RFID 시스템에서 패리티 메카니즘을 이용한 충돌방지 알고리즘

(An Anti Collision Algorithm using Parity Mechanism in RFID Systems)

김 성 수 † 김 용 환 † 안 광 선 ††
(Sungsoo Kim) (Yonghwan Kim) (Kwangseon Ahn)

요 약 RFID(Radio Frequency IDentification) 시스템에서 사물에 부착된 태그에 대한 식별은 리더의 요청으로 시작한다. 리더가 요청을 하면 리더의 인식영역 내에 있는 다수의 태그는 동시에 응답을 하여 충돌이 발생한다. 리더는 인식영역 내의 모든 태그들을 빠르게 인식하는 충돌방지 알고리즘이 필요하다. 본 논문에서는 패리티 메카니즘을 이용한 충돌방지 알고리즘을 제안한다. 제안한 알고리즘에서, 태그는 리더의 요청 프리픽스와 일치하는 태그들 중 프리픽스 다음의 2 비트들의 '1'의 개수가 짝수인 태그 그룹은 '0' 슬롯에 응답하고, '1'의 개수가 홀수인 태그 그룹은 '1' 슬롯에 응답하게 하여 충돌을 방지한다. 해당 슬롯에 응답에 따라 존재하는 태그 아이디만 응답할 수 있도록 요청 프리픽스를 생성한다. 또한 해당 슬롯에 충돌이 2개인 경우에는 패리티 메카니즘을 이용하여 2개의 태그를 인식한다. 즉, 전체적인 질의 횟수를 줄여 인식시간을 단축한다.

키워드 : RFID, 충돌방지 알고리즘, 슬롯, 패리티 메카니즘

Abstract In RFID systems, identifying the tag attached to the subject begins with the request from a reader. When the reader sends a request, multiple tags in the reader's interrogation zone simultaneously respond to it, resulting in collision. The reader needs the anti collision algorithm which can quickly identify all the tags in the interrogation zone. We propose the Anti Collision Algorithm using Parity Mechanism(ACPM). In ACPM, a collision can be prevented because the tags which match with the prefix of the reader's request respond as followings; the group of tags with an even number of 1's in the bits to the prefix + 2nd bits responds in slot '0', while the group of tags with an odd number of 1's responds in slot '1'. The ACPM generates the request prefix so that the only existing tags according to the response in the corresponding slot. If there are two collided bits in tags, then reader identify tags by the parity mechanism. That is, it decreases the tag identification time by reducing the overall number of requests.

Key words : RFID, Anti Collision, Slot, Parity Mechanism

1. 서 론

정보기술의 발전으로 사물이 지능화되고 네트워크화됨으로써 더욱 편리한 정보소통이 가능한 유비쿼터스 사회로 발전되고 있다. 이러한 유비쿼터스 환경은 모든 사물에 부착된 센서를 통해 정보를 습득하고 관리하는 네트워크, 즉 USN(Ubiquitous Sensor Network)을 통해 구현이 가능하다[1,2]. 그러나 USN을 구현하기 위해서는 바코드나 스마트카드 같은 기존의 인식장치 보다 많은 기능을 갖고 전송과 전력효율이 우수한 인식장치가 필요하게 되었고, 그 최적의 대안이 RFID(Radio Frequency IDentification)이다. 모든 사물에 RFID를 부착하고, 무선통신기술을 이용하여 사물의 정보 및 주

† 학생회원 : 경북대학교 컴퓨터공학과
ninny@knu.ac.kr
hypnus@knu.ac.kr
†† 종신회원 : 경북대학교 컴퓨터공학과 교수
gsahn@knu.ac.kr
논문접수 : 2009년 1월 14일
심사완료 : 2009년 7월 14일

정보과학회논문지: 정보통신 제36권 제5호(2009.10)

변 상황 정보를 감지하여 실시간으로 네트워크에 연결하여 정보를 관리하게 되며 모든 사물에 컴퓨팅과 커뮤니케이션 기능이 부여되어 언제, 어디서나, 어떤 것이든 통신이 가능한 유비쿼터스 환경의 바탕이 되고 있다[3].

RFID는 전파를 이용하여 여러 개의 태그들을 일괄적으로 읽어낼 수 있는가 하면, 거리가 떨어진 곳에서도 읽어낼 수 있는 등 현재 일반적으로 널리 사용되고 있는 바코드에서는 볼 수 없었던 여러 가지 기능과 특징을 가지고 있다. 더욱이 얇고 작아지면서 또한 저렴한 태그의 등장으로 거의 모든 제품에 부착하는 것이 가능해지고 있어 빠르게 바코드의 기능을 대체하고 있다.

RFID는 유통/물류뿐만 아니라 식품, 의료/약품, 도로/교통, 우정, 문화, 생산자동화, 소방/방재, 금융, 환경, 정보유통 등 다양한 분야에 적용이 가능하다[3,4].

RFID 시스템은 크게 리더와 태그로 구성 된다. 태그는 트랜스폰더(transponder)라고도 하며 태그의 고유한 아이디 정보를 가지고 식별하고자 하는 물품에 부착된다. 리더는 태그와 무선으로 신호를 주고받음으로써 태그로부터 필요한 정보를 수신한다. 리더와 태그와의 통신은 무선채널을 공유하기 때문에 충돌이 발생한다. 충돌은 리더 충돌과 태그 충돌로 나눌 수 있다. 리더 간 충돌은 다수의 리더가 하나의 태그에 요청 신호를 보냈을 때 발생하며, 리더 상호간에 신호의 간섭으로 태그는 잘못된 요청 신호를 받게 된다. 태그 간의 충돌은 하나의 리더에 두 개 이상의 태그들이 동시에 응답할 때 발생하며, 리더는 어떠한 태그도 식별할 수 없다. 이러한 충돌은 리더가 식별 영역 내의 태그들을 식별하는데 많은 시간이 걸리게 하고, 심지어 하나의 태그도 식별할 수 없게 한다. 따라서 최대한 충돌이 적게 발생하고, 충돌을 적절히 중재할 수 있는 충돌 방지 알고리즘이 필요하다. 현재 많은 충돌방지 알고리즘이 연구 중이다[5-10].

