# Desarrollo de un Módulo MD5 para un Sistema Criptográfico Reconfigurable en un FPGA

Ignacio Algredo-Badillo, René Armando Cumplido-Parra, Claudia Feregrino-Uribe

Coordinación de Ciencias Computacionales
Instituto Nacional de Astrofísica Óptica y Electrónica, INAOE.
e-mail: talion00z@ccc.inaoep.mx, rcumplido@inaoep.mx, cferegino@inaoep.mx.
Luis Enrique Erro #1, CP 72840, Sta. Ma. Tonantzintla, Puebla, México.

**Abstract**: This work reports on the implementation of the MD5 hash algorithm [1] achieving a throughput of 1 Gbps. This module will be integrated into a library of configurations for a reconfigurable system that will allow managing diverse authentication and cipher standards to obtain a highest possible security and to reduce the risk of rapid obsolescence. The main objective is to obtain an implementation that processes data to a speed of 1Gbps for integration into high-speed networks. In this article, initial designs of the implementation are presented as well as strategies that were used to reach the required throughput.

**Resumen**: En este trabajo se reporta la implementación del algoritmo MD5 [1] para un procesamiento de 1 Gbps. Este modulo será integrado en una biblioteca de configuraciones para una plataforma reconfigurable con el objetivo principal de obtener una implementación que procese datos a una velocidad mayor a 1Gbps. Las motivaciones se deben a que las redes de comunicación manejan diversos estándares de autenticación y de cifrado para obtener una mayor seguridad, considerando que en la actualidad las velocidades de transmisión manejan niveles superiores a los Mbps y los Gbps. En este artículo se presentan las estrategias de implementación que se siguieron para alcanzar la capacidad de procesamiento requerida.

Palabras Clave: FPGA, criptografía, *hash*, MD5.

## I. Introducción

Una red de comunicaciones es el conjunto de dispositivos hardware y software que permiten el intercambio de información digital entre los distintos elementos que se encuentran conectados en dicha red [2]. En las redes de comunicación es necesario prevenir la extracción sin autorización de información y asegurar la privacidad. Existen varios problemas que los sistemas de seguridad deben evitar como la interrupción, modificación, interceptación y usurpación de información, los cuales son resueltos por la criptografía. Los criptosistemas (ver figura 1) permiten establecer cuatro aspectos fundamentales de la seguridad informática: confidencialidad, integridad, autenticación y no repudio entre emisor y receptor.

La autenticación permite comprobar de manera segura alguna característica sobre un objeto, como su origen, su integridad, su identidad, etc. [3]. Existen tres tipos de autenticación: autenticación de mensaje, para asegurar que el mensaje no es falsificado,

autenticación de usuario mediante contraseña, para garantizar la presencia de un usuario legal en el sistema y autenticación de dispositivo que garantiza la presencia de un dispositivo válido.

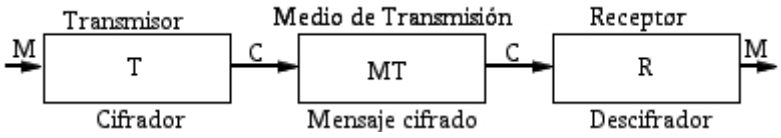


**Fig 1.** Sistema de cifrado.

Las funciones *hash* son utilizadas para proveer el servicio de seguridad de autenticación. Estas funciones comprimen una cadena de bits de longitud arbitraria para obtener una cadena de longitud fija. El propósito de las funciones *hash* es producir un resumen de un archivo, un mensaje u otro bloque de datos.

## II. Algoritmo MD5

MD5 (*Message Digest Algorithm* 5, Algoritmo de Ordenación de Mensajes 5) es un algoritmo seguro desarrollado por RSA Data Security, Inc. MD5 es una función *hash* de 128 bits, que toma como entrada un mensaje de tamaño arbitrario y produce como salida un resumen del mensaje de 128 bits. El MD5 no sirve para cifrar un mensaje ya que lo destruye completamente, la información no es recuperable de ninguna manera ya que hay pérdida de información.

