MicroBlaze en Sistemas Embebidos para Aplicaciones Criptográficas

González I, Gómez-Arribas F. J, Martínez J

Escuela Politécnica Superior, Madrid, España,
{Ivan.Gonzalez, Francisco.Gomez, Javier.Martinez}@ii.uam.es
http://www.ii.uam.es

Resumen. En este trabajo se presenta un estudio sobre el "Soft Core Processor" MicroBlaze como microprocesador en sistemas embebidos para aplicaciones que involucran algoritmos de criptografía. Se analiza el rendimiento alcanzado para un conjunto representativo de los algoritmos simétricos más empleados: DES, 3DES, IDEA, BLOWFISH y AES. Se proponen diferentes arquitecturas del sistema que aprovechen mejor los recursos internos del microprocesador. Se estudia la mejora de rendimiento asociada con cada variación de la arquitectura interna, y se compara con la integración en el sistema de un *core* de cifrado específico, implementado en el *hardware* reconfigurable. Los resultados obtenidos permiten concluir que MicroBlaze puede ser considerado un serio candidato para este tipo de aplicaciones.

1 Introducción

La seguridad y el procesamiento de datos en tiempo real son dos requisitos básicos en la transmisión de información y en las aplicaciones de los futuros sistemas de comunicaciones. En lo referente a seguridad, las aplicaciones que hacen uso de algoritmos de cifrado deben cumplir un conjunto de requerimientos y retos. Es necesario que el rendimiento de los procesos de cifrado sea compatible con la transmisión de datos por las líneas de comunicación. Por otro lado, las aplicaciones actuales se basan en los nuevos paradigmas de los protocolos de seguridad, en muchos de los cuales se permite separar la elección del algoritmo de cifrado del protocolo a utilizar, por lo que los usuarios de la aplicación pueden negociar el algoritmo a emplear para una sesión de cifrado particular. Por ejemplo IPSec, el estándar de seguridad en Internet, permite elegir de una lista de diferentes algoritmos simétricos y asimétricos. La lista de los algoritmos de clave simétrica mas empleados se extiende de cada vez más, incluyendo DES, 3DES, Blowfish, IDEA, AES y RC4 entre otros.

Para dar soporte a este tipo de aplicaciones con diferentes algoritmos involucrados, los sistemas de criptografía deben presentar una enorme flexibilidad. Una solución es ejecutar los algoritmos de cifrado sobre microprocesadores de propósito general, por ejemplo, en un PC. El rendimiento de un PC de altas prestaciones es suficiente para un gran número de aplicaciones y la flexibilidad es inherente al sistema. Sin embargo, la vulnerabilidad de los sistemas operativos que utilizan estas plataformas y la propia seguridad física que viene determinada por la ubicación donde se encuentra el equipo, no garantiza el almacenamiento seguro de las claves.

Otra aproximación que permite obtener altos rendimientos es un diseño *hardware* a medida, que normalmente resulta más robusto en términos de seguridad que el clásico PC. No obstante, el diseño de un sistema específico es costoso en tiempo de desarrollo, supone un alto precio del producto y es difícil conseguir la misma flexibilidad que tiene un sistema basado en un procesador de propósito general.

En la actualidad se están desarrollando sistemas embebidos que proporcionan una solución integrada que combina las ventajas de las aproximaciones anteriores. Para evitar la perdida de flexibilidad en los diseños *hardware*, se emplean dispositivos reconfigurables [1]. Las soluciones basadas en *hardware* reconfigurable, sobretodo en el caso particular de los algoritmos de criptografía, permiten obtener rendimientos elevados siendo posible realiza adaptaciones específicas de las operaciones de cada algoritmo [2].

En este trabajo se propone una variante que consiste en desarrollar un sistema embebido en el que se integra un procesador de propósito general en *hardware* reconfigurable, en el que a su vez, es posible implementar diseños específicos. Este tipo microprocesadores se conocen como Soft Core Processors (SCP). La principal ventaja de los SCP es precisamente su flexibilidad, ya que permite la configuración a medida los recursos internos para una aplicación específica y también la integración de *cores hardware* a medida, consiguiendo en conjunto mejorar notablemente el rendimiento del sistema. Uno de los SCP más conocidos es MicroBlaze. Para estudiar la viabilidad del microprocesador MicroBlaze como procesador en sistemas embebidos en el campo de las aplicaciones criptográficas, se ha realizado un estudio del rendimiento de un conjunto de algoritmos que forman parte de la mayoría de las aplicaciones de seguridad.

