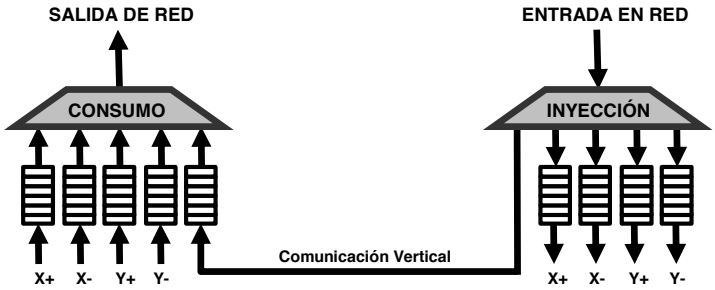

Figura 5.11: Nuevo camino para los mensajes con tránsito únicamente vertical.

Las porciones de los paquetes que sólo necesitan desplazarse en la vertical serán inyectadas del mismo modo que se hacía hasta ahora. En cambio, en lugar de utilizarse un puerto de salida y ser enviadas al router vecino, en este caso pasarán directamente a consumo por medio del nuevo camino, donde se mantendrán a la espera para poder acceder junto con el resto de las porciones.

Con la red presentando un comportamiento aceptable desde el punto de vista funcional, el siguiente objetivo comprende la evaluación del sistema frente a otros planteamientos para comparar el rendimiento obtenido.

6 Evaluación

El análisis de la arquitectura de red diseñada persigue dos objetivos diferenciados:

- En primer lugar, se busca determinar si efectivamente la distribución de los elementos del chip en varias capas es beneficiosa desde el punto de vista de una mejora del rendimiento frente a redes 2D.
- En segundo lugar, se desea determinar la pérdida de rendimiento que se está asumiendo al utilizar el particionado de paquetes para el envío, principalmente causado por el reensamblado de dichos paquetes en destino.

A continuación se definen los tipos de red y los resultados obtenidos en cada una de las pruebas realizadas, así como su discusión.

6.1 Configuraciones de red

Dos tipos de redes se compararán frente el sistema 3D diseñado:

- **Red 3D con particionado ideal**: En esta red, el movimiento de los paquetes por la vertical es instantáneo. Se representará con una red de una sola capa, en la que la anchura de los flits será del doble con respecto a las demás redes (para que los paquetes transmitan la misma información, de acuerdo con las opciones de la Figura 4.2).

  Lo que se intenta representar es que ambas capas de la red se aglutinan en una sola, anexionando los nodos verticales y ofreciendo por tanto latencia cero en particionado y tránsito entre niveles. No obstante, esa es la única parte que es ideal en esta red, ya que posteriormente se comprobará que en ciertas pruebas no ofrece un mejor rendimiento que las demás.

- **Red 2D**: Se utilizarán redes 2D convencionales, en las que el número de flits por paquete es el doble, de nuevo para hacer que los paquetes transporten la misma cantidad de información.

La denotación de las redes será la siguiente: la red 3D con particionado ideal será en adelante 3DP; la red 3D con particionado estándar se denominará 3D y por último a las redes bidimensionales se las denotará simplemente como 2D. En la siguiente tabla se resumen las características de cada una de las redes:

Red	3DP	3D	2D
Anchura de flit	2W	W	W
Longitud de paquete	L	L (dos porciones)	2L

Tabla 6.1: Parámetros de cada una de las redes.

En todas las configuraciones de red se utilizará el mismo router LIGERO, que incorpora los cambios introducidos para mejorar el rendimiento general del sistema.

En cuanto a topologías, se realizarán las pruebas sobre redes toro, con unas dimensiones de planos que variarán del siguiente modo:

- En ciertas pruebas se compararán los tres tipos de redes. Generalmente, en una red se obtiene un mejor rendimiento si ésta es cuadrada (misma longitud para

todas las dimensiones). Para que las comparaciones no sean injustas con la red 2D, siempre será esta la que tenga la topología cuadrada, mientras que las redes tridimensionales serán representaciones de la red 2D "doblada". En la siguiente tabla se muestra la relación de redes utilizada en cada prueba:

Red	3DP	3D	2D
Dimensión 4	4x2(x2)	4x2x2	4x4
Dimensión 6	6x3(x2)	6x3x2	6x6
Dimensión 8	8x4(x2)	8x4x2	8x8

Tabla 6.2: Redes utilizadas en las comparaciones de los tres tipos de red.

En estas pruebas se utilizará solamente tráfico uniforme, ya que al variar las topologías entre las redes 3D y 2D el tráfico aplicado en caso de los dirigidos cambiaría de igual forma, haciendo imposible la comparación.

- En las pruebas restantes solamente se utilizarán las redes 3D. Esta vez serán redes cuadradas, con las mismas dimensiones que en el caso anterior:

Red	3DP	3D
Dimensión 4	4x4(x2)	4x4x2
Dimensión 6	6x6(x2)	6x6x2
Dimensión 8	8x8(x2)	8x8x2

Tabla 6.3: Topologías utilizadas en la comparación de las redes 3D.

En este caso, sí es posible incorporar tráficos dirigidos en la comparación.

