블루투스 베이스밴드 홉 시퀀스의 효율적 구현 방안

*김 현 미, *김 진 숙, *임 재 윤
*제주대학교 통신공학과
전화 : 064-754-3635 / 핸드폰 : 017-691-4110

The efficient implementation of the Hop Sequence in Bluetooth Baseband

Hyun Mi Kim, Jin Sook Kim, Jea Yun Lim
Dept. of Telecommunication Engineering, Jeju University
E-mail : dasom0411@hanmail.net

Abstract

This paper has been understanded the whole concept of hop sequence in bluetooth baseband and studied algorithm of its each type. It is implemented and verified both hardware and firmware of hop sequence according to Sepcification ver.1.1, and compared and examined two schemes with performance, size, cost, and time-to-market, etc.

This paper is suggested that it is the efficient system division to design optimum system depending on developing environment.

I. 서론

근래에 근거리 무선 통신의 새로운 표준으로 각광받고 있는 블루투스(Bluetooth)는 소형, 저가, 저전력의 기술을 전제로 한다. 블루투스는 2.4GHz 대역에서 1MHz 대역폭에 79채널을 설정하고, 초당 1600회 채널을 바꾸는 주파수 호핑(frequency hopping) 방식의 스펙트럼 확산(spread spectrum) 기술을 사용하여 전파를 송수신한다. 이러한 주파수 홉(hop) 설계는 장치들이 전자기 장해가 있는 지역에서도 통신 성능이 고르게 유지 될 수 있는 장점을 갖게 한다[1][2].

블루투스 기기간의 원활한 통신과 최적의 설계를 위해서는 안정적이고 효율적인 주파수 홉 설계방안이 요구된다. 본 논문에서는 블루투스 규격 중 기저 대역에서의 홉 선택 과정에 대해 분석하고 이에 대한 알고리즘을 하드웨어(Hardware)와 펌웨어(Firmware)로 구현하여 비교함으로써 효율적인 설계 방안을 제시하였다.

II. 홉 시퀀스

2.1 블루투스의 무선 인터페이스

블루투스의 주파수는 ISM 대역(2,400~2,483.5MHz)으로, 주파수 호핑 방식의 스펙트럼 확산 방식을 사용한다. 이를 위한 정확한 블루투스 클럭은 Fig.1과 같다. 채널은 79개의 RF 채널을 통한 의사 랜덤 호핑 시퀀스로, 블루투스 기기 어드레스에 의해 결정된다. 또한 이는 625μs 타임 슬롯으로 나뉘어지고 TDD (Time Division Duplex)방식을 채택하여 시간에 따라 교대로 송수신을 반복하게 된다.

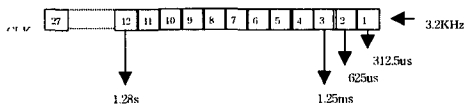


Fig.1 Blutooth clock

2.2 홉 선택의 일반적인 체계

Fig.2는 79 홉 시스템에서의 홉 선택 커널이다. X는 세그먼트에서 위상을 결정하고 Y1, Y2는 전송 방향을 선택하게 된다. 또한 A~D는 세그먼트내의 순서를, E와 F는 호핑 주파수를 결정하고 커널은 모든 짝수 호핑 주파수 다음에 홀수 호핑 주파수를 작성한다.

