(19)대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl.
H04B 7/15 (2006.01)

(45) 공고일자 2007년03월09일
(11) 등록번호 10-0691054
(24) 등록일자 2007년02월27일

(21) 출원번호	10-2000-7009996	(65) 공개번호	10-2001-0041755
(22) 출원일자	2000년09월08일	(43) 공개일자	2001년05월25일
심사청구일자	2004년03월05일		
번역문 제출일자	2000년09월08일		
(86) 국제출원번호	PCT/US1999/004969	(87) 국제공개번호	WO 1999/46942
국제출원일자	1999년03월05일	국제공개일자	1999년09월16일

(81) 지정국 국내특허 : 알바니아, 아르메니아, 오스트리아, 오스트레일리아, 아제르바이잔, 보스니아 헤르체고비나, 바베이도스, 불가리아, 브라질, 벨라루스, 캐나다, 스위스, 중국, 쿠바, 체코, 독일, 덴마크, 에스토니아, 스페인, 핀란드, 영국, 그루지야, 헝가리, 이스라엘, 아이슬랜드, 일본, 케냐, 키르키즈스탐, 북한, 대한민국, 카자흐스탄, 세인트루시아, 스리랑카, 리베이라, 레소토, 리투아니아, 룩셈부르크, 라트비아, 몰도바, 마다가스카르, 마케도니아공화국, 몽고, 말라위, 멕시코, 노르웨이, 뉴질랜드, 슬로베니아, 슬로바키아, 타지키스탄, 투르크맨, 터어키, 트리니아드토바고, 우크라이나, 우간다, 우즈베키스탄, 베트남, 폴란드, 포르투칼, 루마니아, 러시아, 수단, 스웨덴, 싱가포르, 그라나다, 크로아티아, 인도네시아, 인도, 가나, 감비아, 시에라리온, 세르비아 앤 몬테네그로, 짐바브웨,

AP ARIPO특허 : 케냐, 레소토, 말라위, 수단, 스와질랜드, 우간다, 시에라리온, 가나, 감비아, 짐바브웨,

EA 유라시아특허 : 아르메니아, 아제르바이잔, 벨라루스, 키르키즈스탐, 카자흐스탄, 몰도바, 러시아, 타지키스탄, 투르크맨,

EP 유럽특허 : 오스트리아, 벨기에, 스위스, 독일, 덴마크, 스페인, 프랑스, 영국, 그리스, 아일랜드, 이탈리아, 룩셈부르크, 모나코, 네덜란드, 포르투칼, 스웨덴, 핀란드, 사이프러스,

OA OAPI특허 : 부르키나파소, 베닌, 중앙아프리카, 콩고, 코트디브와르, 카메룬, 가봉, 기니, 말리, 모리타니, 니제르, 세네갈, 차드, 토고, 기니 비사우,

(30) 우선권주장 09/036,941 1998년03월09일 미국(US)

(73) 특허권자 콸콤 인코포레이티드
미국 캘리포니아 샌디에고 모어하우스 드라이브5775 (우 92121-1714)

(72) 발명자 퀵,로이,에프.
미국92107캘리포니아샌디에고델몬테애브뉴4502

(74) 대리인 남상선

심사관 : 남옥우

전체 청구항 수 : 총 52 항

(54) 브로드캐스트 챌린지 값을 생성하는 방법

(57) 요약

브로드캐스트 챌린지 값을 생성하는 방법은 브로드캐스트 챌린지 값의 최상위 비트들에 제 1 업데이팅 알고리즘을 적용하는 단계 및 브로드캐스트 챌린지 값의 최하위 비트들에 제 2 의 구별되는 업데이팅 알고리즘을 적용하는 단계를 포함한다. 상기 업데이팅 알고리즘들은 최대길이 시프트 레지스터들 또는 의사랜덤 잡음 발생기의 다른 형태들일 수 있다. 연속적인 비반복 값들의 시퀀스는 업데이트 값이 순환하기 전의 시간 주기동안 올 제로 값을 삽입함으로써 비트들의 양쪽 세트들 또는 한 세트에 대해 늘어날 수 있다. 시스템 와이드 동기화는 글로벌 시간 기준을 통해 달성될 수 있다.

대표도

도 4a

특허청구의 범위

청구항 1.

제어 채널을 통해서 셀룰라 기지국에 의해 전송되는 브로드캐스트 챌린지 값(broadcast challenge value)을 업데이팅하는 방법으로서,

상기 브로드캐스트 챌린지 값의 다수의 최상위 비트들에 제 1 업데이팅 알고리즘을 적용하는 단계; 및

업데이팅된 브로드캐스트 챌린지 값을 생성하기 위해서 상기 브로드캐스트 챌린지 값의 다수의 최하위 비트들에 제 2 업데이팅 알고리즘을 수행하는 단계를 포함하며,

상기 제2 업데이팅 알고리즘 수행 단계는 업데이트 시퀀스가 상기 다수의 최하위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최하위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 제 2 업데이팅 알고리즘을 상기 다수의 최하위 비트들에 수행하는 단계를 포함하는 업데이팅 방법.

청구항 2.

제 1 항에 있어서, 상기 적용 단계는 제 1 최대길이 시프트 레지스터 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 3.

제 1 항에 있어서, 상기 적용 단계는 제 1 의사랜덤 잡음 생성 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 4.

제 1 항에 있어서, 상기 수행 단계는 상기 다수의 최하위 비트들에 제 2 최대길이 시프트 레지스터 알고리즘을 수행하는 단계를 포함하는 업데이팅 방법.

청구항 5.

제 1 항에 있어서, 상기 수행 단계는 상기 다수의 최하위 비트들에 제 2 의사랜덤 잡음 생성 알고리즘을 수행하는 단계를 포함하는 업데이팅 방법.

청구항 6.

제 1 항에 있어서, 상기 적용 단계는 업데이트 시퀀스가 상기 다수의 최상위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최상위 비트들에 대한 업데이트로서 올 제로(all-zeroes) 값을 한번 삽입하는 제 1 업데이팅 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 7.

삭제

청구항 8.

제 1 항에 있어서, 상기 적용 단계는 상기 다수의 최상위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 제 1 업데이팅 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 9.

제 1 항에 있어서, 상기 수행 단계는 상기 다수의 최하위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 제 2 업데이팅 알고리즘을 상기 다수의 최하위 비트들에 수행하는 단계를 포함하는 업데이팅 방법.

청구항 10.

셀룰라 시스템에서, 다수의 이동 가입자 유니트들을 인증하기 위해 다수의 기지국에 의해 사용되는 2진수들을 동기적으로 업데이팅 하는 방법으로서,

상기 다수의 기지국들 각각에 대한 2진수의 다수의 최상위 비트들에 제 1 업데이팅 알고리즘을 적용하는 단계;

상기 다수의 기지국들 각각에 대한 2진수의 다수의 최하위 비트들에 제 2 업데이팅 알고리즘을 수행하는 단계; 및

상기 적용 및 수행 단계들에서 발생하는 2진수의 업데이트 값들을 상기 다수의 기지국들에 걸쳐 동기화하는 단계를 포함하며,

상기 제2 업데이팅 알고리즘 수행 단계는 업데이트 시퀀스가 상기 다수의 최하위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최하위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 제 2 업데이팅 알고리즘을 상기 다수의 최하위 비트들에 수행하는 단계를 포함하는 업데이팅 방법.

청구항 11.

제 10 항에 있어서, 상기 동기화 단계는 상기 다수의 기지국들에 이용가능한 전 시스템 시간 기준(system-wide time reference)을 통해 이루어지는 업데이팅 방법.

청구항 12.

제 11 항에 있어서, 상기 전 시스템 시간 기준은 GPS 시간인 업데이팅 방법.

청구항 13.

제 11 항에 있어서, 상기 적용 단계는 제 1 최대길이 시프트 레지스터 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 14.

제 11 항에 있어서, 상기 적용 단계는 제 1 의사랜덤 잡음 생성 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 15.

제 11 항에 있어서, 상기 수행 단계는 상기 다수의 최하위 비트들에 제 2 최대길이 시프트 레지스터 알고리즘을 수행하는 단계를 포함하는 업데이팅 방법.

청구항 16.

제 11 항에 있어서, 상기 수행 단계는 상기 다수의 최하위 비트들에 제 2 의사랜덤 잡음 생성 알고리즘을 수행하는 단계를 포함하는 업데이팅 방법.

청구항 17.

제 11 항에 있어서, 상기 적용 단계는 업데이트 시퀀스가 상기 다수의 최상위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최상위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 제 1 업데이팅 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 18.

삭제

청구항 19.

제 11 항에 있어서, 상기 적용 단계는 상기 다수의 최상위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 제 1 업데이팅 알고리즘을 적용하는 단계를 포함하는 업데이팅 방법.

청구항 20.

제 11 항에 있어서, 상기 수행 단계는 상기 다수의 최하위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 제 2 업데이팅 알고리즘을 상기 다수의 최하위 비트들에 수행하는 단계를 포함하는 업데이팅 방법.

청구항 21.

셀룰라 시스템에서 이동 가입자 유니트를 인증하기 위해 2진수를 업데이트하기 위한 셀룰라 기지국으로서,

소프트웨어를 운용할 수 있는 집적 회로; 및

업데이팅된 2진수를 생성하기 위해서, 2진수의 다수의 최상위 비트들에 제 1 업데이팅 알고리즘을 적용하며 상기 2진수의 다수의 최하위 비트들에 제 2 업데이팅 알고리즘을 적용할 목적으로 상기 집적회로에 의해 실행되는 소프트웨어 명령 세트를 포함하며,

상기 제 2 업데이팅 알고리즘은 업데이트 시퀀스가 상기 다수의 최하위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최하위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 셀룰라 기지국.

청구항 22.

제 21 항에 있어서, 상기 제 1 업데이팅 알고리즘은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 1 최대길이 시프트 레지스터 알고리즘을 포함하는 셀룰라 기지국.

청구항 23.

제 21 항에 있어서, 상기 제 1 업데이팅 알고리즘은 제 1 의사랜덤 잡음 생성 알고리즘을 포함하는 셀룰라 기지국.