본 논문에서는 하나의 리더가 다수의 태그를 식별하는 다수의 태그 환경에서 더 빠른 식별을 위해 패리티 메카니즘을 이용한 충돌방지 알고리즘을 제안한다. 기존의 알고리즘은 태그들 간 충돌이 발생하지 않도록 중재하여 특정시점에 하나의 태그만이 응답하도록 하여 모든 태그를 식별한다. 하지만, 제안한 알고리즘은 기존에 연구되어진 2개의 충돌비트에 대해 2개의 태그를 예측하는 방법[10]을 개선하였다. 제안한 알고리즘은 패리티 메카니즘을 이용하여 2개의 태그를 예측할 뿐 아니라, 태그 아이디의 정보로 슬롯을 할당하여 불필요한 질의 생성을 방지한다. 이는 리더의 불필요한 요청 수를 줄여 전체 태그 인식시간을 줄인다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 기존에 선행 되어온 충돌 방지 알고리즘에 대한 연구를, 3장에서는 제안하는 빠른 태그 인식을 위한 충돌 방지 알고리즘에 대해 살펴본다. 4장에서는 제안된 알고리즘에 대한 성능평가를 하고, 마지막으로 5장에서 결론을 도출한다.

2. 관련연구

2.1 쿼리 트리 알고리즘

쿼리 트리 알고리즘(Query Tree Algorithm)은 트리 기반의 대표적인 메모리래스 알고리즘으로 태그의 응답에 따라 리더가 전송하는 질의가 결정되고 동작 방식이 간단하여 쉽게 구현할 수 있다. 쿼리 트리 알고리즘은 초기에 ϵ 이라는 질의를 태그에게 전송하며, 각 태그들은 자신의 아이디를 전송하게 된다. 인식과정이 반복될 때 리더는 k 비트 길이를 갖는 프리픽스를 태그에게 전송하며, 각 태그들은 프리픽스 매칭(prefix matching)을 위한 회로를 통해 프리픽스가 일치하는 태그 아이디를 응답한다.

쿼리 트리 알고리즘은 리더의 질의와 태그의 응답을 한 라운드(round)로 하여 라운드를 반복(iteration)하여 태그를 식별 한다[11]. 표 1은 쿼리 트리 알고리즘의 동작 과정의 예를 나타낸다.

표 1에서는 4개의 태그가 리더의 질의에 따라서 응답하는 과정을 보여준다. 응답한 태그는 자신의 아이디를 해당란에 나타내게 되며, 두 개 이상의 태그가 동시에 응답한 경우는 충돌 표시만 한다. 라운드 R1, R2, R3, R4는 충돌이 발생하여 식별할 수 없다. 라운드 R6, R7, R8, R9는 하나의 태그가 응답하여 태그를 식별할 수 있다. 쿼리 트리 알고리즘에서는 충돌이 발생하면 '0' 과 '1'을 추가한 프리픽스를 태그들에게 질의를 한다. 단계 5와 같이 응답이 없는 경우도 발생한다. 쿼리 트리는 충

표 1 쿼리 트리 알고리즘의 동작 과정 예

반복 단계	R1	R2	R3	R4	R5	R6	R7	R8	R9
리더의 요청	ε	0	1	00	01	10	11	000	001
태그의 응답	충돌	충돌	충돌	충돌	무응답	101	110	000	001
태그1(000)	000	000		000				000	
태그2(001)	001	001		001					001
태그3(101)	101		101			101			
태그4(110)	110		110				110		
큐	0 1	1 00 01	00 01 10 11	01 10 11 000 001	10 11 000 001	11 000 001	000 001	001	
메모리						101	101 110	101 110 000	101 110 000 001

돌 감지식이다. 즉 태그의 응답에서 아이디 전체 비트들 중에 하나의 비트라도 충돌이면 리더는 충돌로 인한 처리로 프리픽스를 한 비트 늘여서 다음 인식 라운드를 반복 하게 된다.

질의어는 0과 1을 계속 더하여 만들어지는데 동일한 질의어가 발생하지 않는다. 따라서 태그를 활성 상태에서 비활성 상태로 변경할 필요가 없다. 또한 충돌이 많이 발생 할수록 프리픽스의 수가 늘어나게 되어 요청-응답 반복 횟수가 늘어나게 되며, 각 프리픽스의 길이도 늘어나 전송될 비트도 늘어나게 되어 비효율적인 라운드를 반복하게 된다. 쿼리 트리 알고리즘의 효율성은 짧은 질의어에 의해 태그인식이 가능할 때 높은 성능을 보인다. 쿼리 트리 알고리즘은 큐를 이용하여 트리의 넓이우선 탐색을 행하게 되므로 질의어에 따라 전혀 응답이 없는 경우가 발생한다. 이것은 응답의 충돌 감지만 고려하여 질의어를 생성하기 때문에 발생한다. 즉 여러 개의 태그들의 동시에 응답하였을 때 충돌이 일어났는지 감지만 하고 충돌비트의 위치를 고려하지 않고 프리픽스를 생성하였기 때문이다.

