고속 모듈라 멱승 연산 프로세서

이성순*, 최광윤*, 이계호***, 김정호**, 한승조*

*조선대학교 전자정보통신공학부
**조선이공대학 전산과
***한국통신 인력 개발 본부

A High Speed Modular Exponentiation Processor

Seong Soon Lee, Kwang Yun Choi, Kye Ho Lee, Jung Ho Kim, Seung Jo Han

*School of Electronic and Information Communication Eng., Chosun University
**Dept. of Computer Science, Chosun College of Science and Technology
***Korea Telecommunication

요 약

RSA 암호 시스템에서 512비트 이상의 큰 정수 소수의 모듈라 멱승 연산이 필요하기 때문에 효율적인 암호화 및 복호화를 위해서는 모듈라 멱승 연산의 고속 처리가 필수적이다. 따라서 본 논문에서는 몫을 추정하여 모듈라 감소를 실행하고 carry-save 덧셈과 중간 곱의 크기를 제한하는 interleaved 모듈라 곱셈 및 감소 기법을 이용하여 모듈라 멱승 연산을 수행하는 고속 모듈라 멱승 연산 프로세서를 논리 자동 합성 기법을 바탕으로 하는 탑다운 설계 방식으로 VHDL을 이용하여 모델링하고 SYNOPSYS 툴을 이용하여 합성 및 검증한 후 XILINX XC4025 FPGA에 구현하여 성능을 평가 및 분석한다.

1. 서론

1978년 컴퓨터 시스템을 이용하여 구현될 수 있는 RSA 공개키 암호 시스템이 Rivest, Shamir와 Adleman에 의해 제안되었다. RSA 암호 시스템은 암호화와 전자서명 모두를 제공할 수 있고, 512비트 이상의 큰 정수의 소인수 분해가 어렵고 많은 계산을 요구한다는 사실에 안전도의 근간을 두고 있으며, 암호화 및 복호화 과정에서 모듈라 멱승 연산이 주된 연산으로서 모듈라 멱승 연산은 계층적으로 모듈라 곱셈과 모듈라 감소 연산으로

세분되고 전체 연산에서 모듈라 곱셈이 대부분을 차지한다. 또한 RSA 암호 시스템은 매우 큰 정수 소수의 멱승 및 모듈라 연산을 필요로 하고 모듈라 곱셈에서 많은 수의 메시지 블록이 연속적으로 계산되어야 하므로 암호화 및 복호화 때 처리 속도의 지연이 가장 큰 문제가 되므로 모듈라 멱승 연산의 고속화를 위한 하드웨어 구현이 필요하다.

본 논문에서는 암호화를 고속으로 처리하기 위한 관점에서 몫을 추정하여 모듈라 감소를 수행하고 중간 값의 크기를 제한하는 interleaved 모듈라 곱셈 방식을 이용하여 RSA의 주된 계산 방식인 정수의 모듈라 및 멱승 연산을 고속 처리하는 프로세서를 VHDL을 이용하여 모델링하고 EDA Tool을 사용하여 시뮬레이션 및 합성을 하여 FPGA에 구현하여 성능을 평가 및 분석한다.

본 논문의 구성은 다음과 같다. 2장에서 interleaved 모듈라 곱셈을 이용한 고속 모듈라 멱승 연산 프로세서의 하드웨어를 설계하고, 3장에서 VHDL을 이용한 설계 흐름과 FPGA에 구현된 프로세서의 성능을 분석하였으며, 제4장에서 결론을 맺고자 한다.

2. 하드웨어 설계

모듈라 멱승 연산을 고속으로 수행하기 위하여 본 논문에서 설계한 프로세서의 구조는 그림 1과 같이 전체 블록을 제어하기 위한 Control 블록, 데이터의 입출력을 위한 Data Interface 블록, 키와 입출력 텍스트를 저장하기 위한 Key/Text Shift Register, 모듈라 멱승 연산을 수행하는 Modular Exponentiation 블록으로 구성된다.

먼저 사용될 키를 먼저 입력 받아 저장하고 이 키를 이용하여 입력되는 데이터를 암·복호화한다. 내부에서 데이터는 512비트 단위로 처리되며 외부와의 입출력은 8비트 단위로 이루어지는 범용성 프로세서를 설계하였다.