청구항 24.

제 21 항에 있어서, 상기 제 2 업데이팅 알고리즘은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 2 최대길이 시프트 레지스터 알고리즘을 포함하는 셀룰라 기지국.

청구항 25.

제 21 항에 있어서, 상기 제 2 업데이팅 알고리즘은 제 2 의사랜덤 잡음 생성 알고리즘을 포함하는 셀룰라 기지국.

청구항 26.

제 21 항에 있어서, 상기 제 1 업데이팅 알고리즘은 업데이트 시퀀스가 상기 다수의 최상위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최상위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 셀룰라 기지국.

청구항 27.

삭제

청구항 28.

제 21 항에 있어서, 상기 제 1 업데이팅 알고리즘은 상기 다수의 최상위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 셀룰라 기지국.

청구항 29.

제 21 항에 있어서, 상기 제 2 업데이팅 알고리즘은 상기 다수의 최하위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 셀룰라 기지국.

청구항 30.

다수의 이동 가입자 유니트; 및

상기 다수의 이동 가입자 유니트와의 무선 통신을 위해 구성되는 다수의 기지국들을 포함하며,

다수의 기지국들 각각은,

소프트웨어를 운용할 수 있는 집적 회로; 및

2진수의 다수의 최상위 비트들에 제 1 업데이팅 알고리즘을 적용하고, 상기 2진수의 다수의 최하위 비트들에 제 2 업데이팅 알고리즘을 적용하고 -상기 2진수는 기지국과의 통신을 요청하는 임의의 이동 가입자 유니트를 인증하기 위해서 사용됨-, 상기 다수의 기지국들에 걸쳐 전 시스템 시간 기준 신호를 이용하여 상기 2진수의 연속적인 업데이트 값들을 동기시키기 위해서 상기 집적 회로에 의해 실행되는 소프트웨어 명령 세트를 포함하며,

상기 제 2 업데이팅 알고리즘은 업데이트 시퀀스가 상기 다수의 최하위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최하위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 셀룰라 시스템.

청구항 31.

제 30 항에 있어서, 상기 전 시스템 시간 기준 신호는 상기 다수의 기지국들 각각에 GPS 시간의 측정치를 전달하는 셀룰라 시스템.

청구항 32.

제 30 항에 있어서, 상기 제 1 업데이팅 알고리즘은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 1 최대길이 시프트 레지스터 알고리즘을 포함하는 셀룰라 시스템.

청구항 33.

제 30 항에 있어서, 상기 제 1 업데이팅 알고리즘은 제 1 의사랜덤 잡음 생성 알고리즘을 포함하는 셀룰라 시스템.

청구항 34.

제 30 항에 있어서, 상기 제 2 업데이팅 알고리즘은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 2 최대길이 시프트 레지스터 알고리즘을 포함하는 셀룰라 시스템.

청구항 35.

제 30 항에 있어서, 상기 제 2 업데이팅 알고리즘은 제 2 의사랜덤 잡음 생성 알고리즘을 포함하는 셀룰라 시스템.

청구항 36.

제 30 항에 있어서, 상기 제 1 업데이팅 알고리즘은 업데이트 시퀀스가 상기 다수의 최상위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최상위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 셀룰라 시스템.

청구항 37.

삭제

청구항 38.

제 30 항에 있어서, 상기 제 1 업데이팅 알고리즘은 상기 다수의 최상위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 셀룰라 시스템.

청구항 39.

제 30 항에 있어서, 상기 제 2 업데이팅 알고리즘은 상기 다수의 최하위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 셀룰라 시스템.

청구항 40.

브로드캐스트 챌린지 값의 다수의 최상위 비트들을 업데이팅하는 제 1 수단; 및

업데이팅된 2진수를 생성하기 위해 상기 브로드캐스트 챌린지 값의 다수의 최하위 비트들을 업데이팅하는 제 2 수단을 포함하며,

상기 업데이팅하는 제 2 수단은 업데이트 시퀀스가 상기 다수의 최하위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최하위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 기지국.

청구항 41.

제 40 항에 있어서, 상기 업데이팅하는 제 1 수단은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 1 최대길이 시프트 레지스터를 포함하는 기지국.

청구항 42.

제 40 항에 있어서, 상기 업데이팅하는 제 1 수단은 제 1 의사랜덤 잡음 발생기를 포함하는 기지국.

청구항 43.

제 40 항에 있어서, 상기 업데이팅하는 제 2 수단은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 2 최대길이 시프트 레지스터를 포함하는 기지국.

청구항 44.

제 40 항에 있어서, 상기 업데이팅하는 제 2 수단은 제 2 의사랜덤 잡음 발생기를 포함하는 기지국.

청구항 45.

제 40 항에 있어서, 상기 업데이팅하는 제 1 수단은 업데이트 시퀀스가 상기 다수의 최상위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최상위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 것을 포함하는 기지국.

청구항 46.

삭제

청구항 47.

제 40 항에 있어서, 상기 업데이팅하는 제 1 수단은 상기 다수의 최상위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 것을 포함하는 기지국.

청구항 48.

제 40 항에 있어서, 상기 업데이팅하는 제 2 수단은 상기 다수의 최하위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 것을 포함하는 기지국.

청구항 49.

전 시스템 시간 동기화를 수행하는 수단;

다수의 이동 가입자 유니트들; 및

상기 다수의 이동 가입자 유니트들과의 무선 통신을 위해 구성되는 다수의 기지국들을 포함하며,

상기 다수의 기지국들 각각은:

기지국과의 통신을 요청하는 임의의 이동 가입자 유니트를 인증하는데 사용되는 2진수의 다수의 최상위 비트들을 업데이팅하는 제 1 수단;

상기 2진수의 다수의 최하위 비트들을 업데이팅하는 제 2 수단; 및

상기 전 시스템 시간 동기화를 수행하는 수단을 이용하여 상기 다수의 기지국들에 걸친 상기 2진수의 연속적인 업데이트 값들을 동기화하는 수단을 포함하며,

상기 업데이팅하는 제 2 수단은 업데이트 시퀀스가 상기 다수의 최하위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최하위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 셀룰라 시스템.

청구항 50.

제 49 항에 있어서, 상기 전 시스템 시간 동기화를 수행하는 수단은 다수의 기지국들 각각에 GPS 시간 측정치를 전달하는 셀룰라 시스템.

청구항 51.

제 49 항에 있어서, 상기 업데이팅하는 제 1 수단은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 1 최대길이 시프트 레지스터를 포함하는 셀룰라 시스템.

청구항 52.

제 49 항에 있어서, 상기 업데이팅하는 제 1 수단은 제 1 의사랜덤 잡음 발생기를 포함하는 셀룰라 시스템.

청구항 53.

제 49 항에 있어서, 상기 업데이팅하는 제 2 수단은 컴퓨터 소프트웨어를 통해 시뮬레이트되는 제 2 최대길이 시프트 레지스터를 포함하는 셀룰라 시스템.

청구항 54.

제 49 항에 있어서, 상기 업데이팅하는 제 2 수단은 제 2 의사랜덤 잡음 발생기를 포함하는 셀룰라 시스템.

청구항 55.

제 49 항에 있어서, 상기 업데이팅하는 제 1 수단은 업데이트 시퀀스가 상기 다수의 최상위 비트들에 대한 반복 값을 생성하는데 필요한 시간 주기동안 상기 다수의 최상위 비트들에 대한 업데이트로서 올 제로 값을 한번 삽입하는 셀룰라 시스템.

청구항 56.

삭제

청구항 57.

제 49 항에 있어서, 상기 업데이팅하는 제 1 수단은 상기 다수의 최상위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 셀룰라 시스템.

청구항 58.

제 49 항에 있어서, 상기 업데이팅하는 제 2 수단은 상기 다수의 최하위 비트들에 대한 모든 업데이트 값이 제로가 아닌 값이 되도록 보장하는 셀룰라 시스템.

명세서

기술분야

본 발명은 일반적으로 무선 통신 분야에 관한 것이며, 특히 셀룰라 기지국에서의 브로드캐스트 챌린지 값(broadcast challenge value)의 생성에 관한 것이다.

배경기술

무선 통신 분야는 예를 들어, 무선(cordless) 전화, 페이징, 무선 가입자 회선들 및 위성 통신 시스템들을 포함하여 많은 응용들을 갖는다. 특히 중요한 응용은 이동 가입자들에 대한 셀룰라 전화 시스템이다. (여기서 사용되는, "셀룰라" 시스템이라는 용어는 셀룰라 및 PCS 주파수들을 포함한다.) 다양한 무선(over-the-air) 인터페이스들이 예를 들어, 주파수 분할 다중 접속 방식(FDMA), 시분할 다중 접속 방식(TDMA) 및 코드 분할 다중 접속 방식(CDMA)을 포함한 셀룰라 전화 시스템들을 위해서 개발되었다. 이와 관련해서, 예를 들어, AMPS(Advanced Mobile Phone Service), GSM(Global System for Mobile) 및 잠정 협정 표준 95(IS-95)를 포함하는 다양한 국내 및 국제 표준들이 설립되었다. 특히, IS-95 및 그것의 파생물들인, IS-95A, ANSI J-STD-008 등등은(여기서는 총칭적으로 IS-95로 지칭되는) 통신 산업 협회(TIA) 및 다른 잘 알려진 표준기관들에 의해 공표되었다.

IS-95 표준의 사용에 따라 형성된 셀룰라 전화 시스템들은 매우 효율적이며 신뢰성있는 셀룰라 전화 서비스를 제공하기 위해 CDMA 신호 처리 기술들을 사용한다. IS-95 표준의 사용에 따라 형성된 전형적인 셀룰라 전화 시스템은 본 발명의 양수인에게 양도되었고 여기서 참조되는 미국 특허 No. 5,103,459에 기술되어 있다. 전술한 특허는 CDMA 기지국에서의 전송 신호 처리 또는 순방향 링크 신호 처리를 설명한다. CDMA 기지국에서의 전형적인 수신 신호 처리 또는 역방향 링크 신호 처리는 본 발명의 양수인에게 양도되었고 여기서 참조된, 1997년 12월 9일에 출원된 "다중 채널 복조기"라 명칭된 미국 특허 출원 No. 08/987,172에 기술되어 있다. CDMA 시스템에서는, 전력 제어가 중요한 문제이다. CDMA 시스템에서 전형적인 전력 제어 방법은 본 발명의 양수인에게 양도되었고 여기서 참조되는 미국 특허 No. 5,056,109에 기술되어 있다.