2.2 충돌 비트 위치를 활용한 RFID 다중 태그 인식 알고리즘

충돌 비트 위치를 활용한 RFID 다중 태그 인식 알고리즘(QT-CBP)은 리더가 충돌 비트 위치를 알 수 있다.

QT-CBP 알고리즘[5]에서는 태그 아이디 중 한 개의 비트 충돌이 발생한 경우 쿼리 트리 알고리즘과는 달리 이를 다시 질의를 보내지 않고 충돌비트에 0, 1을 예측하여 두 개의 태그가 있는 것으로 인식한다. 이렇게 함으로써 불필요한 질의생성을 줄일 수 있게 된다.

표 2는 QT-CBP 알고리즘을 사용하여 3비트 태그들을 식별하는 과정을 보여준다. 쿼리 트리 알고리즘과의 차이점은 라운드 R5에서 쿼리 트리 알고리즘은 000, 001의 질의를 생성하는 반면, QT-CBP에서는 한 개의 충돌 비트에 0, 1을 예측하여 000, 001의 태그를 인식한다.

표 2 QT-CBP 알고리즘의 동작 과정 예

반복 단계	R1	R2	R3	R4	R5
리더의 요청	ε	1	11	10	0
태그의 응답	충돌 xxx	충돌 1xx	110	101	충돌 00x
태그1(000)	000				000
태그2(001)	001				001
태그3(101)	101	101		101	
태그4(110)	110	110	110		
스택	0 1	0 10 11	0 10	0	
메모리			110	110 101	101 110 000 001

3. 패리티 메카니즘을 이용한 충돌 방지 알고리즘(ACPM)

제안된 알고리즘에서 리더는 충돌비트의 위치와 수를 감지해야한다. 이는 맨체스터 부호화를 이용하면 해결할 수 있다. 그림 1은 맨체스터 부호화를 위한 개별 비트의 충돌 검출방법을 보여준다. 그림 1에서 태그1과 태그2의 비트 값이 0, 1로 서로 다른 값(비트1, 비트5, 비트6)이 동시에 들어오는 경우 맨체스터 부호화에서는 이를 에러로 인식을 하게 된다. 결국 개별 비트의 충돌 감지가 가능하게 된다[3].

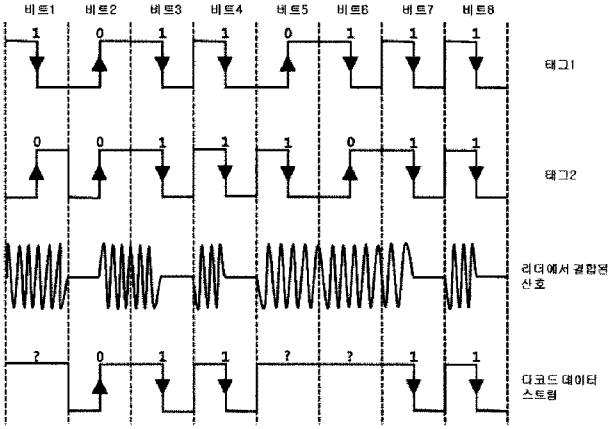


그림 1 맨체스터 부호화를 위한 개별 비트의 충돌 검출 방법

3.1 제안된 태그 아이디의 구조

표 3은 제안된 알고리즘에서 사용하는 태그 아이디의 구조를 보여준다.

표 3 태그 아이디의 구조

태그 아이디	
패리티 비트	아이디 비트
c_0	$d_1 d_2 \cdots d_k$
P_e	D

D(아이디 비트): 태그가 가진 k 비트의 고유 아이디를 $d_1 d_2 \cdots d_k$비트열로 나타낸다. k개의 아이디 비트로 가질 수 있는 서로 다른 태그의 최대 개수는 $n = 2^k$ 개이다. 즉, 아이디 비트 k = 8일 경우, 서로 다른 태그들의 최대 개수는 $n = 2^8 = 256$개이며, 각 태그들의 아이디 비트의 값은 8자리 이진수 값 00000000~11111111을 가지게 된다.

P_e(패리티 비트): c_0를 의미하며 패리티 비트와 같다. 아이디 비트의 '1' 의 개수를 2 로 나눈 나머지 값인 $c_0 = (\sum_{i=1}^{k} d_i) \% 2$는 '0' 또는 '1'의 값을 가지므로 ($c_0 \in \{0 \text{ or } 1\}$), 아이디 비트의 '1'의 개수가 홀수인지 짝수인지 판단하는데 사용된다. $c_0 = 0$이면 아이디 비트의 '1'의 개수가 짝수이고, $c_0 = 1$ 이면 홀수이다.

3.2 리더가 관리하는 인식정보

제안한 알고리즘은 태그 아이디를 인식하기 위해서 태그들의 응답 신호를 분석한다. 리더는 응답 신호에 대하여 다음의 4 가지 인식 정보를 유지한다.

- N_p = 패리티 비트(P_e)
- N_1 = 아이디 비트에서 확인된 '1' 의 수
- N_c = 아이디 비트에서 충돌된 비트의 개수
- N_{1r} = N_c에 포함되어야 하는 '1' 의 개수

리더는 위의 4 가지 인식 정보를 가지고 태그를 예측한다. N_c가 2인 경우는 N_p를 이용하여 충돌된 두 개의 태그를 인식한다. 리더는 이러한 정보를 유지하여 태그를 인식하게 된다.

3.3 충돌비트 패턴에 대한 분석

리더의 질의로 부터 응답한 태그들의 비트패턴에 대한 예측을 위해 다음의 변수를 정의한다.