2 El Core Xilinx MicroBlaze

MicroBlaze es un SCP (Soft Core Processor) de 32 bits con arquitectura Harvard tipo RISC. Este procesador, descrito en VHDL, está especialmente desarrollado para FPGAs de Xilinx. Dispone de 32 registros de 32 bits que están basados en *look-up-table* RAM, lo que garantiza un tiempo de acceso corto. El sistema de memoria permite, además de la memoria BRAM, emplear memoria externa. El tiempo de acceso a la memoria BRAM es reducido debido a que se emplean recursos específicos para su acceso. MicroBlaze hace un uso eficiente de los recursos disponibles en la FPGAs actuales, por lo que es posible llegar hasta frecuencias de reloj de 150 MHz [3].

La arquitectura interna de MicroBlaze puede ser adaptada para cualquier aplicación. El empleo de unidades *hardware* de desplazamiento y división, memorias caché de datos e instrucciones y el bus FSL (Fast Simples Link) son opcionales. Los tamaños de las memorias caches son configurables desde los 2 a los 64 Kbytes. Un conjunto de periféricos estándar como controladores de memoria, UARTs, controladores de interrupciones, etc. están disponibles para ser conectados a través del estándar CoreConnect [4].

Generalmente hay dos formas de integrar cores propios en un sistema embebido basado en MicroBlaze. Un modo es conectar el core al bus OPB (On-Chip Peripheral Bus), que forma parte del estándar *CoreConnect*. El segundo modo es mediante el bus FSL, especialmente indicado si el tiempo de acceso a la aplicación es crítica, siendo posible usar funciones C predefinidas para usar el *core* desde una aplicación software.

3 Algoritmos de Criptografía

En criptografía se diferencian dos clases de algoritmos: los de clave privada o simétrica y los de clave pública. A su vez, los algoritmos de clave simétrica se clasifican, por su modo de funcionamiento, en cifradores de bloques y cifradores de flujo.

Este estudio se centra en los algoritmos de cifrado de bloques. Los algoritmos elegidos son DES, IDEA, 3DES, Blowfish y AES. Todos ellos han sido ampliamente estudiados e implementados de forma eficiente para su ejecución en procesadores de propósito general [5] y en *hardware* específico [1, 2, 6, 7]. Muy brevemente enumeramos sus características principales:

Algoritmo	XOR AND OR	Suma Resta Mod.	Rotación Fija	MUL Mod.	MUL GF Const.	LUT	Tam. Bloque Interno
DES/3DES	•		•			6-to-4	32
IDEA	•	2^{16}		$2^{16}+1$			16
BLOWFISH	•	2^{32}				8-to-32*	32
AES	•		•		$GF(2^8)$	8-to-8	32

Tabla 1. Operaciones básicas de los algoritmos seleccionados

DES/3DES. El algoritmo DES utiliza claves de 56 bits y un cifrado de bloques de 64 bits. La complejidad del algoritmo reside en las transformaciones basadas en *SBoxes*, y en las permutaciones a nivel de bit. El algoritmo 3DES es una variante que realiza el cifrado triple de los datos mediante DES.

IDEA. El algoritmo IDEA trabaja con bloques de 64 bits de longitud y emplea una clave de 128 bits. La operación mas compleja es la multiplicación módulo $2^{16}+1$, que viene acompañada de otras dos operaciones, la suma módulo 2^{16} y la exor módulo 2^{16}.

BLOWFISH. Es un algoritmo de cifrado simétrico que utiliza una clave de longitud variable, de 32 bits a 448 bits, y cifra bloques de 64 bits de longitud. Las operaciones básicas son similares a DES.

AES. En Octubre de 2000 el algoritmo Rijndael fue elegido como el estándar de cifrado. Actualmente, se usa con claves de longitud 128, 192, o 256 bits para cifrar bloques de longitud de 128, 192 o 256 bits. Las operaciones involucradas son la transformación mediante SBox y la multiplicación de Galois.

