

Diseño e Implementación de un Multiprocessor Systems-on-Chip (MPSoC) Interconectado por una Networks-on-Chip (NoC)

Wilson Chicaiza
Departamento de Eléctrica y Electrónica
Escuela Politécnica del Ejército (ESPE)
Sangolquí, Ecuador
cmoswilson@gmail.com

Daniel Verdezoto
Departamento de Eléctrica y Electrónica
Escuela Politécnica del Ejército (ESPE)
Sangolquí, Ecuador
dgverdezoto@gmail.com

Resumen— En el presente documento se presenta una breve caracterización de los medios de comunicación empleados en arquitecturas multiprocesadas. Esta caracterización tiene como objetivo principal el mostrar un nuevo modelo de comunicación basado en conmutación de paquetes a los cuales se les denomina como Networks-On-Chip (NoC). Esta publicación muestra una arquitectura de red llamada NoC Hermes, la cual fue interconectada a un Multiprocessor-Systems-on-Chip (MPSoC) compuesto de cuatro procesadores MicroBlaze. Esta conexión se la realizó gracias al diseño y desarrollo de una Interfaz de Red generada en código VHDL. Por medio de la Interfaz de Red se consiguió que los procesadores MicroBlaze interactúen con los Switches de Hermes a fin de crear una arquitectura multiprocesada interconectada por una NoC. Con el motivo de realizar comparaciones también se creó otra arquitectura de multiprocesadores interconectados por buses. Para ambas arquitecturas se desarrolló una aplicación de Esteganografía en la que existe multiprocesamiento de dos procesadores trabajando simultáneamente. Lamentablemente sobre dicha aplicación no fue posible medir directamente la latencia y el consumo de energía, razón por la cual se utilizó simuladores que permitieron estimar dichas mediciones.

Palabras Claves; *Buses, System-on-Chip (SoC), Multiprocessor System-on-Chip (MPSoC), IP-Core (Bloque de Propiedad Intelectual), Field Programmable Gate Array (FPGA), Networks-on-Chip (NoC), Embedd Development Kit (EDK), Xilinx Platform Studio (XPS), Software Development Kit (SDK), Network Interface (NI), Network Element (NE), Process Element (PE), Fast Simple Link (FSL), Digital Video Interface (DVI), Compact Flash (CF), System ACE Interface Controller (SYSACE), Thin Film Transistor (TFT), Multi-Port Memory Controller (MPMC), MicroBlaze (MB o μ B), Xilinx Caché Link (XLC).*

I. INTRODUCCIÓN

En la actualidad, el nivel de integración en los circuitos integrados está creciendo rápidamente gracias a la utilización de nuevas técnicas y herramientas de diseño e implementación, lo que ha permitido disminuir la brecha entre la capacidad de integración de transistores dentro de un chip y el diseño de los mismos. Sin embargo, aparecen nuevos desafíos para los diseñadores de dispositivos electrónicos modernos que requieren de la implementación de sistemas completos dentro

de un circuito integrado SoC [1]. Entre los principales retos a los que se enfrentan, se pueden nombrar la disipación de potencia, la disminución de la latencia y el aumento del ancho de banda en la comunicación intrachip.

Para satisfacer los requerimientos del mercado actual, se utilizan sistemas embebidos basados en SoC que incorporan un solo procesador o chip maestro. A fin de solventar una aplicación de alto nivel computacional, se utilizaban varios SoCs que debían compartir el control y procesamiento general de la aplicación creándose un problema de paralelismo, desperdicio de transistores, consumo de potencia y limitación física. De allí que aparece la necesidad de desarrollar los MPSoC.

El MPSoC es un SoC compuesto por múltiples procesadores, memorias y circuitería especializada, interconectados a través de una infraestructura de comunicaciones.

Tradicionalmente, los MPSoC se han comunicado a través de arquitecturas basadas en buses. Lamentablemente, los buses no satisfacen los requerimientos actuales debido a que la convergencia de aplicaciones demanda la mezcla de varios tipos de tráfico en el mismo SoC. Es así que al compartir el medio de transporte, se requiere manejar el concepto de QoS. Por tanto, las arquitecturas modernas exigen una estructura flexible, reconfigurable y reprogramable. Esta exigencia ha dado paso al nacimiento de un nuevo paradigma de interconexión, el empleo de Networks-On-Chip. [2]

En este ámbito, la arquitectura NoC se proyecta como una solución prometedora en la comunicación On Chip. Características como la eficiencia energética, fiabilidad, escalabilidad, ancho de banda y reutilización, son factores que sustentan su desarrollo. Cabe señalar que, las NoC emplean teorías y métodos de comunicación por medio de redes de conmutación de paquetes dentro del chip, introduciendo notables mejoras. Los módulos intercambian información entre sí gracias a la intervención de las NoC que establecen una red como un subsistema de transporte público para el tráfico de información. De esta forma la información pasa por varios enlaces, de acuerdo a las decisiones que se tomen dentro del algoritmo de la NoC. [3]

II. MULTIPROCESSOR SYSTEMS-ON-CHIP

Un MPSoC representa una rama distinta de multiprocesadores tradicionales que cumplen con requerimientos específicos de aplicaciones embebidas que utilizan varios procesadores interconectados junto con otros subsistemas de hardware implementados en un solo sistema. [4]

Un MPSoC incorpora varios elementos de procesamiento necesarios para una aplicación que utiliza multiprocesadores programables como parte de los elementos de un sistema, unidos por una estructura de interconexión, ajustándose a las necesidades y evolución de la arquitectura de computadores, tales como: operación en tiempo real, consumo de baja potencia y aplicaciones multitarea. [5]

A. Tipos de MPSOC

En el estudio de MPSoC se argumenta la formación de dos importantes familias de multiprocesadores: La familia de Multiprocesadores Homogéneos cuyo primer prototipo fue Lucent Daytona [6] [4] y la familia de Multiprocesadores Heterogéneos, en la que destaca el C-5 Network Processor [7], Nexperia y OMAP [8].

TABLA 1. COMPARACIÓN DE MPSOC HOMOGÉNEOS Y HETEROGÉNEOS.