X: 다수의 태그로부터 수신된 응답에서 충돌비트만 모은 비트열 $X_1X_2...X_i$이다. 각 X는 0과 1이 동시에 수신된 미결정 값을 가진다. $X_1, X_2, ..., X_i \in \{0 \text{ and } 1\}$이며 $1 \le i \le k$이다. i의 값에 따라 충돌비트의 수도 달라지며, 아이디 비트가 k개일 경우 최대 충돌비트 수는 k개이다. 태그가 응답하는 충돌비트의 수를 N_c라고 하면 충돌비트열에서 비트수 i는 N_c와 같다. 즉 $N_c = i$이다. $N_c = 0$은 응답에서 충돌이 없는 경우이며 X_i가 존재하지 않는다.

B: 다수의 태그로부터 수신된 응답에서 충돌이 없는 비트만 모은 비트열 $b_{p1}b_{p2}...b_{ph}$이다. 여기서 h는 k의 크기보다 작거나 같으며($1 \le h \le k$), 각 b_{ph} 비트는 0 또는 1 중에 하나의 값만 가진다. p=0일 때, $b_{0h} \in \{0\}$이고 p=1이면, $b_{1h} \in \{1\}$인 고정된 값을 가진다. 아이디 비트가 k개일 경우 수신응답은 충돌이 없는 비트수 h와 충돌비트 수 i의 합으로 표현된다. $k = h + i$이다. $k = h$일 경우 모든 비트가 충돌 없이 인식된 경우이다. B의 비트열 중에 $b_h = 1$에 속하는 $b_{1h} \in \{1\}$의 수를 N_1이라고 하고, $b_h = 0$에 속하는 $b_{0h} \in \{0\}$의 수를 N_0이라고 하면, k=h 일 때, 비트 수 $h = N_1 + N_0 = k$가 된다. N_0는 $h - N_1$으로 구해지므로 리더가 관리하는 정보에서 제외된다. 충돌비트 X_i가 있는 경우($1 \le h \le k$)에는 $k = h + X_i$가 되며, X_i가 N_c라고 할 때 응답비트 수는 $k = N_1 + N_0 + N_c$가 된다.

정리. 패리티 N_p 값을 알고 있을 때, 태그의 응답에서 충돌비트의 개수 $N_c = 2$인 경우 식 (1)을 만족하는 2 개의 태그를 동시에 인식할 수 있다.

$$N_{1r} = (N_1 + N_p) \% 2, \quad where \quad N_{1r} \in \{0 \text{ or } 1\} \qquad (1)$$

증명. 충돌비트의 개수 $N_c = i = 2$이므로 태그의 응답에 충돌비트 X_1X_2가 가지는 패턴의 모든 조합은 2^2개이며 $X_1X_2 \in \{00 \text{ or } 01 \text{ or } 10 \text{ or } 11\}$ 중에 하나의 비트패턴을 가진다(표 4 참고). 그런데 X_1X_2가 미결정의 값 $X_i \in \{0 \text{ and } 1\}$이 되기 위해서는 태그$a_0(00)$와 태그$a_3(11)$가 동시에 응답한 경우이거나, 태그$a_1(01)$와 태그$a_2(10)$가 동시에 응답한 경우뿐이다. 여기서 $P_e = N_p = 0$일 경우 태그$a_0(00)$와 태그$a_3(11)$가 인식되고 $P_e = N_p = 1$일 경우, 태그$a_1(01)$와 태그$a_2(10)$가 동시에 인식할 수 있다. 만약에 태그$a_0(00)$와 태그$a_1(01)$가 동시에 응답한 경우에는 $X_1X_2 = 0X$가 되므로 X_1은 충돌 없이 인식된 b_h와 같으며 $X_1 = b_{0h} \in \{0\}$ 이므로, 조건 $X_i \in \{0 \text{ and } 1\}$에 모순이 된다. 태그$a_2(10)$와 태그$a_3(11)$가 동시에 응답한 경우에도 $X_1X_2 = 1X$가 되므로, X_1은 충돌 없이 인식된 b_h와 같으며 $X_1 = b_{1h} \in \{1\}$이므로, 조건 $X_i \in \{0 \text{ and } 1\}$에 모순이 된다.

표 4 태그들의 충돌비트 패턴과 인식

태그 종류	패리티 비트	충돌비트 패턴
	$P_e = N_p$	X_1X_2
태그a_0	0	0 0
태그a_1	1	0 1
태그a_2	1	1 0
태그a_3	0	1 1

3.4 적합한 태그 그룹 분리 및 질의 생성

1) 적합한 태그 그룹 분리

- 리더의 요청 프리픽스와 일치하는 태그들 중 프리픽스 다음의 2 비트들의 '1'의 개수가 짝수인 태그 그룹은 '0' 슬롯에 프리픽스 다음 비트부터 마지막 비트까지 응답
- 요청 프리픽스와 일치하는 태그들 중 프리픽스 다음의 2 비트들의 '1'의 개수가 홀수인 태그 그룹은 '1' 슬롯에 프리픽스 다음 비트부터 마지막 비트까지 응답
- 요청 프리픽스의 길이가 태그 아이디의 길이보다 1이

작다면, 마지막 한 비트의 값에 따라 0이면 '0' 슬롯에 1이면 '1' 슬롯에 응답

2) 질의 생성

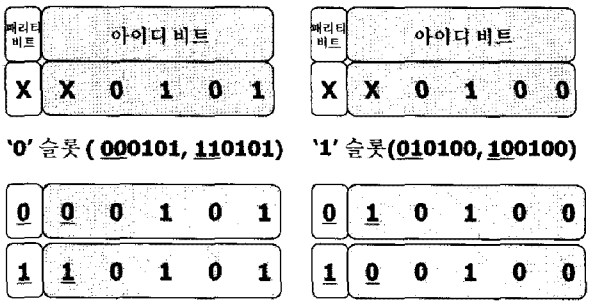
- 해당 슬롯에 프리픽스 다음의 2 비트가 충돌되었다면 응답 슬롯에 따라 '00', '11' 또는 '01', '10'의 질의를 생성한다.
- 해당 슬롯에 프리픽스 다음의 3번째 비트 이후에서 충돌이 발생되었다면 해당 비트 충돌 이전 비트까지 질의를 생성한다.

