韓國IT서비스學會誌
第12卷 第3號
2013年 9月, pp.331-343
http://dx.doi.org/10.9716/KITS.2013.12.3.331

RFID 시스템에서 다중 태그 인식 개선을 위한 충돌방지 알고리즘

김용환* · 류명춘** · 박준호***

Anti-Collision Algorithm for Improvement of Multiple Tag Identification in RFID System

Yong-Hwan Kim* · Myung-Chun Ryoo** · Joon-Ho Park***

■ Abstract ■

In RFID systems, the anti-collision algorithm is being improved to recognize Tag's ID within recognition area of the reader quickly and efficiently. This paper focuses on Tag collision. Many studies have been carried out to resolve Tag collision. This paper proposes a new N-ary Query Tree Algorithm to resolve more than Tag collision simultaneously, according to the value of m(2 ~ 6). This algorithm can identify more tags than existing methods by treating a maximum 6 bit collision, regardless of the continuation/non-continuation Tag's ID patterns. So, it extracts maximumly different 2^6 bit patterns per single prefix in recognition process. The performance of N-ary Query Tree Algorithm is evaluated by theoretical analysis and simulation program.

Keyword : RFID, Anti Collision, Query Tree Algorithm

논문투고일 : 2013년 07월 26일 논문수정완료일 : 2013년 09월 05일 논문게재확정일 : 2013년 09월 10일

* 경운대학교 컴퓨터공학과 연구교수, 제1저자
** 경운대학교 컴퓨터공학과 교수, 제2저자
*** 경운대학교 컴퓨터공학과 교수, 교신저자

1. 서 론

RFID(Radio-Frequency Identification)은 사물에 대한 접촉 없이 리더를 통해 태그 정보를 획득하거나 인식하는 기술이다. 이 시스템은 리더와 태그로 구성되고, 리더는 다양한 종류의 물품이나 동·식물에 부착된 태그 ID를 읽어 들여, 이에 대한 정보를 추적하여, 사용자에게 다양한 정보를 제공한다. 따라서 RFID 기술은 여러 분야에서 사용될 수 있다[7].

기존의 바코드, 교통카드, 무선결제시스템과 같은 접촉식/비접촉식 방식은 일대일 통신환경에서 유용하게 사용되고 있다. 그러나 이 방식들은 동시에 다수 개의 물품 정보를 식별하지 못하기 때문에 효율적이지 못하다[1, 3].

RFID 기술이 실 응용 분야에서 성공적인 도입을 위해서는 신뢰성 있는 태그의 식별이 보장되어야 한다. 특히 실시간으로 대량의 물품을 동시에 식별해야 하는 대규모 전자물류시스템에서 하나의 리더 식별 영역 내에 다수 개의 태그가 존재할 경우, 이를 적절하고 빠르게 인식할 수 있어야 한다. 따라서 RFID 충돌 방지 기술에 대한 연구는 태그 인식 및 인식 속도와 데이터 전송률의 특성 및 정확도를 향상시키도록 요구되고 있다[1, 3].

RFID의 충돌 문제는 리더가 태그 ID를 인식할 수 없는 경우를 나타낸다. 이 경우는 실 환경에서 다양한 상황이 만들어지고, 많은 연구들이 이루어지고 있다. 모든 상황을 고려할 수는 없지만, RFID 충돌 문제는 크게 리더 충돌 문제와 태그 충돌 문제로 나뉠수 있다. 리더 충돌은 다수의 리더에 의한 전파 간섭으로 인해 발생하고, 태그 충돌은 하나의 리더 인식 영역 내에 다수의 태그가 존재할 때 발생한다. 이런 충돌로 인해 리더는 태그를 인식하는데 많은 시간이 소요되며, 최악의 경우 태그를 인식하지 못하는 경우가 있다. 따라서 리더에서 태그를 인식하기 위한 충돌방지(anti-collision) 방법이 필요하다.