CDMA 무선 인터페이스를 사용하는 주된 이점은 통신들이 동일한 RF 대역을 통해 수행된다는 것이다. 예를 들어, 주어진 셀룰라 전화 시스템에서 각 이동 가입자 유니트(일반적으로 셀룰라 전화기)는 동일한 1.25 MHz의 RF 스펙트럼을 통해 역방향 링크 신호를 전송함으로써 동일한 기지국과 통신할 수 있다. 비슷하게, 상기 시스템의 각 기지국은 또 다른 1.25 MHz의 RF 스펙트럼을 통해 순방향 링크 신호를 전송함으로써 이동 유니트들과 통신할 수 있다.

동일한 RF 스펙트럼을 통해 신호들을 전송하는 것은 예를 들어, 셀룰라 전화 시스템의 주파수 재사용의 증가 및 두개나 그 이상의 기지국들 사이의 소프트 핸드오프(soft handoff)를 수행하는 능력을 포함하여 다양한 이점들을 제공한다. 증가된 주파수 재사용은 주어진 양의 스펙트럼을 통해 훨씬 더 많은 수의 통화가 이루어지도록 한다. 소프트 핸드오프는 두 기지국들과 동시에 인터페이스하는 둘 이상의 기지국들의 서비스 범위 영역으로부터 이동 유니트를 전환시키는 효율적인 방법이다. (반대로, 하드 핸드오프는 제 2 기지국과 인터페이스를 형성하기 전에 제 1 기지국과의 인터페이스를 종료하는 것과 관련된다.) 소프트 핸드오프를 실행하는 전형적인 방법은 본 발명의 양수인에게 양도되었고 여기서 참조되는 미국 특허 No. 5,267,261에 기술되어 있다.

당업자들에 의해 이해된대로, CDMA 기술은 셀룰라 시스템이외에 무선 가입자 회선 시스템들 및 위성 통신 시스템들에 적용될 수 있다.

일반적으로 셀룰라 전화 시스템들에서, 이동 가입자 유니트 또는 이동국들은 전화 연결들과 같은 서비스들에 대한 액세스가 허용되기 전에 기지국에 의해 인증되어야 한다. 셀룰라 통신 표준들은 일반적으로 셀룰라 하부구조(기지국들 및/또는 기지국 제어기들)에 의해 제공되는 서비스를 사용하여 이동국들의 인증에 대한 절차들을 정의한다. TIA 에 의해 공표된 셀룰라 표준들은 이동국들을 인증하는 두 가지 방법을 제공한다. 그 방법들은 "유니크 챌린지(unique challenge)"방법 및 "브로드캐스트 챌린지(broadcast challenge)"방법이라 불린다. 이런 방법들을 사용하는 TIA 표준들은 IS-91(AMPS 표준), IS-54(아날로그 제어 채널들을 정의하는 TDMA 표준), IS-136(디지털 제어 채널들을 정의하는 TDMA 표준) 및 IS-95를 포함한다.

유니크 챌린지 방법은 당업자들에게 잘 알려져 있다. 유니크 챌린지 방법하에서는, 셀룰라 하부구조 장치는 챌린지 값을 이동국에 송신하고, 이동국은 이러한 챌린지 값, 이동국 식별자, 및 특정 식별자를 갖는 적법한 이동국과 기지국에만 알려진 비밀 데이터로부터 계산된 응답을 되돌려보낸다. 그 응답이 맞으면, 셀룰라 하부구조장치는 전화 연결과 같은 서비스들에 대한 액세스를 제공한다. 유니크 챌린지는 챌린지-응답 처리를 완성하는데 요구되는 시간이 상대적으로 길고 콜 셋업(call setup)을 심각하게 지연시킬 수 있는 단점을 갖는다. 이런 이유로, 브로드캐스트 챌린지 방법이 셀룰라 서비스들에 대한 액세스의 요구들의 빠른 인증을 제공하는 수단으로 TIA 셀룰라 표준들에서 포함되었다.

브로드캐스트 챌린지 방법하에서는, 챌린지 값(일반적으로 "RAND"라 지칭됨)은 셀룰라 제어 채널들을 통해 방송된다. 셀룰라 서비스들에 대한 액세스를 요구하는 이동국은 챌린지에 대한 응답을 계산하는데 브로드캐스트 챌린지 값을 사용하며, 그 응답은 챌린지 값, 이동국 식별자 및 그 식별자를 갖는 이동국 및 기지국에만 알려진 비밀 정보를 사용하여 계산된다. 이동국은 서비스에 대한 요청에 그 응답을 포함한다.

브로드캐스트 방법은 부정한 이동국이 적법한 이동국들로부터 통신들을 모니터하고 적법한 이동국에 대한 식별자와 브로드캐스트 챌린지에 대한 그 이동국의 응답 둘다를 재사용하는 "재생" 어택(attack)을 받을 수 있다. 재생 어택을 방지하는 여러 공지된 방법들이 있다. 그럼에도 불구하고, 재생 어택을 방지하는 주요 종래 방법들은 브로드캐스트 챌린지 값을 자주 바꾸는 것이다. 브로드캐스트 챌린지 값이 종래의 전화 통화의 지속시간에 필적하는 업데이트(update) 시간간격으로 변하면, 재생 어택들은 이동국으로부터의 통화가 이미 진행중인 동안에 같은 이동국에서 나온 것 같은 액세스들을 거절함으로써 단순히 차단될 수 있다. 요즘은 셀룰라 전화 통화의 예상 지속시간은 약 1분이다.

그러나, RAND의 이러한 빈번한 변경들은 중앙에서 관리되는 하부구조 장치에 대해서는 어려울 수 있는데, 왜냐하면 RAND 값은 많은 수의 셀 사이트로부터 전송되며 모든 셀 사이트들 내의 모든 장치들이 RAND 를 바꾸기 위해 업데이트되어야 하기 때문이다. 이것은 셀룰라 하부구조의 내부 제어 시스템상에 상당한 통신 부담을 준다. 또한, RAND의 업데이트는 어떤 RAND 값이 응답을 계산하는데 사용되었는지를 이동국이 식별할 것을 요구한다. 이동국이 RAND 갱신이 시작되고 바로 액세스를 시작하였다면, 이동국이 업데이트된 값이 아니라 이전의 RAND값을 사용할 가능성이 존재한다. 따라서, 셀룰라 하부구조는 최근의 모든 RAND 값들에 대한 응답을 계산 및 수용하지 않는 것이 바람직한데, 그 이유는 예상된 응답의 계산이 느릴 수 있기 때문이며, 이것은 랜덤하게 선택된 응답이 성공할 수 있는 가능성을 증가시킴으로써 RAND의 효율성을 떨어뜨리기 때문이다.

그러나, 신호 전송의 신뢰도를 높이고 대역폭을 보존하기 위해서는 무선 인터페이스상에 전송되어야만 하는 비트 수를 최소화하는 것이 바람직하다. 그러므로, 이동국 액세스 요구에 대한 TIA 표준들은 일반적으로 액세스 요구에 완전한 RAND 값을 포함하고 있지 않다. 대신에, RAND의 가장 중요한 부분만이 액세스 요구에 포함되어 전송되고, 따라서 어떤 RAND 값이 사용되었는지를 식별하기 위해 더 적은 수의 비트들을 사용한다. TIA 표준들에서는, RAND의 최상위 8개 비트들("RANDC"라 지칭되는)이 사용된다. 그러나 이 기술은 RAND가 업데이트될 때 RAND의 최상위 비트들이 변경되는 경우에만 성공한다. 따라서, RANDC가 각각 새로운 값의 RAND에 대해 구별되도록 RAND 업데이트 프로세스가 실행되는 것이 요구된다.

TIA 표준들에서, RAND는 일반적으로 32 비트 길이이며(0 비트에서 31 비트까지), RANDC라 지칭되는 최상위 8개 비트(24 비트에서 31 비트까지)를 갖는다. 이동국은 액세스 요구 메세지를 통해 RAND에 대한 응답과 함께 RANDC를 리턴시킨다. 기지국은 유효 RAND 값들의 리스트를 유지하여야 하며, RANDC만을 이용하여 이동국에 의해 리턴된 응답을 계산하는데 어떤 RAND 값이 사용되었는지를 결정하여야 한다. 따라서 최근에 사용된 모든 RAND 값들은 고유한 RANDC 값들을 가질 것이 요구된다.

RAND 값들을 선택하는데 있어서 상기와 같은 것들을 고려하는 것 이외에, RAND 값이 재사용되기 전의 시간 주기를 최대화하는 것이 보안상의 이유로 바람직하며, 그로인해 인증 서명이 재생될 수 있기 전에 대기 시간이 길어지게 된다. 이것은 사실, RAND에 대해 진정한 랜덤 숫자를 사용하는 것이 바람직하지 않다는 것을 의미한다. 대신, 가능한 RAND 값들에 대한 최대 사이클을 보장하는 결정성 알고리즘을 사용하는 것이 바람직할 것이다.

또한, 대부분 셀룰라 시스템들에서는 RANDC에 대해 제로(zero)값을 갖는 것이 허용되지 않는데, 왜냐하면 이동국이 RAND의 현재 값을 갖지 않으며, 이에 대한 응답을 계산하는데 모든 제로값을 사용했다는 것을 표시하기 위해 제로값이 이동국에 의해 사용되기 때문이다. 그러므로 RAND 업데이트 프로세스의 특성은 RANDC가 각 갱신에 대해 구별되며 제로가 아니라는 것이 보장되어야 한다.