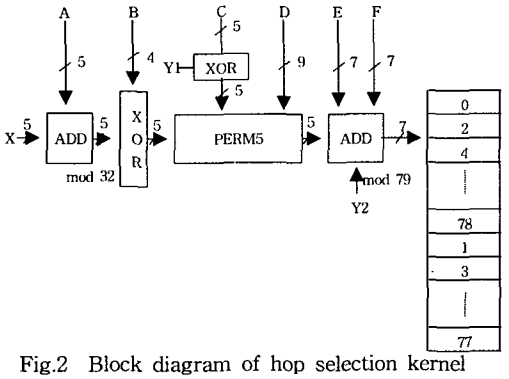


Fig.2 Block diagram of hop selection kernel

첫 번째 가산기(ADD)는 세그먼트 내의 위상만을 바꾸는데 이 출력은 배타적 논리합(XOR)에서 B와 연산을 수행하게 된다. 다음 단계의 순열연산(PERM5)은 Fig.3과 같고 Table 1은 제어 신호 P('1'일 때)의 동작을 보여준다. 여기서 P_{0-8}은 D_{0-8}, P_{9-13}은 $C_{0-4} \oplus$ Y1 이고 Z는 배타적 논리합의 출력이다.

Table 1. Control of the butterflies.

Control signal	Butterfly	Control signal	Butterfly
P_0	(Z_0, Z_1)	P_8	(Z_4, Z_4)
P_1	(Z_2, Z_3)	P_9	(Z_0, Z_3)
P_2	(Z_1, Z_2)	P_{10}	(Z_2, Z_4)
P_3	(Z_3, Z_4)	P_{11}	(Z_1, Z_3)
P_4	(Z_0, Z_4)	P_{12}	(Z_0, Z_3)
P_5	(Z_1, Z_3)	P_{13}	(Z_1, Z_2)
P_6	(Z_0, Z_1)		
P_7	(Z_2, Z_4)		

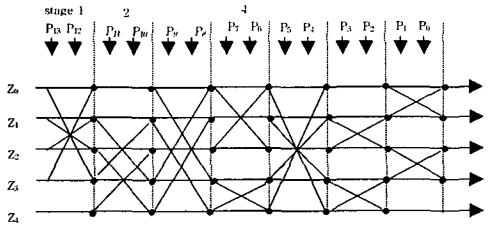


Fig.3 Permutation operation

두 번째 가산기(ADD)는 세그먼트의 다른 호핑 주파수를 결정하고 이 출력은 79 나머지 연산을 수행하여 79 레지스터의 뱅크에 주소로 지정된다. 레지스터는 0에서 78의 호핑 주파수에 해당하는 합성 코드 워드를 싣고 상위에는 짝수 호핑 주파수를, 하위에는 홀수 호핑 주파수를 구성한다.

2.3 홉 시퀀스 알고리즘

Table 2는 선택커널의 제어워드를 나타낸 것이고 각 상태를 (1)~(6)에서 설명하였다. 이때 사용되는 클럭의 형태는 다음과 같다.

① CLK_{27-0} : 현재의 피코넷(piconet)의 마스터 클럭

② $CLKN_{27-0}$: 기기의 고유 클럭

③ $CLKE_{27-0}$: page된 기기의 고유 클럭에 옵셋을 더한 클럭

Table 2. Control of hop selection kernel

	Page scan/ Inquiry scan	Page/Inquiry	Page response (master/slave)and Inquiry response	Connection state
X	$CLKN_{16-12}$	$X_{p_{4-0}}/X_{i_{4-0}}$	$X_{prm_{4-0}}/X_{prs_{4-0}}$	CLK_{6-2}
Y1	0	$CLKE_1/CLKN_1$	$CLKE_1/CLKN_1$	CLK_1
Y2	0	32 × $CLKE_1$/ 32 × $CLKN_1$	32 × $CLKE_1$/ 32 × $CLKN_1$	32 × CLK_1
A	"	A_{27-23}	A_{27-23}	" CLK_{22}
B	"	A_{22-19}	A_{22-19}	"
C	"	$A_{18,4,2,0}$	$A_{18,4,2,0}$	" CLK_{16}
D	"	A_{18-10}	A_{18-10}	$A_{18-10} \oplus CLK_{15-7}$
E	"	$A_{13,11,9,7,5,3,1}$	$A_{13,11,9,7,5,3,1}$	"
F	0	0	0	$16 \times CLK_{27-7}$ mod79

(1) 호출 주사(Page scan)와 조회 주사(Inquiry scan) 상태 대기중인 기기의 BD_ADDR이 입력 주소로 사용된다.