- Adicionalmente, se realizará una comparación entre las redes 3D real y 2D bajo un sistema más realista, utilizando para ello el simulador GEMS. En estas simulaciones de nuevo la red 2D será la cuadrada, mientras que la red 3D representará un plegado de la misma.

6.2 Tráfico sintético

Las simulaciones realizadas en este apartado se dividen en dos tipos:

- **Modal**: Configuración de tráfico base, con paquetes de la misma longitud (5 flits). En estas simulaciones se aplicarán diferentes cargas sobre la red durante un número de ciclos determinado, suficiente para garantizar el estado estable de la red. En el punto estable se obtendrán métricas de latencia.
- **Reactivo**: Se generarán peticiones que serán contestadas con respuestas, siendo las peticiones de tamaño mínimo (uno o dos flits, dependiendo de si la red es 3D o 2D) y las respuestas del tamaño base (5 flits redes 3D, 10 flits redes 2D). Cada uno de los nodos activos solamente podrá tener un número determinado de peticiones en vuelo, que se verá reducido según vaya recibiendo las respuestas pertinentes. Este número ha sido seleccionado de forma empírica, eligiendo aquel que mejoraba los resultados de las pruebas para cada red.

  La mitad de los nodos se comportarán de forma activa, enviando peticiones al resto de nodos además de contestando a las peticiones que reciban. La otra

mitad de nodos actuarán de forma pasiva, siendo su única función el envío de respuestas a las peticiones. Esta configuración de red es más realista, ya que se asemeja a lo que sucede en la realidad (los procesadores serían los nodos activos, mientras que los bancos de cache serían los pasivos). Con la inclusión de un límite de peticiones por nodo se busca modelar el comportamiento de un sistema real. En el mismo, existe una estructura denominada MSHR (Miss Status Holding Register), utilizada para almacenar el estado de los fallos de memoria pendientes de resolución. Al ser un elemento hardware, esta estructura es finita, lo que determina el número de peticiones en vuelo por nodo.

La colocación de los nodos activos y pasivos en la red se realizará del siguiente modo: en las redes 3D, todos los nodos activos se encontrarán en una capa, y los pasivos en la otra. En la red 2D, se colocarán por columnas de manera alterna para optimizar el acceso, tal como se muestra en la siguiente representación de los cores y los bancos de memoria:

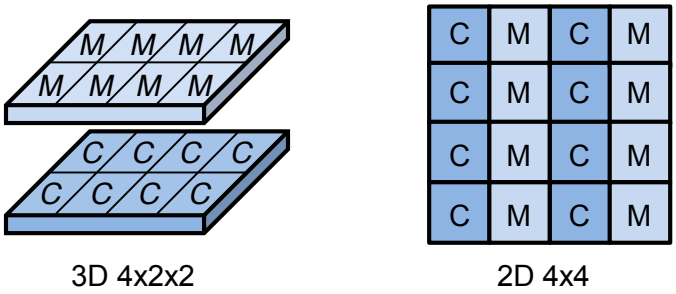

Figura 6.1: Organización de los nodos activos y pasivos (cores (C) y memoria (M) respectivamente) en las redes 3D y 2D.

Los nodos activos dispondrán de una ráfaga de peticiones que deben solventar. La simulación concluirá cuando todos los nodos hayan recibido respuesta a sus peticiones, siendo el tiempo total de simulación el dato utilizado en la comparación de las redes. Las diferentes ráfagas utilizadas en las pruebas son de 200, 400, 600, 800 y 1000 peticiones por nodo activo.

6.2.1 Comparación de todas las redes

Para estas pruebas se han realizado simulaciones con tráfico reactivo. El número de peticiones pendientes por nodo que se ha utilizado en las redes es el siguiente:

Red	3DP			3D			2D		
Dimensiones	4x2	6x3	8x4	4x2x2	6x3x2	8x4x2	4x4	6x6	8x8
Peticiones pendientes	16	14	14	16	14	14	8	8	14

Tabla 6.4: Número de peticiones pendientes máximas por nodo.

Los resultados de esta comparación se encuentran en la siguiente imagen:

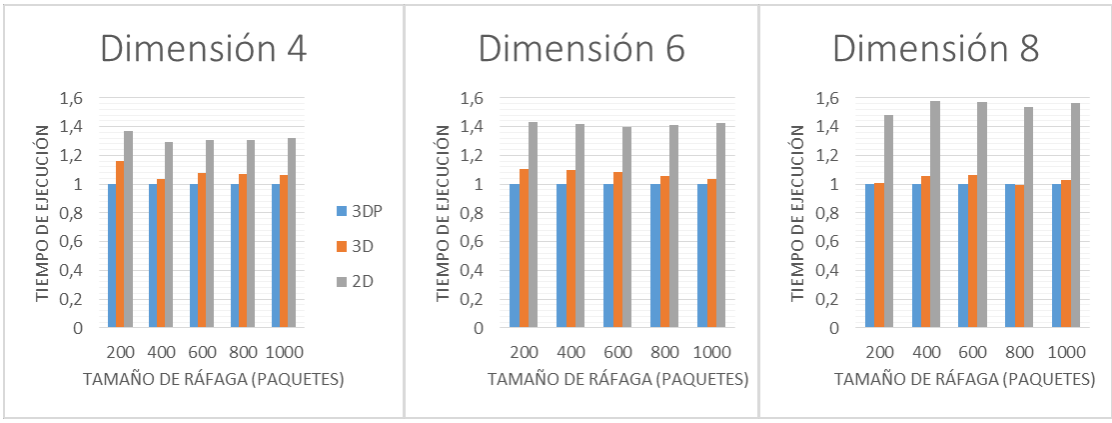

Figura 6.2: Tiempos de ejecución normalizados para las pruebas reactivas con las tres redes.