또한, 응답 생성 프로세스상의 상이한 어택들에 대한 성공 가능성을 최소화하도록 연속적인 RAND 값들이 가능한 한 상이한 값을 갖는 것이 바람직하다. 이것은 단순한, 카운터기반의 방식은 충분하지 않으며, 연속적인 값들 사이에 낮은 상관 관계를 갖는 업데이트 방법들이 바람직하다는 것을 의미한다.

마지막으로, 새로운 RAND 값들의 계산을 분산화(de-centralizing)해서 셀룰라 시스템의 상호연결 망내에서의 메시징을 최소화하는 것이 바람직하다.

따라서, 연속적인 RAND 값들사이의 상관관계를 최소화하고, RAND 와 RANDC에 대한 최대 주기성을 확보하며, 업데이트 프로세스를 트리거하기 위해 모든 셀 사이트가 중앙 제어 장치로부터의 메시징없이 동일한 업데이트들을 실행하도록 하는 생성 방법이 필요하다.

발명의 상세한 설명

본 발명은 연속적인 RAND 값들사이의 상관관계를 최소화하고, RAND 및 RANDC에 대한 최대 주기성을 확보하며, 모든 셀 사이트가 업데이트 프로세스를 트리거하기 위해 중앙 제어 장치로부터의 메시징없이 동일한 업데이트들을 실행하는 것을 허용하는 생성 방법에 관한 것이다. 따라서, 브로드캐스트 챌린지 값을 생성하는 방법은 2진수의 최상위 비트들에 제 1 업데이트 알고리즘을 적용하는 단계 및 2진수의 최하위 비트들상에 제 2 업데이트 알고리즘을 수행하는 단계를 포함한다. 바람직하게도, 이러한 업데이트 알고리즘들은 별개의 최대-길이 시프트 레지스터들을 시뮬레이트한다. 바람직하게는, 모두 0인 값은 업데이트 시퀀스가 반복 값을 생성하는데 필요한 시간 주기동안 비트들의 각 세트에 대한 업데이트로서 한번 삽입된다.

본 발명의 첫번째 특징은, 2진수가 최대 주기성 및 연속적인 업데이트들 사이의 최소 상관관계를 가지고 업데이트되는 것이다. 업데이트들은 2진수의 개별 부분들상에 별개의 알고리즘들을 동작시킴으로써 유리하게 실행된다.

본 발명의 두번째 특징에서, 2진수가 셀룰라 시스템의 모든 기지국들에 걸쳐 동기적으로 업데이트된다. 바람직하게는, 업데이트들은 전 시스템의 타임 클럭 기준 신호로 동기화된다.

실시예

본 발명의 특징들을 구현하는 RAND 생성 방법은 당업자가 이해하는 바와 같이 여러 셀룰라 전화 시스템들 중 어느 한 시스템에서 수행될 수 있다. 이런 셀룰라 시스템들은 AMPS(아날로그), IS-54(북미 TDMA), GSM(이동 통신에 관한 세계 시스템 TDMA), IS-95(CDMA)를 포함한다.

도 1에 도시된 대로, CDMA 무선 전화 시스템은 일반적으로 다수의 이동 가입자 유니트(10), 다수의 기지국(12), 기지국 제어기(BSC)(14) 및 이동 스위칭 센터 (MSC)(16)를 포함한다. MSC(16)는 종래의 공중 교환 전화망(PSTN)(18)과 인터페이스하도록 구성된다. MSC(16)는 또한 BSC(14)와 인터페이스하도록 구성된다. BSC(14)는 각 기지국(12)과 연결된다. 기지국(12)은 또한 기지국 송수신기 서브시스템(BTSs) (12)으로 알려져 있다. 대안적으로, "기지국"은 집합적으로 BSC (14) 및 하나 이상의 BTSs(12)를 지칭할 수 있으며, BTSs(12)는 또한 "셀 사이트"(12)라 지칭될 수 있다. (대안적으로, 주어진 BTS(12)의 섹터들은 셀 사이트로 지칭될 수 있다.) 이동 가입자 유니트(10)는 일반적으로 셀룰라 전화기(10)이며, 셀룰라 전화 시스템은 IS-95 표준에 따라 구성된 CDMA 시스템인 것이 유리하다.

셀룰라 전화 시스템의 일반적인 동작동안, 기지국(12)은 이동 유니트(10) 세트로부터 역방향 링크 신호들 세트를 수신한다. 이동 유니트(10)는 전화 통화나 다른 통신들을 수행한다. 주어진 기지국(12)에 의해 수신된 각 역방향 링크 신호는 그 기지국(12)내에서 처리된다. 상기 처리된 데이터는 BSC(14)로 보내진다. BSC(14)는 기지국(12)들 간의 소프트 핸드오프의 조정을 포함하는 이동성 관리 기능 및 통화 자원 할당을 제공한다. BSC(14)는 또한 수신된 데이터를 MSC(16)로 라우트(route)시키며, MSC(16)는 PSTN(18)과 인터페이스하도록 부가적인 라우팅 서비스들을 제공한다. 비슷하게, PSTN (18)은 MSC(16)와 인터페이스하며, MSC(16)는 BSC(14)와 인터페이스하며, BSC(14)는 차례로 순방향 링크 신호들 세트를 이동 유니트(10) 세트에 전송하도록 기지국(12)을 제어한다.

도 1의 CDMA 시스템에서, 각 기지국(12)은 적어도 하나의 섹터(미도시)를 포함하며, 각 섹터는 기지국(12)으로부터 방사상으로 멀어지는 특정 방향으로 지향되는 안테나를 포함한다. 바람직하게는, 각 기지국(12)은 세개의 섹터들 및 방사성 방향들을 포함하고, 각 섹터 안테나 포인트는 120도만큼 이격된다.

바람직하게도, 새로운 브로드캐스트 챌린지 값들을 생성하거나 RAND를 업데이팅하기 위해서 최대 길이 시프트 레지스터들 혹은 그것들의 소프트웨어 시뮬레이션들이 기지국(12)에서 이용될 수 있다. 바람직한 실시예에서는, 갈로이스(Galois) 시프트 레지스터들이 최대길이 시프트 레지스터들로서 사용된다. 예를 들어, 선형 피드백 시프트 레지스터들(LFSRs)과 같이 공지된 최대 길이 시프트 레지스터들이 갈로이스 시프트 레지스터들 대신 사용될 수 있다.

당업자가 이해하는 바와 같이, 갈로이스 시프트 레지스터는 각각의 비트의 위치를 각 클럭 펄스를 통해 한 칸씩 왼쪽으로 시프트시키는데, 여기시 미리 결정된 비트들이 피드백 탭 비트와 XOR 연산된다. 따라서, 도 2에서 도시된 것처럼, 8비트 갈로이스 시프트 레지스터(20)는 비트 0,4,5위치 다음에 피드백 탭들을 포함한다. 각각 왼쪽으로 시프트 될 때, 비트 위치(1)는 비트(7)와 비트(0)의 XOR 결과를 수신한다. 비슷하게, 비트 위치(5)는 비트(7)와 비트(4)의 XOR 결과를 수신하며, 비트 위치(6)는 비트(7)와 비트(5)의 XOR 결과를 수신한다.

도 3에 도시된 특정한 실시예에서, RAND 업데이트 방법은 제 1 및 제 2 갈로이스 레지스터들(30),(32)에 기반을 두고 있다. 단일 업데이트 클럭 신호(34)는 두 시프트 레지스터들(30),(32)의 업데이트를 발생시킨다. 제 1 시프트 레지스터(30)는 RANDC의 연속적인 값들을 결정하는데 사용되는 8비트 시프트 레지스터(30)이다. 제 2 시프트 레지스터(32)는 RAND의 잔여 비트들의 연속적인 값들을 결정하는데 사용되는 24비트 시프트 레지스터(32)이다. 제 1 및 제 2 시프트 레지스터들 (30),(32)은 둘다 동일한 클럭 신호(34)에 결합되지만, 서로 연결되지는 않는다. 그러므로, RANDC는 RAND의 나머지와는 별개로 생성된다.

도시된 실시예에서, 제 1 및 제 2 시프트 레지스터들(30),(32) 각각은 레지스터들(30),(32)에서 특정 비트 위치들에 인가되는 피드백 탭들 또는 전기적인 접속들을 갖는다. 8차 및 24차 원시 다항식들은 각각 제 1 및 제 2 시프트 레지스터들(30),(32)의 피드백 탭들을 결정하는데 사용될 수 있다는 것을 당업자는 알고 있다. 도시된 구체적 실시예에서, 제 1 및 제 2 시프트 레지스터들(30),(32)에 대한 원시 다항식들은 각각, $x^8 + x^6 + x^5 + x + 1$ 및 $x^{24} + x^4 + x^3 + x + 1$이다.

각 시프트 레지스터(30),(32)가 제로가 아닌 값으로 초기화되면, 각 시프트 레지스터(30),(32)가 생성하는 값은 항상 제로가 아닐 것이다. 유리하게도, 이것은 RANDC가 제로가 아니라는 조건을 충족시킨다. 그러나, 이런 배열은 RAND 값들의 시퀀스의 길이를 최대화하지 않는데 그 이유는 두개의 시프트 레지스터들(30),(32)이 각각 서로에 대해 소수(prime)가 아닌 길이들의 시퀀스들, 즉 $2^8-1 = 255 = 3*5*17$ 및 $2^{24}-1 = 16777215 = 3*3*5*7*13*17*241$을 생성하기 때문이다. 따라서, 이런 배열로 생성된 RAND 값들의 시퀀스는 길이가 겨우 65793이며, 이 길이는 단일의 16 비트 시프트 레지스터에 의해 생성되는 것보다 단지 조금 더 긴 것이다.

