암호화 칩의 설계 및 시뮬레이션에 관한 연구

(A study on the design and simulation of an encryption chip)

류승석*, 오재곤, 정연모
경희대학교 공과대학 전자공학과

요 약

본 논문에서는 암호화 알고리즘의 하나인 GOST (Government Standard)를 칩으로 구현했을 경우에 차지하는 면적과 속도에 대해 DES와 비교 분석하고, GDES의 구조를 이용하여 GOST 알고리즘을 빠르게 처리할 수 있도록 설계하였다. 합성한 것을 최종적으로 MAX+plus II를 이용하여 시뮬레이션을 통해 검증하였다.

I 서 론

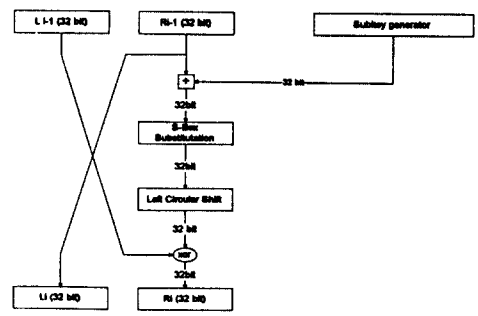
정보화 사회를 맞이하여 개인 정보의 보호는 중요성을 더해가고 있다. 그러나 최근에는 개인 정보의 유출 사례가 빈번하게 발생하여 문제가 되는 경우가 많이 있다. 이런 중요한 정보는 암호화 과정을 통하면 타인의 접근이 어려워지므로 보안성이 향상된다. 자료의 암호화에 대한 연구는 주로 기존의 CPU를 이용하고 있으며 실질적인 하드웨어 칩을 이용한 접근은 많지 않다. 최근에 인터넷을 이용한 전자 상거래가 많아지면서 사용자에게 암호화된 대량의 데이터가 전송되는 경우가 많이 발생한다. 이때, CPU가 복호화(decryption)에 사용된다면 부하가 과중하게 되어 CPU 속도가 느려진다. 이를 위해서 별도의 암호화/복호화 칩을 이용하면 CPU의 부하를 상당히 줄일 수 있을 것이다.

미국 정부가 암호화의 표준으로 사용하고 있는 DES(Data Encryption Standard)알고리즘은 안정성 면에서 아직까지 인정받고 있다. 하지만 56비트의 key 값을 사용하는 이 알고리즘을 key 크기 면에서 작기 때문에 공격(attack) 당하기 쉽다. 그래서 DES가 발표된 후 이러한 문제점을 보완하려는 노력이 계속 연구되었다. 본 논문에서는 DES 알고리즘과 유사한 구조를 가지면서 key 값이 증가된 GOST(Government Standard)알고리즘을 칩으로 구현했을 경우에 차지하는 면적과 속도에 대해 DES(Data Encryption Standard)와 비교 분석하였다. 또 GDES(General DES)[3] 구조를 기본 GOST(64비트)알고리즘을 적용하여 한번에 처리하는 데이터 양을 증가시켰다. 즉 GOST에서 64비트 단위로 처리하던 것을 128비트로 처리할 수 있는 구조로 변환하였다. 최종적으로 이 구조를 Altera 툴인 MAX+plus II를 이용하여 시뮬레이션을 통해 검증하였다.

II GOST 알고리즘의 이론적 배경

DES 알고리즘은 블록 암호화 알고리즘의 가장 기본 되는 표준으로서 어느 정도 안전성에 대해서는 인정받고 있다. 하지만 방대한 양의 자료를 처리해야 하는 경우에 속도를 고려하고 불법적인 해독을 방지하기 위해 key의 크기를 증가해야 하는 경우에는 한 쪽을 희생해야 되는 경우가 발생한다. 따라서 DES 알고리즘이 발표가 된 이후 속도를 빠르게 하면서 보안적인 측면의 강화에 대한 연구가 활발히 진행되어 왔다.

GOST는 구 소련의 정부 표준으로서 28147-89라는 표준 번호를 가지고 있다. GOST 알고리즘은 64비트로 이루어진 블록을 한꺼번에 암호화하는 블록 암호화 방식 중의 하나로서 256비트의 key값을 사용하여 암호화한다. <그림 1>과 같이 연산하는 과정의 round가 32회 반복되는데 이 알고리즘은 DES보다 key값이 상대적으로 많지만, round부분이 덜 복잡하다.