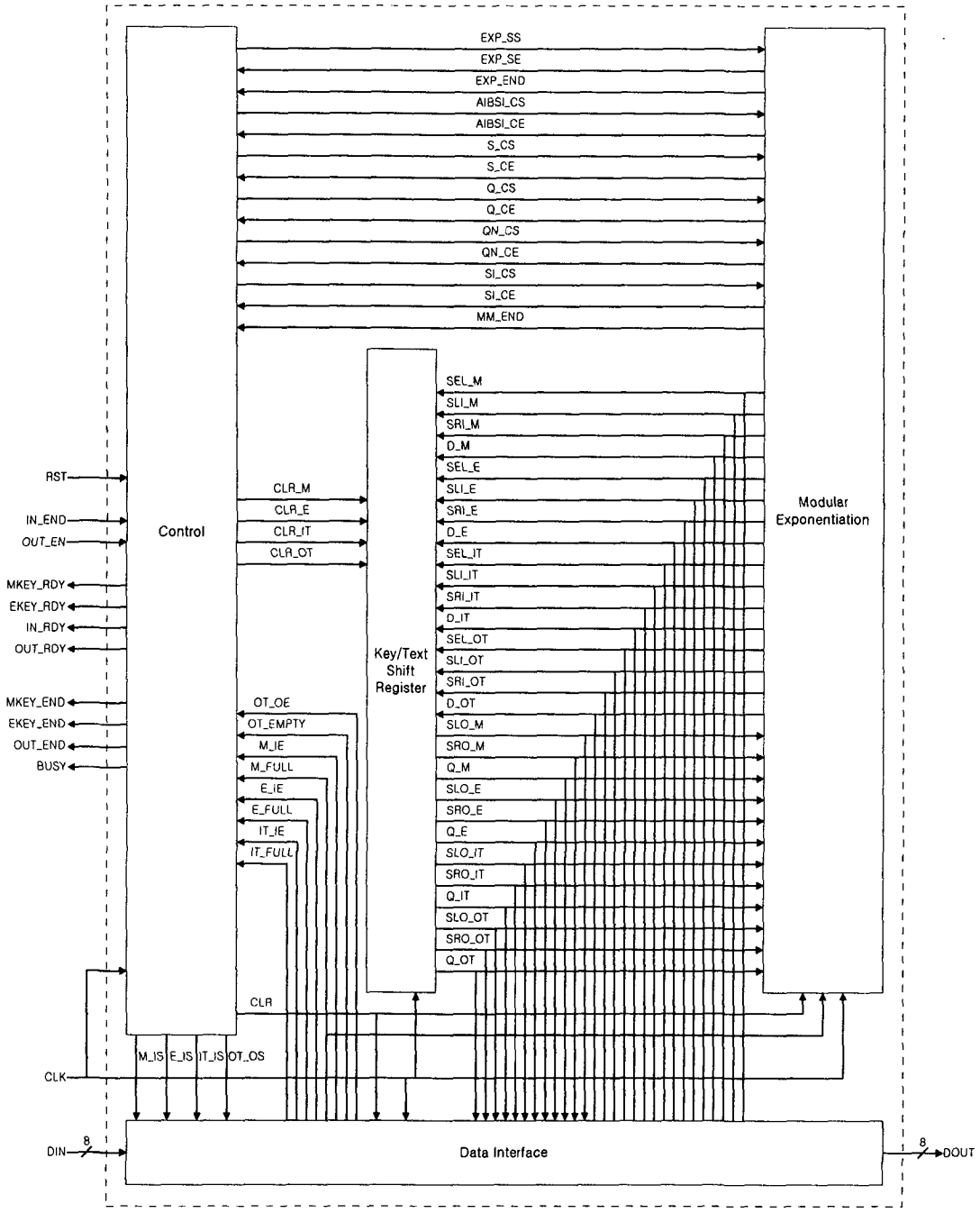

그림 1. 프로세서 설계 블록도

2.1. Control

Control 블록은 각 블록의 동작을 제어하기 위해 제어 신호를 생성한다. Control 블록을

FSM으로 설계하기 위한 상태 천이도는 그림 3과 같다.