최대 길이 시퀀스는 전체가 제로인 값들을 포함하지 않기 때문에, RAND 값들의 시퀀스 길이는 전체가 제로인 값을 삽입함으로써 확장될 수 있다. 그러므로, 24 비트 시프트 레지스터(32)의 전체가 제로인 값은 시퀀스의 임의의 지점에서 유리하게 삽입될 수 있으며, 그로인해 24 비트 값들의 시퀀스 길이를 2의 거듭제곱인 16777216까지 증가시키며, 이 값은 RANDC 시퀀스의 길이인 255에 대해 소수이다. 이것은 유리하게 RAND 값들의 시퀀스를 $2^{32} - 2^{24}$의 결과와 같은 255*16777216 까지 연장한다. 이것은 RANDC가 반드시 제로가 아닌 경우의 최대 가능 시퀀스 길이이다.

도 4의 흐름도는 제 1 및 제 2 갈로이스 시프트 레지스터들(RANDC 및 RANDL로 지칭되는 잔여 RAND 비트들에 대응함)이 컴퓨터 소프트웨어로 시뮬레이트되는 특정 실시예에 따른 RAND 업데이트 방법을 설명한다. 도 4의 방법은 시뮬레이트되는 24비트 RANDL 갈로이스 시프트 레지스터에 의해 생성되는 시퀀스에 전체가 제로인(all-zeros) 값을 편리하게 삽입시키며, 상기 방법은 당업자에게 공지된 C 코드나 C++ 코드를 포함하는 임의의 종래 소스 코드로 실행될 수 있다. 셀 사이트들은 일반적으로 집적회로들을 포함하며, 상기 집적회로들은 소프트웨어를 운용하는 프로세서들을 갖는 응용 주문형 집적회로들(ASICs)이다.

단계(40)에서, 알고리즘은 "INIT"라 지칭되는 8자리 16진수(32비트 2진수)와 8자리 16진수 00FFFFFF의 AND 결과를 계산한다. 이 계산은 INIT의 24 최하위 비트들이 모두 0인지 아닌지를 결정한다. INIT의 24 최하위 비트들이 모두 0이라면, 알고리즘은 단계(42)로 진행한다. 그렇지 않으면, 알고리즘은 단계(44)로 진행한다.

단계(42)에서, INIT는 INIT와 1의 OR 결과와 같은 값으로 고정된다. 이 단계는 INIT의 최하위 비트에 1의 값을 제공한다. 단계(40) 및 (42)는 레지스터가 제로 아닌 값으로 초기화되는 것을 보장하며, 만약 시작단계에서 그것이 제로라면 레지스터를 1의 값으로 강제고정시킨다. 그리고 나서, 알고리즘은 단계(44)로 진행한다. 단계(44)에서 알고리즘은 INIT와 16진수 00FFFFFF의 AND 결과값을 계산한다. 이 결과를 "PREINIT"라 부른다. 따라서, PREINIT는 최상위 8비트들이 0인 32비트 2진수이다. 위의 값을 PREINIT에 제공한후에, 알고리즘은 단계(46)로 진행한다.

단계(46)에서, PREINIT 와 1의 AND 결과값이 결정된다. 이것은 알고리즘이 PREINIT의 최하위 비트가 1인지 아닌지를 체크하도록 허용한다. 만약 PREINIT와 1의 AND 결과값이 1이면, 알고리즘은 단계(48)로 진행한다. PREINIT와 1의 AND 결과값이 1이 아니면, 알고리즘은 단계(50)로 진행한다.

단계(48)에서, 알고리즘은 PREINIT와 8자리 16진수 0100001B의 XOR 값을 계산한다. PREINIT와 0100001B의 XOR 결과값은 PREINIT의 새로운 값이 된다. 그 다음에 알고리즘은 단계(50)로 진행한다. 단계(50)에서 PREINIT 값은 한 비트씩 오른쪽으로 즉, 레지스터가 시간상으로 한 클럭 펄스 이전에 가졌던 값으로 시프트된다. 그 다음에 알고리즘은 단계(52)로 진행한다. 단계(48) 및 (50)은 RANDL에 대해 갈로이스 시프트 레지스터 기능들을 반대 순서로 효과적으로 시뮬레이트한다. 따라서, PREINIT의 24 최하위 비트들은 먼저 19개의 0과 후속하는 시퀀스 11011로 구성되는 2진값과(즉, 16진법 숫자 0100001B의 마지막 6개 숫자들인 16진수 00001B) XOR 연산 되었으며, 25번째 비트는 1로 설정된다. 시퀀스 11011은 비트 위치 0,1,3 및 4에서 피드백 탭들을 시뮬레이트한다. 그 후에, 다음 명령으로, 시뮬레이트된 시프트 레지스터는 한 비트씩 오른쪽으로 시프트되며, 따라서 25번째 비트는 24번째 비트위치로 시프트한다. (대조적으로, 도 2에서 갈로이스 시프트 레지스터는 왼쪽으로 시프트하고, 피드백 탭 비트들을 XOR 연산한다)

단계(52)에서, 32 비트 2진 RAND 값과 32 비트 2진수 00FFFFFF의 AND 결과값이 얻어진다. 이 AND 결과값은 "LSMASK"라 불린다. 이 계산은 LSMASK가 8개의 0들과 후속하는 RANDL의 24비트들(RAND의 최하위 24비트들)과 같은 값을 갖도록 한다. 그런 다음 알고리즘은 단계(54)로 진행한다.

단계(54)에서, 알고리즘은 LSMASK가 0인지 아닌지, 즉 LSMASK의 최하위 24비트들이(RANDL을 구성하는) 모두 0인지 아닌지를 체크한다. 만약 LSMASK가 0이면, 알고리즘은 단계(56)로 진행한다. 그렇지 않다면, 알고리즘은 단계(58)로 진행한다.

단계(56)에서, RAND 와 PREINIT의 OR 결과값이 얻어지며 새로운 RAND 값이 만들어진다. 따라서, RAND는 8개의 0들과 후속하는 PREINIT의 최하위 24비트들이 된다.(즉, RANDC가 0이고 RANDL이 PREINIT의 최하위 24비트들이다) 그 다음에 알고리즘은 단계(62)로 진행한다.

단계(58)에서, 알고리즘은 LSMASK가 PREINIT와 같은지 아닌지를 체크한다. LSMASK가 PREINIT와 같지 않으면, 알고리즘은 단계(62)로 진행한다. 반대로, LSMASK와 PREINIT이 같으면, 알고리즘은 단계(60)로 진행한다. 단계(60)에서 RAND 와 8자리 16진수 FF000000(8개의 1들과 뒤이은 24개의 0들)의 AND 결과값이 계산되고 새로운 RAND 값이 만들어진다. 따라서, RAND는 24개의 0이 후속하는(즉, RANDL이 0인) RANDC가 된다. 그 다음에 알고리즘은 단계(62)로 진행한다.

단계(62)에서, "MASK"라 불리는 값은 0으로 세팅된다. 그 다음에 알고리즘은 단계(64)로 진행한다.

단계(64)에서, 알고리즘은 RAND와 8자리 16진수 80000000(1개의 1과 후속하는 31개의 0들)의 AND 결과값을 계산한다. 그 다음에 알고리즘은 이 AND 결과값이 0인지 아닌지를 체크한다. 다시말해 알고리즘은 RAND의 최상위 비트가 1인지 아닌지를 결정한다. 이 AND 결과값이 0이 아닌 것으로 결정되면, 알고리즘은 단계(66)로 진행한다. 그러나 만약, AND 결과값이 0으로 결정되면, 알고리즘은 단계(68)로 진행한다.

단계(66)에서, MASK는 8자리 16진수 63000000으로 주어진다.(RANDC는 위에서 검토된대로, 0x163의 원시 다항식, 즉 2진수 101100011, 혹은 원시 다항식 $x^8 + x^6 + x^5 + x + 1$ 을 갖는 8비트 레지스터로 정의된다. MASK 값은 163000000일 필요는 없는데, 이것은 최상위비트는 시뮬레이트되는 RANDC 레지스터의 끝에서 오프이기 때문이다.) 그 다음에 알고리즘은 단계(68)로 진행한다. 단계(68)에서 알고리즘은 RAND와 8자리 16진수 00800000(8개의 0들, 뒤이은 1개의 1, 및 뒤이은 23개의 0들)의 AND 결과값을 계산한다. 그 다음에 알고리즘은 이 AND 결과값이 0이 아닌지를 체크한다. 다시말해 알고리즘은 RANDL의 최상위비트가 1인지 아닌지를 결정한다. AND 결과값이 0이 아닌 것으로 결정되면, 알고리즘은 단계(70)로 진행한다. 그러나 만약 AND 결과값이 0으로 결정되면, 알고리즘은 단계(72)로 진행한다. 단계(70)에서 MASK와 8자리 16진수 0100001B의 XOR 결과값이 새로운 MASK 값으로 된다. 이 XOR 결과는 위에서 검토된대로, 다항식 $x^{24} + x^4 + x^3 + x + 1$ (0x0100001B)의 24비트 RANDL 시뮬레이트된 레지스터에 대한 MASK 값을 산출하는 것으로 이해될 수 있다. 그 다음에 알고리즘은 단계(72)로 진행한다.

단계(72)에서, RAND는 한 비트씩 왼쪽으로 시프트된다. 그 다음에 알고리즘은 단계(74)로 진행한다. 단계(74)에서 알고리즘은 RAND와 MASK의 XOR 결과값을 계산한다. 이 XOR 결과값은 RAND의 새로운 값으로 된다. 그 다음에 알고리즘은 단계(76)로 진행한다. 단계(76)에서 알고리즘은 RAND의 업데이트된 값으로 단계(74)에서 유도된 RAND 값을 리턴시킨다.

따라서, 도 4에 관하여 기술된 방법에서, RANDC는 0x163의 원시 다항식을 갖는 8비트 갈로이스 시프트 레지스터로 정의되며, RANDL은 0x0100001B의 원시 다항식을 갖는 24비트 갈로이스 시프트 레지스터로 정의된다. 0값은 일반적으로 0x00000001과 같지 않는 INIT 바로 전에 삽입되며, INIT는 , 따라서 그 주기는 2^{24}까지 증가하게 되어, RANDC의 주기 255에 대해 소수인 주기를 만든다. 2개의 시뮬레이트된 갈로이스 시프트 레지스터들은 상이한 피드백 탭들을 가지며(왜냐하면 이 레지스터들은 상이한 원시 다항식들을 갖기 때문에), 따라서 구별되는 별개의 알고리즘들을 유리하게 구성한다.