<그림 1> GOST 알고리즘의 round

64비트의 입력 값을 32비트씩 둘로 나누어 왼쪽(L_{i-1})과 오른쪽(R_{i-1})에 제공한다. 오른쪽 32비트 R_{i-1}과 256비트의 key값 중 32비트를 선정하여 두 개의 32비트를 덧셈연산을 한다. 덧셈연산 결과 중 carry는 무시하고 다음과정에 32비트를 제공한다. 앞으로 carry를 무시한 덧셈 연산과정을 Modulo 2^{32} Adder라 부르기로 한다. 또 256비트의 key값에서 32비트를 선정하는 과정을 subkey generator라고 부른다. Modulo 2^{32} 과정을 거친 다음 불규칙적인 배치를 하는 S-Box substitution이라는 과정을 거친 후 32비트의 값을 왼쪽으로 회전시키는 left circular shift과정을 거친다. 최종적으로 shift후 결과 값과 L_{i-1}과 xor 과정을 실행하여 R_i에 저장하고 L_i에는 R_{i-1}값을 저장한다.

위와 같은 과정을 round라 하고 총 32 반복하는 것이 GOST 알고리즘이다.

DES와 GOST의 기본적인 구조는 다음과 같은 차이점을 가지고 있다.

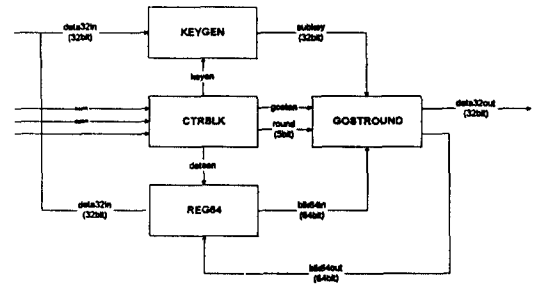
(1) DES는 입력된 key 값에서 subkey를 생성하는데 복잡한 과정을 통해 얻어지는 반면 GOST는 간단한 과정을 통해 생성된다.
(2) DES는 56비트의 key 값을 가진 반면 GOST는 256비트의 key 값을 가진다.
(3) DES에서 S-box는 6비트를 입력하여 4비트를 출력하는데 비하여 GOST의 S-box는 4비트를 입력받아 4비트를 출력한다.
(4) DES는 P-box라 불리는 불규칙적 치환 과정을 거치지만 GOST는 11비트 왼쪽으로 rotation하는 과정이 있다.
(5) DES는 16개의 round를 거치지만 GOST는 총 32 round를 통해서 결과 값이 나온다.

III GOST 칩의 구성

GOST 알고리즘을 각 블록으로 나누어 칩으로 구성하였다. 또 기본적인 GOST구조를 GDES와 같은 구조로 확장하여 한번에 처리할 수 있는 데이터 양을 많게 처리할 수 있는 구조로 설계하였다.

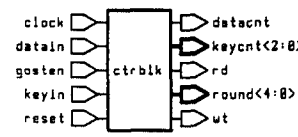
3.1 기본적인 GOST 구조

GOST 칩은 <그림 2>처럼 크게 CTRBLK 블록, REG64 블록, KEYGEN 블록, GOSTROUND 블록으로 나눌 수 있다. 칩 외부로부터 클럭을 입력해 주는 clock(1비트), 암호화와 복호화를 알리는 EnDec(1비트), 칩 전체를 초기화하는 reset(1비트), key값과 데이터 값을 입력받는 data32in(32비트), 64비트의 값이 key값임을 알리는 keyin(1비트), 64 비트의 값이 데이터 값임을 알리는 datain(1비트)의 입력 값과 최종적으로 암호화 또는 복호화 값을 내보내는 data32out(32비트)의 출력 값 그리고 데이터의 출력을 알리는 dataout(1비트)로 구성되어있다. 64비트의 데이터 값은 data32in으로부터 2번에 나누어 입력받고 256비트의 key값은 data32in으로부터 8번에 나누어 입력받는다. 또 암호화/복호화 된 출력 값은 data32out으로 2번에 나누어 출력된다. 각 블록에 대한 기능은 다음과 같다.