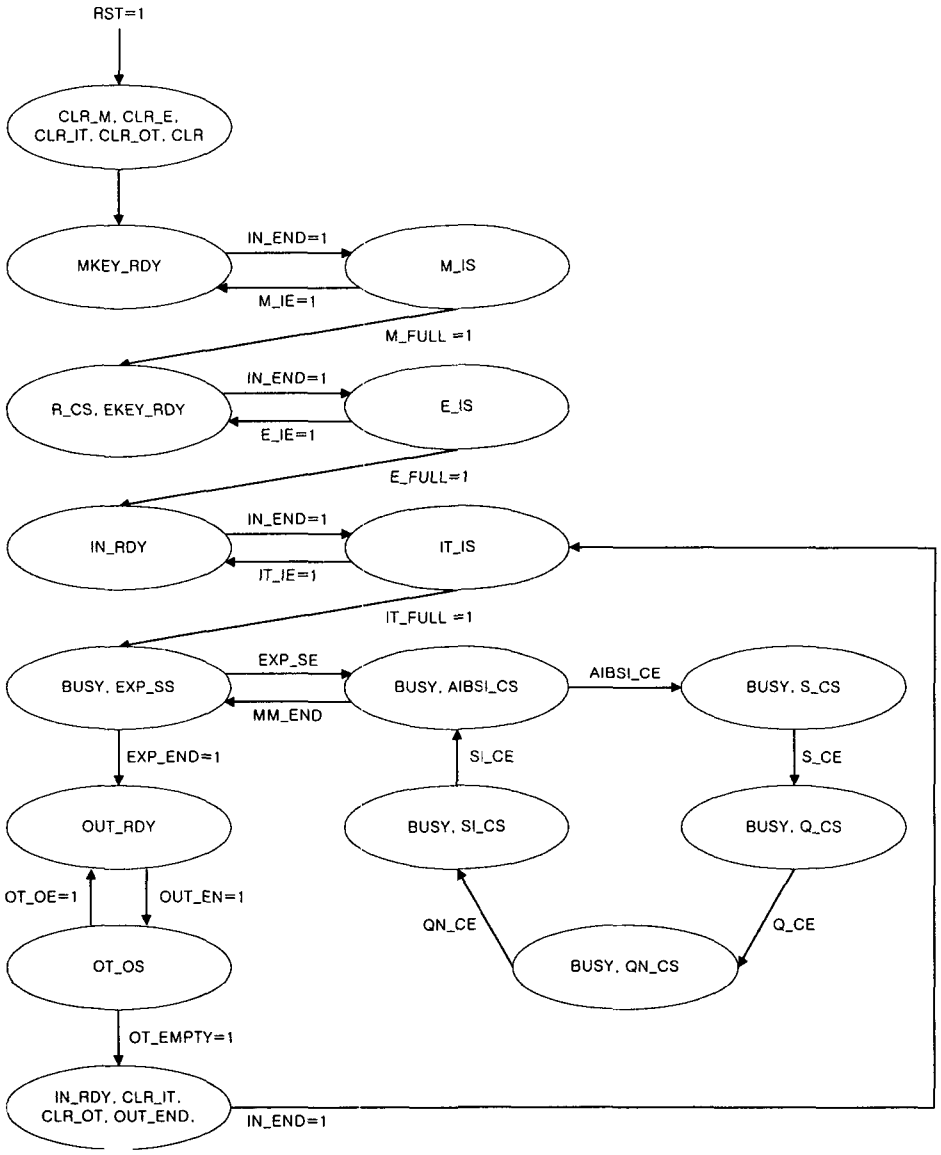

그림 2. Control 블록의 상태 천이도

2.2. Data Interface

Data Interface 블록은 암·복호화에 사용될 키 N, E를 입력 받아 Modulus Shift Register와 Exponent Shift Register에 저장하고 입력 텍스트를 Input Text Shift

Register에 저장하고 Output Text Shift Register의 값을 출력한다.

2.3. Key/Text Shift Register

512비트 RSA 연산($C = M^E \bmod N$)을 수행하는데 필요한 오퍼랜드를 저장하기 위해 네 개의 512비트 시프트 레지스터를 사용하였다. Modulus Shift Register에 공개키 N이 저장되고, Exponent Shift Register에 지수 E가 저장된다. Input Text Shift Register와 Output Text Shift Register에 각각 메시지 M과 암호문 C가 저장된다. Key/Text Shfit Register의 블록도는 그림 4와 같다.