Homogéneos	Heterogéneos
Son conjuntos de procesadores basados en el modelo de arquitectura computacional paralela.	Están conformados por procesadores interconectados con diferentes funcionalidades.[6]
Con la arquitectura paralela se divide el tiempo de ejecución de cada recurso, para hacer más eficiente el tiempo de procesamiento.[7]	Cada uno de los procesadores está dedicado a un propósito específico, como aceleradores de hardware, procesadores digitales de señales, periféricos, etc.
Los MPSoC Homogéneos son utilizados comúnmente en consolas de videojuegos, servidores y computadores de escritorio.	Los MPSoC Heterogéneos tienen aplicaciones en sistemas portátiles y en sistemas integrados.

B. Arquitecturas MPSOC

La gran capacidad de implementar varios procesadores dentro de un mismo diseño, permite desarrollar diversas arquitecturas dentro de los MPSoC, con la finalidad de aumentar el rendimiento de una aplicación específica.

TABLA 2. DESCRIPCIÓN DE LA ARQUITECTURAS EXISTENTES EN MPSOC.

Arquitecturas de MPSoCs	Descripción

1. Procesadores no interconectados.	Incluyen varios procesadores pero cada uno se dedica a una tarea independiente y no existe comunicación entre ellos. Pudiendo ser el caso que los procesadores sean diferentes.
2. De propósito general y específico.	Son MPSoC interconectados en los que se hace necesario la ejecución de tareas de forma conjunta y el intercambio de información entre los procesadores, mediante una memoria local (2) o a una compartida (3).
3. Memoria Compartida.	
4. Configuración Maestro – Esclavo.	Todos los procesadores esclavo ejecutan el mismo código y es el procesador maestro el encargado de recibir la información y repartirla entre los esclavos para su procesamiento.
5. Segmentada.	Cada procesador realiza una parte de la tarea y pasa sus resultados al siguiente procesador. Esta arquitectura se puede utilizar en aplicaciones en donde no existe paralelismo, sin embargo pueden dividir las tareas entre los procesadores de forma independiente.[13]

III. INTERCONEXIÓN EN UN MPSOC

La arquitectura de comunicación en un MPSoC es uno de los principales retos tecnológicos en la actualidad debido al aumento de frecuencias de operación y el incremento del tamaño del chip. Por lo cual la dependencia de rendimiento en el diseño de un MPSoC recae en la eficiencia de su estructura de comunicación [9]. Por ende el presente documento presenta a continuación dos modelos de MPSoC, una interconectada por buses FSL y la otra interconectada por una NoC, que fueron diseñados para compararse entre sí y demostrar cual se desempeña mejor, a través de resultados obtenidos en latencia y disipación de potencia por medio de simulaciones.

Cabe mencionar que todo el trabajo de creación del MPSoC se lo realizó en un FPGA Virtex 6 ML605, que posee un encapsulado XC6VLX240T-1FFG1156, y que además incluye el software EDK 13.2 para el co-diseño de hardware y software, el cual provee un entorno flexible para el diseño de aplicaciones de alto rendimiento. [10][11]

A. Interconexión mediante buses FSL

La implementación de buses para la interconexión de IP-Cores dentro un MPSoC es la más simple y ampliamente utilizada. De la gran variedad existente en buses se utiliza el FSL, que es un bus que se emplea para la comunicación dentro

de un MPSoC, debido a que puede soportar efectivamente topologías como mesh, anillo y estrella, a medida que se incrementa el número de procesadores en el FPGA [12]. Se menciona además que este bus es utilizado también para la comunicación de IP-Cores de diseño específico (dentro del XPS de Xilinx) como se verá más adelante en la interconexión de la NoC.

La arquitectura del hardware que ha sido constituido se detalla en la Figura 1 en donde se puede apreciar la interconexión de los elementos necesarios para cumplir con la aplicación de Esteganografía, de la cual se hablará posteriormente.

Esta arquitectura consta de 4 procesadores MicroBlaze comunicados entre sí por buses FSL. Destaca en esta arquitectura la Interfaz MPMC, que es un IP-Core que permite la comunicación entre los procesadores con una memoria externa, en este caso una DDR3 de 512MB, soporta hasta 8 conexiones (un par para cada MicroBlaze) de manera personalizada, haciendo uso de la interfaz XCL o bien por conexión directa al bus PLB [12] [14]. Por XCL se asigna un acceso de memoria directa para el procesador y por bus PLB la memoria se comparte para todos los procesadores.

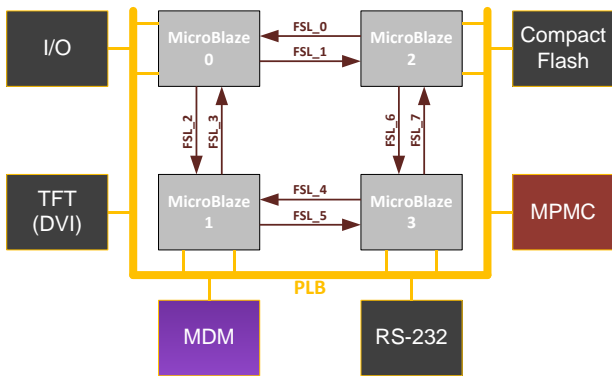


Figura 1. Arquitectura de 4 procesadores embebidos interconectados por Buses FSL para aplicación de Esteganografía

Los componentes más relevantes en esta arquitectura además del MPMC, son el TFT y el lector de CF, los cuales se detallan a continuación:

- El IP-Core TFT, es el medio por el cual se controla una pantalla LCD o CTR, consta de una conexión maestra conectada de forma independiente hacia la memoria DDR3 y una conexión esclava conectada al bus PLB para adquirir la información de los píxeles que serán proyectados. El TFT además trabaja a una frecuencia de 25MHz con una resolución de 640x480 píxeles y tolera conexiones para DVI y VGA [15].
- El lector de CF o SYSACE, es un IP-Core flexible que permite obtener información de una memoria CF [16]. El SYSACE usualmente es empleado para verificar que el FPGA está funcionando de manera correcta al tener acceso a archivos ejecutables ubicados dentro de

la CF que permiten realizar un test de periféricos. Para el caso de la arquitectura mostrada en la Figura 1 el módulo SYSACE fue adaptado para poder adquirir información necesaria para la aplicación de Esteganografía.