3.5 태그 예측 방법

1) 해당 슬롯에 프리픽스 다음의 2번째 비트까지만 충돌이 발생하고 프리픽스 다음의 3번째 비트부터 일치할 경우

- 해당 슬롯에 한 개의 태그가 응답하였다면 인식(한 개 태그 인식)
- 해당 슬롯에 프리픽스 다음의 2번째 비트까지만 충돌이 발생했다면 응답 슬롯에 따라 해당 충돌비트에 '00', '11' 또는 '01', '10'인 2개의 태그를 예측

그림 2는 해당 슬롯에 처음 두 비트만 충돌인 경우에 태그 예측을 보여준다. 그림 2(a)는 '0' 슬롯에 응답한 경우에는 리더의 요청 프리픽스와 일치하는 태그 중 프리픽스 다음 두 비트가 00이거나 11인 태그가 응답한 경우로써 처음 두 비트만 충돌이 발생하였다면, 00과 11을 예측하여 000101과 110101의 두 개의 태그를 인식한다. 그림 2(b)의 경우는 '1' 슬롯에 응답한 경우로써 01, 10을 예측하여 010100, 100100의 두 개의 태그를 인식한다.



(a) '0' 슬롯에 응답한 경우 (b) '1' 슬롯에 응답한 경우

그림 2 해당 슬롯에 따른 태그 예측

2) 해당 슬롯에 프리픽스 다음의 2번째 비트 이후의 2개 충돌에 따른 태그 예측

- $N_c=0$: 모든 비트들의 충돌이 없는 경우로서, 리더의 요청에 하나의 태그가 응답한 경우이므로 리더는 하나의 태그를 인식할 수 있다.
- $N_c=2$: 충돌된 비트의 수가 2 개인 경우로서, 식 (1)을 이용하여 태그를 예측한다.

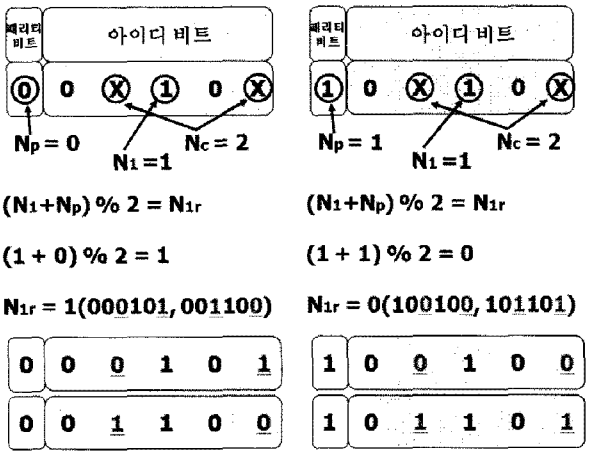
N_{1r}이 0인 경우

- X_1X_2에 '0'과 '0'인 태그 예측
- X_1X_2에 '1'과 '1'인 태그 예측

N_{1r}이 1인 경우

- X_1X_2에 '0'과 '1'인 태그 예측
- X_1X_2에 '1'과 '0'인 태그 예측

그림 3은 패리티 비트를 이용하여 아이디 비트의 충돌비트의 값을 구하는 방법을 나타낸다. 그림 3(a)는 '0' 슬롯에 응답하며, 패리티 비트가 0인 경우이다. 이 경우는 아이디 비트의 '1'의 개수가 짝수 개인 경우이다. $N_p=0, N_1=1, N_c=2$인 경우로써, 식 (1)을 이용하여 N_{1r}의 값을 구하면 1이 된다. N_{1r}이 1인 경우는 2 개의 충돌비트(X_1X_2)에 '0'과 '1'인 태그(000101)와 2 개의 충돌비트(X_1X_2)에 '1'과 '0'인 태그(001100)을 예측할 수 있다. 그림 3(b)는 '1' 슬롯에 응답하며, 최상위 비트인 패리티 비트가 1 인 경우이며, 이 경우는 아이디 비트의 '1'의 개수가 홀수 개인 경우이다. $N_p=1, N_1=1, N_c=2$인 경우로써, 식 (1)을 이용하여 N_{1r}의 값을 구하면 0이 된다. 아이디 비트에 남아있는 '1'의 개수가 0인 경우는 2 개의 충돌비트(X_1X_2)에 '0'과 '0'인 태그(100100)와 2 개의 충돌비트(X_1X_2)에 '1'과 '1'인 태그(101101)을 예측할 수 있다. 이처럼 한 비트의 패리티 비트를 이용하여 2 개의 태그를 예측할 수 있다.



(a) N_p=0인 경우 (b) N_p=1인 경우

그림 3 N_p값에 따른 태그 예측

3.6 인식 예

그림 4는 표 5에 있는 5개의 태그를 인식하기 위한 제안된 알고리즘의 진행순서를 나타낸다.