De los resultados anteriores pueden extraerse dos conclusiones:

- En primer lugar, la red 2D no puede igualar los tiempos de las redes tridimensionales, principalmente a causa de una mayor distancia media entre los nodos. Esta diferencia se incrementa conforme aumentan las dimensiones de la red, ya que el plegado de la misma es cada vez más beneficioso.
- En segundo lugar, el tiempo de ejecución de la red 3D se acerca cada vez más a la red 3DP, llegando a igualarse en algunas simulaciones con dimensión 8. Las pérdidas por reensamblado en la red 3D siguen existiendo, pero posee una ventaja añadida con respecto a la red 3DP: como se describió anteriormente, en la red 3D la inyección es simultánea, siendo necesario que esté disponible para todos los planos antes de que cualquiera de ellos pueda realizarla. Esto provoca que la inyección sea más restrictiva que en las redes que solo están formadas por un plano (como la red 3DP). Por lo tanto, hay menos paquetes en la red, lo que hace que avancen con mayor facilidad y reduce la latencia. En general, el tiempo total de envío de las ráfagas se ve reducido. Esta disminución oculta las pérdidas por reensamblado hasta el punto de compensarlas completamente.

6.2.2 Comparación de redes 3D

Se realizarán pruebas tanto con tráfico tanto modal como reactivo, incorporando en el caso modal el uso de los patrones dirigidos explicados anteriormente en el documento. No se han realizado pruebas reactivas con estos patrones, principalmente a causa de su naturaleza, ya que someten a la red a cargas muy exigentes. Esto provoca el desvío de ciertos paquetes hacia el camino de escape que conforman los routers LIGERO. La latencia de estos paquetes puede ser muy alta, y su aparición ocasional en la simulación de ráfagas reactivas hace que los resultados finales de éstas no sean fiables, existiendo una gran divergencia entre simulaciones idénticas que utilizan otra semilla de generación de números aleatorios.

Para las pruebas modales, se han utilizado los patrones uniforme, permutación, perfect shuffle y bit-reversal. En la Figura 6.3, la Figura 6.4 y la Figura 6.5 se puede observar la latencia media de los paquetes para cada una de las dimensiones de la red.