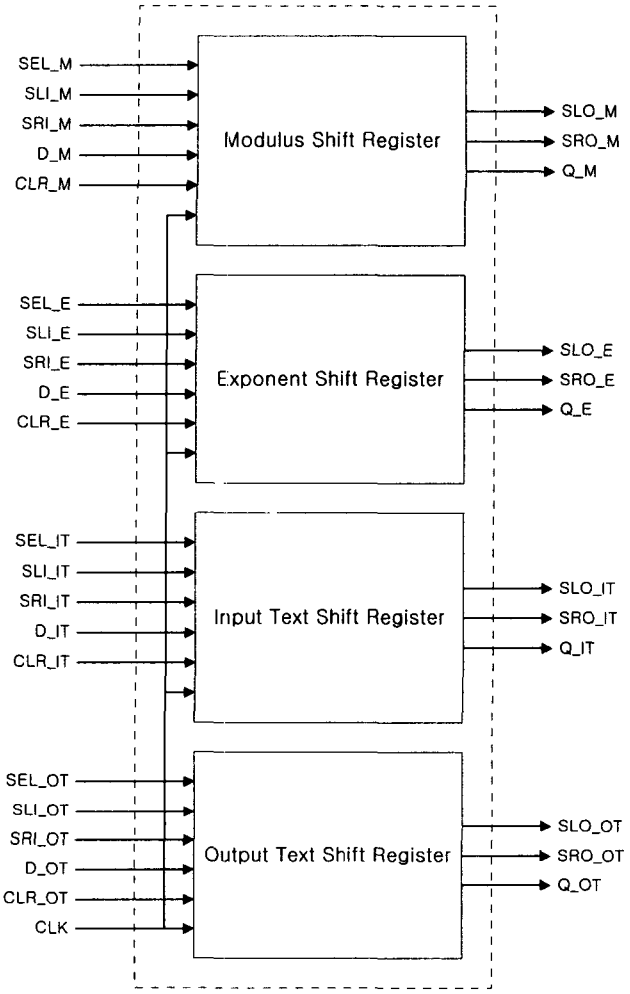

그림 3. Key/Text Shift Register 블록의 구조

2.4. Modular Exponentiation

RSA 암호 시스템을 구현하기 위한 가장 핵심적인 부분은 모듈라 멱승 연산으로, 이를 효율적으로 구현하는 것이 RSA 암호 시스템의 속도 향상에 있어 중요한 문제이다. 모듈라 멱승 연산은 모듈라 곱셈의 반복으로서, 전체 연산 시간을 단축시키기 위해서는 모듈라 곱셈의 수행 시간을 단축시키거나, 모듈라 곱셈의 반복 회수를 줄이는 것이 필요하다.

모듈라 곱셈의 반복 회수를 감소시키는 방법 중에서 잘 알려진 이진 알고리즘(binary algorithm)은 지수 E의 이진 표현에 근거하여 반복하여 제곱과 곱셈을 수행한다.

($E_{n-1}, E_{n-2}, E_{n-3}, \cdots\cdots, E_1, E_0$)를 지수 E의 이진 표현이라고 하면 지수 E는 n비트 길이의 수이고 E_{n-1}는 MSB(Most Significant Bit)가 되고 E_0은 LSB(Least Significant Bit)가 되며, 식 (3.1)은 지수 E의 이진 표현과 모듈라 멱승 연산을 나타낸다.

$$E = \sum_{i=0}^{n-1} E_i \cdot 2^i$$
$$C \equiv M^E \bmod N \equiv \prod_{i=0}^{n-1} M^{E_i \cdot 2^i} \bmod N \tag{3.1}$$

이진 알고리즘은 지수의 이진 표현을 왼쪽에서 오른쪽으로 검색하면서, 즉 MSB에서 LSB 쪽으로 검색하면서, 곱셈과 제곱을 수행하게 된다.

모듈라 멱승 계산이 수행되면 대단히 큰 수가 얻어지므로 이 큰 숫자를 저장하는데 소요되는 기억용량과 모듈라 감소 연산에 소요되는 계산 시간을 줄이기 위한 효율적인 모듈라 곱셈 방법이 요구된다. 본 논문에서는 모듈라 곱셈을 빠르게 수행하기 위하여 모듈라 곱셈을 다정도 곱셈과 모듈라 감소 연산으로 나누고 멱승 계산 도중에 생성되는 중간 곱에 대하여 모듈라 감소 연산을 수행한다.