첫 번째 반복에서, 리더는 엡실론(ε)을 인수로 태그들에게 요청을 보낸다. 엡실론은 모든 프리픽스와 일치하

표 5 사용된 태그들의 태그 아이디

태그 종류	태그 아이디	
	패리티 비트(C)	아이디 비트(D)
태그1	1	11100
태그2	0	11000
태그3	1	01000
태그4	1	10101
태그5	1	11111

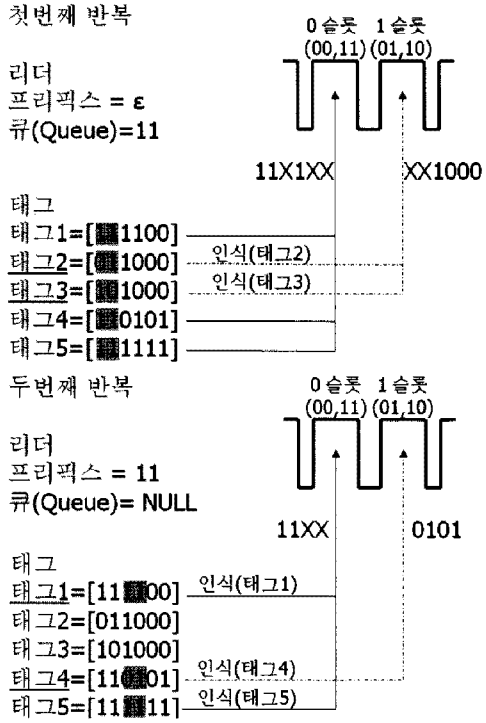


그림 4 리더의 요청과 태그의 응답

는 것으로 간주하여 모든 태그는 프리픽스 다음의 2 비트들의 '1'의 개수에 따라 해당 슬롯에 응답하게 된다. 프리픽스 다음의 2 비트들의 '1'의 개수가 짝수인 태그 그룹은 '0' 슬롯에 응답, '1'의 개수가 홀수인 태그 그룹은 '1' 슬롯에 응답한다. 이때 태그2와 태그3은 '1' 슬롯에 응답하게 되고 처음 두 비트만 충돌이 발생한 상태이다. 이 경우는 '1' 슬롯에 해당하는 경우이기 때문에 충돌비트에 0,1과 1,0을 예측하여 011000, 101000 인 두 개의 태그를 인식한다. 태그1, 태그2와 태그5는 '1' 슬롯에 응답하며, 예측이 불가능한 상태이다. 이 경우는 인식이 된 '11'을 질의로 생성한다. 두 번째 반복에서, 리더의 요청 프리픽스 값 '11'을 전송하게 되면 태그들 중 프리픽스가 일치하는 태그1, 태그4, 태그5가 프리픽스 다음의 2 비트들의 값에 따라 해당 슬롯에 응답한다. 이 경우는 해당 슬롯에 프리픽스 다음의 2번째 비트 이후의 2개 충돌에 따른 태그 예측 가능한 경우이다. 리더는 '0' 슬롯에 $11X_1X_2$의 응답에 대해 식 (1)을 이용하여 N_{1r}의 값을 구한다. N_{1r}의 값은 0이다. 0인 경우는 충돌비트 X_1X_2에 0,0과 1,1을 예측하여 111100, 111111을 인식한다.

제안한 알고리즘을 사용하면 예측을 통해 다수의 태그를 인식하는데 전체적인 인식시간을 줄일 수 있다.

4. 실 험

본 논문에서 성능평가를 위한 태그는 EPC™ Tag Data Standards Version 1.4를 따르는 태그에서 시리얼 넘버에 해당되는 부분만을 사용하였다[12]. 표 6은 General Identifier(GID-96)의 헤더 필터를 보여준다.

표 6 General Identifier(GID-96)의 헤더 필터

	Header	General Manager Number	Object Class	Serial Number
GID-96	8	28	24	36
	00110101 이진값	268,435,455 최대10진값	16,777,215 최대10진값	68,719,476,735 최대10진값

Header, General Manager Number, Object Class의 경우는 유사성이 많은 비트들로 구성되어 있기 때문에 본 실험에서는 실제 각 상품에 대한 고유 식별번호를 나타내는 시리얼 넘버(Serial Number)만을 생성하여 실험을 하였다. 본 논문에서 실험을 하기 위한 시뮬레이션 프로그램은 C# 으로 작성하였으며, 제안된 알고리즘이 한 비트의 여분의 비트를 사용하기 때문에 다른 알고리즘보다 한 비트 길어진 아이디를 가지고 실험하였다. 실험에서 사용한 태그 아이디 할당 방법은 랜덤 할당(random assignment) 방법을 사용하였다. 랜덤 할당 방법은 태그의 아이디를 랜덤 값을 생성하여 할당하는 방법이다. 랜덤 할당 방법을 사용하면 태그 아이디가 가질 수 있는 전체 범위에 아이디가 고르게 분포된다. 이는 태그들의 아이디 간의 비트 값의 유사성이 적다는 것을 의미한다.

랜덤 할당 방법에서 태그 아이디가 가지는 값의 범위는 균일분포(uniform distribution)를 갖는 난수발생 함수를 사용하였다. 본 논문에서는 태그들이 가지는 패턴의 유사성이 없는 상태를 표현하기 위해 시간변수를 종자(seed)값으로 하여 아이디 비트 크기의 균일분포를 가지는 랜덤함수를 사용한다. 랜덤함수로 발생한 비트패턴은 36비트일 경우 0에서 2^{36}-1의 값을 가진다.