RAND 업데이트들을 생성하깅 위해, 도 4에 기술된 방법은 셀룰라 시스템의 하부구조 어디에서도 실행될 수 있을 만큼 충분히 간단하다는 것을 알 수 있다. 구체적인 실시예에서, 그 방법은 도 1의 CDMA 셀룰라 시스템의 셀 사이트들(미도시)에서 실행될 수 있으며, 그로 인해 중앙 집중 방식으로 새로운 RAND 값들을 생성하여 브로드캐스트를 위해 셀 사이트들에 그 값들을 분배할 필요성을 제거한다.

셀 사이트들이 RANDC로부터 RAND의 국부적(local) 복원을 실행하면서, 동기화없이 주기적으로 RAND를 업데이트하도록 하는 것이 가능하다. 그러나, 이것은 보안의 차원에서는 바람직하지 않는데, 왜냐하면 이는 RAND 값들의 시퀀스 길이를 사실상 감소시키기 때문이다. 즉, 주어진 셀 사이트가 RAND 값들의 최대길이 시퀀스를 생성할지라도, 비동기적으로 이러한 값들을 생성하는 근접한 셀 사이트는 제 1 셀 사이트보다 더 빨리 제 1 셀 사이트의 RAND 값들 중 하나를 반복시킬 것이다. 최악의 경우에는, 근접한 셀 사이트는 하나의 셀 사이트에서 다른 사이트로 재생들이 쉽게 수행되는 것을 허용하면서, 단지 조금의 RAND 값들만 남기게 될 것이다.

그럼에도 불구하고, 그러한 셀 사이트간 재생 어택들은 모든 RAND 값들이 주어진 모든 시간에서 동일하도록 하는 방식으로 모든 셀 사이트들에 걸쳐 RAND 업데이트들을 동기화시킴으로써 간단하게 방지될 수 있다. 구체적인 실시예에서는, 글로발 타임 클럭이 이 동기화를 달성하는데 사용된다. 모든 셀 사이트들이 이미 10 μs 내에서 동기화되는 CDMA 확산 시퀀스들을 반드시 생성하여야 하기 때문에, 그러한 글로발 타임 클럭은 TIA 표준 IS-95에 기술된 CDMA 시스템에서 쉽게 이용가능하다. IS-95 CDMA 표준은 글로발 위치 시스템(GPS)의 시간 기준에 맞춰지는 글로발 "시스템 시간"을 정의한다. 당업자는 셀 사이트들 간의 시간 기준을 맞추는 임의의 비슷한 방법이 사용될 수 있음을 잘 이해할 것이다.

모든 셀 사이트들이나 기지국들에 이용가능한 글로발, 또는 시스템 와이드 시간 기준이 존재하기 때문에 모든 셀 사이트들이 동일한 RAND를 사용하는 것을 보장할 수 있다. 기준 시간 이후로 발생했던 업데이트들의 총 숫자를 계산함으로써, 그리고 업데이트들의 그 총 숫자로 인해 RAND 값을 결정함으로써 각 셀 사이트는 RAND 값을 셀 사이트 초기화 동안에 세팅할 수 있다. 도 1의 CDMA 시스템에서, 예를 들어, 1980 년 1월 6일 자정과 같이, GPS 시간이 제로로 정의됨과 동시에 시스템 시간도 제로로 정의된다.

RAND의 현재 값은 도 5의 흐름도에 의해 기술된 특정한 실시예에 따라 유도될 수 있는데, 상기 흐름도는 당업자에 의해 숙지된대로, C 코드나 C++ 코드를 포함하는 임의의 종래의 소스 코드로 실행될 수 있다. 셀 사이트들은 일반적으로 집적 회로들을 포함하는데, 이 집적 회로는 소프트웨어를 운용하는 프로세서들을 갖는 응용 주문형 집적 회로들 (ASICs)이다. 도 5에서 도시된 방법은 클럭킹없이 현재 값들을 계산하는데 소프트웨어 시뮬레이트된 갈로이스 시프트 레지스터의 특성들을 사용하며, 그로인해 RAND의 현재값에 도달하는데 계산상의 비효율성을 방지한다.

도 5에 관해 기술된 실시예에서, "SYSTIME"로 지칭되는 64 비트 필드는 시스템이 가동된 이후의 프레임들 수로 현재 시스템 시간을 표시한다. "UPDATE_TIME"이라 지칭되는 32 비트 필드는 RAND 업데이트 간격당 시간을 분 단위로 나타낸다. 따라서, 분 단위의 시스템 시간이 제로 모듈로(modulo) UPDATE_TIME일 때 RAND 업데이트들이 발생한다. 32 비트 필드 "INIT"는 시스템 개시에서의 RAND의 초기 값을 준다. 위에서 기술된대로, RANDC는 RAND의 최상위 8비트들을 나타내며 RANDL은 RAND의 최하위 24비트들을 표시한다(RAND는 32비트 2진수이다). 바람직하게는, 도 1에 관해 기술된 CDMA 시스템이 사용됨으로써 프레임 속도가 20ms/frame 또는 3000 frames/min로 정의되어 사용되며, 시스템 시간의 개시는 1980년 1월 6일 자정으로 정의된다.

단계(80)에서는, 알고리즘은 INIT와 16진수 00FFFFFF의 AND 결과값을 결정한다. 이 AND 결과값은 "LSINIT"라 불리는 32 비트 변수로 할당되며, LSINIT는 8개의 0들과 후속하는 INIT의 최하위 24비트들과 같다. 그 다음에 알고리즘은 단계(82)로 진행한다. 단계(82)에서 INIT는 24비트 만큼 오른쪽으로 시프트된다. 그 다음에 알고리즘은 단계(84)로 진행한다. 단계(84)에서 INIT 값은 "MSINIT"라는 32비트 변수에 할당되고, 따라서 MSINIT는 24개의 0들과 후속하는 INIT의 최상위 8비트와 같다. 그리고 나서 알고리즘은 단계(86)로 진행한다. 단계(86)에서 알고리즘은 MSINIT가 0인지 아닌지를

체크한다. MSINIT가 0이면, 알고리즘은 단계(88)로 진행한다. 그렇지 않으면 알고리즘은 단계(90)로 진행한다. 단계(88)에서 알고리즘은 MSINIT의 최하위 비트를 1로 세팅한다. 그 다음에 알고리즘은 단계(90)로 진행한다. 따라서, 단계(80)에서 단계(88)까지에서, 알고리즘은 RANDC와 RANDL의 초기값들을 결정하였다.

단계(90)에서, 알고리즘은 숫자 3000을 나타내는 32비트 2진 필드로 SYSTIME를 나눈다. 그로 인한 몫은 "UPDATES"라 불리는 64비트 변수에 할당된다. 따라서, 알고리즘은 3000frames/min의 프레임 속도로 시스템 개시후의 프레임들 수를 나눔으로써 프레임 단위의 시스템 시간을 분 단위의 시스템 시간으로 변환하였다. 다음에 알고리즘은 단계(92)로 진행한다.

단계(92)에서는, 알고리즘은 UPDATE_TIME으로 UPDATES를 나눈다. UPDATES는 최종 몫과 같도록 세팅된다. 따라서, 알고리즘은 RAND 업데이트당 분 단위 시간으로 시스템 개시후의 총 분 단위 시간을 나눔으로써 시스템을 개시한 이후의 RAND 업데이트 횟수를 계산하였다. 다음에 알고리즘은 단계(94)로 진행한다.

단계(94)에서, 숫자 255를 나타내는 32비트 2진 필드로 UPDATES를 나눔으로써 몫과 나머지를 산출한다. 몫은 버려지고 나머지는 "CLOCKS"라는 32비트 변수에 할당된다. 위에서 기술된대로, 숫자 255는 RANDC(동일한 값을 되풀이하기 이전에 RANDC가 변하는 횟수)의 주기이다. 그 다음에 알고리즘은 단계(96)로 진행한다.

단계(96)에서, MSINIT는 CLOCKS의 거듭제곱으로 늘어나며, 그 결과는 다항식 길이 분할을 사용하여 16진수 00000163(RANDC에 대한 원시 다항식)으로 나뉘어진다. 최종 몫은 버려지고 나머지는 RANDC라 한다. 그 다음에 알고리즘은 단계(98)로 진행한다. 단계(98)에서 RANDC는 24비트 만큼 왼쪽으로 시프트되며, RANDC와 후속하는 24개의 0들로 이뤄지는 32 비트 2진수를 산출한다. 그 다음에 알고리즘은 단계(100)로 진행한다.

단계(100)에서, UPDATES(즉, UPDATES[0])의 최하위 32비트들과 16진수 00FFFFFF의 AND 결과값이 얻어진다. CLOCKS는 이 AND 결과값과 같도록 세팅된다. 이 단계는 2^{24}(제로 삽입된 RANDL의 주기는 2^{24}이다) 에 의해 나누어지는 UPDATES의 나머지를 효율적으로 계산한다. 그 다음에 알고리즘은 단계(102)로 진행한다.

단계(102)에서, 알고리즘은 LSINIT가 0인지 아닌지를 체크한다. LSINIT가 0이면, 알고리즘은 단계(104)로 진행한다. 그렇지 않으면, 알고리즘은 바로 단계(112)로 간다. 단계(104)에서 알고리즘은 CLOCKS가 0인지 아닌지를 체크한다. 만약 CLOCKS가 0이면, 알고리즘은 단계(106)로 진행한다. 그렇지 않으면, 알고리즘은 단계(108)로 진행한다. 단계(106)에서는 알고리즘은 RANDL이 0이기 때문에, RAND르 위해 RANDC의 값을 리턴시킨다. 단계(108)에서는 CLOCKS는 CLOCKS - 1로 세팅되며, 그로인해 CLOCKS 값을 1만큼 감소시킨다. 그 다음에 알고리즘은 바로 단계(110)로 간다. 단계(110)에서는 LSINIT의 최하위 비트가 1로 세팅된다. 다음으로, 알고리즘은 단계(116)로 바로 간다. 만약 현재 RAND 업데이트가 사이클의 처음에 이루어진다면, RAND의 최하위 부분(RANDL)은 0이 되기 때문에 단계(104)에서 단계(110)까지가 필요하다.