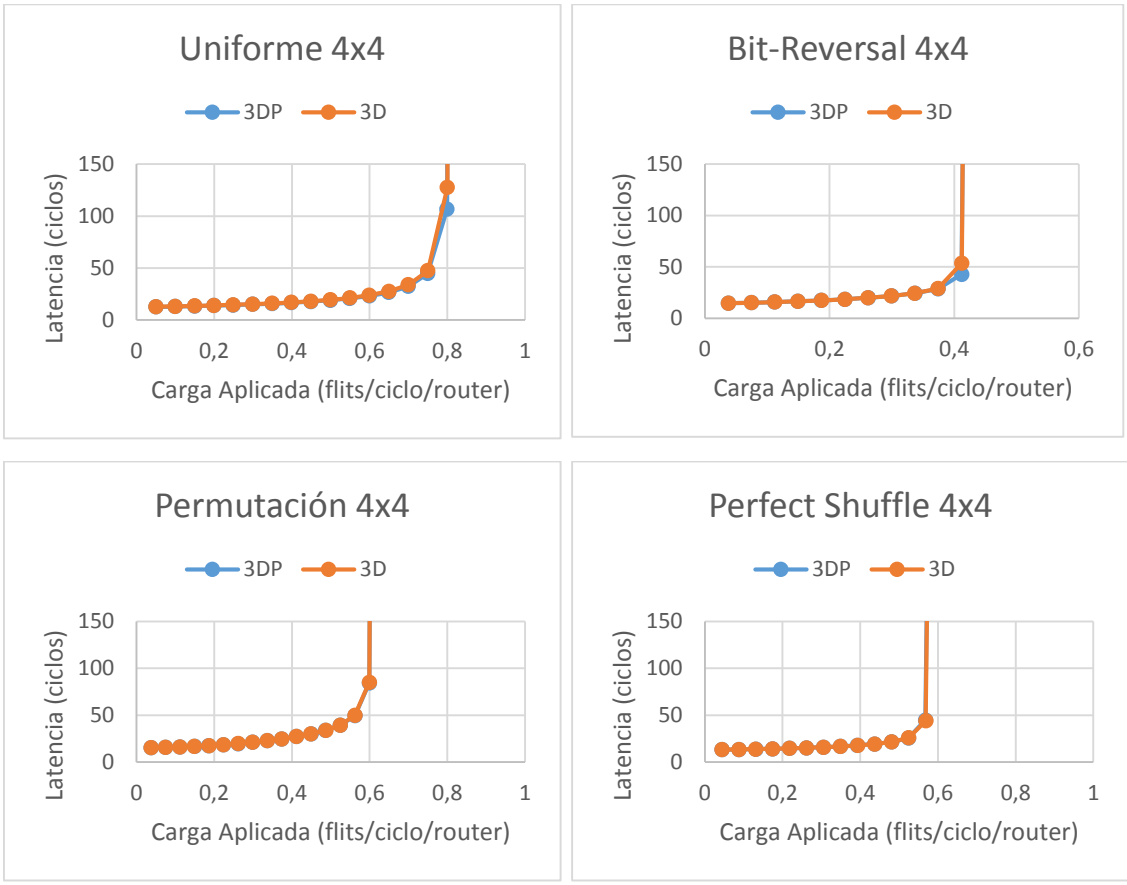

Figura 6.3: Resultados del tráfico modal para las redes con dimensión 4.

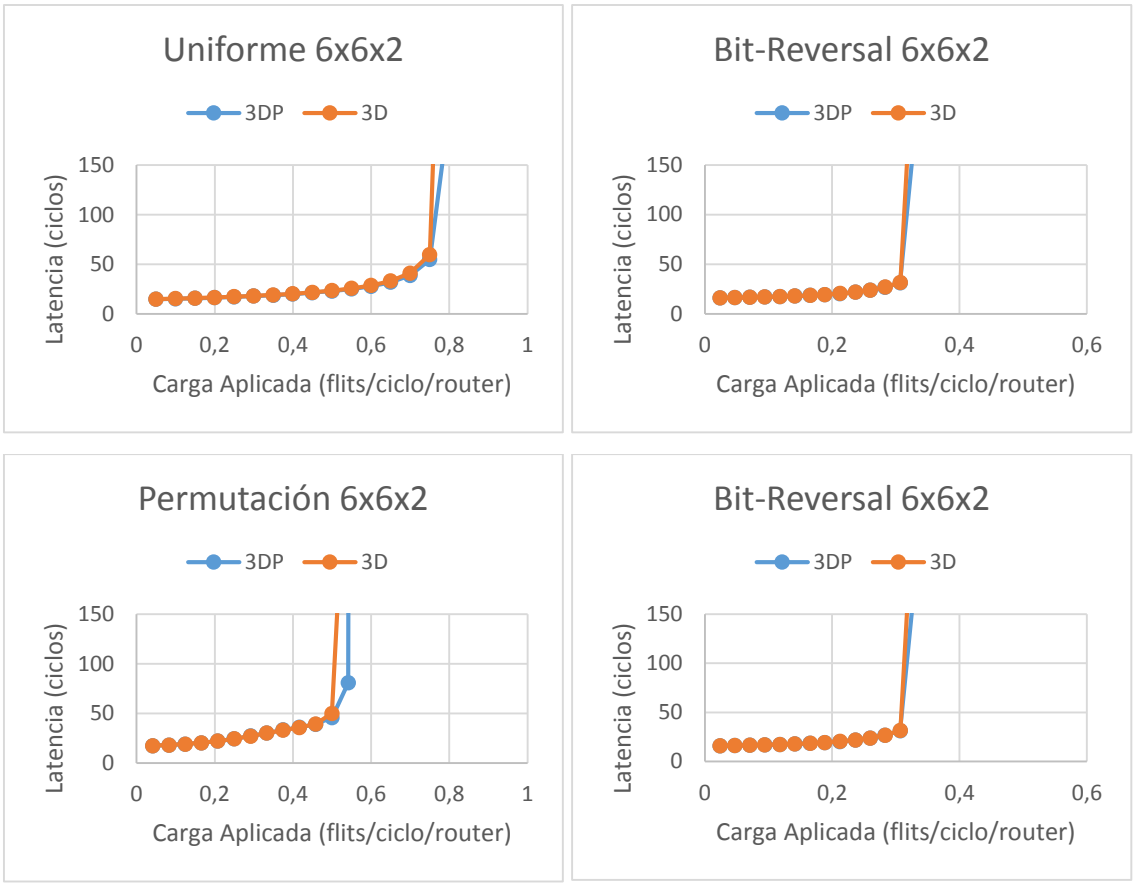

Figura 6.4: Resultados del tráfico modal para redes de dimensión 6.