En la tabla 1 se muestra un resumen de las operaciones básicas que utiliza cada algoritmo. La columna denominada LUT, representa las operaciones basadas en *look-up-table*. Estas tablas o memorias de traducción rápida son muy adecuadas para realizar las transformaciones basadas en *SBox* como las del algoritmo DES, o incluso la multiplicación cuando un operando es constante (*) en el algoritmo Blowfish.

4 Trabajo Experimental

Se proponen diferentes experimentos para medir el rendimiento alcanzado en un sistema criptográfico basado en Microblaze con los distintos algoritmos de cifrado seleccionados. Para ello se han definido 2 conjuntos de experimentos que básicamente se diferencian en la arquitectura del sistema que les da soporte, como se muestra en la figura 1.

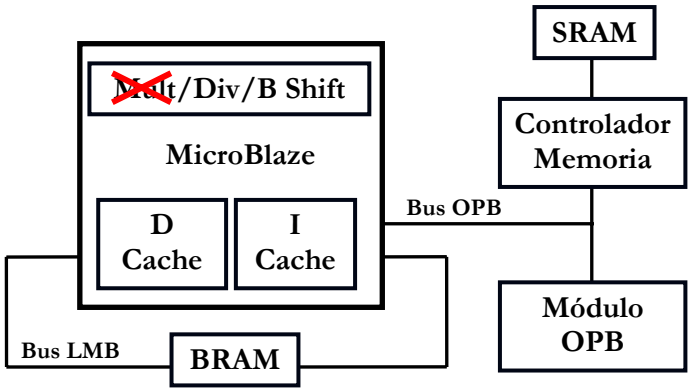

Fig. 1. Arquitectura de MicroBlaze. El interior del *core* muestra los recursos internos opcionales. En la parte derecha de la figura se indica la conexión al bus OPB de un módulo específico de cifrado.

En el primero se ejecuta un código en lenguaje C de cada algoritmo criptográfico lo más sencillo posible para favorecer su ejecución de forma óptima en cualquier procesador de propósito general, sin incluir mejoras específicas. Al mismo tiempo, se estudia el efecto de modificar la arquitectura interna del SCP, con los recursos opcionales del *core* Microblaze: multiplicador, divisor, *barrel shifter* y memoria caché para instrucciones y datos. El tamaño de las dos caches de instrucciones y de datos, con organización de correspondencia directa, aunque admite varias configuraciones se ha fijado con un tamaño de 8192 bytes y 256 entradas. Las arquitecturas resultantes se indican en la tabla 2.

En el segundo conjunto de experimentos se añade a las arquitecturas anteriores un módulo específico para cada algoritmo, que se encarga de realizar el proceso de cifrado completo. La comunicación con el SCP se realiza mediante el bus OPB como se muestra en la figura 1. En este caso el programa a ejecutar incluye las líneas de código que realizan el cifrado mediante el envío y recepción de datos al módulo específico. El uso de estos periféricos pretende acelerar la ejecución de cada uno de los algoritmos. Aunque es posible que el *hardware* sea muy eficiente, existen dos limitaciones a tener en cuenta: por un lado la frecuencia de operación del bus OPB que será equivalente a la del sistema, en este caso 50 MHz; y por otro lado, la complejidad de la implementación *hardware* de cada uno de los algoritmos, que implica diferentes latencias y tiempos de procesamiento.

El equipo experimental en el que se han desarrollado los distintos sistemas MicroBlaze es la plataforma RC1000PP de Celoxica, basada en una FPGA XCV2000E, y que dispone de 4 x 2 MB de memoria SRAM. Adicionalmente, aprovechando los pines I/O disponibles se ha configurado un puerto serie RS-232 que actúa de consola para la entrada/salida. Como se comprueba en la tabla 2 se ha evitado una arquitectura que incluya el multiplicador como recurso interno. Esto es debido a que el dispositivo empleado, FPGA Virtex-E, no

dispone de multiplicadores embebidos. En cualquier caso, de la tabla 1 se deduce que la ausencia de multiplicación como recurso *hardware* solo afecta al rendimiento del algoritmo IDEA, ya que es el único que utiliza esta operación.