단계(112)에서, 알고리즘은 CLOCKS가 16진수 00FFFFFF와 같은지 아닌지를 체크한다. 만약 CLOCKS가 00FFFFFF와 같다면, 알고리즘은 단계(114)로 진행한다. 그렇지않으면, 알고리즘은 단계(116)로 진행한다. 단계(114)에서 알고리즘은 RANDL이 0이기 때문에 RAND를 위해 RANDC 값을 리턴시킨다. 만약 현재 RAND 업데이트가 사이클의 끝에서 이루어지면, RAND의 최하위 부분(RANDL)은 0이기 때문에 단계(112)및 단계(114)가 요구된다.

단계(116)에서는, LSINIT는 CLOCKS 거듭제곱까지 올라가고, 그 결과는 다항식 길이 분할을 사용하여 16진수 0100001B(RANDL에 대한 원시 다항식)로 나뉘어진다. 최종 몫은 버려지고 나머지는 RANDL이라 한다. 그 다음에 알고리즘은 단계(118)로 진행한다.

단계(118)에서, 알고리즘은 RANDC와 RANDL의 OR 결과값을 계산한다. 이 OR 결과값은 32비트 변수 RAND에 할당된다. 즉, RAND의 최상위 8비트들은 RANDC와 8개 0들과의 OR 결과, 즉 RANDC와 같으며, RAND의 24 최하위 비트들은 24개의 0들과 RANDL의 OR 결과, 즉 RANDL과 같다. 그 다음에 알고리즘은 단계(120)로 진행한다. 단계(120)에서 알고리즘은 다음에 업데이트되는 RAND값으로써 단계(118)의 RAND 값을 리턴시킨다.

따라서, 도 5의 실시예에 따르면, 알고리즘은 현재 시스템 시간에서의 현재 RAND 값 및 분 단위의 RAND 업데이트 주기(분)를 결정한다. 도 4의 실시예에 따르면, 알고리즘은 RAND 부분을 갈로이스 레지스터들처럼 다루며, 따라서 레지스터 컨텐츠들은 공식 파라미터 x(1은 x^0 를 나타내며, 2는 x^1 을 나타냄) 다항식들로 다루어질 수 있다. 현재 레지스터 값은 초

기상태 이후로 클럭 펄스들의 총수의 거듭제곱으로 늘어난 초기상태와 같다. 유리하게도, 도 5의 방법은 도 4의 업데이트 시퀀스를 통해 RAND를 단순히 스텝핑하는 것보다 더 빠른데 그 이유는 업데이트들의 총숫자가 RAND 값 시퀀스 길이 모듈로로 계산되기 때문이다.

특정 실시예에서, 도 5의 방법은 당업자에게 공지된 다양한 연산자들을 사용하여 실행될 수 있다. 그러한 연산자들은 예를 들어, 64비트 숫자를 32비트 숫자로 나누는 다중 정밀 분할 및 다항식을 정수 거듭제곱 모듈로 원시 다항식으로 늘리는 모듈라 지수함수를 포함한다. 모듈로 지수함수는 당업자가 이해할 수 있는 $\log_2(N)$ 단계들에서 계산될 수 있다. 그러한 연산자들은 알고리즘의 속도를 증가시키는데 기여한다.

당업자는 상기에 기술된 실시예들에서 최대 길이 시프트 레지스터들 대신에 임의의 비슷한 형태의 의사랜덤 잡음 발생기가 대용될 수 있다는 것을 쉽게 알 것이다. 게다가, 상기 기술된 실시예들은 셀룰라 전화 시스템들에 또, RANDC가 제로가 아닌 값으로 제약되는 CDMA 시스템들에 적합하지만, 특정 시스템이 이런 조건으로 지정하지 않는다면 RANDC가 제로가 아닐 필요는 없음을 주목하여야 한다. 따라서 시스템의 제약에 따라, RANDC 또는 RANDL 중 하나 또는 둘다는 시퀀스들 중 하나 또는 둘다의 길이를 연장하도록 삽입되는 올 제로 값들을 가질 수 있다. 게다가, 상기 기술된 실시예들의 셀룰라 시스템 브로드캐스트 챌린지 값은 연속적인 업데이트들 사이의 상관관계가 최소화되도록 또 반복 값이 발생하기 전의 업데이트들의 총수가 최대화되도록 주기적인 업데이트들을 요구하는 임의의 2진수일 수 있다.

본 발명의 바람직한 실시예들이 상술되었다. 그러나 본 발명의 정신이나 범위를 이탈하지 않고서 밝혀진 상기 실시예들에 다양한 변형들이 만들어질 수 있다는 것을 기술분야에서 통상의 지식을 가진 자에게 명백할 것이다. 따라서, 본 발명은 이하의 청구항들에 특별히 제한되는 것은 아니다.

도면의 간단한 설명

도 1은 셀룰라 전화 시스템의 블록선도이다.

도 2는 갈로이스(Galois) 시프트 레지스터의 블록선도이다.

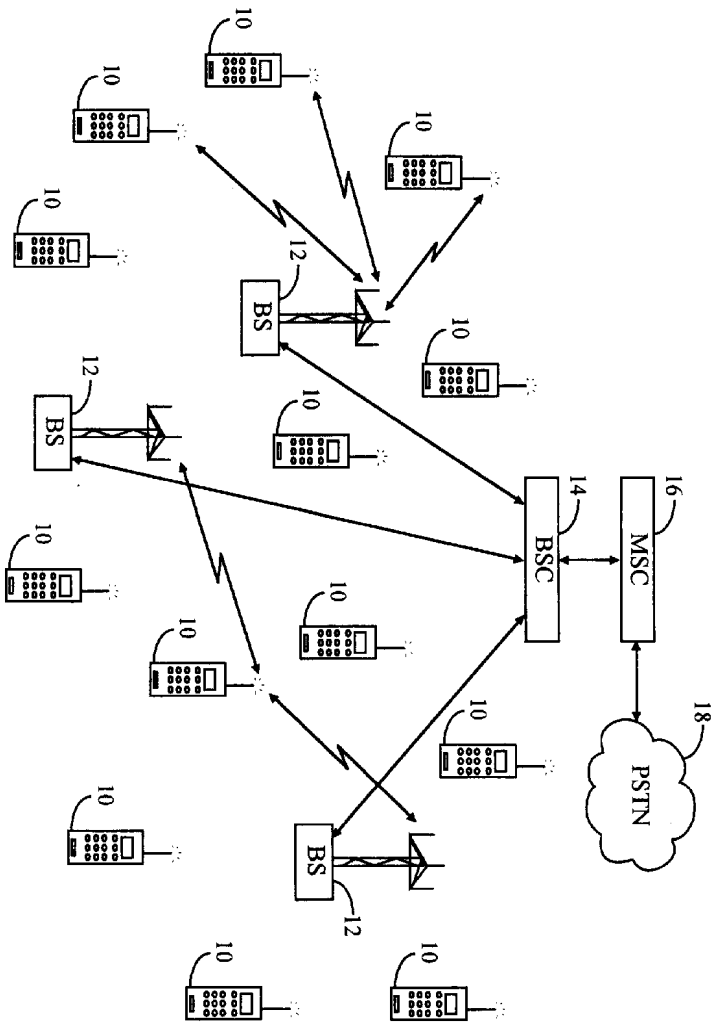
도 3은 두개의 갈로이스 시프트 레지스터들의 블록선도이다.

도 4는 RAND 업데이트 방법의 흐름도이다.

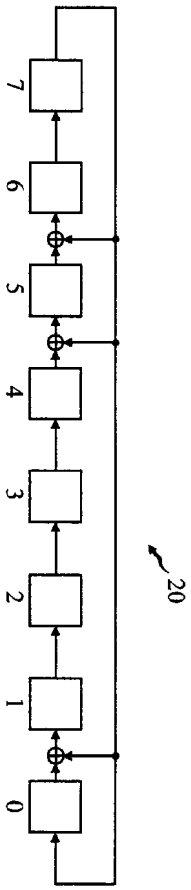
도 5는 현재 RAND 값을 계산하는 방법의 흐름도이다.