(2) 호출 상태(Page substate)

호출과 대기 기기 사이에 반복되는 부적당한 가능성을 피하기 위해 주기적인 천이가 필요하다. X-입력은 식 (1)과 같고 옵셋값은 식 (2)와 같다.

$$X_{prs} = [CLKE_{16-12} + k_{offset} + (CLKE_{4-2,0} - CLKE_{16-12}) \bmod 16] \bmod 32 \quad (1)$$

$$k_{offset} = \begin{cases} 24 & A-train \\ 8 & B-train \end{cases} \quad (2)$$

(3) 호출 응답(Page response)

가) 슬레이브 응답(Slave response)

$CLKN^{*}_{16-12}$은 링크 손실의 가능성을 제거하기 위해 현재 값으로 고정한 것이고 N은 0에서 시작해서 $CLKN_1$이 0이 되는 순간마다 1씩 증가하게 된다. X-입력은 식 (3)과 같다.

$$X_{prs} = [CLKN^{*}_{16-12} + N] \bmod 32 \quad (3)$$

나) 마스터 응답(Master response)

마스터 응답은 클럭값과 현재의 k_{offset} 값이 고정되어야하고, N은 1에서 시작해서 $CLKE_1$이 0이 되는 순간마다 1씩 증가한다. X-입력은 식 (4)와 같다.

$$X_{prm_{4-0}} = [CLKE^{*}_{16-12} + k^{*}_{offset} + (CLKE^{*}_{4-2,0} - CLKE^{*}_{16-12}) \bmod 16+N] \bmod 32 \quad (4)$$

(4) 조회 상태(Inquiry substate)

X-입력은 특정 기기가 주소로 지정되지 않았기 때문에 조회 측의 CLKN이 사용되고, 이는 식 (5)와 같이 표현되며 이 때 k_{offset} 은 식 (2)와 같다.

$$X_i = [CLKN_{16-12} + k_{offset} + (CLKN_{4-2,0} - CLKN_{16-12}) \bmod 16] \bmod 32 \quad (5)$$

(5) 조회 응답(Inquiry response)

X-입력은 식 (6)과 같고 N은 FHS 패킷이 전송된 후에 증가한다.

$$X_{ir} = [CLKN^{*}_{16-12} + N] \bmod 32 \quad (6)$$

(6) 연결 상태(Connection state)

이 상태에서는 마스터의 Access Code와 클럭을 사용한다. 마스터의 전송은 짝수(CLK_{1-0} =00) 슬롯에서, 슬레이브는 홀수(CLK_{1-0} =10) 슬롯에서 시작된다.

Ⅲ. 블루투스 베이스밴드 홉 시퀀스의 하드웨어/펌웨어 구현 및 검증

3.1 블루투스 기술의 구현

블루투스 솔루션을 모두 구현하려면 신중하게, 그리고 최적으로 설계해야 한다. 그러므로 성능, 크기, 가격, 판매시기 등을 적당히 고려하여 블루투스 시스템 내부의 하드웨어와 펌웨어간의 역할 분담을 시켜야 한다. 따라서 본 논문에서는 홉 시퀀스의 보다 나은 설계를 위해 FPGA(Altera)를 이용한 하드웨어 구현과 MCU(EISC)를 이용한 펌웨어 구현을 통해 효율적인 구현 방안을 제시하였다.

3.2 홉 시퀀스의 하드웨어 구현 및 검증

홉 시퀀스의 하드웨어 구현은 Fig.2와 Table 2로부터 각 형태의 알고리즘을 이용하여 하드웨어 설계 언어인 VHDL(Very-high speed integrated-circuit Hardware Description Language)로 하였고 이를 테스트한 환경을 Fig.4에 나타내었다.