- Los demás IP-Cores que también forman parte de la arquitectura son: dos General Purpose IO (GPIO's) constituidos por los Led 8 Bits y Dip-Switchs, un RS232_Uart16550 y un Microblaze Debug Module (MDM). Los dos últimos IP-Cores son elementos indispensables en cualquier diseño, pues el RS232 permite una comunicación entre el computador y el FPGA [17], y el MDM provee soporte para depuraciones en el software que se está desarrollando desde uno hasta ocho procesadores MicroBlaze [18].

B. Interconexión mediante NoC Hermes

En la Figura 2 se detalla la interconexión de los cuatro MicroBlaze interconectados por NoC Hermes. El proceso para el desarrollo de esta arquitectura es similar a la realizada en la conexión de buses, salvo por el IP-Core que fue desarrollado por la Facultad de Informática de la Pontificia Universidad Católica de Rio Grande (Brasil) y adaptado a través de una NI desarrollada en la Escuela Politécnica del Ejército.

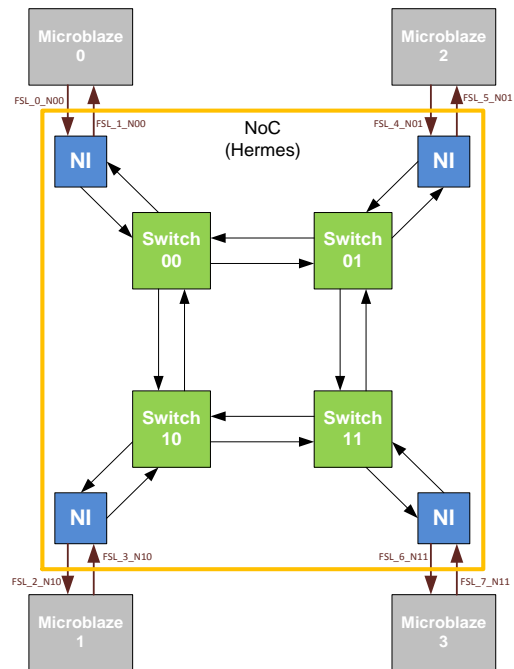


Figura 2. Arquitectura de 4 de procesadores interconectados por una NoC para aplicación de Esteganografía

1) Diseño y desarrollo de una NI para NoC Hermes

Para el planteamiento del diseño se parte del modo de operación de los elementos involucrados. Uno es el medio por el cual se puede transmitir información desde el MicroBlaze

hacia un IP-Core cualquiera, este medio es el bus FSL, y el otro es el Switch Hermes [19] al que le hace falta una interfaz de red que le permita interactuar con el procesador a través del mencionado bus.

a) Operación FSL

FSL es un bus unidireccional de punto a punto utilizado para aplicaciones desarrolladas en FPGA's. La interfaz FSL puede ser habilitada desde el propio MicroBlaze y se lo utiliza para transferir datos hacia y desde los archivos de registro del procesador al hardware implementado. [20]

Los puertos de entrada y salida del bus FSL que son utilizados en el diseño se especifican en la siguiente tabla.

TABLA 3. PUERTOS DEL FSL NECESARIOS PARA EL DISEÑO.

Fast Simplex Link (FSL)			
Maestro	I/O	Esclavo	I/O
FSL_M_DATA	In	FSL_S_DATA	Out
FSL_M_WRITE	In	FSL_S_READ	In
FSL_M_FULL	Out	FSL_S_EXISTS	Out

De la tabla anterior se analiza el modo de operación, empezando por el Maestro, en donde la forma de transmisión de datos (FSL_M_DATA) se la realiza cuando se activa en alto el FSL_M_WRITE, y mientras se mantenga así los datos se mantendrán circulando a través del FSL Maestro hasta que el FSL_M_WRITE vuelva a un nivel bajo o se emita una señal en alto por el FSL_M_FULL indicando que el bus está lleno [20].

En el Esclavo los datos (FSL_S_DATA) se trasladan cuando el FSL_S_EXISTS se activa en alto y únicamente bajo esta condición se puede activar el FSL_S_READ, caso contrario si no existen datos en el bus, el FSL_S_READ se mantendrá en un nivel bajo [20].

b) Operación Switch Hermes

La topología de Hermes es una 2D-Mesh que emplea control de flujo por Handshake y técnica de ruteo por algoritmo XY para la realización de la transmisión de información desde un nodo de origen hacia un nodo de destino al conmutar los paquetes por medio de la técnica Wormhole. [19]

La distribución de entradas y salidas presentes en el puerto local del Switch Hermes se muestran en la siguiente tabla.

TABLA 4. PUERTOS DEL SWITCH HERMES NECESARIOS PARA EL DISEÑO.

Switch Hemes (Local Port)			
Receptor	I/O	Transmisor	I/O
rxlocal	In	txlocal	Out
data_inlocal	In	data_outlocal	Out
ack_rxlocal	Out	ack_txlocal	In

En el nodo de origen se emplea Handshake a través de rx y ack_rx, en donde cada flanco positivo va acompañado de un

flit del paquete de datos por data_in. Este paquete va encabezado por la dirección de destino y por el tamaño del paquete. [19]

Entonces cada vez que se hace una transmisión se pone en alto la señal rx y a través de ack_rx se confirma que el dato transmitido por data_in ha sido enrutado, así sucesivamente hasta que todo el paquete de datos haya sido enviado. [19]

Los datos al ser receptados en el nodo de destino presentan el mismo comportamiento, es decir que tx se activa en alto cada vez que llega un nuevo dato, del mismo modo la recepción se confirma por medio de ack_tx. La recepción de los flits finaliza una vez que se haya receptado todo el paquete [19].

c) Desarrollo de la Network Interface

Entre todas las posibles arquitecturas que pueden existir en diferentes literaturas que tratan sobre NoC, se puede distinguir la presencia de tres principales componentes que lo conforman, los cuales son los elementos de red (Networks Elements, NE) (también llamados routers o switches), las interfaces de red (Networks Interface, NI) (llamados también adaptadores de red) y los nodos o elementos de proceso (Process Elements, PE) [21]. De estos componentes, el interés se centra en el NI puesto que es un elemento faltante que tuvo que ser creado para la arquitectura de red que aquí se expone.