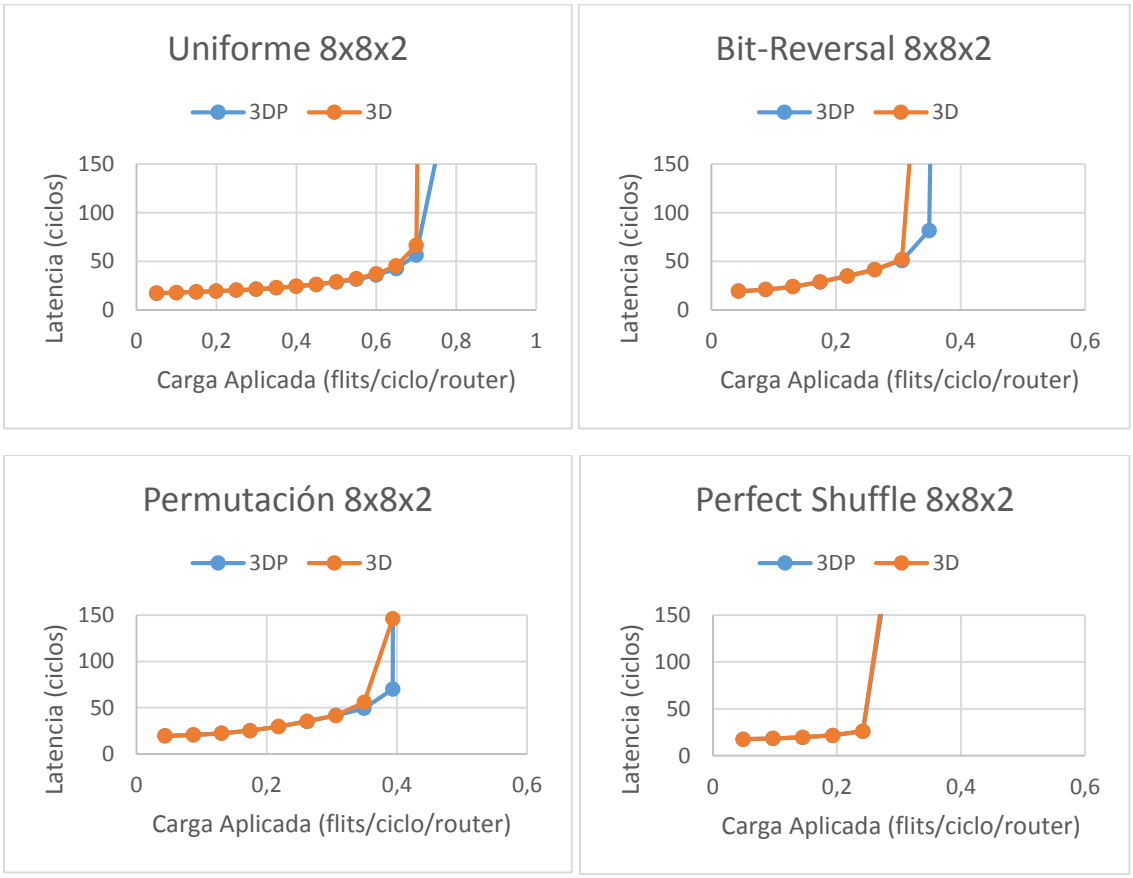

Figura 6.5: Resultados del tráfico modal para redes de dimensión 8.

Las conclusiones de estos resultados son las siguientes:

- Conforme aumenta el tamaño de la red, se observan mayores dificultades a la hora de transmitir los mensajes, lo que disminuye la carga constante que es capaz de admitir y se traduce en un aumento de latencia hasta alcanzar el punto de saturación.
- Con una carga gestionable por parte de la red no existen diferencias de latencia, ya que las porciones viajan de forma simultánea a destino sin que existan divergencias que las hagan llegar en instantes de tiempo diferentes.
- Cuando la carga aproxima a la red a saturación es cuando se observan las diferencias entre las redes provocadas por el desajuste de las particiones. Estos desajustes provocan que la latencia aumente ligeramente para el caso de la red 3D, pero al encontrarse en un punto tan cercano a la saturación no son realmente problemáticos.

Para las pruebas reactivas con tráfico uniforme, se han asignado estos límites de peticiones:

Red	3DP			3D		
Dimensiones	4x4	6x6	8x8	4x4x2	6x6x2	8x8x2
Peticiones pendientes	10	12	14	12	12	16

Tabla 6.5: Límite de solicitudes pendientes por inyector.

Los resultados se muestran en la siguiente figura:

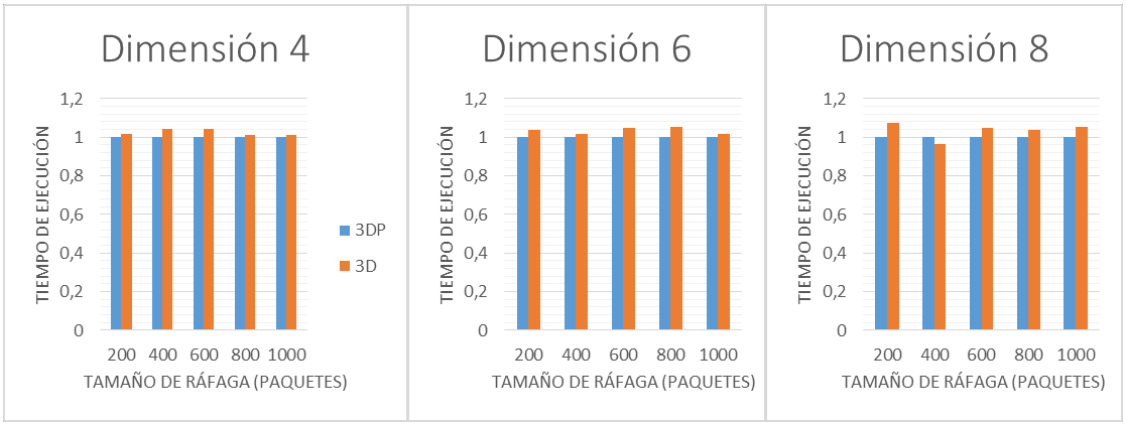

Figura 6.6. Comparación reactiva de redes 3DP y 3D.