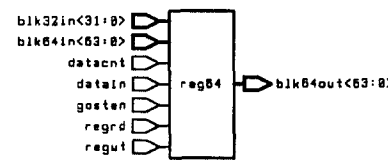
<그림 2> GOST 칩의 구성

CTRBLK은 KEYGEN 와 REG64 블록에 데이터를 저장할 것인지 읽어올 것인지를 여부를 제어하고 매 round마다 GOSTROUND에 제어신호를 제공하는 역할을 한다.



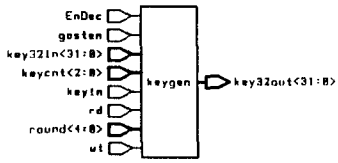
<그림 3> CTRBLK의 블록도

REG64 블록은 외부로부터 읽어들인 64비트의 데이터 값을 저장하고 매 round마다 값을 GOSTROUND 블록에 데이터 값을 공급해주고 매 round를 거쳐 나온 값을 저장한다.



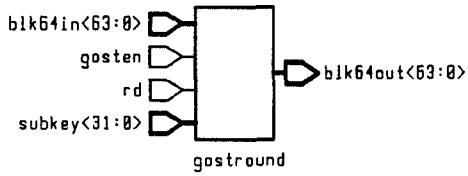
<그림 4> REG64의 블록도

KEYGEN 블록은 외부로부터 읽어들인 256비트의 key값을 저장하고 매 round마다 32비트의 key값을 공급한다.



<그림 5> KEYGEN의 블록도

GOSTROUND 블록은 GOST 암호화 칩의 핵심으로 실질적인 암호화 과정의 부분이다. 64비트의 데이터 값 blk64in<63:0>을 REG64로부터 받고 KEYGEN부터 32 비트의 subkey 값 subkey<31:0>을 입력받아 암호화를 한다. 또 CTRBLK 블록의 rd 신호를 이용하여 새로운 값을 읽어오고 결과 값을 REG64에 보낸다

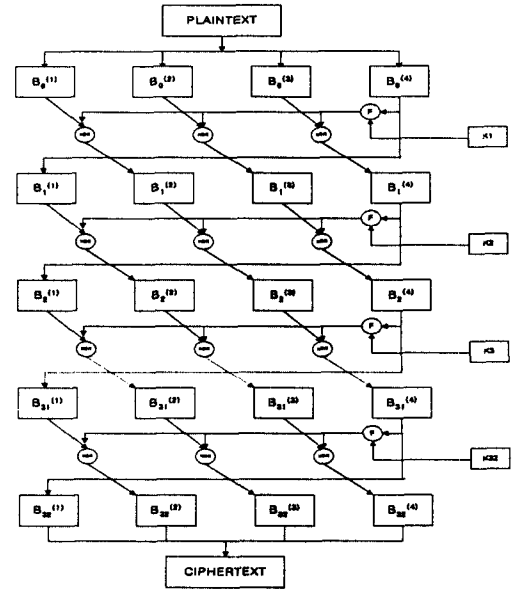


<그림 6> GOSTROUND의 블록도

이 블록은 앞에서 설명했던 것처럼 크게 Modulo 2^{32} Adder, S-Box, Left Circular Shift 부분으로 나눌 수 있다. 이 세 부분이 GOST알고리즘의 가장 핵심이며 칩으로 구현하였을 때 가장 지연이 많이 되는 요소이다. 앞으로 특별히 이 부분을 축약하여 " F "라 하기로 한다.

