블루투스 암호화 모듈의 설계 및 구현

황 선 원, 조 성, 안 진 우, 이 상 훈, 신 위 재
경남대학교 전기전자공학부

Design and Implementation of a Bluetooth Encryption Module

Sun-Won Hwang, Sung Cho, Jin-Woo Ahn, Sang-Hoon Lee, Wee-Jae Shin
Div. of Electrical & Electronic Engineering, Kyungnam University
E-Mail: sanghoon@kyungnam.ac.kr

요 약

본 논문에서는 블루투스 장비 간 암호화를 위해 사용되는 암호화 모듈의 설계 및 구현에 관한 내용을 다룬다. 암호화 모듈은 기저 대역내에 암호화 키 생성 모듈과 암호화 엔진 모듈로 구성된다. 암호화 키 생성 모듈은 Cylink사에서 제안한 공개 도메인인 SAFER+(Secure And Fast Encryption Routine) 알고리즘을 사용하여 128bit 키를 생성한다. 그 구성은 키 치환을 위한 치환 함수(key-controlled substitution)와 선형 변환을 위한 PHT(Pseudo-Hadamard Transform)와 Armenian Shuffle 변환기로 구성된다. 암호화 엔진 모듈은 전송 패킷내의 페이로드 데이터와 생성된 사이퍼 키 스트림 데이터와 XOR연산을 통하여 암호화를 행하며 그 구성은 LFSR (Linear Feedback Shift Register)와 합 결합기로 구성된다. 이 중 암호화 키 생성 모듈은 LM(Link Manager)의 PDU(Protocol Data Unit) 패킷을 통해 상호 정보가 교환되므로 암호화 키를 생성하는데 있어 시간적 제약이 덜하다. 따라서 본 논문에서는 변형된 SAFER+ 알고리즘 구현하는데 있어 치환 함수의 덧셈과 XOR, 로그, 지수 연산을 바이트 단위의 순차 계산을 수행함으로써 소요되는 하드웨어 용량을 줄이도록 설계하였다. 본 논문에서 제시한 모듈은 블루투스 표준안 버전 1.1에 근거하여 구현 하였으며 시뮬레이션 및 테스트는 Xilinx FPGA를 이용하여 검증 하였다.

Ⅰ. 서 론

블루투스 무선 전송 기술은 기존의 유선망 대체와 이동성, 이종간 장비와의 상호 접속성을 꾀하기 위해 저가격성, 저전력, 소형화를 만족하는 무선 통신 프로토콜로 1998년 블루투스 SIG (Special Intest Group)를 통해 제안되었다. 블루투스 무선 인터페이스는 이동전화, PDA, 노트북, 프린터, 무선 헤드셋과 같은 개인 정보 기기간 근거리 10M 범위내에 PAN(Personal Area Network)를 형성하여 멀티미디어 정보를 최대 723.2Kb/s로 유선 접속 장치 없이 송수신할 수 있다. 전송 대역은 무면허 대역인 2.4GHz ISM(Industry Scientific Medicine)밴드를 사용하며 79채널에 대해 주파수 호핑 방식을 사용한다. 변조 방식은 각 채널에 대해 1Mb/s 전송속도를 갖는 GFSK (Gaussian Frequency Shift Keying) 방식을 사용한다. 송수신 장치는 $625\mu s$ 간격의 슬롯 단위로 TDD (Time Division Duplex) 방식을 통해 전이중 통신을 행한다. 데이터는 패킷 단위로 전송되며 패킷은 1~3개의 슬롯을 점유할 수 있다. 패킷의 구조는 액세스 코드, 헤더, 페이로드(payload)로 구성되어 있다. 이 중 페이로드는 상호 장치 간 암호 알고리즘을 이용하여 데이터를 보호 할 수 있다. 이는 블루투스 프로토콜이 개인 정보 기기간 인터페이스를 제공하므로 개인 정보 보호를 위해서는 매우 중요한 것이다. 블루투스 표준안은 암호화를 위해 Cylink 사에서 제시한 128bit 암호화를 키를 생성하는 SAFER+ (Secure And Fast Encryption Routine)알고리즘을 사용한다. 이 알고리즘은 공개 도메인에서 얻을 수 있는 강력한 최신 암호화 알고리즘이다[1][2].