El primer paso es dividir el mensaje en bloques de 512 bits. El último bloque o si el mensaje completo es menor a 512 bits, se formatea para tener un tamaño de 512 bits mediante el agregado de bits 0 más la longitud del tamaño del mensaje. Además, se tiene un búfer estado de 128 bits manejado como cuatro palabras de 32 bits. La función compresión tiene cuatro rondas y en cada ronda el bloque de mensaje y el búfer son combinados en el cálculo, mediante el uso de sumas modulares, XOR's, AND's, OR's y operaciones de rotaciones sobre palabras de 32 bits [4]. Para detalles ver [1].

Cada ronda combina el bloque de 512 bits del mensaje con el búfer estado, así que cada palabra del mensaje es usado cuatro veces. Después de las cuatro rondas de la función compresión, el búfer estado y el resultado son sumados (sumas módulo 232) para obtener la salida [2], ver figura 2.

## III. Trabajo Relacionado

La búsqueda de información se realizó en base a implementaciones en hardware, especialmente en FPGA's, tanto trabajos de investigación como productos comerciales que utilizarán diferentes técnicas de diseño.

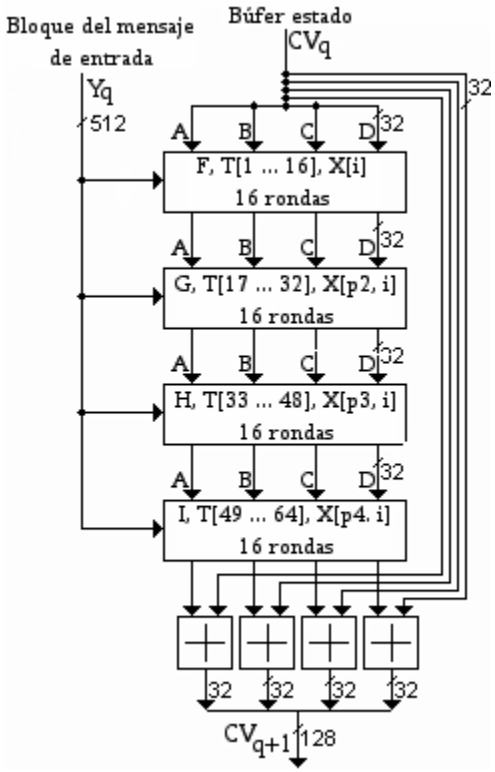


**Fig. 2**. Procesamiento MD5 de un sólo bloque.

El trabajo en [5] se muestran las implementaciones de tres funciones *hash* (SHA-1, HAS-160 y MD5). La estrategia de diseño consideró utilizar un mínimo número de compuertas. Las implementaciones se realizaron en el dispositivo Altera EP20K1000EBC652-3. El resultado procesamiento presentado es de 142 Mbps a una frecuencia de reloj de 18MHz utilizando 65 ciclos de reloj para el proceso de un bloque.

En [6] se tienen las implementaciones de las funciones MD5 y SHA-1 un FPGA Xilinx Virtex 2V3000. Presentan resultados de la síntesis en ISE 4.1 para la implementación del algoritmo MD5, el cual procesa 467.3 Mbps con un diseño simple sin alguna técnica de optimización. El trabajo [7] presenta dos diseños para la implementación del algoritmo MD5, sobre un Virtex V1000FG680-6. El diseño iterativo procesa a 165 Mbps a 21 MHz., mientras que el diseño con lazos desenrollados trabaja a 71.4 MHz con un procesamiento de 354 Mbps. Los diseños utilizan el 6% y 38% de recursos del FPGA que contiene 12288 *slices*.

Amphion presenta en [8] la implementación comercial del algoritmo MD5. El mayor procesamiento reportado es de 472 Mbps en un dispositivo Xilinx Virtex-II a 60 MHz, utilizando 844 *slices*. En [9] tiene la implementación del MD5 y del SHA-1 en un

FPGA Virtex-II, donde el MD5 procesa a 400 Mbps a 56 MHz. En [10] se tienen implementaciones comerciales del algoritmo MD5. Hi/fn Inc., ofrece varios procesadores alcanzando el 7811 el más alto procesamiento a 376 Mbps a una frecuencia de reloj de 90MHz.

SecuCore Consulting Services [11] ofrece *cores* comerciales usando tecnología ASIC de 0.18µm y trabajando a una frecuencia de reloj de 166 MHz. El procesamiento es de 1.25 Gbps usando 68 ciclos de reloj para el proceso de un bloque de 512 bits.