Dentro de las herramientas de co-diseño de hardware y software (EDK) se encuentra incluido también el ISE, el cual fue utilizado para programar el VHDL del NI que fue diseñado a partir de diagramas de estados, para poder crear las máquinas secuenciales que conforman el NI que se muestra en la siguiente figura.

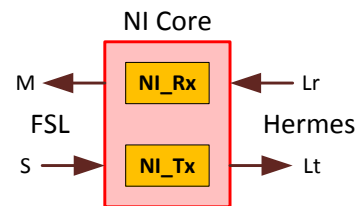


Figura 3. Módulos internos del NI.

En la Figura 3 se puede observar que el NI consta de dos módulos internos NI_Rx y NI_Tx. Estos módulos son las máquinas secuenciales encargadas de administrar la información que proviene tanto de los Switches de Hermes como de los MicroBlaze's. Cada uno de estos subsistemas tiene terminaciones que se conectan al maestro (M) y al esclavo (F) del FSL, y al puerto local de recepción (Lr) y transmisión (Lt) de la NoC.

En la siguiente tabla se puede apreciar todas las entradas y salidas listas para ser conectadas al MicroBlaze (por medio del FSL) y a Hermes (Por medio del Puerto Local), completando así la arquitectura red.

TABLA 5. PUERTOS DE LA NETWORK INTERFACE

Network Interface			
FSL	I/O	Local Port	I/O
FSL_S_DATA	In	Data_out_Local	Out
FSL_S_EXISTS	In	txLocal	Out
FSL_S_READ	Out	Ack_txLocal	In
FSL_M_DATA	Out	Data_in_Local	In
FSL_M_FULL	In	rxLocal	In
FSL_M_WRITE	Out	Ack_rxLocal	Out

2) Interconexión Procesador – Noc a Través de la Network Interface

El NI aquí descrito es integrado a NoC Hermes dentro de la plataforma ISE como un todo, antes de ser importado al XPS, que es el software en donde se encuentra el MPSoC. Ya en el XPS la integración del nuevo IP-Core con el MPSoC es más sencilla.

El proceso de importación se lo realiza desde el XPS al utilizar la herramienta Create or Import Peripheral Wizard, el cual permite a diseños realizados en el ISE tomar la forma de un IP-Core mucho más flexible para su interacción con otros IP-Cores [22].

IV. APLICACIÓN DE ESTEGANOGRAFÍA

La Esteganografía trabaja con meta información, es decir información dentro de otra información y ha sido empleada desde tiempos remotos dentro de la vida del ser humano. En donde a partir de una imagen portadora se ocultan caracteres ASCII codificados en binario dentro de los pixeles que constituyen una imagen. [23]

Para el desarrollo de la aplicación se ha seleccionado el formato PPM por ser ideal para trabajos relacionados con procesamiento de imágenes [24] [25] [26]. Su utilización es sencilla así como la programación realizada para adquirir su información. Además se mantiene bajo los recursos de memoria lo cual lo hace apto para ser empleado en el FPGA Virtex-6.

La aplicación de esteganografía se ejecuta entre cada par de procesadores de forma simultánea. A continuación se explica este proceso.

En la Figura 4 se aprecia una arquitectura de cuatro procesadores MicroBlaze comunicados por buses FSL. El MicroBlaze 0 y el MicroBlaze 2 realizan el proceso de esteganografía de forma paralela sobre dos imágenes distintas. Mientras que el MicroBlaze 1 y el MicroBlaze 3 realizan el proceso de decodificación de forma paralela para extraer la información contenida en las imágenes.

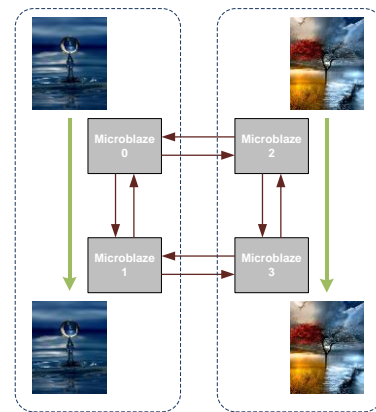


Figura 4. Proceso de Esteganografía y de Decodificación de mensaje sobre arquitectura MPSoC interconectada por buses FSL.

En la Figura 5 se puede apreciar una arquitectura de cuatro procesadores MicroBlaze comunicados por NoC Hermes. El MicroBlaze 0 y el MicroBlaze 2 realizan el proceso de esteganografía de forma paralela sobre dos imágenes distintas. Mientras que el MicroBlaze 1 y el MicroBlaze 3 realizan el proceso de decodificación de forma paralela para extraer la información contenida en las imágenes.

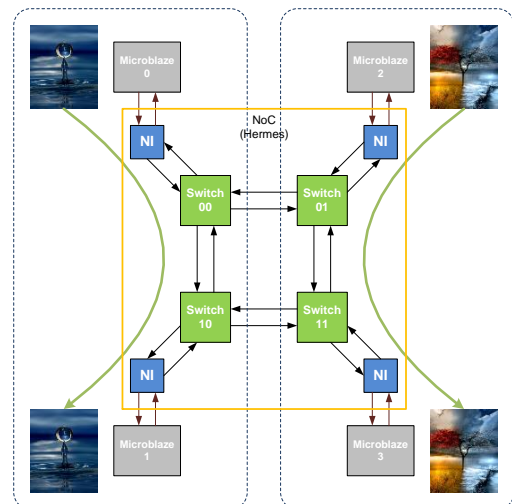


Figura 5. Proceso de Esteganografía y de Decodificación de mensaje sobre segunda arquitectura

En ambas arquitecturas se interrumpe el paralelismo cuando los MicroBlaze se comunican entre sí para acceder a recurso compartidos del FPGA. Los procesadores comparten el acceso a memoria CF para la adquisidor de imágenes y mensajes, y la proyección de resultados en un monitor a través del puerto DVI.