그림 5는 랜덤 할당 방법으로 36비트의 태그를 가지고 태그의 개수를 증가시키면서 리더와 태그간의 요청-응답 수를 비교하였다. 36 비트 태그의 경우는 2^{36}개의 태그 아이디를 가질 수 있지만 실험에서는 10^1개에서 10^5개까지의 태그를 리더영역에 동시에 있다고 간주하고

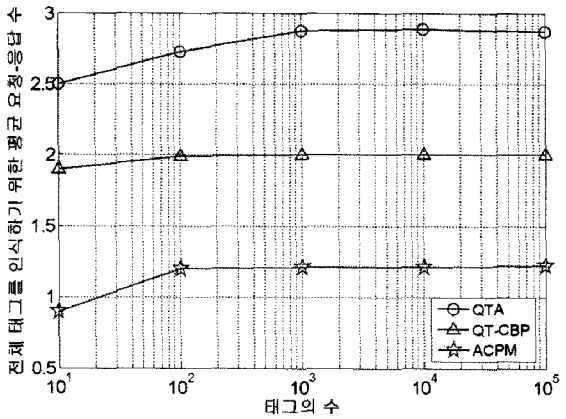


그림 5 태그수별 요청-응답 수

표 7 요청-응답 수 비교

알고리즘	요청-응답 수의 비례상수(a=y/x)					횟수 비교
	10^1	10^2	10^3	10^4	10^5	
QTA	2.50	2.73	2.88	2.89	2.87	2.35
QT-CBP	1.90	1.99	1.99	2.00	2.00	1.64
ACPM	0.90	1.20	1.21	1.21	1.22	1.00

인식실험을 진행하였다. 태그 개수가 증가 할수록 요청-응답의 수도 비례하여 증가되었다. x축을 태그의 수, y축을 요청-응답 수라고 할 때, 비례상수 $a=\Delta y/\Delta x$를 조사하면, 제안된 알고리즘(ACPM)은 증가되는 비례상수가 a=0.90(태그 수 10^1개)에서 a=1.22(태그 수 10^5개)까지 증가하고 있으나 쿼리 트리 알고리즘(QTA)는 a=2.50(태그 수 10^1개)에서 a=2.87(태그 수 10^5개)로 제안된 알고리즘보다 큰 값에서 증가되고 있다.

제안된 알고리즘은 36비트 랜덤 할당에서 태그 수가 10^5개의 경우 표 7에 나타낸 바와 같이 QTA, QT-CBP에 비해 각각 2.35배와 1.64배 이상의 질의 반복 횟수를 줄였다.

그림 6은 랜덤 할당 방법을 사용한 36 비트의 태그 아이디를 가지고 리더의 요청에 대한 전송비트 수와 태그의 응답에 대한 전송비트 수를 보여준다. 그림 7은 리더의 요청과 태그의 응답에 대한 전체 전송비트 수를 보여준다.

실험을 통해서, 모든 경우에서 제안된 알고리즘이 다른 알고리즘에 비해 전송비트 수가 줄었음을 볼 수 있다.

각 알고리즘에 대한 전송비트 수를 비교하여 표 8에 나타내었다. 각 필드는 리더의 요청비트수와 태그의 응답비트수 및 리더와 태그의 요청-응답의 총 전송비트 수를 비례상수 a=y/x와 함께 ACPM을 1로 하였을 때 상대적인 전송량을 나타내었다. 제안된 알고리즘인 ACPM은 리더의 요청비트수의 상대적인 전송량에서 QTA에 비해 2.35배, QT-CBP에 비해 1.59배 적다는 것을 알수 있다. ACPM은 리더의 요청 횟수가 줄어들 뿐만 아니라 프리픽스만 전송하므로 요청 전송비트 수를 줄일

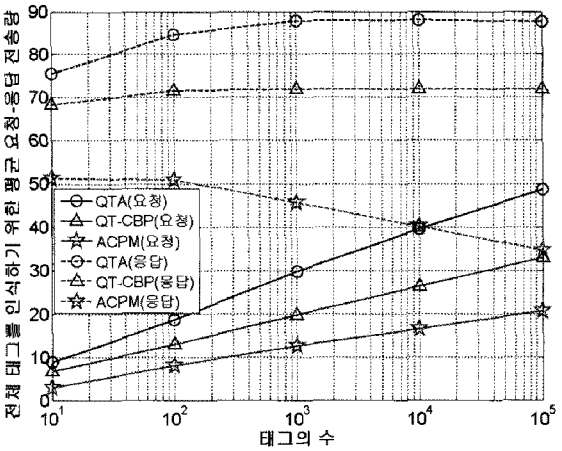


그림 6 태그수별 요청과 응답에 대한 전송비트 수

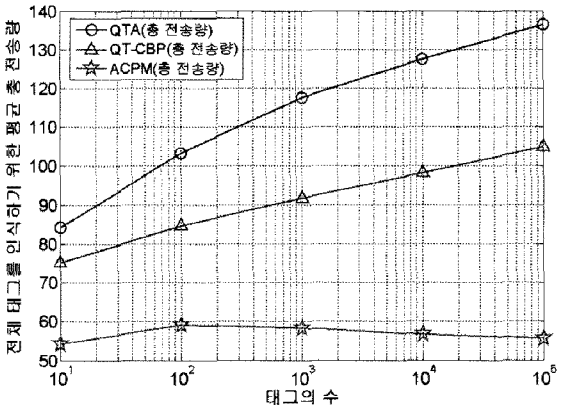


그림 7 태그수별 전체 전송비트 수

표 8 전송량 비교

알고리즘	요청		응답	
	a=y/x	전송량	a=y/x	전송량
QTA	48.76	2.35	87.69	2.52
QT-CBP	32.99	1.59	71.99	2.07
ACPM	20.74	1.00	34.76	1.00

수 있다. 그리고 태그의 응답의 전송량은 QTA에 비해 2.52배, QT-CBP에 비해 2.07배 적으며, 응답은 태그 아이디의 모든 비트를 수신하기보다는 상대적으로 태그로부터 프리픽스 다음비트부터 아이디의 끝까지만 부분 전송되기 때문에 전송량을 줄일 수 있다. 따라서 패리티 비트로 인한 예측으로 리더의 요청 횟수를 줄이는 것은 물론이고, 태그의 응답에서도 프리픽스가 일치하는 아이디의 나머지 비트만 수신함으로서 요청-응답에 대한 전체 전송비트 수를 줄이고 있다.