## IV. Implementación y Resultados

La estrategia de diseño se fundamentó en realizar un diseño que utilizará los mínimos recursos hardware del FPGA que proveyera información de *paths* críticos y de esta manera seguir estrategias de diseño para alcanzar el requerimiento de procesamiento de 1Gbps.

### 1. Diseño óptimo en recursos hardware

La primera implementación basada en un diseño modular estableció usar un mínimo de recursos hardware. El diagrama general se puede ver en la figura 3.

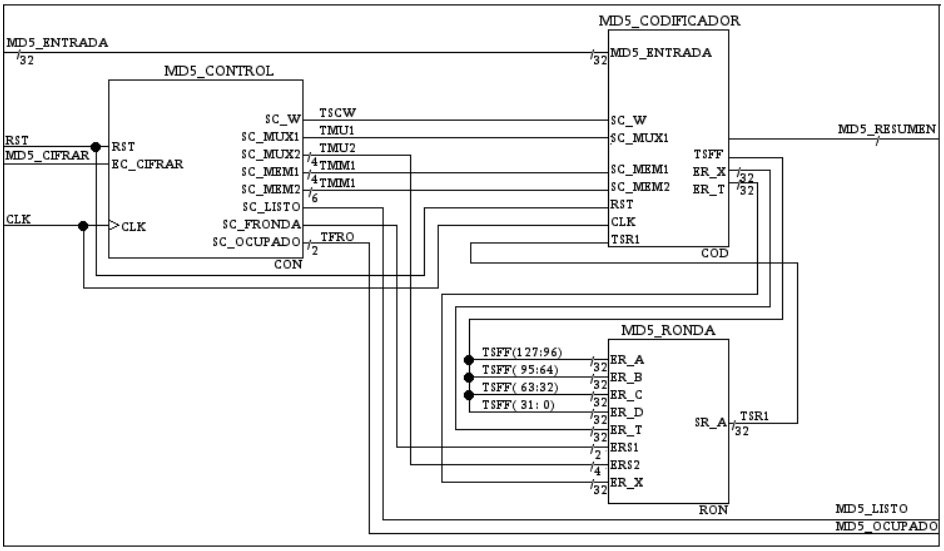


**Fig. 3.** Diagrama a bloques de la implementación del MD5.

A partir del esquema de la figura 3 fundamentado en el RFC-1321, se realizaron dos diseños diferentes de la unidad de control (basados en dos distintas máquinas de estados) y la colocación de registros. Estos diseños manejaban una latencia de 66 y 65

ciclos para el proceso de un sólo bloque de 512 bits, sin la realización del *padding* descrito en [1].

Los resultados de la implementación pueden verse en las tablas 1 y 2. Los diseños fueron escritos y simulados en Active-HDL e implementados en Xilinx ISE 6 para la medición de parámetros de hardware tales como uso de lógica y frecuencia de operación. Los diseños fueron sintetizados, mapeados, colocados y ruteados en un FPGA Xilinx XC2V1000-FG456 con la herramienta Xilinx ISE 6, además se creó un modelo de simulación "Post-Place & Route" que validó el funcionamiento de cada diseño mediante Active-HDL 5.1. Los datos de la tabla 1 son calculados en base a la información generada por los reportes del proceso Place & Route.

| Diseño | Frecuencia | IOBs | *Slices* | LUTs 4-E | Ciclos | Procesamiento |
|---|---|---|---|---|---|---|
| MD5v1 | 36.45MHz | 165/324 | 833/5120 | 1081/10240 | 66 | 282.81 Mbps |
| MD5v2 | 51.60MHz | 165/324 | 899/5120 | 1229/10240 | 65 | 406.48 Mbps |

**Tabla 1**. Resultados de la implementación.

El diseño MD5v1 es la implementación con latencia de 66 ciclos y la restricción de síntesis en velocidad. MD5v2 difiere de MD5v1 en el módulo unidad de control y la colocación de registros, con una latencia de 65 ciclos y una frecuencia de reloj más rápida. Estos diseños fueron conformados, en general, por los módulos que se ven en la tabla 2.