3.2 일반화시킨 GOST 구조

<그림 7>은 앞에서 구현한 구조를 기본으로 하여 한 번에 처리할 수 있는 데이터 처리 양을 두 배로 증가시킨 구조이다. 즉 입력 데이터 값을 64비트에서 128비트로 증가시켜 한번에 처리할 수 있는 능력을 두 배로 증가시켰다. 단순히 GOST를 나란히 두 번 배열하는 경우는 GOSTROUND 부분이 2개가 필요하지만 여기에서는 GOSTROUND를 부분을 하나로 구성하고 하나의 32 비트 블록에만 적용하였다. 즉, 128비트 값을 32비트 씩 네 개로 나누어 가장 우측에 있는 32비트만을 F(Modulo 2^{32} Adder -> S-Box -> Left Circular Shift)연산 과정에 적용시킨 형태이다. 이 과정에 의해 나온 결과 값을 각각의 나머지 32비트 블록에 XOR 시켜 하나의 round를 처리한다. 이런 방법을 이용해서 계속적으로 입력 데이터를 증가시키면 처리 량을 증가되지만 32비트 블록에 적용되는 GOSTROUND 회수가 적어지므로 상대적으로 데이터를 불규칙적으로 흩어지게 하는 round 과정이 줄어 암호화가 약해지기도 한다. 이것을 이용하면 필요에 따라서 빨리 처리하길 원하는 경우에 활용할 수 있다.



<그림 7> 128비트 입력의 GOST 칩

IV 결 과

4.1 합성결과

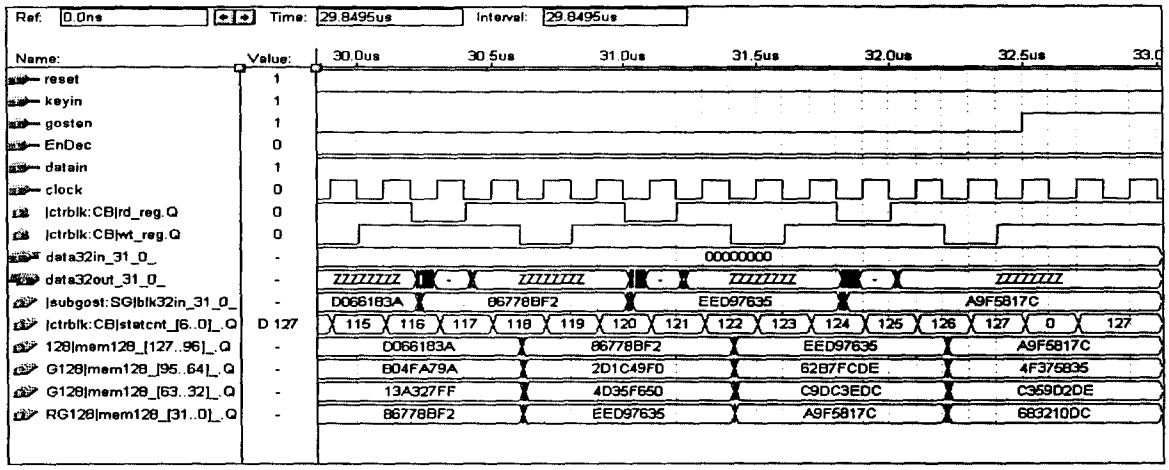
합성은 samsung의 KG75000 SOG cell library를 이용하여 synopsys analyzer을 사용하여 합성하였다. 합성결과 <표 1> 같은 결과 값을 얻을 수 있었다. 전체적으로 볼 때 KEYGEN 부분이 256 비트의 저장공간이 필요하여 가장 많은 면적을 차지함을 볼 수 있으며 그리고 다음으로 핵심인 GOSTROUND부분이 가장 많은 면적을 차지함을 볼 수 있다.