모듈라 감소를 수행할 때 나눗셈 몫 q의 정확한 값을 계산하는 대신에 q의 값을 추정하는데, q의 값을 추정하기 위해 본 논문에서는 개선된 Barrett 알고리듬을 이용한다.

$n=|M|$이면, 몫 q는 다음 식 (4.5)와 같이 표현될 수 있다.

$$q = \lfloor \frac{U}{N} \rfloor = \lfloor \frac{\frac{U}{2^{n+\beta}} \frac{2^{n+\alpha}}{N}}{2^{\alpha-\beta}} \rfloor \tag{4.5}$$

몫의 추정은 다음과 같이 된다.

$$\hat{q} = \lfloor \frac{U}{N} \rfloor = \lfloor \frac{\lfloor \frac{U}{2^{n+\beta}} \rfloor \lfloor \frac{2^{n+\alpha}}{N} \rfloor}{2^{\alpha-\beta}} \rfloor \tag{4.6}$$

여기서

$$R = \lfloor \frac{2^{n+\alpha}}{N} \rfloor \tag{4.7}$$

는 고정된 모듈러스 N에 대해 상수가 되고 미리 계산될 수 있다. $|M| = n$이므로 $2^{\alpha} < R < 2^{\alpha+1}$이 된다. $\alpha = t+3$, $\beta = -2$인 개선된 Barrett 알고리즘을 interleaved 모듈라 곱셈에 사용한다.

Modular Exponentiation 블록의 구조는 그림 5와 같다.