본 논문은 태그 충돌 문제에 집중한다. 태그 충돌의 RFID 충돌 방지 알고리즘은 크게 3분류로 나뉜다. 트리 기반의 결정적(deterministic) 알고리즘, 슬롯 알로하 기반의 확률적(probability) 알고리즘, 그리고 두 기반 알고리즘의 단점들을 보완하기 위한 하이브리드 쿼리 트리 기반의 알고리즘이 있다.

트리기반의 주요 알고리즘은 쿼리 트리 알고리즘, 4-ary 쿼리 트리 알고리즘, 개선된 쿼리 트리 프로토콜 등이 있다[7, 10, 11, 13]. 슬롯 알로하 기반의 알고리즘은 프레임 슬롯알로하, 동적 프레임 슬롯 알로하 등이 있다[4, 5, 7, 16]. 하이브리드 쿼리 트리 기반의 알고리즘은 하이브리드 쿼리 트리 프로토콜, 효율적인 슬롯 할당을 통한 추측 알고리즘 등이 있다[1, 2].

본 논문은 트리 기반의 알고리즘들의 동작 방식과 특성 분석을 통해, 각 알고리즘들의 장점을 적용하고, 도출된 단점을 보완하여, 최소의 질의로 보다 많은 태그를 인식할 수 있는 N-ary 쿼리 트리 알고리즘을 제안한다.

본 논문의 연구는 다음과 같다. 첫째, 트리 기반 결정적인 방식을 채택한다. 이 기반의 최대 단점인 태그 ID의 동시 응답 특성을 활용하고, 슬롯 알로하와 하이브리드 쿼리 기반의 제약사항인 무응답 또는 무한 충돌 타임 슬롯에 의한 지연시간을 해결한다. 둘째, Manchester code를 적용하여 충돌 비트 위치와 충돌 비트 수를 추출한다. 셋째, 충돌 비트와 충돌 비트 수를 활용하여, 리더가 태그 인식과정에서, 한 번의 질의에 더 많은 정보를 얻을 수 있도록, 이전 연구[10]보다 향상된 확장된 비트 변형 방식을 고안한다. 이로 인해 최대 2^6개의 서로 다른 태그 ID를 인식할 수 있다. 넷째, 제안한 알고리즘의 이론적 정리와 수학적 접근법으로 타당성을 검증하고 있으며, 실험을 통해 알고리즘의 논리성을 증명한다. 다섯째, 본 논문에서 제안한 N-ary 쿼리 트리 알고리즘과 기존의 발표된 알고리즘들과 비교 분석한다. 성능 분석은 태그 수에 따른 리더의 질의 수, 질의-응답 수를 도출하여, 태그 인식속도를 비교 검증한다.

2. 관련 연구

2.1 트리 기반에서의 태그 ID 인식

트리 기반의 알고리즘들은 계층적인 트리 구조를 구성한다. 트리의 단말 노드는 태그 ID에 대응되며, 노드는 태그 ID의 일부 비트를 나타낸다. 태그 ID를 인식하는 과정에서, 비트 충돌이 발생할 경우, 트리의 차수(degree)는 늘어난다. 최대 차수는 트리의 항(N-ary)에 따라 결정된다. 또한 트리의 루트('£')에서 단말 노드까지의 경로를 탐색하면, 각 태그 ID를 인식할 수 있다. 이때 트리의 깊이(높이)는 질의-응답 수를 산출할 때 이용된다. 트리의 노드는 3가지 상태를 가지며, 상태는 다음과 같다.

- 무응답 상태(Idle state, S_I) : 리더의 프리픽스(태그 ID의 일부분) 질의에 대해, 태그 ID가 일치하지 않는 경우, 태그는 응답을 하지 않는다. 따라서 S_I의 증가는 불필요한 질의가 증가되는 것으로 태그 인식 시간의 지연을 초래한다.

- 인식 상태(Success state, S_S) : 리더의 프리픽스 질의에 대해, 하나의 태그만이 응답하는 경우이다. 이 때 리더는 태그 ID를 인식한다.