5. 결 론

RFID 시스템에서 리더의 인식영역 내에 있는 다수의 태그를 식별하는 과정에서 다수의 태그가 응답을 하여

충돌이 발생한다. 리더는 인식영역 내의 모든 태그들을 빠르게 인식하는 충돌방지 알고리즘이 필요하다. 우리는 패리티 메카니즘을 이용한 충돌방지 알고리즘(ACPM)을 제안한다. 제안된 알고리즘에서 적절한 슬롯 할당을 통해 태그 그룹을 효율적으로 나누어 응답하도록 하여 충돌을 방지한다. 또한 태그 예측을 통해 전체적인 질의 횟수를 줄여 태그 인식시간을 단축한다. 실험 결과에서 알 수 있듯이, ACPM은 쿼리 트리 알고리즘(QTA)보다는 약 2.35배의 질의 반복 횟수를 감소와 2.52배의 전송 비트 수를 줄였고, 충돌 비트 위치를 활용한 RFID 다중 태그 인식 알고리즘(QT-CBP)보다는 약 1.64배의 질의 반복횟수를 감소와 약 1.89배의 전송 비트 수를 줄였다.

참 고 문 헌

[1] D. W. Jang, P. D. Cho, "Analysis of Technical Regulations for Radio Frequency Identification," *전자통신동향분석*, vol.18, no.6, Dec., 2003. (in Korea)

[2] H. C. Kim, C. P. Hong, "RFID/USN 기술 분석 및 전망," *Journal of KICS*, vol.21, no.6, pp.665-678, Jun. 2004. (in Korea)

[3] K. Finkenzeller, *RFID Hand Book: Fundamentals and Applications in Contactless Smart Card and Identification*, Second Edition, John Wiley & Sons Ltd, 2003.

[4] P. H. Cole, "Fundamentals in RFID part1," Korean RFID course, 2006, Available: http://autoidlab.eleceng-.adelaide.edu.au/education/FundamentalsInRfidPart1.pdf.

[5] H. Lee, J. Kim, "QT-CBP: A New RFID Tag Anti-collision Algorithm Using Collision Bit Positioning," *Emerging Directions in Embedded and Ubiquitous Computing(EUC), LNCS*, Springer vol.4097, pp.591-600, August, 2006.

[6] S. Kim and P. Park, "An efficient tree-based Tag Anti-collision protocol for RFID systems," *IEEE Commun. Lett.*, vol.11, pp.449-451, May. 2007.

[7] T. Wang, "Enhanced binary search with cut-through operation for anticollision in RFID systems," *IEEE Commun. Lett.*, vol.10, no.4, pp. 236-238, Apr. 2006.

[8] D. Shih, P.L. Sun, D.C.Yen and S. M.Huang, "Taxonomy and Survey of RFID Anti-collision protocols," *Computer and communicatios*, vol.29, pp.2150-2166, 2006.

[9] J. Eom, T. Lee, R. Rietman, and A. Yener, "An efficient framed-slotted ALOHA algorithm with pilot frame and binary selection for anti-collision of RFID tags," *IEEE Commun. Lett.*, vol.12, no.11, pp.861-863, Nov. 2008.

[10] S. Kim, Y. Kim, S. Lee, and K Ahn, "An Improved Anti Collision Algorithm using Parity Bit in RFID System," *Seventh IEEE International Symposium on Network Computing and Applications(NCA)*, pp.224-227, 2008.

[11] C. Law, K. Lee, and K. Y. Siu, "Efficient Memoryless protocol for Tag Identification," In *Proceedings of the 4th international workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM'00)*, ACM, pp.75-84, 2000.

[12] EPCglobal, EPCglobal Tag Data Standards Version 1.4, [Online]. Available: http://www.epcglobalinc.org /standards/tds/tds_1_4-standard-20080611.pdf

김 성 수

2002년 금오공과대학교 컴퓨터공학과(학사). 2005년 경북대학교 대학원 컴퓨터공학과(석사). 2006년~현재 경북대학교 대학원 컴퓨터공학과 박사과정. 관심분야는 RFID, 임베디드 시스템, 센서 네트워크

김 용 환

2004년 경운대학교 컴퓨터공학과(학사) 2006년 경북대학교 대학원 컴퓨터공학과(석사). 2006년~현재 경북대학교 대학원 컴퓨터공학과 박사과정. 관심분야는 RFID, 임베디드 시스템, 센서 네트워크

안 광 선

1972년 연세대학교 전기공학과(학사). 1975년 연세대학교 대학원 전자공학과(석사) 1980년 연세대학교 대학원 전자공학과(박사). 1975년~1976년 스페리유니백 근무 1977년~현재 경북대학교 공과대학 컴퓨터공학과 교수. 관심분야는 RFID, 임베디드 시스템, 센서 네트워크 등