| Componentes | Descripción | MD5 Unibloque |
|---|---|---|
| AND2TO1_32BIT | Compuerta AND de 32 bits, 2 entradas. | 4 |
| NOT_32BIT | Compuerta NOT de 32 bits. | 2 |
| OR2TO1_32BIT | Compuerta OR de 32 bits, 2 entradas. | 3 |
| XOR2TO1_32BIT | Compuerta XOR de 32 bits, 2 entradas. | 3 |
| REGISTRO_32BIT | Registro de 32 bits. | 16 |
| REGISTRO_128BIT | Registro de 128 bits. | 1 |
| SUMADOR2TO1_32BIT | Sumador módulo $2^{32}$, 2 entradas. | 8 |
| MUX4TO1_32BIT | Multiplexor de 32 bits, 4 entradas. | 1 |
| MUX2TO1_128BIT | Multiplexor de 128 bits, 2 entradas. | 1 |
| BLOQUES RAM 64X32 | Memoria RAM 64x32. | 1 |
| FSM 7 ESTADOS | Máquina de estados. | 1 |
| CONTADOR_4BCD | Contador. | 1 |
| CONTADOR_64BCD | Contador. | 1 |

**Tabla 2**. Módulos utilizados en la implementación.

## 2. Diseño con réplica de lógica

Los resultados obtenidos de la primera implementación indican que el *path* crítico se encuentra en las rondas para el cálculo de los nuevos datos. Esta conclusión es la

misma que la reportada en [6], donde el *path* crítico se encuentra en el cálculo del nuevo dato:

$$A = B + ((A + f(B, C, D) + Xi + k1) ,<<< k2 )$$

El operando A representa la retroalimentación de la ronda para tener un *path* no optimizable al realizar un diseño con rondas desenrollados. Los otros operandos están disponibles y su suma puede ser hecha en paralelo con el cálculo de A, es decir, se puede tener una arquitectura parcialmente desenrollada. La ganancia de desenrollar una ronda es evitar el retardo de dos sumas y no de las cuatro sumas de una ronda completa, ver figura 4.

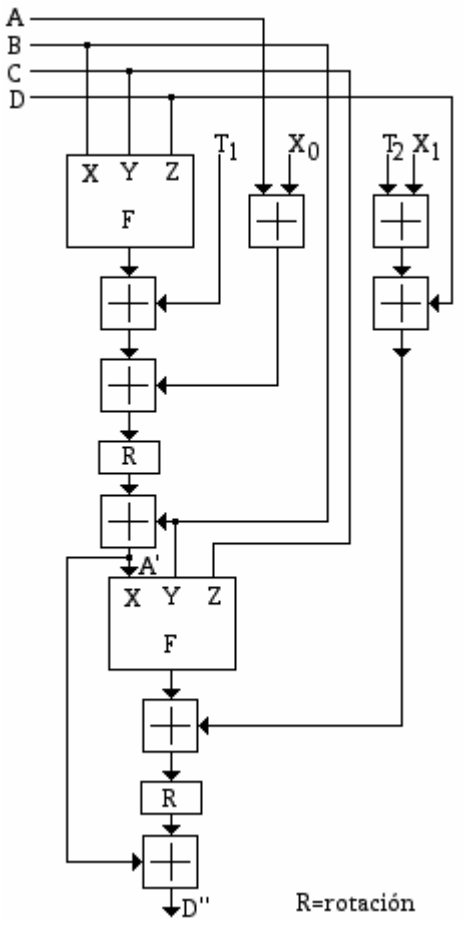


**Fig. 4**. Estructura de dos rondas desenrolladas

Las implementaciones usando esta técnica de replicación de lógica generaron aumentos en la capacidad de procesamiento de información. Los resultados de los reportes del Place & Route se pueden ver en la tabla 3.

| Diseño | Frecuencia | IOBs | *Slices* | LUTs 4-E | Ciclos | Procesamiento |
|---|---|---|---|---|---|---|
| MD5vA1 | 39.77MHz | 261/324 | 1101/5120 | 1181/10240 | 32 | 636.43 Mbps. |
| MD5vA2 | 26.10MHz | 261/324 | 1387/5120 | 2392/10240 | 16 | 835.37 Mbps. |
| MD5vA3 | 12.25MHz | 263/324 | 2404/5120 | 3855/10240 | 8 | 784.15 Mbps. |

**Tabla 3**. Resultados de la implementación en la segunda etapa.

MD5vA1 es la implementación con dos rondas desenrolladas y para el procesamiento se utiliza iterativamente durante 32 ciclos de reloj. MD5vA2 utiliza cuatro rondas desenrollados con una latencia de 16 ciclos y MD5vA3 usa ocho rondas desenrolladas en 8 ciclos de reloj.