블루투스 암호화는 키 생성 모듈과 암호화 엔진 모듈로 구성된 암호화 모듈을 사용하여 이루어진다. 키 생성 모듈은 변형된 SAFER+ 알고리즘을 이용하여 표준안에서 제시한 4개의 키와 암호화 키를 생성하고 인증(authentication)과정을 수행한다. 암호화 엔진 모듈은 생성된 암호화 키의 사이즈를 조정하고 사이퍼 키 스트림 값을 생성하며 이를 페이로드 데이터와 XOR 연산을 통해 암호화를 수행 하고 수신된 데이터를 암호화 해독한다[1].

본 논문에서는 블루투스 암호화를 위해 필요한 키 생성 모듈과 암호화 엔진 모듈의 설계를 제안하고 이를 하드웨어 묘사언어인 VHDL을 이용해 구현한다

II. 본 론

1. 블루투스 암호화 키 생성 모듈

그림 1은 블루투스 장치간 암호화 키를 생성하는 과정을 나타낸다. 이 과정에서 사용되는 링크 키와 암호화 키는 SAFER+ 알고리즘과 이를 변형한 SAFER+ 알고리즘을 통해 생성된다. 블루투스 표준안은 SAFER+ 알고리즘을 A_r로 변형된 SAFER+ 알고리즘을 A'_r로 표기한다.

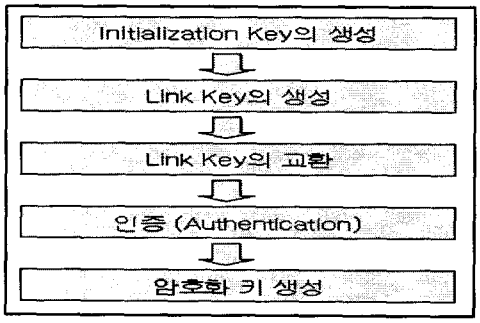

그림 1. 암호화 과정의 블록도

그림 1에서 사용되는 링크 키는 4개의 종류가 있다. 초기화 키(K_{init}), 임시 키(K_{master}), 유닛 키(K_A), 조합 키(K_{AB})이다. 각각 키는 A'_r가 사용되는 E_2알고리즘을 통해 생성된다. 초기화 키(K_{int})는 128bit 길이를 가지며 암호화 초기화 과정에서 블루투스 어드레스에 의해 확장된 PIN (Personal Identification Number)값과 랜덤 수(128bit)를 가지고 E_{22}알고리즘을 통해 생성된다. 이 키는 링크 키의 상호 교환시 링크 키를 보호하기 위해 XOR 연산에 사용된다. 임시 키(K_{master})는 피코넷(piconer)의 점 대 다 broadcast 패킷의 암호화를 위해 사용되는 링크 키이며 유닛 키를 대신하여 한정된 시간에 사용된다. 임시 키는 두 개의 랜덤 수를 가지고 초기화 키와 동일 하게 E_{22}알고리즘을 통해 생성된다. 수식 (1)은 E_{22}알고리즘을 나타낸다.

$$PIN = \begin{cases} PIN[0...L-1] \cup BD\_ADDR[0...min\{5, 15-L\}], & L < 16 \\ PIN[0...L-1], & L = 16 \end{cases}$$

$$\begin{cases} X = \bigcup_{i=0}^{15} PIN'[i \,(mod\, L')], \\ Y = RAND[0...14] \cup (RAND[15] \oplus L') \end{cases}$$

$$L' = min\{16, L+6\}$$

$$E_{22} : (PIN, RAND, L') \mid \rightarrow A'_r(X, Y) \qquad (1)$$

유닛 키(K_A)는 반영구적 수명을 가진 키로 각 유닛 마다 ROM에 저장되어 있으며 공장 셋업 기간 중에 생성된다. 이 키는 사용자의 요구에 따라 변경 될 수 있으나 매우 드물게 변경 된다. 이 키는 암호화 키를 생성하는데 있어 중요한 파라미터로 사용된다. 유닛 키는 128bit 램덤 수와 블루투스 48bit 어드레스에 의해 생성된다. 조합 키(K_{AB})는 각 유닛의 유닛 키를 조합한 키로 암호화 특성을 개선하기 위해 유닛 키 대신에 종종 사용된다. 조합 키와 유닛 키의 기능은 동일하며 단지 생성 과정에 차이가 난다. 이 두 키의 사용은 블루투스 디바이스의 메모리 용량에 따라 결정되며 E_{21}알고리즘을 통해 생성된다. 수식 (2)는 E_{21}알고리즘을 나타낸다.