V. ANÁLISIS DE RESULTADOS

Las dimensiones de las imágenes empleadas en la aplicación de esteganografía son de 240x320 pixeles, con capacidad para ocultar 25600 caracteres. Por los recursos de memoria del FPGA únicamente se pudo ocultar un 2% de los 25600 caracteres que se pueden contener dentro de la imagen.

Esta limitante únicamente permite trabajar con imágenes de un tamaño menor o igual a 240x320 píxeles. Debido a que no se pueden realizar mediciones directas de latencia y consumo de energía sobre el FPGA, se realizaron simulaciones por medio de las herramientas ISim (Simulador del ISE) y XPower Analyzer que están incluidos en el kit de desarrollo EDK. Por medio de estas herramientas se realizaron una serie de pruebas con las cuales se obtienen resultados que se exponen a continuación.

A. Análisis de Resultados en Latencia

a) Análisis de Resultados para NoC Hermes en Dos y Tres Nodos

En la Figura 6 se puede apreciar la curva de tendencia de latencia simulada en NoC Hermes para dos y tres nodos.

Latencia Simulada en NoC Hermes para 2 y 3 Nodos

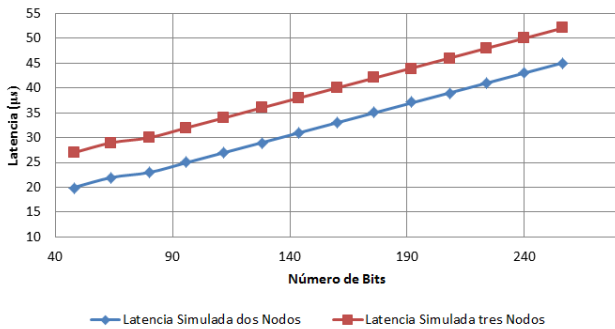


Figura 6. Gráfica de comparación entre NoC Hermes para 2 y 3 nodos.

Como se puede observar en la figura existe un incremento en latencia por aumento del número de bits transmitidos. También existe un diferencial constante debido al número de nodos. Por lo tanto se deben realizar pruebas en topologías de más de cuatro nodos con el objetivo de verificar si la diferencia se mantiene o cambia.

b) Análisis de Resultados entre FSL y NoC Hermes a Dos Nodos

La Figura 7 muestra los valores de los datos simulados para latencia entre NoC Hermes a dos nodos y bus FSL. Esto se debe principalmente a que el tiempo de dispersión es mayor en NoC Hermes con respecto a FSL. Sin embargo estas simulaciones fueron realizadas de punto a punto. Por este motivo se evidencia que es imperceptible evaluar el rendimiento en latencia para una arquitectura de multiprocesamiento MPSoC interconectado por NoC Hermes de 2x2 en topología 2D-Mesh. De manera que se recomienda el desarrollo de trabajos futuros sobre arquitecturas MPSoC

más grandes en donde se pueda percibir el potencial real en latencia de NoC Hermes con respecto a buses FSL.

Latencia Simulada en NoC Hermes y en Bus FSL

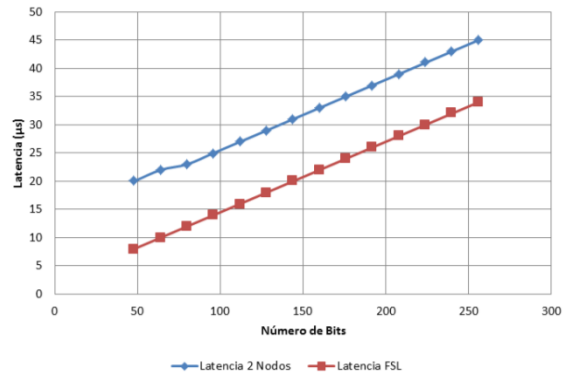


Figura 7. Gráfica de comparación entre NoC Hermes y FSL.

c) Análisis de Resultados entre FSL y NoC Hermes con Interfaz de Red

La Tabla 6 muestra los resultados obtenidos de la simulación con NoC Hermes a dos nodos y NoC Hermes con interfaz de red para un paquete de 80 bits. Dado que solo se hicieron simulaciones, en donde aparentemente se incrementa la latencia, los valores obtenidos no interpretan la realidad.

TABLA 6. DATOS DE SIMULACIÓN EN NOC SIN NI Y NOC CON NI.

Latencia Simulada		
Núm. Bits	NoC Hermes sin NI (µs)	NoC Hermes con NI (µs)
80	23	25

De la tabla anterior se puede concluir que la latencia en NoC Hermes con NI se incrementa en 2µs adicionales. Por lo tanto se demuestra que aparentemente la interfaz de red diseñada incrementa la latencia en el valor mencionado.

Por otra parte la simulación en NoC Hermes con Interfaz de Red también permite tomar en cuenta los valores de latencia sobre el NI diseñado. La Tabla 7 muestra las latencias de los NI00 y NI10.

TABLA 7. LATENCIA EN NI.

Latencia Simulada			
Núm. Bits	FSL (µs)	NI00 (µs)	NI10 (µs)
80	12	11	10

En la Tabla 7 se muestra que la latencia en FSL es de 12µs. Este valor al ser comparado con las latencias de la Tabla 6 muestra claramente que el bus FSL conmuta paquetes de 80 bits de forma más efectiva que NoC Hermes con o sin NI. Además las simulaciones prueban la latencia de 2µs en el

FSL. AL bus FSL le toma 1 μ s para escribir en el lado del maestro y 1 μ s en leer en el lado del esclavo. Por lo tanto la latencia en el NI00 corresponde a la latencia en lectura del bus FSL. Es decir, que le toma tan solo 1 μ s menos que al FSL para transmitir información una vez que adquiere los datos. En otras palabras la latencia del NI00 depende mucho del tiempo de respuesta del esclavo del FSL ante datos presentes en su memoria FIFO interna. Lo mismo se puede concluir con el NI10 pues en este caso a la memoria FIFO le toma 1 μ s en aceptar nuevos datos escritos sobre el maestro del FSL.