<표 1> 합성된 GOST 칩의 결과 값

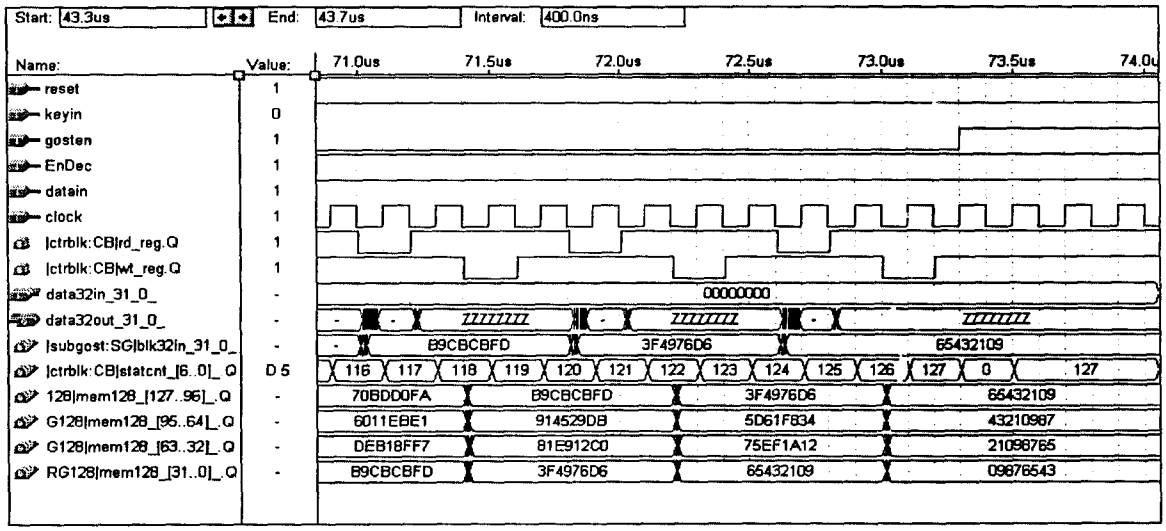
	Cell Area	Combination	Non-combination
CTRBLK	165	88	77
GOSTROUND	1603	803	800
KEYGEN	2415	1135	1280
REG128	1376	736	640
Total	5626	2765	2861

4.2 Round블록의 비교

DES와 GOST의 암호화 핵심부분인 round 블록부분을 비교하면 <표 2>과 같다. 한 개의 round를 구성하는데 면적 측면에서 본다면 GOST(64비트 입력)가 DES보다 50% 정도 적게 사용됨을 볼 수 있다. 또 128 비트 입력인 경우 면적이 64비트를 두개 사용한 것의 75% 정도만



<그림 8> 암호화된 결과 값



<그림 9> 복호화된 결과 값

사용했다. 지연 측면에서 GOST(128비트)의 지연과 GOST(64비트)를 비교해 보면 거의 차이가 없음을 알 수 있다. 즉 속도 지연 없이 한번에 처리할 수 있는 양을 증가시킬 수 있음을 볼 수 있다.

<표 2> DES와 GOST의 비교

	DES	GOST (64비트)	GOST (128비트)
cell Area	2084	1080	1610
data arrival time	9.07	19.95	20.28

4.3 시뮬레이션 결과

본 논문에서 제시한 구조에 대하여 구현된 ALTERA flex8000 칩에 256비트의 key(12345678 90123456 78901234 56789012 34567890 12345678 90123456 78901234 : HEX)값과 128비트의 데이터(09876543 21098765 43210987 65432109 : HEX)를 입력하면 <그림 8>과 같이 128비트의 출력 값 데이터(683210DC C359D2DE 4F375835 A9F5817C : HEX)값을 얻을 수 있었다. 이것을 다시 입력하고 복호화를 하면 <그림 9>와 같이 암호화 이전에 입력된 값(09876543 21098765 43210987 65432109 :HEX)을 얻어 제시된 구조가 정확하게 수행되고 있음을 확인할 수 있었다.

일반적으로 한문서나 기타 자료에서 한글이면 한글, 영어면 영어 중에서 자주 사용되는 문자가 많은데 이런 편중된 데이터가 입력되었을 때 암호화 알고리즘이 어느 정도 분산시키는지에 대해 분석하기 위해서 다음과 같은 임의의 값을 입력하였다.

사용 key값 : 12345678 90123456 78901234 56789012

(HEX) 34567890 12345678 90123456 78901234

입력 데이터 : 00000000 00000000 00000000 00000000
(HEX) 22222222 22222222 22222222 22222222
33333333 33333333 33333333 33333333
bbbbbbbb bbbbbbbb bbbbbbbb bbbbbbbb
ffffffff ffffffff ffffffff ffffffff
09876543 21098765 43210987 65432109

출력 데이터 : cbac7204 662eeb1f c312650e 7e008c35
(HEX) 33358d78 95d947fd 112a6384 73430ce6
a31395b9 06a88f1e 63d18244 33de7551
c98512bb adeebc46 1bb01e04 ddd7291f
eaa4635c 34cf83d2 1a22a303 093c7b15
683210de c359d2de 4f375835 a9f5817c