$$\begin{cases} X = RAND[0...14] \cup (RAND[15] \oplus 6) \\ Y = address[i \,(mod 6)] \end{cases}$$

$$E_{21} : (RAND, adderss) \mid \rightarrow A'_r(X, Y) \qquad (2)$$

인증 과정은 상호 교환된 링크 키의 유효성 판단을 위해 사용되며 출력 값으로 ACO(Authentication Ciphering Offset)과 SRES(Signed Response)를 생성한다. 이 중 ACO는 암호화 키를 생성하기위해 사용된다. 인증 과정은 요구(challenge)와 응답(response) 방식을 사용한다. 요구자(claimant)는 인증을 위하여 입증(verifier)자에게 인증 랜덤 수(AU-RAND)값을 요구 한다. AU-RAND값이 교환 된 후 두 유닛은 같은 AU-RAND값과 요구자의 어드레스, 링크 키를 E_1알고리즘을 이용해 계산한다. 요구자는 생성된 SRES값을 인증자에게 다시 전송하여 그 값이 동일한지 검사하여 상호 교환된 링크 키의 유효성을 판단한다. 수식 (3)은 Hash 함수로 정의된 E_1알고리즘의 수식을 나타낸다. 그림 2는 인증 과정에 사용되는 E_1알고리즘 계산의 데이터 흐름을 나타낸다.

$$Hash(K, RAND, address) \mid \rightarrow (SRES, ACO)$$

$$E : (X[0, ..., L-1], L) \mid \rightarrow (X[i \,(mod L)] \text{ for } i = 0...15$$

$$E_1 : (K, I_1, I_2, L) \mid \rightarrow A'_r([K], [E(I_2, L) +_{16} (A_r(K, I_1) \oplus_{16} I_1)]) \qquad (3)$$

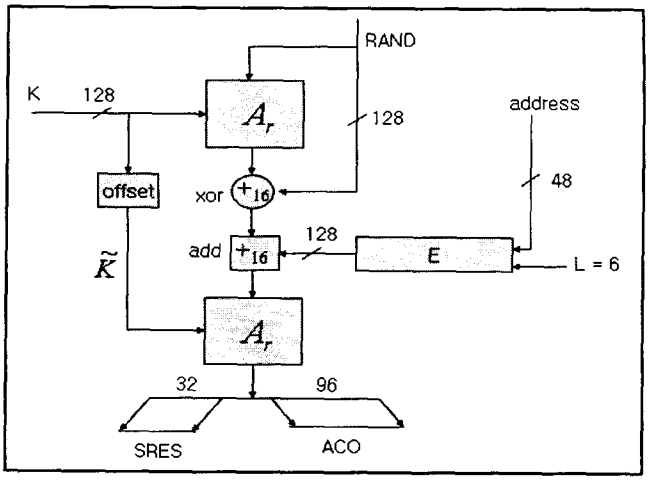

그림 2 E_1 알고리즘 계산을 위한 데이터 흐름도

암호화 키 생성은 교환된 링크 키와 인증 과정에서 생성된 ACO 그리고 상호 교환된 랜덤 수를 E_3알고리즘을 이용해 생성한다. 수식 (4)는 E_3알고리즘을 나타낸다.

$$COF = \begin{cases} BD\_ADDR \cup BD\_ADDR, & \text{if } Link\ key\ is\ a\ master\ key \\ ACO, & otherwise \end{cases}$$