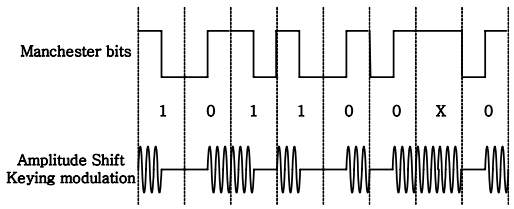
- 충돌 상태(Collision state, S_C) : 리더의 프리픽스 질의에 대해, 태그 ID가 일치하는 다수의 태그들이 응답해 충돌이 발생한 경우이다. 이 때 리더는 각 알고리즘마다 다양한 방식으로 태그 ID를 추출한다.

2.2 Manchester Code

제안 알고리즘은 태그로부터 ID를 응답 받은 후, 응답 받은 비트열의 이상 유무를 Manchester code를 사용하여, 판별한다.

Manchester code는 하나의 비트 구간에서 반드시 위상의 변화를 발생시켜 data-0 또는 data-1을 판별한다. [그림 1]과 같이, low→high로 위상이 바뀌면 data-0이고, high→low로 위상이 바뀌면 data-1이다[7].

[그림 1]은 이진 데이터를 Manchester code로 부호화 과정을 거친 후 ASK(Amplitude Shift Key ing)로 변조한 신호를 나타낸다. 이 code는 하나의 비트 셀에서 위상이 변하지 않는 상태를 충돌('X') 상태로 하여, data-0 또는 data-1 또는 비트 충돌('X')를 구분 할 수 있다[3, 8, 9].



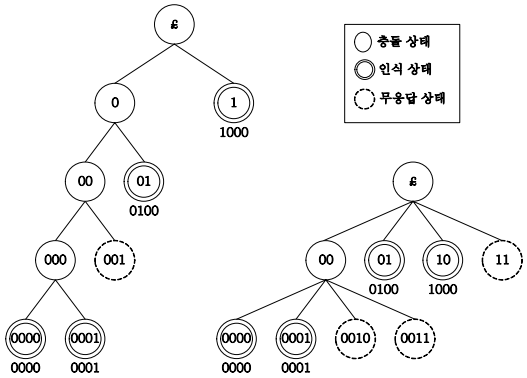
[그림 1] Manchester 비트와 ASK 변조

2.3 쿼리 트리 알고리즘과 4-ary 쿼리 트리 알고리즘

쿼리 트리 알고리즘(Query Tree Algorithm, QTA)은 트리 기반의 알고리즘으로 트리의 항에 따라 명칭을 달리한다. 기본적인 트리의 항은 2이고, 쿼리 트리 알고리즘으로 명명된다. 트리의 항이 4인 경우, 4-ary 쿼리 트리 알고리즘이다[10, 11, 12, 14].

QTA는 리더의 질의와 태그의 응답을 한 과정으로 구성된다. 리더는 일부분의 태그 ID 비트열을 프리픽스 $q_1q_2q_3$ q_x(q_x = {0, 1}, $1 \leq x \leq b$, b는 태그 ID의 비트 수)로 생성하고, 이를 태그로 질의한다. QTA는 리더의 프리픽스 질의에 하나의 태그가 응답한다면, 리더는 인식한다. 그리고 하나 이상의 태그가 응답할 경우, 리더는 인식할 수 없는 비트열이 존재함을 감지하고, 현재의 프리픽스 비트열에 '0'과 '1'을 추가($q_1q_2q_3 \cdots q_x0$, $q_1q_2q_3 \cdots q_x1$)하여 재 질의한다. 인식 영역 내의 모든 태그를 인식할 때까지 반복하게 된다.