B. Análisis de Resultados en Estimación de Potencia

a) Análisis de Resultados en Estimación de Potencia Dinámica para las Arquitecturas Interconectadas por NoC Hermes y por Buses FSL

En la Figura 8 se compara el consumo total de potencia para cada una de las arquitecturas implementadas. Al realizar un análisis comparativo de estimación de potencia dinámica consumida por los elementos del FPGA, se determina que en la infraestructura conectada por NoC Hermes, el consumo de potencia es mayor a la infraestructura conectada por buses FSL en todos los escenarios de estimación. También se puede observar un incremento de potencia directamente proporcional al incremento de la tasa conmutación en cada uno de los escenarios de prueba tanto para la arquitectura interconectada por NoC Hermes como para la arquitectura interconectada por Buses FSL.

Para determinar un modelo de estimación se utiliza la variación de la tasa de conmutación tanto para Flip Flops, como para elementos I/O. En la Tabla 8 se enlistan los escenarios de pruebas bajo los cuales se realizó la estimación de potencia tanto para una arquitectura interconectada por NoC Hermes, como para una arquitectura interconectada por buses FSL.

TABLA 8. ESCENARIOS DE ESTIMACIÓN DE POTENCIA

Tasa I/Os	Tasa Flip Flops	Escenarios de Prueba
12.5 %	12.5 %	Escenario 1
12.5 %	25 %	Escenario 2
25 %	12.5 %	Escenario 3
25 %	25 %	Escenario 4
50 %	50 %	Escenario 5
75 %	75 %	Escenario 6

Estimación de Potencia

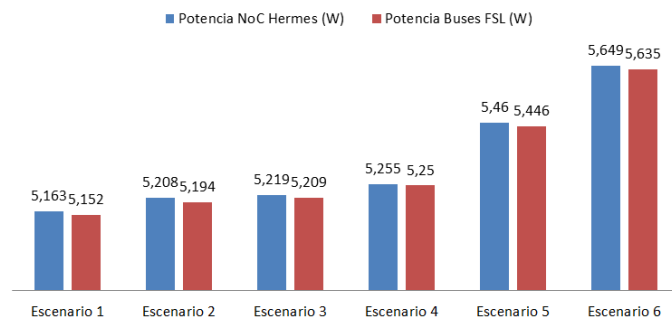


Figura 8. Análisis comparativo de estimación de potencia en función de los escenarios de estimación.

En conclusión el incremento en el consumo de potencia en la arquitectura interconectada por NoC Hermes se debe a la cantidad de elementos lógicos presentes en los Switches de Hermes y en las Interfaces de Red diseñadas para las interconexiones con los distintos procesadores. Estos elementos se obviaron en la arquitectura de buses y por esta razón, tanto instanciar, como configurar dichos elementos, ya conlleva un incremento de consumo de potencia.

b) Análisis de Resultados en Estimación de Potencia Estática para las Arquitecturas Interconectadas por NoC Hermes y por Buses FSL

En la Figura 9 se puede observar que a medida que se incrementa el consumo de potencia estática también se incrementa la temperatura de juntura. Es así que la arquitectura interconectada por NoC Hermes consume 33mW más que la arquitectura interconectada por buses FSL. En consecuencia la temperatura de juntura de la arquitectura interconectada por NoC Hermes es 0.1 °C más que la arquitectura interconectada por buses FSL. Este aumento en potencia y temperatura sobre la arquitectura interconectada por NoC Hermes se debe a la cantidad de recursos utilizados en el FPGA tal como se puede observar en la Figura 9.

Comparación de Temperatura de Juntura y Potencia Estática Consumida por el FPGA

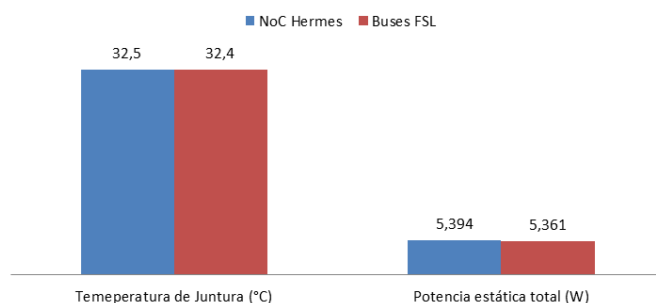


Figura 9. Comparación de temperatura de juntura y de potencia estática total.

En la Figura 10 se observa la comparación de recursos utilizados en el FPGA para la arquitectura interconectada por

NOC Hermes y para la arquitectura interconectada por buses FSL.

Comparación de Recursos Utilizados en el FPGA

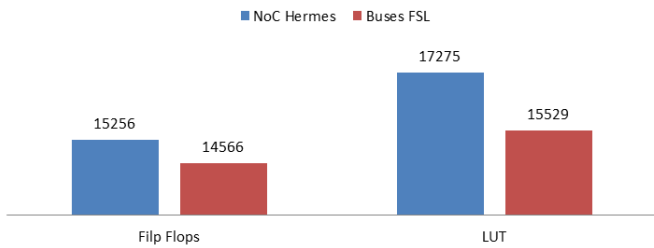


Figura 10. Comparación de los recursos utilizados en el FPGA Virtex-6.

En base a la herramienta Xilinx Power Estimator, la cual es una hoja de cálculo para evaluar potencia, se puede realizar un análisis comparativo de consumo de potencia estática para cada una de las arquitecturas, en donde se evalúan los valores de estabilización con respecto a la fuente de alimentación Vccint.

Potencia en NoC Hermes y en FSL Vs Vccint

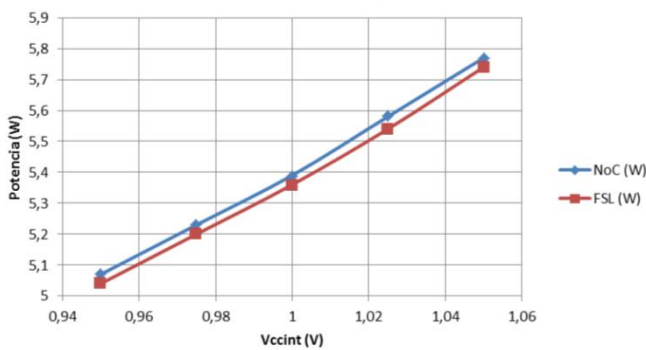


Figura 11. Potencia en NoC Hermes y en FSL Vs. Vccint.