$$E_3 : (K, RAND, COF) \mid \rightarrow Hash(K, RAND, COF, 12) \qquad (4)$$

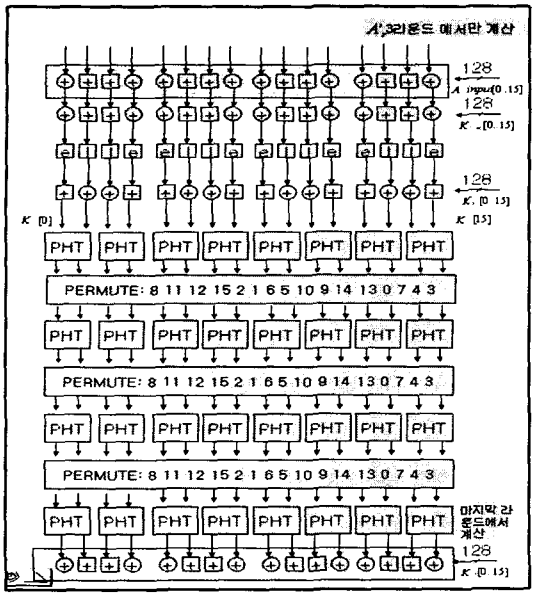

그림 3. A'_r와 A_r 알고리즘의 1라운드 구성도

암호화 키를 생성하기 위해 사용되는 A_r, A'_r 알고리즘은 8개의 라운드를 통해 키를 생성하며 128bit 두 개의 입력 변수에 대해 128bit 키를 출력 한다. 1 라운드의 구성은 그림 3과 같다. 각 라운드는 2개의 서브 키가 필요하다. 이 서브 키는 키 스케줄 모듈을 통해 입력 받게 된다. 각 라운드의 연산은 byte 중심의 연산을 수행하며 치환을 위해 사용되는 e, l함수는 $y=(45^x \bmod 257) \bmod 256$, $y = (log_{45}(x) \bmod 257) mod 256$에 각각 대응된다. 실제 e, l함수는 역 함수 관계를 가지며 구현은 복잡한 연산을 줄이기 위해 룩업 테이블 방식을 이용한다. PHT(Pseudo-Hadamard Transformation)는 PHT(x, y) = (2x+y mod 256, x+y mod 256)연산을 수행한다. 그림 3의 치환을 나타내는 부분은 Armenian Shuffle 치환을 의미하며 이는 단순히 데이터의 위치 교환이다. 표시된 숫자는 치환 되어질 입력데이터의 위치를 나타낸다[2][3][4].

A'_r, A_r 알고리즘을 이용한 인증 과정과 링크 키, 암호화 키의 생성은 LM(Link Manager)의 PDU(Protocol Data Unit)패킷을 통해 이루어지는 비교적 시간이 많이 걸리는 과정이며 키 생성의 시간적 제약이 덜하다. 따라서 A_r, A'_r 알고리즘 모듈은 치환 함수 (key-controlled substitution)의 덧셈, XOR, 지수, 로그 연산 부분을 바이트 중심의 순차 계산함으로써 하드웨어 연산을 줄일 수 있다. 그림 4, 5는 구현된 A'_r과 A_r 모듈과 키 스케줄 모듈을 나타내고 있다. 키 스케줄 모듈의 바이어스 함수는 $B_p[i] = ((45^{(45^{17p+i+1} mod 257)} mod 257) mod 256), \text{for } i = 0, .., 15$의 2중 지수 계산을 행한다. 이 연산 역시 룩업 테이블을 이용하여 구현한다. 그림 4의 모듈은 모드 선택 입력을 통해 A_r과 A'_r을 다르게 실행된다. A'_r은 처음 입력 값을 3라운드에서 다시 사용하는 것이 A_r과 다르다[1].