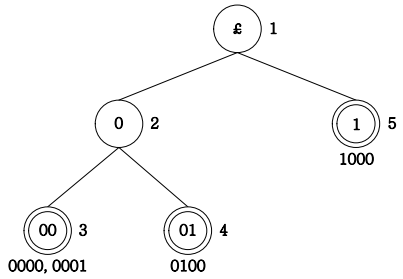
4-ary QTA의 기본적인 동작 과정은 쿼리 트리 알고리즘과 동일하다. 그러나 리더가 인식하지 못할 경우, 현재의 프리픽스 비트열에 “00”, “01”, “10”, “11”을 추가($q_1q_2q_3 \cdots q_x00$, $q_1q_2q_3 \cdots q_x01$, $q_1q_2q_3 \cdots q_x10$, $q_1q_2q_3 \cdots q_x11$)하여 재질의 한다.



[그림 2] 쿼리 트리와 4-ary 쿼리 트리 알고리즘

2.4 개선된 쿼리 트리

개선된 쿼리 트리(Improved Query-Tree, IQT)은 QTA와 유사하다. QTA는 태그 ID의 충돌 여부만을 감지 하지만, IQT는 충돌 비트 위치를 추출할 수 있다[13, 15]. 따라서 리더는 다수 개의 태그가 응답하여, 태그 ID의 비트열에 충돌이 발생한다면, 첫 번째 충돌 비트 위치를 추출한다.

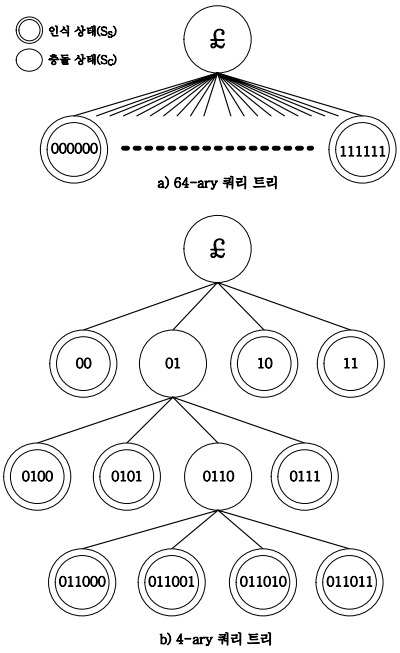


[그림 3] 개선된 쿼리 트리 알고리즘

IQT는 QTA와 4-ary QTA 달리, 첫 번째 충돌 비트를 위치를 이용하여, 첫 번째 충돌 비트 이전까지 인식된 비트열과 충돌 비트를 프리픽스로 생성을 하게 된다. 즉 질의한 프리픽스+정상적으로 수신된 비트+충돌 비트 (‘0’과 ‘1’)로 2개의 프리픽스를 생성한다.

3. N-ary 쿼리 트리 알고리즘

N-ary 쿼리 트리 알고리즘(N-ary Query Tree Algorithm, N-ary QTA)은 맨체스터 코드를 사용하여 비트 충돌에 대해 충돌 위치와 개수를 추출 할 수 있으며, N($2^2 \le N \le 2^6$)는 [그림 4]와 같이 최대 64-ary까지 확장된다.



[그림 4] 4-ary 쿼리 트리와 64-ary 쿼리 트리

3.1 리더의 프리픽스

리더와 태그간의 성능개선을 위해서는 하드웨어적인 부분과 소프트웨어적인 부분의 향상이 필요하다. 본 연구는 소프웨어적으로 RFID 리더와 태그간의 성능을 개선하기 위한 방법을 제안한다. 이 방법은 통신과정에서 프로토콜을 수정하여, 리더가 한 번의 질의에 대해, 태그들로부터 전달받은

데이터에서 태그 ID를 인식하기 위한 최대한의 정보를 얻고자한다. 따라서 리더의 질의문과 태그의 응답문은 필히 수정이 필요하다. 제안된 알고리즘은 리더로부터 전송된 프리픽스에 따라 자신의 원본 비트들을 전송하는 것과 변형된(change)비트들을 판단해서 전송할 수 있도록 프리픽스 format을 설계한다. N-ary QTA은 충돌 비트들의 분석에 따라 N_P을 생성한다. <표 1>과 같이, 리더는 2종류(N_{P_1}, N_{P_2})의 N_P를 생성한다.