Arquitectura	Barrel Shifter	Mult	Divisor	Caché Datos	Caché Instrucciones	FPGA Slices	BRAM
A						1102	32
B	•					1213	32
C				•	•	1216	74
D	•			•	•	1321	74
E	•		•	•	•	1385	74

Tabla 2. (a) Arquitecturas MicroBlaze propuestas (b) Recursos para cada Arquitectura MicroBlaze

La aplicación de test para cada algoritmo mide el tiempo empleado en el cifrado/descifrado de un conjunto de datos de 2 MBytes. La ejecución de estos programas de test sobre el sistema MicroBlaze se realiza a partir del código cargado en la memoria SRAM externa en vez de la BRAM interna, porque el tamaño de esta memoria suele ser insuficiente para las aplicaciones que se ejecutan en sistemas embebidos. La frecuencia de operación de las diferentes arquitecturas es de 50 MHz, en todos los casos. La medida del tiempo se realiza mediante la integración de un *core* contador en la FPGA. Otros detalles del proceso, como la configuración del tamaño de bloque de datos a cifrar y el tamaño de la clave empleado para cada caso se indican en las tablas de la siguiente sección con los resultados obtenidos.

5 Resultados

Para disponer de un valor de referencia al realizar los diferentes experimentos se han tomado los resultados obtenidos en un PC con procesador Intel Pentium IV a 2.4 GHz. Aunque las arquitecturas MicroBlaze y PC no son comparables y la frecuencia de operación es 48 veces superior, conviene tener los resultados del PC como referencia de los rendimientos necesarios que se exigen en las aplicaciones actuales.

5.1 Arquitecturas Configurables de MicroBlaze

Los resultados de implementación del sistema para cada arquitectura de la tabla 2 se muestran en la misma tabla y los rendimientos obtenidos para cada algoritmo en la tabla 3. La implementación de las caches se realiza en BRAM lo que se corresponde con el incremento de 32 a 74 bloques de BRAM. El tamaño de la caché de instrucciones es suficiente para guardar el código completo de todos los algoritmos.

En términos de rendimiento, con un análisis preliminar se puede determinar que la arquitectura E, que incluye un divisor *hardware* no aporta ventajas a ninguno de los algoritmos como cabría esperar según la tabla 1. Por el contrario, la unidad *barrel shifter* que aparece en la arquitectura B, se revela como el componente fundamental para la mejora del rendi-

miento en todos los algoritmos, con excepción del algoritmo IDEA, donde las operaciones de permutación de bits no son básicas. El uso de memoria caché favorece sobretodo a los algoritmos IDEA y Blowfish, ya que para el resto la mejora de rendimiento es inferior que al incluir la unidad *barrel shifter*.

Algoritmo	Clave / Bloque	PC (Mbit/s)	Arq_A (Mbit/s)	Arq_B (Mbit/s)	Arq_C (Mbit/s)	Arq_D (Mbit/s)	Arq_E (Mbit/s)
DES	64 / 64	80,854	0,038	0,256	0,139	1,791	1,791
3DES	192 / 64	35,099	0,014	0,090	0,033	0,168	0,168
IDEA	128 / 64	52,429	0,067	0,093	0,537	0,760	0,760
BLOWFISH	448 / 64	72,472	0,135	0,358	0,952	1,942	1,942
AES	128 / 128	342,392	0,068	0,435	0,168	0,904	0,905

Tabla 3. Rendimiento de las diferentes arquitecturas MicroBlaze.

La combinación de caché y *barrel shifter*, denominada arquitectura D, presenta el mejor resultado, como se deduce del análisis de los diferentes componentes por separado.

5.2 Arquitectura Mejorada mediante un Módulo Específico de Cifrado.

Como era previsible, los rendimientos de la sección anterior obtenidos exclusivamente mejorando la arquitectura de MicroBlaze no alcanzan los valores necesarios para ser usados en las aplicaciones típicas de seguridad de datos, por ejemplo de cifrado de sesiones a través de Internet, donde al menos se requiere trabajar a velocidades de unos MBits/seg.