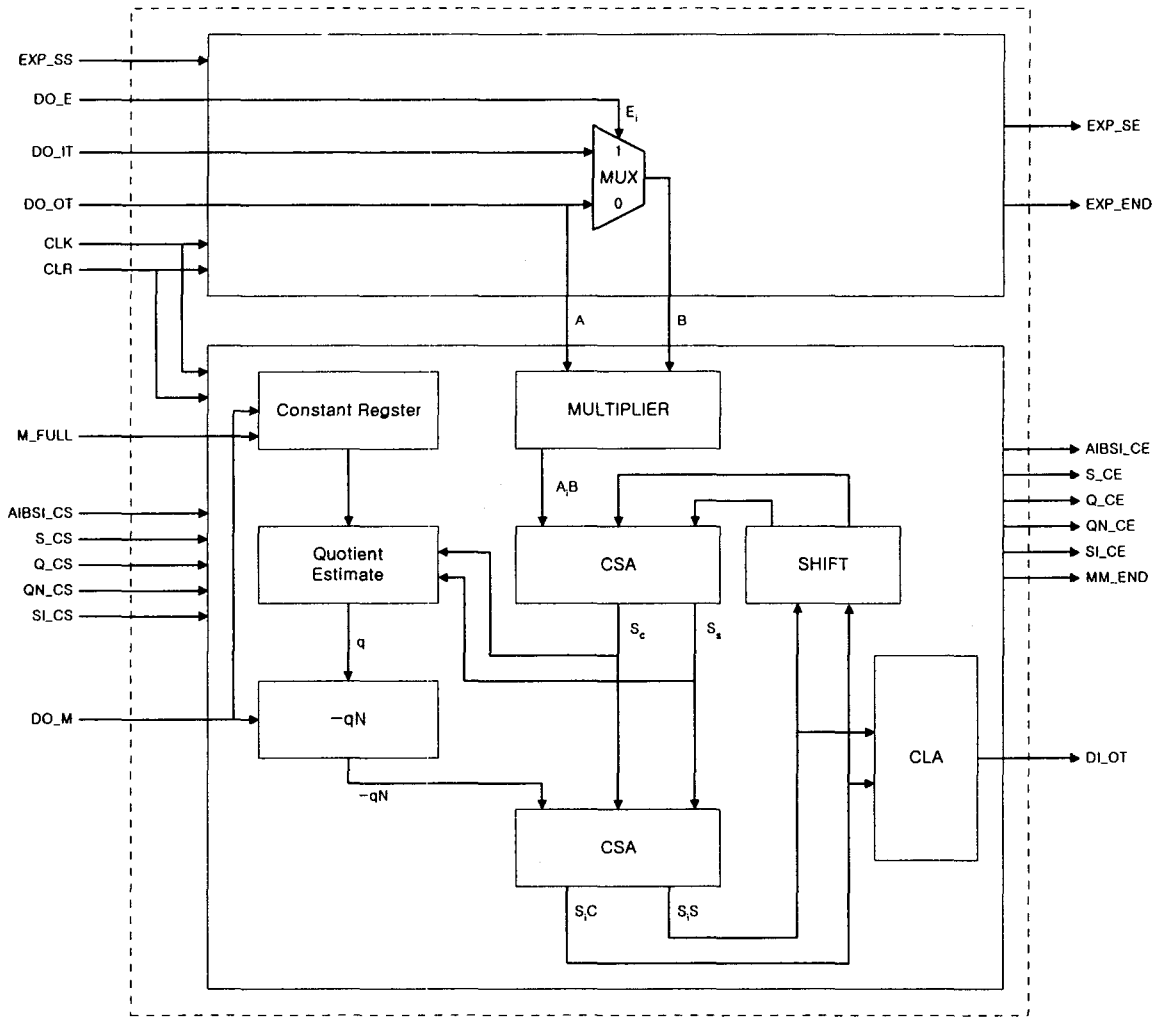

그림 4. Modular Exponentiation 블록의 구조

3. 설계 흐름 및 분석

FPGA를 이용한 구현은 미리 설계된 칩에 프로그램밍을 함으로써 원하는 기능의 회로를 구현하는데 목적이 있으며, IC 설계 제작에 비하여 소요되는 비용과 시간을 절감할 수 있을 뿐만 아니라, 원하는 회로에 대한 신속한 실험 제작(prototyping)이 가능하다는 이득이 있다. XILINX XC4025 FPGA는 조합 논리 회로와 메모리를 구현할 수 있는 CLB(Configurable Logic Block), 입출력을 위한 IOB(Input/Output Block), 그리고 CLB들과 IOB들을 서로 연결하여 회로를 구현하기 위한 배선 구조로 이루어져 있다.본 논문에서 사용한 탑다운 설계 방식의 설계 흐름은 그림 2와같다.