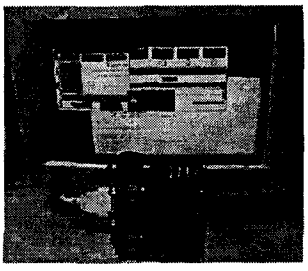
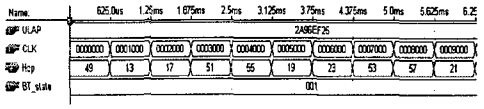
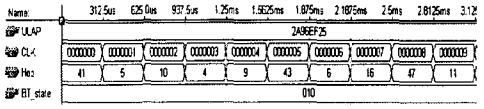


Fig.4 Environment for hardware test

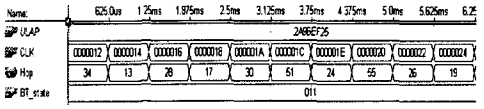
Fig.5는 블루투스 기기의 고유 주소 ULAP를 '0x2A96EF25'라 가정하여 구현한 시뮬레이션 결과이고 블루투스 SIG(Special Interest group) 규격의 샘플 데이터와 비교하였다. 구현 결과 홉 시퀀스의 하드웨어 크기는 대략 10,000 Gates임을 알 수 있었다.



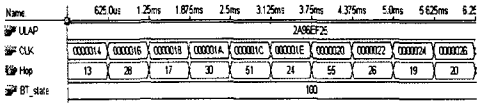
(a) Page Scan / Inquiry Scan



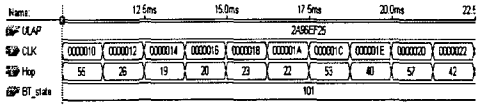
(b) Page State / Inquiry State



(c) Slave Page Response State



(d) Master Page Response State



(e) Connection State

Fig.5 Simulation result of the hardware(a)(b)(c)(d)(e)

3.3 홉 시퀀스의 펌웨어 구현 및 검증

홉 시퀀스의 펌웨어 구현은 MCU 사용에 편리한 C로 하였고 이를 테스트한 환경을 Fig.6에 나타내었다.

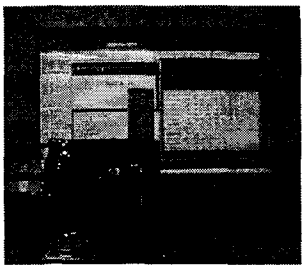
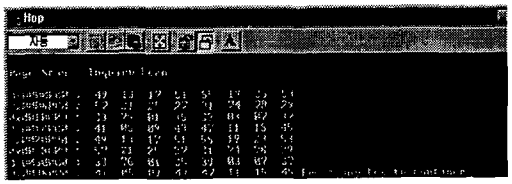
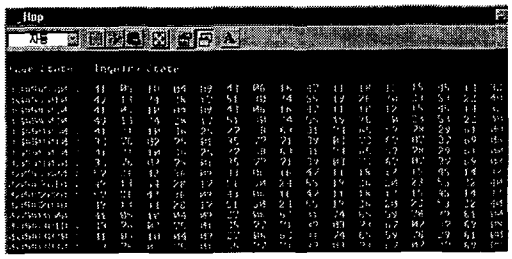


Fig.6 Environment for hardware test

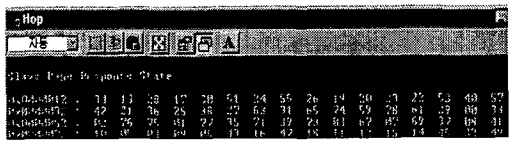
Fig.7은 3.2절과 같은 조건에서 시뮬레이션 한 결과이고 이 또한 샘플데이터와 비교하였다. 구현 결과 홉 시퀀스의 펌웨어 크기는 대략 28KB임을 알 수 있었다.