De nuevo, los tiempos de ejecución de ambas redes son bastante similares, encontrándose algún caso en que la red 3D iguala e incluso supera a la red 3DP, como ya se ha mencionado a causa de poseer una inyección más restrictiva.

6.3 Aplicaciones

Por último, en este apartado se muestran los resultados correspondientes a la ejecución de aplicaciones reales sobre la red 3D implementada, que es comparada contra la red 2D. A causa de los *checkpoints* disponibles para las pruebas, solamente ha sido posible realizar pruebas para las redes de dimensión 4, utilizando un sistema con 8 procesadores y 8 bancos de cache compartida que se distribuyen en una red 4x2x2 en el caso de la arquitectura 3D y en una red 4x4 para la arquitectura 2D. Los tiempos obtenidos son los siguientes:

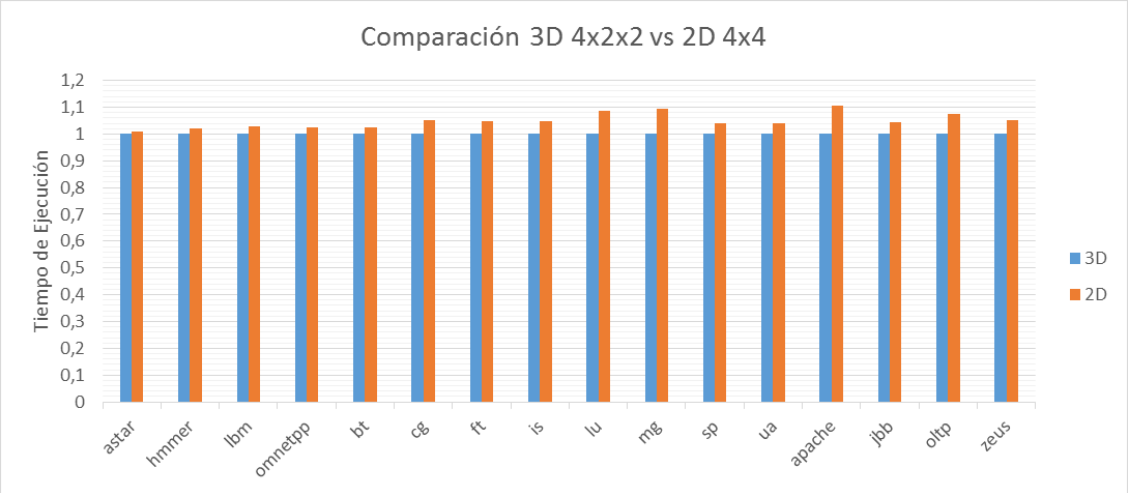

Figura 6.7: Comparación de redes 3D y 2D de dimensión 4 con aplicaciones reales.

El análisis de los resultados de cada suite por separado sería el siguiente:

- Para las aplicaciones SPEC (astar-omnetpp), se observa una moderada mejoría de un 1-2%. Esto se debe a que estas aplicaciones presentan un uso bastante reducido de la cache de último nivel, y por lo tanto de la red de interconexión.

- En el caso de NPB (bt-ua), se aprecia una mejora mayor, cercana al 10% para las aplicaciones “lu” y “mg”. Esta suite se ve más favorecida por la mejora de la red debido a su naturaleza paralela.
- Por último, las aplicaciones transaccionales (apache-zeus) también se ven favorecidas por la arquitectura de red, encontrándose el mejor resultado en la aplicación “Apache” con un 10,5% de mejora.

No obstante, la distancia media entre nodos en una red 4x2 y una red 4x4 es muy similar, lo que hace que no se obtengan grandes mejoras. Tal y como se comprobó en las pruebas sintéticas, la diferencia de rendimiento entre la red 3D y la red desplegada 2D es mayor conforme aumentan las dimensiones de la red, ya que la disminución de la distancia media es más significativa. Por ello, la mejora introducida por esta arquitectura de red sería mayor si se utilizaran redes de dimensiones superiores (por ejemplo, las dimensiones 6 y 8 que se han utilizado en las anteriores pruebas).

7 Conclusiones

Con este trabajo se ha conseguido diseñar una arquitectura de red tridimensional capaz de gestionar la comunicación entre planos de una forma eficiente sin requerir un número desorbitado de enlaces verticales, fundamentalmente gracias a la autonomía que poseen cada uno de estos planos para la toma de decisiones.

La inclusión del router LIGERO en la red ha permitido solventar los problemas causados por la utilización del sistema de particionado en base a su versatilidad a la hora de responder ante situaciones de congestión o proclives al bloqueo.

Este diseño ha sido incluido en un simulador de redes en chip, lo que ha permitido llevar a cabo una evaluación de su rendimiento. Se han realizado pruebas con tráfico sintético básico y simulaciones más complejas con el objetivo de representar situaciones reales, comprobándose que se conserva gran parte del rendimiento que LIGERO presenta en una red bidimensional, y que la respuesta de la red 3D mejora frente a una red 2D que se encuentre en las mismas condiciones. Esta mejora es más apreciable conforme aumentan las dimensiones de la red, alcanzando hasta un 40% de mejoría al utilizar redes de dimensión 8.