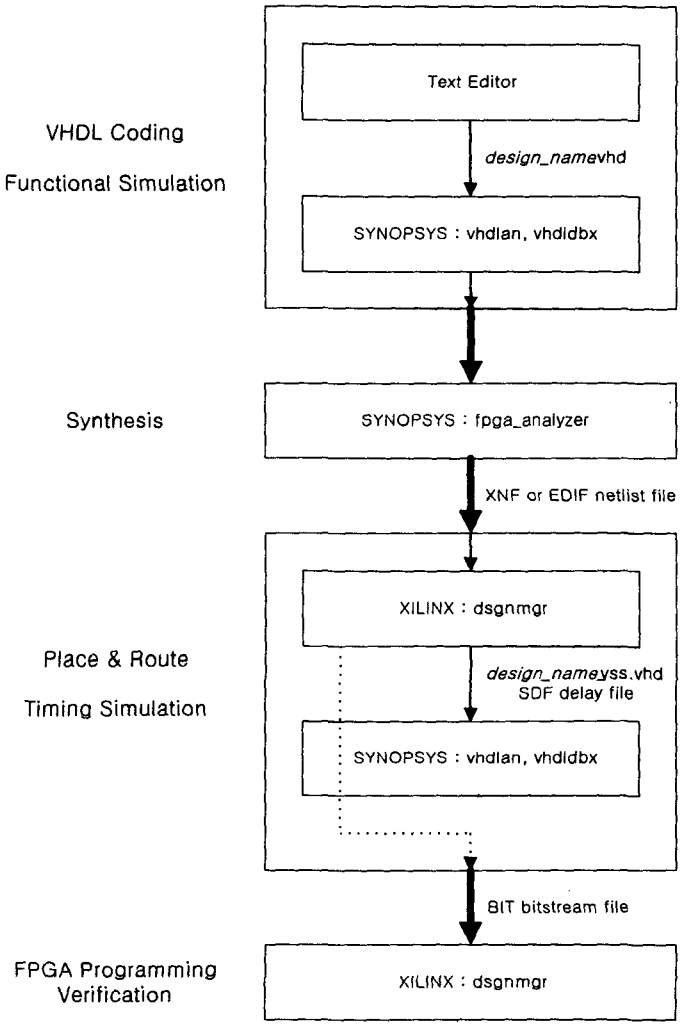

그림 5. 설계 흐름도

설계 초기 단계에서 VHDL 모델을 작성하고 SYNOPSYS사의 VHDL 컴파일러 vhdlan

을 사용하여 VHDL 소스 코드를 컴파일하였다. 이렇게 컴파일된 파일은 시뮬레이션 및 다른 여러 정보를 가지고 있는 목적 파일들을 생성한다. 컴파일을 거쳐 생성된 파일을 이용하여 VHDL 시뮬레이터 vhdldbx를 사용하여 기능 시뮬레이션을 수행하였고, 설계에 부합되는 동작이 검증된 후 FPGA 컴파일러 fpga_analyzer를 사용하여 합성하였다. 이렇게 합성된 설계를 XILINX 배선 및 배치 툴 dsgnmgr을 사용하여 배선 및 배선을 수행하였고, 생성된 SDF 파일과 VHDL 네트리스트 파일을 VHDL 시뮬레이터에 입력하고 XILINX VITAL 라이브러리를 이용하여 타이밍 시뮬레이션을 수행하였다. 타이밍 시뮬레이션 후 생성된 비트스트림 파일을 이용하여 XILINX사의 XC4025 FPGA를 프로그래밍하고 테스트하였다.

4. 결론

RSA 암호 시스템은 매우 큰 정수 소수의 멱승 및 모듈라 연산을 필요로 하고 모듈라 곱셈에서 많은 수의 메시지 블록이 연속적으로 계산되어야 하므로 암호화 및 복호화 때 처리 속도의 지연이 가장 큰 문제가 되므로 모듈라 멱승 연산의 고속화를 위한 하드웨어 구현이 필요하다. 따라서 본 논문에서는 파이프라인 모듈라 멱승 연산 기법과 몫 추정 모듈라 곱셈 기법을 이용하고 carry-save 덧셈을 이용하여 모듈라 멱승 연산을 수행하는 고속 모듈라 멱승 연산 프로세서를 VHDL을 사용하여 설계하고 합성 및 검증하여 XILINX XC4025 FPGA에 구현하였다.

참고문헌

[1] William Stallings, "Network and Internetwork Security Principles and practice," IEEE PRESS, 1995.

[2] Bruce Schneier, "Applied Cryptography", pp 466-474, John Wiely & Sons, Inc.

[3] Arto Salomaa, "Public-Key Cryptography", pp 125-157, Splinger-verlag.

[4] N. Takagi and S. Yajima, "Modular multiplication hardware algorithms with a redundant representation and their application to RSA cryptosystem," IEEE transaction on Computers, vol. 41, no. 7, pp. 887-891, July 1992.

[5] P. L. Montgomery, "Modular multiplication without trial division," Mathematics of Computation, vol. 42, no. 3, pp. 376-378, Mar, 1993.

[6] Ernest F. Brickell. A Survey of Hardware Implementation of RSA. In CRYTO '89, 1989.

[7] Holger. Orup, "Simplifying quotient determination in high-radix modular

multiplication," in Proceedings of the 12th Symposium on Computer Arithmetic, pp. 193-9, July 1995.

[8] Ishii. S., Ohyama. K., Yamanaka. K., "A single-chip RSA processor implemented in a 0.5 ㎛ rule gate array," in Proceedings of 7th Annual IEEE International ASIC Conference and Exhibit, pp. 433-6, 1994.

[9] S.Y.Kung, VLSI array processors. Prentice-Hall, 1988.

[10] Z. Navadi, "Using VHDL for model and design of processing unit," Proceeding of the 1992 ASIC Conf. and Ex., pp.315-326. 1992.

[11] James R. Armstrong & F. Gail, Structured Logic Design with VHDL, Prentice-Hall.

[12] David R. Coelho, The VHDL Handbook, Kluwer Academic Publishers, Norwell, Massachusetts, 1989.

[13] Roger Lipsett, Carl Schaefer, Cary Ussery, VHDL : Hardware Description and Design, Kluwer Academic Publishers, Norwell, Massachusetts, 1989.