En la Figura 11 se puede apreciar que la potencia es mayor en una arquitectura interconectada por NoC Hermes con respecto a una arquitectura interconectada por buses FSL en 32mW aproximadamente. Esto se debe a que ambas arquitecturas están conectadas en la misma topología de malla, siendo NoC Hermes la arquitectura de comunicación que más elementos emplea para su interconexión a nivel intrachip.

Finalmente, la Figura 12 muestra un análisis comparativo de Temperaturas de Juntura para la arquitectura conectada por NoC y para la arquitectura conectada por buses FSL, tomando en cuenta la estimación de potencia para cada uno de los diseños.

Potencia en NoC Hermes y en FSL Vs Temperatura de Juntura

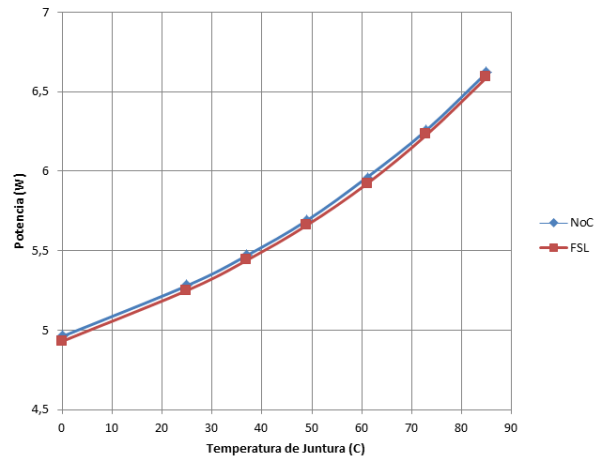


Figura 12. Potencia en NoC Hermes y en FSL Vs. Temperatura de Juntura.

Como se mencionó anteriormente ambas arquitecturas utilizan una topología de malla en su infraestructura de comunicación. Por lo tanto en la Figura 12 se puede apreciar que la disipación de potencia por temperatura de juntura para una arquitectura interconectada por NoC Hermes es mayor que para una arquitectura interconectada por buses FSL en 31mW aproximadamente. Sin embargo es posible que conforme se realicen trabajos futuros sobre topologías en donde se incrementen más dispositivos con buses FSL se vea un aumento en la disipación de potencia por temperatura de juntura.

VI. CONCLUSIONES

Se implementó dos arquitecturas MPSoC de 4 procesadores MicroBlaze sobre la tarjeta de desarrollo ML605 Virtex-6. La primera tiene una arquitectura de comunicación por medio de Buses FSL y la otra está comunicada por medio de NoC. Estas arquitecturas fueron comparadas con el fin de demostrar que NoC es más eficiente en latencia que buses FSL en la transmisión de paquetes para una aplicación que demande alto tráfico de datos y mínima pérdida de información.

Después de realizar una caracterización del estado del arte de NoC se encontró que existe una amplia variedad de modelos de switches interconectados en diferentes topologías a nivel intrachip. Gracias al aporte del Área de Sistemas Digitales de la Escuela Politécnica del Ejército (ESPE) se planteó la utilización de NoC Hermes desarrollada por la Facultad de Informática de la Pontificia Universidade do Rio Grande do Sul (PUCRS, Porto Alegre, Brasil) y la implementación sobre la FPGA mediante código VHDL a través de la herramienta ATLAS desarrollada por GAPH (Hardware Design Support Group-PUCRS). Esta herramienta configura únicamente los switches de NoC Hermes en una topología 2D-Mesh en código abierto VHDL.

Con la finalidad de crear una arquitectura multiprocesada basada en MicroBlaze que emplea NoC Hermes para su comunicación fue necesario diseñar y desarrollar una interfaz de red (NI). El IP-Core resultante es de código abierto VHDL, puede ser modificado y es exclusivamente para comunicación por FSL. Este IP-Core fue desarrollado en la ESPE como contribución al presente proyecto de grado.

A pesar de que se logró implementar las dos arquitecturas MPSoC interconectadas por buses e interconectadas por NoC, no se pudo medir físicamente la latencia y el consumo de energía debido a que la plataforma de desarrollo no permite realizarlas. Por este motivo se recurrió a la estimación de estos parámetros mediante el uso de simuladores. Las simulaciones se realizaron utilizando generadores de tráfico randómico que carecen de un patrón de comportamiento que emule las características de una arquitectura multiprocesada. Por lo tanto, lo ideal para trabajos futuros es que se desarrollen generadores de tráfico que permitan simular de forma paralela diversos números de nodos dentro de una topología NoC Hermes superior a una 2D-Mesh de 2x2. Por otra parte se deberán crear generadores de tráfico que permita evaluar a NoC Hermes con patrones de tráfico de aplicaciones reales.

Mediante la utilización de generadores de tráfico se realizaron simulaciones a dos y tres nodos encontrando como resultado que la arquitectura NoC Hermes aumenta su latencia a medida que se incrementa el número de nodos involucrados en la simulación. Esto se debe principalmente al tiempo de dispersión que le toma a cada switch para establecer una ruta de comunicación mediante el algoritmo XY.

En las pruebas simuladas entre NoC Hermes a dos nodos y buses FSL, en la transmisión de paquetes de punto a punto se demostró que el bus FSL es más rápido que la arquitectura NoC Hermes, debido a que los tiempos de transmisión son más prolongados en NoC Hermes que en FSL. En conclusión una arquitectura NoC Hermes de 2x2 en topología 2D-Mesh no es suficiente para ser evaluada. Por lo cual se requiere topologías más grandes en donde las latencias se incrementen a tal punto que se pueda demostrar la potencial mejora del desempeño al utilizar NoC Hermes frente a buses FSL.

En la arquitectura de comunicación NoC Hermes sin NI, conectada con generadores de tráfico, la latencia es de 23 μ s para un paquete de 5 flits. Mientras que para una arquitectura de comunicación NoC Hermes con NI, la latencia es de 25 μ s para el mismo tamaño de paquete. En consecuencia los NI diseñados para comunicar a los procesadores con NoC Hermes presentan una latencia de 2 μ s adicionales.