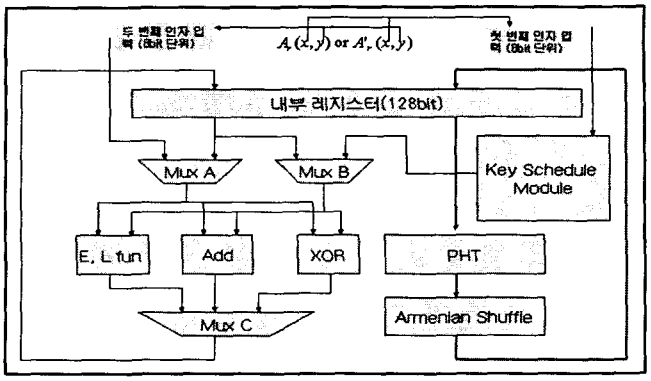

그림 4. 설계된 A_r와 A'_r 모듈의 블록 다이어그램

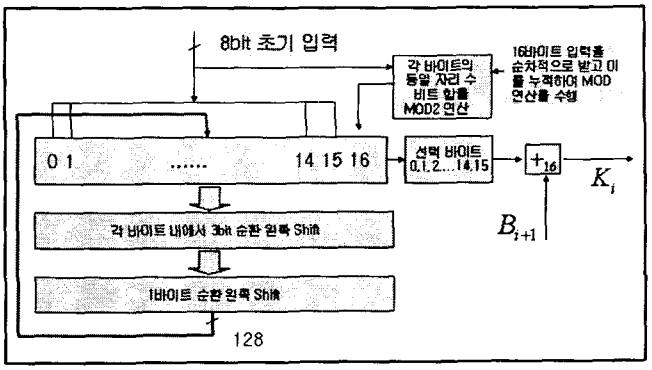

그림 5. 제안된 키 스케줄 모듈의 블록 다이어그램

2. 암호화 엔진 모듈

전송 패킷내의 페이로드 데이터 암호화는 암호화 엔진 모듈의 사이퍼(cipher) 키 스트림 값과 XOR 연산을 통해 이루어진다. 암호화 엔진의 입력 값은 생성된 암호화 키, 마스터 어드레스, 클럭 값이다. 암호화 키의 유효한 길이는 8~128bit 중 8의 배수 길이를 가질 수 있다. 하지만 E_3 알고리즘을 통해 생성되는 암호화 키의 길이는 128bit로 고정되어 있다. 이를 유효한 길이를 줄이기 위해 modulo연산을 행하며 수식 (5)는 이 과정을 나타낸다[2].

$$K'_c(x) = g_2^{(L)}(x)(K_c(x) \bmod g_1^{(L)}(x)) \qquad (5)$$

수식(5)의 g_1, g_2 생성 다항식은 16개의 유효한 길이에 대해 블루투스 표준안에서 제공하며 이 들 값은 룩업 테이블 방식을 이용해 연산 과정에 사용하게 된다. 암호화 엔진 모듈의 구성은 4개의 LFSR(Linear Feedback Shift Register) 뱅크와 합 결합기(summation combiner logic)으로 구성되어 있다. LFSR 뱅크의 길이는 $L_1 = 25$, $L_2 = 31$, $L_3 = 33$, $L_4 = 39$이며 생성 다항식의 값은 $f_1(t) =$ 0x210111, $f_2(t) =$ 0x810111, $f_3(t) =$ 0x2011000011, $f_4(t) =$ 0x9010000011이다. 그림 6은 암호화 엔진의 블록 다이어그램을 나타낸다. 여기서 T_1, T_2는 선형 일대일 대응을 나타낸다.

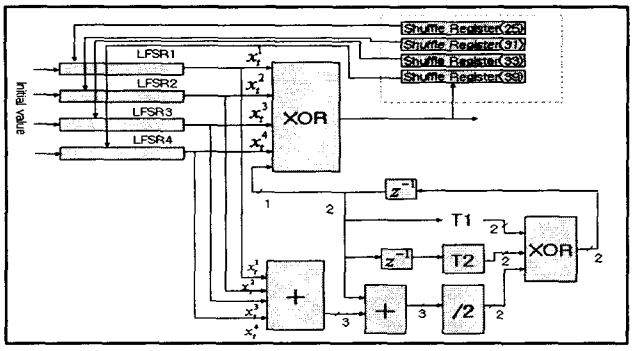

그림 6. 암호화 엔진 모듈의 블록 다이어 그램

3. 시뮬레이션 및 검증

본 논문에서 제시한 암호화 키 생성 모듈과 암호화 엔진 모듈은 하드웨어 묘사언어인 VHDL을 이용하여 구현하였으며 시뮬레이션 및 테스트는 Xilinx FPGA를 통해 검증하였다. Target Device는 20만 게이트 Virtex XCV200 PQ240이다. 각 모듈의 정확한 동작특성은 블루투스 표준안 버전 1.1 부록 4의 샘플 데이터를 통해 확인한다.