Los resultados de las tablas 1 y 3 muestran que el uso de rondas desenrolladas aumenta la capacidad de procesamiento, pero llega a un límite ya que comienza a decrementarse, ver figura 5. Este decremento se debe al ruteo en el FPGA, porque los resultados de la síntesis indican que MD5vA2 procesa a 956.5 Mbps y MD5vA3 procesa a 1.146Gbps, es decir no debería haber decremento.


**Fig. 5**. Comparación de procesamiento de las diversas implementaciones.

El objetivo principal era alcanzar el procesamiento de 1Gbps para la implementación del algoritmo MD5 en un FPGA, por lo que la figura 5 indica que el diseño en base a rondas desenrolladas mediante réplica de lógica no es la solución.

## 3. Diseños con módulos especializados

El diseño MD5vA2 es la implementación que alcana el mayor procesamiento al manejar cuatro rondas desenrolladas, por lo que es diseño fundamental para mejorar la capacidad de procesamiento.

La idea principal es manejar funciones de compresión especiales conformadas de cuatro rondas desenrolladas, ya que es el módulo que mejor comportamiento presentó

en la implementación en un FPGA. Se debe evitar las unidades de corrimiento (son 16 valores de corrimientos).

Esta nueva descripción estructural se puede ver parcialmente en la figura 6. Esta figura muestra dos de las cuatro nuevas funciones rondas de compresión, las cuales están compuestas de cuatro rondas desenrolladas y las unidades de corrimiento quedan descritas como simples recolocaciones de buses de datos. Cada nueva ronda es utilizada cuatro veces para totalizar una latencia de 16 ciclos de reloj y como cada ronda consta de 4 rondas desenrolladas al final se tienen las 64 rondas descritas en [1].

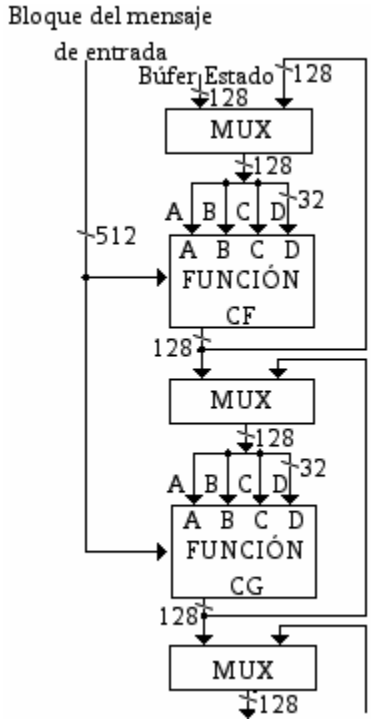


**Fig. 6**. Nueva descripción estructural.

De esta manera, se tienen las ventajas de las cuatro rondas desenrolladas (ver figura 4), se utilizan cuatro memorias RAM de 4x128 y además se evitan unidades de corrimiento que dependían del número de paso (64 pasos descritos en [1] con 16 valores de corrimientos).

Los resultados de la implementación de esta estructura se pueden ver en la tabla 4.

| Diseño | Frecuencia | IOBs | *Slices* | LUTs 4-E | Ciclos | Procesamiento |
|---|---|---|---|---|---|---|
| MD5vB | 32.47MHz | 261/324 | 2104/5120 | 3779/10240 | 16 | 1.039 Gbps |

**Tabla 4**. Resultados de la implementación en la tercera etapa.

La latencia para el procesamiento de un solo bloque de 512 bits es de 16 ciclos de reloj, ya que cada una de las cuatro funciones rondas son utilizadas cuatro veces (4x4 pasos) y cada función ronda se compone de cuatro rondas desenrolladas (4x4x4 pasos = 64 pasos). Los resultados de la síntesis reportan un período 25.395ns para obtener un procesamiento de 1.26 Gbps, pero la implementación del FPGA presenta un procesamiento de 1.039 Gbps.