En cuanto a la simulación del sistema completo, para aquellas aplicaciones en las que había una utilización mayor de la red se obtuvo una mejora del tiempo de ejecución, alcanzando hasta el 10%.

Lo que puede extraerse de estos resultados es que, si se superan las restricciones de producción existentes, el apilado vertical para la creación de chips multicapa se postula como una solución interesante para obtener sistemas que aglutinen un mayor número de componentes, y que por tanto ofrezcan un rendimiento mayor al existente en los sistemas bidimensionales actuales.

8 Trabajo futuro

En primer lugar, se debe estudiar la escalabilidad de esta arquitectura para su utilización en redes de más de dos capas. En ese caso, las TSV se comparten entre todos los nodos de la vertical, siendo necesario incluir un arbitraje para su utilización, así como determinar si el particionado sigue siendo razonable con un número de capas superior o si debe ser sustituido con tránsitos dinámicos en el trayecto de los paquetes de origen a destino.

Por otra parte, el diseño de red planteado no es un fin en sí mismo, sino que se trata del primer paso de un objetivo más amplio. Como se mencionó en la sección introductoria del apilado vertical, el método de fabricación wafer-on-wafer (apilado de obleas previo al corte) presenta problemas causados por la multiplicación de la probabilidad de error en un chip al aumentar el número de capas. En una red en chip que utilice routers convencionales, la no disposición de alguno de los enlaces supondría posibles problemas de interbloqueo o de imposibilidad de comunicación. En cambio, las capacidades internas de LIGERO permitirían su funcionamiento incluso aunque existieran ciertos fallos asumibles en la red.

Por ello, el trabajo posterior a este proyecto se encamina hacia la definición de redes tolerantes a fallos, tanto en los enlaces de comunicación como en los nodos del sistema, de forma similar a lo mostrado en la siguiente figura:

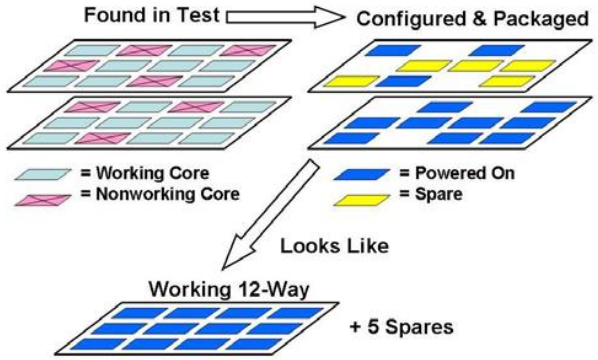

Figura 8.1: Sistema de doble capa que ofrece 12 núcleos operativos.

En la imagen se observa un chip formado por dos capas, que contiene doce núcleos en cada capa. Sin embargo, en las pruebas se detecta que algunos de ellos no se encuentran operativos, pero al menos existe uno que funciona en cada vertical. Si el sistema pudiera tolerar los fallos en los componentes y en la red de interconexión, en lugar de disponer de un sistema con 24 núcleos que posee defectos de fabricación y debe ser desechado, se obtiene un chip que en sus especificaciones posee 12 núcleos funcionales, y además cuenta con una serie de núcleos “spare” o de recambio que podrían utilizarse para algún cometido extra.

Esta clase de permisividad ante la aparición de fallos supondría un aumento importante en el porcentaje de retorno de chips multicapa, y permitiría que técnicas como el wafer-on-wafer fueran viables en la fabricación. La unión de obleas completas no presenta unos requisitos mecánicos tan elevados, pudiendo utilizarse TSVs de menor grosor y, por lo tanto, aumentaría el número de enlaces que se pueden realizar en la misma área, incrementando la densidad de las conexiones verticales.

9 Bibliografía

[1] W. J. Dally and B. Towles, “Route packets, not wires: on-chip inteconnection networks,” in *Proceedings of the 38th conference on Design automation - DAC ’01*, 2001, pp. 684–689.

[2] G. E. Moore, “Cramming more components onto integrated circuits, Reprinted from Electronics, volume 38, number 8, April 19, 1965, pp.114 ff.,” *IEEE Solid-State Circuits Newsl.*, vol. 20, no. 3, pp. 33–35, Sep. 2006.

[3] N. Jerger and L. Peh, “On-chip networks,” *Synth. Lect. Comput. Archit.*, 2009.

[4] “Intel® Xeon Phi™ Coprocessor - the Architecture | Intel® Developer Zone.” [Online]. Available: https://software.intel.com/es-es/articles/intel-xeon-phi-coprocessor-codename-knights-corner. [Accessed: 14-Jul-2014].

[5] “TILE PROCESSOR ARCHITECTURE OVERVIEW FOR THE TILEPRO SERIES.” [Online]. Available: http://www.tilera.com/scm/docs/UG120-Architecture-Overview-TILEPro.pdf. [Accessed: 14-Jul-2014].