Con el fin de evaluar las arquitecturas multiprocesadas interconectadas por buses FSL y por NoC Hermes implementadas sobre el FPGA se desarrolló una aplicación de esteganografía en la que existe multiprocesamiento de dos procesadores trabajando simultáneamente. Esta aplicación fue útil para evaluar ambas arquitecturas dando como resultado que la implementación mediante buses FSL ocupó el 10% de los recursos del FPGA y NoC Hermes el 11%. Mientras que

para las dos arquitecturas se utilizó el 100% de los recursos de memoria disponibles en el FPGA.

Para medir latencia se necesita gran cantidad de tráfico de información pero por las limitaciones de memoria en la plataforma de desarrollo se utilizó imágenes de 320x240 pixeles y con mensajes de 300 caracteres. Las limitantes en dimensión de imágenes y en tamaño de mensajes están condicionadas por las capacidades de memoria del FPGA ML605 Virtex-6 sobre el cual fueron implementados.

Para comparar el consumo de energía de cada una de las arquitecturas multiprocesadas se realizó una estimación de potencia variando las tasas de conmutación tanto para Flip Flops como para elementos I/O. El proceso de estimación da como resultado que el consumo de energía en la arquitectura interconectada por NoC Hermes en topología 2D-Mesh es 33mW mayor que el consumo de energía para la arquitectura interconectada por buses FSL. Este incremento se produce debido a que la arquitectura interconectada por NoC Hermes posee más elementos que son necesarios para su funcionamiento, los mismos que no están presentes en la arquitectura interconectada por buses FSL. Estos elementos corresponden a los Switches de Hermes y las Interfaces de Red que permiten la comunicación con los procesadores MicroBlaze.

AGRADECIMIENTO

Agradecemos a los Autores de la Facultad de Informática PUCR por facilitar el uso de la NoC que desarrollaron y en especial agradecemos a la Escuela Politécnica del Ejército por permitirnos ser parte de este importante proyecto de investigación.

REFERENCIAS

- [1] K.Rajesh , "On-chip memory architecture exploration of embedded Systems on chip", Tesis Doctoral, Facultad de Ingeniería , Instituto de Ciencias de la India, Supercomputer Education and Research Centre, Sep. 2008.
- [2] L. Benini, G. de Micheli, "Network-on-Chip: A new paradigm for component-based MPSoC design", DEIS Università di Bologna Italia, CSL Stanford University Stanford CA.
- [3] T. Tao Ye, "On-chip Multiprocessor Communication Network design and analysis ", Stanford University , December 2003.
- [4] L. Torres, P. Benoit, G. Sassatelli, M. Robert, "An introduction to Multi-Core Systems on Chip – Trends and challenges", University of Montpellier 2, LIRMM UMR CNRS, France.
- [5] A. Schranzhofer, J. Chen, L. Thiele, "Power-Aware Mapping of Probabilistic Applications onto Heterogeneous MPSoC Platforms ", Computer Engineering and Networks Laboratory (TIK), Zurich Switzerland.
- [6] W. Wolf, A. Amine, G. Martin, "Multiprocessor System-On-Chip (MPSoC) Technology", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 27, No 10, October. 2008.
- [7] "Interfaz WISHBONE", Revisión 1.6, OpenCores, 2010.
- [8] R. Ernst, "MPSOC Architecture Modeling", TU Braunschweig, 2003.
- [9] M. Milica, M. Stojcev, "An Overview of On-Chip Buses", Facta Universitatis (NIS), Vol 19, December 2006.

- [10] M. Aparicio, "Diseño de un sistema de lectura y procesado para múltiples sensores embebidos en un FPGA", Proyecto de Fin de Carrera, Universidad Autónoma de Madrid, Diciembre. 2007.
- [11] J. Peña, "Diseño de módulos para el manejo de Puertos, Temporización e Interrupciones para el Núcleo KCPSM3 e Implementación en el FPGA XC3S500 de Xilinx", Proyecto de Fin de Carrera, Universidad Tecnológica de la Mixteca, Junio. 2008.
- [12] S. Pantelopoulos, A. Brokalakis, "High Level Architecture of Parallel System", Information and Communication Technologies (ICT) Programme, May 2012.
- [13] S. Hessabi, "SoC Design", Department of Computer Engineering, Sharif University of Technology.
- [14] Xilinx, "LogiCore IP Multi-Port Memory Controller (MPMC) (v6.04.a)", EDK, DS643 July 6, 2011.
- [15] Xilinx, "XPS Thin Film Transistor (TFT) Controller (v2.01a)", EDK, DS695 May 3, 2010.
- [16] Xilinx, "XPS SYSACE (System ACE) Interface Controller (v1.01a)", EDK, DS583 March 4, 2010.
- [17] Xilinx, "EDK Concepts, Tools and Techniques", UG683 EDK 12.2.
- [18] Xilinx, "MicroBlaze Debug Module (MDM) (v2.00b)", EDK, DS641 March 1, 2011.
- [19] F. Moraes, N. Calazans, A. de Mello, L. Möller, L. Copello Ost, "HERMES: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip", TECHNICAL REPORT SERIES, Facultad de Informática PUCRS-Brazil, Number 034, October, 2003.
- [20] Xilinx, "LogiCORE IP Fast Simplex Link (FSL) V20 Bus (v2.11e)", EDK, DS449, June 2011.
- [21] D. Atienza, F. Angiolini, S. Murali, A. Pullini, L. Benini, G. de Micheli, "Network-on-Chip design and synthesis outlook", INTEGRATION, the VLSI journal 41 (2008) 340–359, Received 26 November 2007
- [22] Xilinx, "Embedded System Tools Reference Manual", EDK 13.2, UG111 July 6. 2011.
- [23] M. Death, "Introducción a la Esteganografía", Death Master, 2004 (GFDL), <http://www.death-master.tk/>.
- [24] "Procesamiento de Imágenes", Universidad Simón Bolívar, Departamento de Comunicación y TI, Abril – Julio 2008.
- [25] "Procesamiento de Imágenes Digitales", Universidad de Málaga, Dpto. Lenguajes y CC. Computación E.T.S.I. Telecomunicación.
- [26] P. Juan, "Proyecto Encriptación de texto sobre imágenes BMP mediante software".