La tabla 5 muestra resultados de las implementaciones revisadas en la sección III y del diseño MD5vB.El resultado de procesamiento de 1.039 Gbps es la mejor opción de una implementación en FPGA, ya que la búsqueda de información de trabajo relacionado no señala implementación alguna que sobrepase el gigabit por segundo. En cambio, la implementación en un ASIC de [11] procesa a 1.25 Gbps trabajando a 166 MHz. Si la implementación presentada en este artículo trabajará a 39.38 MHz (reportada en la síntesis) se tendría un procesamiento parecido (1.26Gbps) pero a una frecuencia de reloj menor.

| Diseño | Frecuencia (MHz) | Ciclos | Procesamiento (Gbps) |
|---|---|---|---|
| [5] - FPGA | 18.00 | 65 | 0.142 |
| [6] - FPGA | 60.20 | 66 | 0.467 |
| [7] - FPGA | 71.40 | 103 | 0.354 |
| [8] - FPGA | 60.00 | 65 | 0.472 |
| [9] - FPGA | 56.00 | 65 | 0.400 |
| [10] -FPGA | 90.00 | 123 | 0.376 |
| [11] - ASIC | 166.00 | 68 | 1.250 |
| Este trabajo | 32.47 | 16 | 1.039 |

**Tabla 5**. Resultados de la implementación en la tercera etapa.

## VI. Conclusiones

Este diseño es para el proceso de bloques de 512 bits, pero agregarle retroalimentación de la salida para procesos de multibloques no debe incluir retardos en el *path* crítico, ya que se registraría la salida inmediatamente.

La implementación en un FPGA del algoritmo MD5 presentado en este artículo ofrece la mejor opción respecto a la capacidad de procesamiento de 1.039Gbps.

Este módulo formará parte de la biblioteca de configuraciones de una plataforma reconfigurable criptográfica y la velocidad de procesamiento es una buena solución entre sistemas configurables.

## VII. Agradecimientos

Se agradece al CONACyT el apoyo otorgado a través de la Beca para Estudios de Maestría # 171489.

## Referencias

[1] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, MIT and RSA Data Security, Inc., Abril 1992.

[2] Stallings, W., “Cryptography and Network Security. Principles and Practices”, Prentice Hall, EUA, 2003.

[3] Lucena, M. J., “Criptografía y Seguridad en Computadores”, Libro Electrónico, Tercera Edición, Junio 2001, wwwdi.ujaen.es/~mlucena/lcripto.html.

[4] Ferguson N., Schneier B., “Practical Cryptography”, Wiley Publishing, EUA, 2003.

[5] Yong K. K., Dae W. K., Taek W. K., Jun R. C., “An Efficient Implementation of Hash Function Processor for IPSec”, The Third IEEE Asia-Pacific Conference on ASICs, Taipei, Taiwan, Agosto 2002.

[6] Diez J. M., Bojaniç S., Stanimiroviç Lj., Carreras C., Nieto-Taladriz O., “Hash Algorithms for Cryptographic Protocols: FPGA Implementations”, 10th Telecommunications Forum TELFOR’2002, Belgrade, Yugoslavia, Noviembre 2002.

[7] Deepakumara J., Heys H. M., Venkatesan., “FPGA Implementation of MD5 Hash Algorithm”, Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering (CCECE 2001), Toronto, Ontario, Mayo 2001.

[8] Amphion, Hoja de especificaciones, “CS5315 High Performance Message Digest 5 Algorithm (MD5) Core”, 2002. www.amphion-semi.com/acrobat/DS5315.pdf.

[9] Amphion, Hoja de especificaciones, “CS5315 High Performance SHA1/MD5 Hashing Algorithm Core”, 2002, www.amphion-semi.com/acrobat/DS5316.pdf.

[10] Hifn Inc., Hoja de especificaciones, “7811 Network Security Processor”, 2002, www.memec-impact.ch/pdf/hifn/7811.pdf.

[11] SecuCore Consulting Services, Hoja de especificaciones, “SecuCore SHA-1/MD5/HMAC Core”, 2001, www.secucore.com/secucore-hmac.pdf.