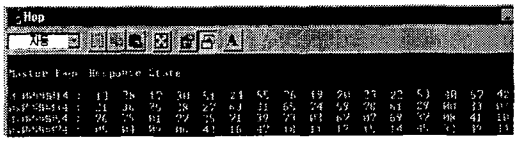
(a) Page Scan / Inquiry Scan



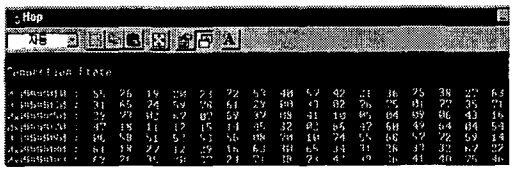
(b) Page State / Inquiry State



(c) Slave Page Response State



(d) Master Page Response State



(e) Connection State

Fig.7 Simulation result of the firmware(a)(b)(c)(d)(e)

3.4 홉 시퀀스 하드웨어/펌웨어 구현의 비교 · 검토

블루투스 클럭은 전원을 켤 때 0으로 리셋되고, 그 후 312.5us를 증가시키면서 자기 발진하는 28비트 카운트이므로 홉 시퀀스는 매시간 계산되어야 한다. 블루투스는 1Mbps의 전송속도를 갖기 때문에 계산 속도가 그리 중요하지는 않지만 차후 전송속도가 증가될 경우 속도를 고려하는 것 또한 매우 중요하게 된다. 펌웨어로 구현할 경우 MCU 사용율이 높아지게 되고 이로 인해 부하의 위험성이 커진다는 점을 감안한다면 하드웨어 구현이 펌웨어 구현보다 시스템 효율에 있어 유리하다는 것을 알 수 있다.

홉 시퀀스의 구현된 크기를 고려할 때 이는 소형, 저가의 블루투스 전체조건과 밀접한 관계가 있다. 홉 시퀀스는 하드웨어 크기가 약 10,000 Gates이고 펌웨어는 약 28KB이기 때문에 판매시기의 시장 현황에 따른 크기와 비용을 비교하여 두 조건이 절충적으로 만족할 수 있는 구현 방식을 선택하는 것이 중요하다 하겠다.

또한 블루투스 기술은 지속적인 발전이 계속되고 있다. 이때 하드웨어는 고정적이므로 교체해야하는 번거로움과 함께 비싼 교체 비용이 들지만 펌웨어는 유연성이 있기 때문에 업그레이드가 용이하게 된다.

개발자는 위 사항을 고려하여 시스템 효율이 높은 최적의 블루투스 기술 개발을 위해 적합한 선택을 해야할 것이다.

Ⅳ. 결론

블루투스의 발전과 함께 효율적인 시스템 구현 분할의 연구는 기기간의 원활한 통신과 소형, 저가, 저전력의 기술을 위해서 중요한 과제가 된다.

본 논문에서는 블루투스 베이스밴드의 홉 시퀀스에 대한 전체적인 개념을 이해하고 각 형태의 알고리즘에 대해 분석하였고, 이를 블루투스 SIG 규격의 절차에 따라 하드웨어와 펌웨어로 구현하고 검증하였다. 또한 성능, 크기, 가격, 판매시기 등을 적당히 고려하여 하드웨어와 펌웨어 두 방식을 비교 · 검토함으로써 개발 환경과 조건에 따른 홉 시퀀스의 적합한 구현 방식을 제시할 수 있었다.

본 논문은 블루투스 시스템의 성능을 향상시킬 수 있는 방안으로 효율적인 시스템 분할 방법을 제시함으로써 블루투스 코어 설계의 기초 자료로 활용될 수 있을 것으로 사료된다.

참고 문헌

[1] Jennifer Bray and Charles F Sturmam, 2001, "BLUETOOTH Connect without Cables", *Prentice-Hall*, pp.(33, 63-64).

[2] Bluetooth Special Interest Group, 2000 , "Specification of the bluetooth system Part B", pp.(127-138).