Algoritmo	DES	3DES	IDEA	BLOWFISH	AES
Slices	4121	12176	1979	2877	5143
BRAM	0	0	12	10	0

Tabla 4. Recursos empleados: OPB *Cores* de cifrados

En esta sección se presentan los resultados obtenidos al intentar acelerar los diferentes algoritmos ampliando la funcionalidad del procesador mediante la creación de *cores hardware* específicos. Para dar una idea de la complejidad, en la tabla 4 se muestran los recursos necesarios de cada uno de los *cores* desarrollados, todos ellos se han diseñado para actuar como esclavos del bus OPB. Dependiendo de la dificultad para portar la funcionalidad de cada algoritmo al *hardware*, se han conseguido diferentes resultados. DES y 3DES se ha desarrollado completamente, siendo necesario emplear el 50% de los recursos disponibles en el caso de 3DES. El mismo caso se da para el algoritmo AES, aunque a diferencia de 3DES, su implementación requiere una menor cantidad de recursos. Por el contrario, en el caso de IDEA y Blowfish se ha optado por un diseño simplificado de una sola de las etapas junto con el mecanismo de control que la utiliza reiteradamente de acuerdo a las sucesivas iteraciones implicadas en cada algoritmo para cifrar un bloque de datos.

La tabla 5 recoge los resultados del test sobre cada una de las arquitecturas presentadas en el apartado anterior, excepto la arquitectura E con divisor embebido, ya que no aporta ninguna mejora significativa con respecto a la arquitectura D que no lo incluye.

Los resultados obtenidos permiten determinar que el recurso fundamental de la arquitectura de MicroBlaze es la caché, debido en mayor medida a que el manejo de los *cores* no requiere de operaciones de rotación, lo que favorecería la dependencia de la unidad *barrel shifter*. Al mismo tiempo, se puede comprobar que los rendimientos son similares, lo que se debe a que el manejo de los diferentes *cores* es prácticamente idéntico. La excepción es la implementación hardware del algoritmo AES, que al ser más sencillo requiere de un menor número de ciclos de operación para realizar un cifrado completo que el resto de algoritmos, lo que permite obtener un rendimiento superior. Es importante destacar en estos resultados de rendimiento, la limitación que supone trabajar a una frecuencia de 50 Mhz.

Algoritmo	Clave / Bloque	PC (Mbit/s)	Arq_A (Mbit/s)	Arq_B (Mbit/s)	Arq_C (Mbit/s)	Arq_D (Mbit/s)
DES	64 / 64	80,854	1,215	1,288	5,079	5,199
3DES	192 / 64	35,099	1,160	1,227	4,414	4,504
IDEA	128 / 64	52,429	1,165	1,222	4,194	4,275
BLOWFISH	448 / 64	72,472	1,013	1,265	4,747	5,356
AES	128 / 128	342,392	2,097	2,326	9,907	10,347

Tabla 5. Rendimiento de las diferentes Arquitecturas con Módulo Específico.

La figura 2 muestra el factor de mejora en rendimiento cuando a la arquitectura Microblaze se le añade un *core* especifico para cada algoritmo. Los resultados comparan la arquitectura D de cada una de las soluciones estudiadas. Los algoritmos DES y Blowfish apenas han mejorado, lo que se debe al buen rendimiento mostrado en la ejecución software. El resto de algoritmos, 3DES, IDEA y AES, presentan mejoras considerables, obteniéndose en el caso de 3DES el mayor incremento de rendimiento.

6 Conclusiones y Trabajo Futuro

Este trabajo se estudia la flexibilidad que presentan los "Soft Core Processors", permitiendo concluir que son una buena solución para su uso en sistemas embebidos. La capacidad de adaptar las características de su arquitectura y poder complementarla mediante nuevos módulos, que se incorporan como periféricos, es fundamental.