<표 3> 입력값의 분포

입력값	0	1	2	3	4	5	6	7
총수	36	3	35	35	3	3	3	3
입력값	8	9	A	B	C	D	E	F
총수	3	4	0	32	0	0	0	32

<표 4> 출력값의 분포

출력값	0	1	2	3	4	5	6	7
총수	11	16	12	23	11	14	10	11
출력값	8	9	A	B	C	D	E	F
총수	11	8	10	10	13	13	12	7

<표 3>와 같이 편중된 값이 입력되더라도 <표 4>과 같이 입력된 값에 비하여 분산된 분포를 가진 출력 값을 얻을 수 있었다. 이것을 통해 S-BOX의 분산기능이 정상적으로 동작함을 볼 수 있었다.

V 결 론

암호에 대한 연구는 오래 전부터 연구되었지만 군사적 목적이나 안보적 목적으로 주로 사용되었다. 최근에 컴퓨터와 통신이 급속도로 발전하면서 상업적, 개인적 목적으로 빈번하게 사용되고 있다.

이에 대한 연구가 계속되는 것에 비례하여 암호를 불법적으로 해독하려는 기술 또한 발전하였다. 암호화한 것을 해독하기 위해서는 얼마나 많은 시간과 비용을 들여서 현실성 있게 할 수 있는가 하는 것이 문제지 해독이 완전히 불가능한 암호화 알고리즘은 아직까지 없다. 즉, 해독하려는 데이터가 시간과 비용을 투자하여 그 이상의 가치를 얻을 수 있는가 하는 것이 관건이다. 앞에서 언급한 비밀 key 암호화 방식으로는 DES알고리즘 그리고 공개 key 암호화 알고리즘으로는 RSA알고리즘이 가장 널리 사용되고 있다. DES 알고리즘은 보안성 면에서는 어느 정도 인정받고 있지만 앞으로 계속 사용될 경우 key 크기 면에서 불안한 요소가 있다.

본 논문에서는 key의 크기가 DES에 사용되는 것보다 4배가 큰 GOST 알고리즘을 이용하여 칩을 구성하여 보안적인 면을 강화하였다. 또 기본 GOST 알고리즘을 개선하여 한번에 처리할 수 있는 데이터 양을 두 배(128 비트)로 처리할 수 있는 구조를 제시하고 ALTERA의 Max+plus II를 이용하여 flex8000 시리즈 칩에 적용하여 기본 GOST 알고리즘과 비교 분석하였다.

구현 결과에 따르면 DES보다 round측면에서 칩 크기를 반정도로 줄일 수 있음을 알 수 있었다. 또 확장된 구조가 기본 GOST구조와 지연 측면에서 거의 차이가 없음을 확인할 수 있다.

앞으로의 연구방향은 기존의 GOST round에서 사용되는 S-box의 table 값을 보다 불규칙적으로 유도하고 공개 key 암호화 알고리즘인 RSA알고리즘과 연계하여 공개 key 암호화 알고리즘과 비밀 key 암호화 알고리즘은 하나의 칩에 구현하는 것에 대해 연구하고자 한다.

<참고문헌>

[1] Altera, *Max+plus II getting started*, Atera corporation, 1995.

[2] Artech House Boston · London, *Smart Cards*, Artech House Inc.

[3] Bruce Schneier, *Applied Cryptography*, John Wiley & Sons, Inc 1996.

[4] GOST, Gosudarstvennyi Standard 28147-89, *Cryptographic Protection for Data Processing Systems*, Government Committee of the USSR for Standards 1989.

[5] Warwick Ford, Computer Communications Security, Prentice-Hall International Inc. 1994.

[6] 류승석 · 정연모, "DES 알고리즘을 이용한 암호화 칩 설계", 경희대학교 산학협력기술연구논문집, 제2집, pp195-199. 1996.

[7] 류승석 "Internet data 암호에 관한 연구", 경희대학교 전자공학과 연구집, 제14권 제 1호, pp141-145. 1996.

[8] 류승석 · 오재곤 · 정연모, "GOST 알고리즘을 이용한 암호화 칩 설계에 관한 연구", CAD 및 VLSI 설계 연구회 학술발표회 논문집, 1997.