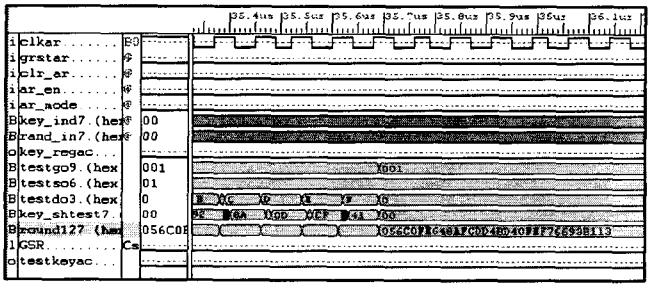

그림 7. E_1알고리즘 시뮬레이션

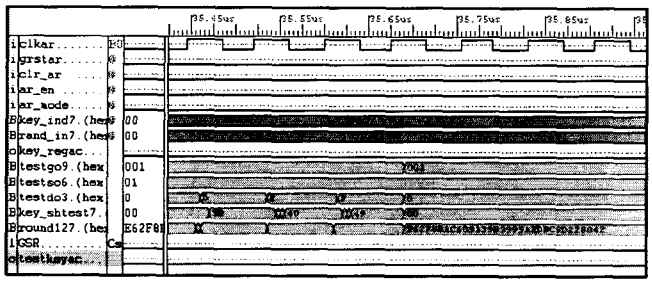

그림 8. E_{21}알고리즘 시뮬레이션

그림 7는 E_1 알고리즘에 모든 입력 값을 0으로 하여 암호화 인증 과정을 수행한 것이다. 샘플 데이터의 예상 출력 값은 0x056C0FE648AFCDD4BD40FEF76693B113이며 이와 정확히 일치함을 볼 수 있다. 그림 8은 어드레스 0xCAC4364303B6와 랜덤 수 0x2DD9A5503431913040 13B2D7E1189D09 값을 넣어 E_{21} 알고리즘을 통해 유닛 키를 생성한 것이다. 이 샘플 데이터의 출력 예상 값은 0xE62F8BAC609139B3999AEDBC9D228042이다. 역시 샘플 데이터의 출력 값과 일치함을 볼 수 있다. 그림 9는 암호화 엔진 모듈에 암호화 키의 값과 어드레스 값을 모두 1로 하고 클럭 값을 CLK = 0xFFFFFF03을 입력하여 초기화 이후 127번째 동작후의 4개의 LFSR 값과 사이퍼 키 스트림 Z값을 검증한 것이다. 출력 예상 값은 L_1= 0x042289, L_2=0x2B9AD56C, L_3=0x02C20B3DE, L_4 =0x315A5B0D20, Z = 1이다.

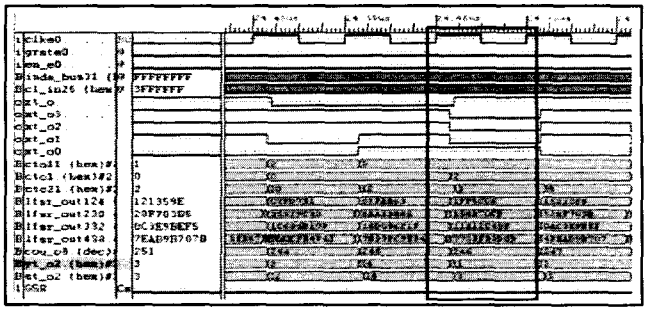

그림 9. 암호화 엔진 모듈 시뮬레이션

Ⅲ. 결 론

본 논문에서는 암호화 과정 중에 인증 과정과 링크 키, 암호화 키를 생성하는 암호화 키 생성 모듈과 생성된 암호화 키를 이용하여 페이로드 데이터를 암호화 하는 암호화 엔진 모듈을 설계하고 검증하였다. 특히 암호화 키 생성 모듈은 LM(Link Manager)의 PDU(Protocol Data Unit) 패킷을 통해 상호 정보가 교환되므로 암호화 키를 생성하는데 있어 시간적 제약이 덜하다. 따라서 본 논문에서는 변형된 SAFER+ 알고리즘 구현하는데 있어 치환함수의 덧셈과 XOR, 로그, 지수 연산을 바이트 단위의 순차 계산을 수행함으로써 소요되는 하드웨어 용량을 줄이도록 설계 하였다. 제시한 모듈은 블루투스 표준안 버전 1.1의 샘플 데이터를 가지고 검증하여 제시한 모듈의 정확성을 테스트 하였다.

참 고 문 헌

[1] Bluetooth Specification Version 1.1, "Core", Specification of the Bluetooth System volume1, Feb 22 2001

[2] Kitsos, P.; Sklavos, N.; Papadomanolakis, K.; Koufopavlou, O.; "Hardware Implementation of Bluetooth Security", Pervasive Computing, IEEE , Vol 2, Issue: 1, pp21 -29, Jan.-Mar 2003

[3] Schubert, A.; Meyer, V.; Anheier, W.; "Reusable cryptographic VLSI core based on the SAFER K-128 algorithm with 251.8 Mbit/s throughput", Signal Processing Systems, SIPS 98. 1998 IEEE Workshop on , 8-10, pp: 437-446 Oct. 1998

[4] J. L. Massey; "SAFER K-64: A Byte-Oriented Block Ciphering Algorithm", Fast Software Encryption, Proceedings of the Cambridge Security Workshop, Cambridge, UK, pp1-17, 1993