[6] P. Emma, A. Buyuktosunoglu, M. Healy, K. Kailas, V. Puente, R. Yu, A. Hartstein, P. Bose, and J. Moreno, “3D stacking of high-performance processors,” *(HPCA), 2014 IEEE 20th Int. Symp. High Perform. Comput. Archit.*, pp. 500–511, 2014.

[7] P. Abad, P. Prieto, L. G. Menezo, A. Colaso, V. Puente, and J. A. Gregorio, “Interaction of NoC Design and Coherence Protocol in 3D-Stacked CMPs,” in *2013 Euromicro Conference on Digital System Design*, 2013, pp. 48–55.

[8] D. Park, S. Eachempati, and R. Das, “MIRA: A multi-layered on-chip interconnect router architecture,” *Int. Symp. Comput. Archit.*, pp. 251–261, 2008.

[9] D. H. Kim, K. Athikulwongse, M. Healy, M. Hossain, M. Jung, I. Khorosh, G. Kumar, Y.-J. Lee, D. Lewis, T.-W. Lin, C. Liu, S. Panth, M. Pathak, M. Ren, G. Shen, T. Song, D. H. Woo, X. Zhao, J. Kim, H. Choi, G. Loh, H.-H. Lee, and S. K. Lim, “3D-MAPS: 3D Massively parallel processor with stacked memory,” in *2012 IEEE International Solid-State Circuits Conference*, 2012, pp. 188–190.

[10] D. Fick, R. G. Dreslinski, B. Giridhar, G. Kim, S. Seo, M. Fojtik, S. Satpathy, Y. Lee, D. Kim, N. Liu, M. Wieckowski, G. Chen, T. Mudge, D. Sylvester, and D. Blaauw, “Centip3De: A 3930DMIPS/W configurable near-threshold 3D stacked system with 64 ARM Cortex-M3 cores,” in *2012 IEEE International Solid-State Circuits Conference*, 2012, pp. 190–192.

[11] “Sony’s PS Vita Uses Chip-on-Chip SiP – 3D, but not 3D | Chipworks Blog,” 2012. [Online]. Available: http://www.chipworks.com/en/technical-competitive-analysis/resources/blog/sonys-ps-vita-uses-chip-on-chip-sip-3d-but-not-3d/. [Accessed: 16-Jun-2014].

[12] T. P. Morgan, “Nvidia to stack up DRAM on future ‘Volta’ GPUs • The Register,” 2013. [Online]. Available: http://www.theregister.co.uk/2013/03/19/nvidia_gpu_roadmap_computing_update/. [Accessed: 16-Jun-2014].

[13] Joel Hruska, “Intel’s next-gen Xeon Phi will be 3x faster, include next-gen Hybrid Memory Cube tech | ExtremeTech,” 2014. [Online]. Available: http://www.extremetech.com/extreme/185007-intels-next-gen-xeon-phi-will-be-3x-faster-include-next-gen-hybrid-memory-cube-tech. [Accessed: 20-Jul-2014].

[14] P. Abad, P. Prieto, L. G. Menezo, A. Colaso, V. Puente, and J.-Á. Gregorio, “TOPAZ: An Open-Source Interconnection Network Simulator for Chip Multiprocessors and Supercomputers,” in *2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*, 2012, pp. 99–106.

[15] M. M. K. Martin, D. J. Sorin, B. M. Beckmann, M. R. Marty, M. Xu, A. R. Alameldeen, K. E. Moore, M. D. Hill, and D. A. Wood, “Multifacet’s general execution-driven multiprocessor simulator (GEMS) toolset,” *ACM SIGARCH Comput. Archit. News*, vol. 33, no. 4, p. 92, Nov. 2005.

[16] A. R. Alameldeen, M. M. K. Martin, C. J. Mauer, K. E. Moore, M. D. Hill, D. A. Wood, and D. J. Sorin, “Simulating a $2M commercial server on a $2K PC,” *Computer (Long. Beach. Calif).*, vol. 36, no. 2, pp. 50–57, Feb. 2003.

[17] J. Y. H. Jin, M. Frumkin, “The OpenMP Implementation of NAS Parallel Benchmarks and its Performance,” *NAS Tech. Rep. NAS-99-011*, 1999.

[18] “SPEC - Standard Performance Evaluation Corporation.” [Online]. Available: http://www.spec.org/. [Accessed: 30-Aug-2014].

[19] B. P. T. William James Dally, *Principles and Practices of Interconnection Networks (The Morgan Kaufmann Series in Computer Architecture and Design)*. 2004.

[20] C. Fallin, C. Craik, and O. Mutlu, “CHIPPER: A low-complexity bufferless deflection router,” in *2011 IEEE 17th International Symposium on High Performance Computer Architecture*, 2011, pp. 144–155.

[21] M. Hayenga, N. E. Jerger, and M. Lipasti, “SCARAB,” in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture - Micro-42*, 2009, p. 244.

[22] P. Abad, V. Puente, and J.-A. Gregorio, “LIGERO: A light but efficient router conceived for cache-coherent chip multiprocessors,” *ACM Trans. Archit. Code Optim.*, vol. 9, no. 4, pp. 24–45, Jan. 2013.