도면

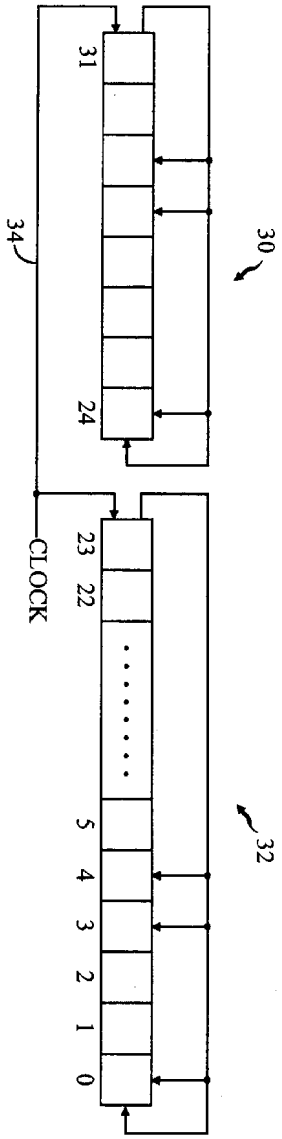
도면1



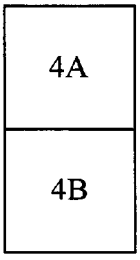
도면2



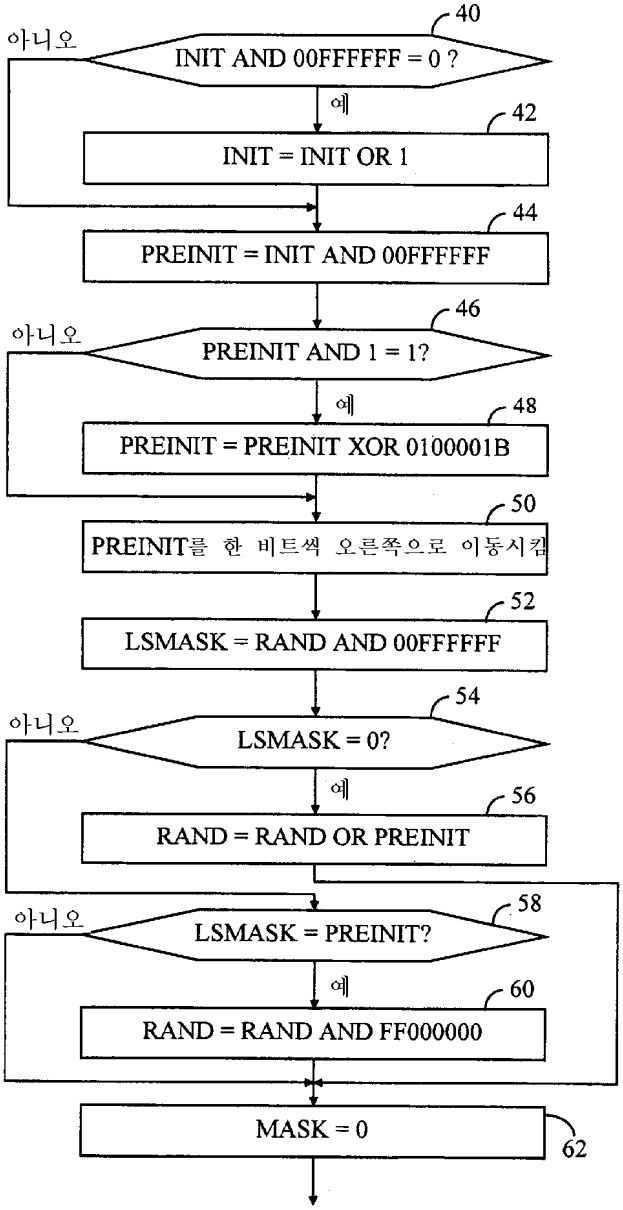
도면3



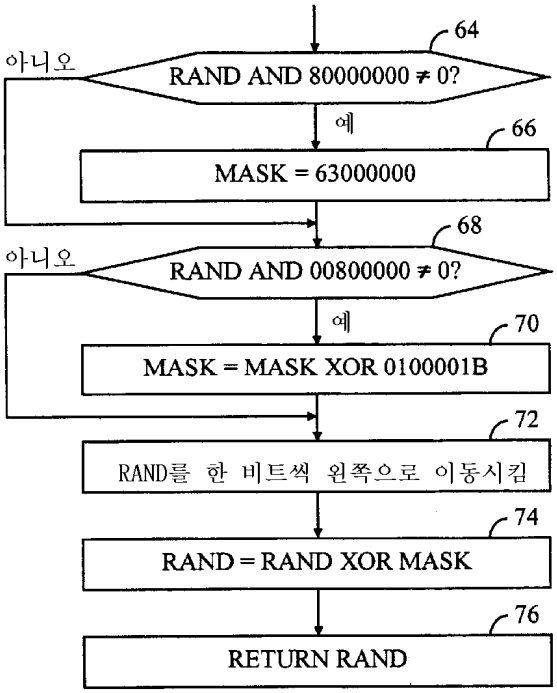
도면4a



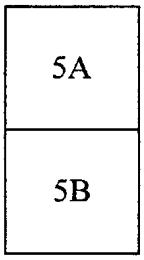
도면4b


40
아니오
INIT AND 00FFFFFF = 0 ?
예
42
INIT = INIT OR 1
44
PREINIT = INIT AND 00FFFFFF
46
아니오
PREINIT AND 1 = 1?
예
48
PREINIT = PREINIT XOR 0100001B
50
PREINIT를 한 비트씩 오른쪽으로 이동시킴
52
LSMASK = RAND AND 00FFFFFF
54
아니오
LSMASK = 0?
예
56
RAND = RAND OR PREINIT
58
아니오
LSMASK = PREINIT?
예
60
RAND = RAND AND FF000000
MASK = 0
62

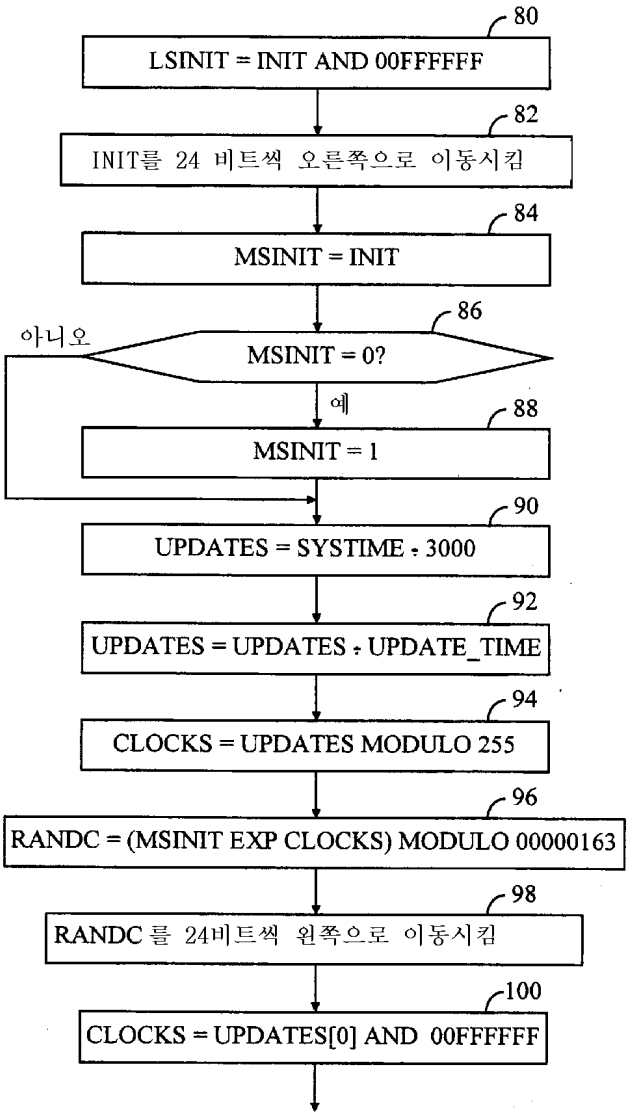
도면4c


64
아니오
RAND AND 80000000 ≠ 0?
예
66
MASK = 63000000
68
아니오
RAND AND 00800000 ≠ 0?
예
70
MASK = MASK XOR 0100001B
72
RAND를 한 비트씩 왼쪽으로 이동시킴
74
RAND = RAND XOR MASK
76
RETURN RAND

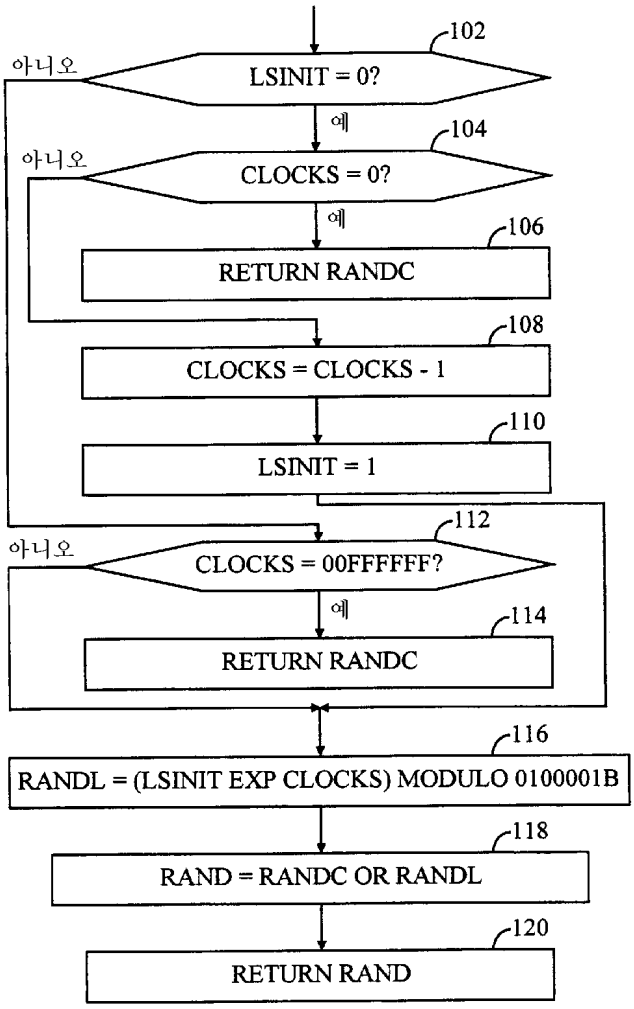
도면5a


5A
5B

도면5b


80
LSINIT = INIT AND 00FFFFFF
82
INIT를 24 비트씩 오른쪽으로 이동시킴
84
MSINIT = INIT
86
아니오
MSINIT = 0?
예
88
MSINIT = 1
90
UPDATES = SYSTIME ÷ 3000
92
UPDATES = UPDATES ÷ UPDATE_TIME
94
CLOCKS = UPDATES MODULO 255
96
RANDC = (MSINIT EXP CLOCKS) MODULO 00000163
98
RANDC 를 24비트씩 왼쪽으로 이동시킴
100
CLOCKS = UPDATES[0] AND 00FFFFFF

도면5c


102
LSINIT = 0?
아니오
예
104
CLOCKS = 0?
아니오
예
106
RETURN RANDC
108
CLOCKS = CLOCKS - 1
110
LSINIT = 1
112
CLOCKS = 00FFFFFF?
아니오
예
114
RETURN RANDC
116
RANDL = (LSINIT EXP CLOCKS) MODULO 0100001B
118
RAND = RANDC OR RANDL
120
RETURN RAND