En el caso particular de MicroBlaze, los diferentes recursos *hardware* disponibles en su arquitectura son determinantes para obtener un buen rendimiento a la hora de ejecutar los diferentes algoritmos de criptografía seleccionados. En concreto, es prácticamente imprescindible incluir memoria caché y en la mayoría de los casos añadir como recurso interno la unidad *barrel shifter*, sobretodo en implementaciones que no hace uso de *cores* específicos. Por otro lado, al incluir periféricos hardware se ha podido mejorar el rendimiento de los diferentes algoritmos posibilitando su uso en aplicaciones de cifrado de datos a través de Internet, aunque la mejora de rendimiento no ha sido igual para todos ellos. Una de las razones de la falta de mejoría se debe a la implementación *hardware* de los *cores* y a la elección del bus OPB. En el caso particular de éste último, por sus características no se recomienda como un buen candidato para transmisión de datos a altas velocidades, ya que hay que tener también en cuenta que el uso de este bus se comparte con el acceso a la memoria de programa y datos del microprocesador.

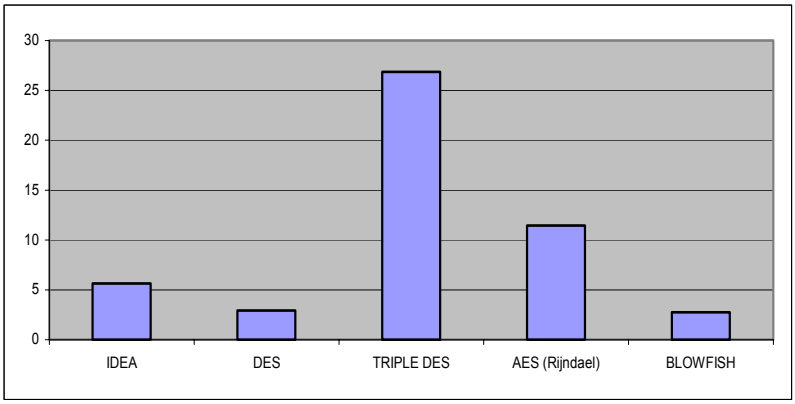

Fig. 2. Mejora del Rendimiento de la Arquitectura MicroBlaze D al añadir un Módulo de cifrado específico, para los distintos algoritmos (Factor_Mejora = Tsin_core/Tcon_core).

Con intención de mejorar los resultados obtenidos, MicroBlaze ofrece otras soluciones, como el Bus FSL (Fast Simples Link), que permite añadir los diferentes *cores hardware* en la propia arquitectura del microprocesador. Otra posibilidad para mejorar el rendimiento es desarrollar *cores* OPB maestros con capacidad de acceso DMA, posibilitando el acceso directo a la memoria y evitando que sea el microprocesador el que lea los datos y los envíe al *core*. Estas dos soluciones se plantean como trabajos futuros a realizar.

Agradecimientos

Trabajo financiado por los proyectos 07T/0052/2003-3 de la Consejería de Educación de la CAM y TIC2001-2688-C03-03 del Ministerio de Ciencia y Tecnología de España.

Referencias

1. J. L. Beuchat, J.O. Haenni, H.F. Restrepo, C. Teuscher, F.J. Gomez and E. Sanchez, "*Approches matérielles et logicielles de l'algorithme de chiffrement IDEA*". Technique et Science Informatiques (TSI), volumen 1, (2001) 203-224
2. Gonzalez, I., Lopez-Buedo, S., Gomez, F. J., Martinez, J.: *"Using Partial Reconfiguration in Cryptographic Applications: An Implementation of the IDEA Algorithm"*. Proc. of FPL03, LNCS, 2778. ISBN 3-540-40822-3 Springer-Verlag, Berlin (2003) 194-203
3. http://www.xilinx.com/ipcenter/processor_central/microblaze/performance.htm
4. http://www-306.ibm.com/chips/products/coreconnect/
5. Schneier, Bruce. Applied Cryptography, 2nd Edition. John Wiley & Sons, 1996.
6. Cameron Patterson, "*High Perfomance DES Encryption in Virtex FPGAs using JBits*", Proc. 2000 IEEE Symposium on Field-Programmable Custom Computing Machines, pp. 113-121, Napa, California, 2000.
7. A. Elbirt, W. Yip, B. Chetwynd, and C. Paar, "*An FPGA-based performance evaluation of the AES block cipher candidate algorithm finalists*," IEEE Trans. VLSI Syst., vol. 9, pp. 545--557, Aug. 2001.