〈표 1〉 N_{P_1}, N_{P_2}의 프리픽스 구성

	$Q_1 \cdots Q_X$			L_1
N_{P_1}	$I_1 \cdots I_X$		$D_1 \cdots D_X$	0
N_{P_2}	$S_1S_2S_3$	$P_1 \cdots P_X$	$R_1 \cdots R_{F-1}$	1
			$R_{X+1} \cdots R_{F-1}$	

리더는 프리픽스 N_p 생성 시 몇 가지 규칙을 가진다. 프리픽스 N_p는 비트 분석을 통해 N_{p_1}과 N_{p_2}로 구성된다. 인식 과정에서 리더는 각 단계별로 N_{p_1}과 N_{p_2}를 반복적으로 전송하면서 태그 ID를 인식한다. <표 2>의 변수 m의 값은 프리픽스의 길이를 결정하는 중요한 요소이다. m는 유동적인 값을 가질 수 있으며, 2~6의 범위를 가진다. 제안 알고리즘에서 이 값은 초기에 설정하며, 인식 과정이 종료될 때까지 변하지 않는다.

m이 2이면, N_{p_1}의 S 비트열 $S_1S_2S_3$은 “010”로 설정된다. 그리고 N_{p_1}의 D 비트열은 D_1D_2가 된다. 또한 m이 6이면, N_{p_1}의 S 비트열 $S_1S_2S_3$은 “110”로 설정된다. 그리고 N_{p_1}의 D 비트열은 $D_1 \cdots D_6$가 된다. 따라서 리더의 프리픽스 N_p는 정의 1의 규칙을 따른다.

정의 1 : 최악의 경우, 리더는 N_{P_1}생성시, Q 비트열의 길이는 m설정 값에 따라 확장된다.

인식 단계에서, N_{p_1}는 인식된 비트열($I_1 \cdots I_X$)과 확장된 비트 변형 방식으로 생성된 비트열 $D_1 \cdots D_X$로 구성된다. N_{p_1}을 태그로 전송한 후, 2비트

〈표 2〉 리더의 프리픽스에 대한 정보

심벌	명세
N_p	리더는 태그에서 전송받은 응답 비트열로 다음에 전송할 새로운 질의를 생성한다. 이 비트열은 N_{p_1}과 N_{p_2}로 나눠진다.
m	이 변수는 태그에서 변환할 비트열의 수를 결정하기 위해 사용된다. m값은 2~6이다.
R	R은 태그가 리더로 전송한 비트열이다. R_1 비트부터 충돌이 발생한다면, N_{p_2} 생성 과정의 충돌 없이 인식된 비트열($R_1 \cdots R_{F-1}$ 또는 $R_{X+1} \cdots R_{F-1}$)은 존재하지 않는다. $R_1 \cdots R_X$(R ∈ {0, 1}, 1 ≤ X ≤ b, b는 태그 ID의 비트 수)
I	I는 N_{p_2}의 $P_1 \cdots P_X+R_1 \cdots R_{F-1}$로서, 충돌 없이 인식된 비트열이다. $I_1 \cdots I_X$(D ∈ {0, 1}, 1 ≤ X ≤ b-m)
D	D는 태그에서 리더로 전송한 $E_1E_2 \cdots E_X$를 실제 ID로 변환한 비트열이다. $D_1 \cdots D_X$(D ∈ {0, 1}, 1 ≤ X ≤ 6)
S	S는 m의 값에 따라 이진수로 변환한 비트열이다. 이 비트열의 길이는 항상 3비트이다. $S_1S_2S_3$(S ∈ {0, 1})
P	L_1 = 0일 경우의, Q 비트열이다. 리더가 태그로 전송한 비트열이다. $P_1 \cdots P_X$(P ∈ {0, 1}, 1 ≤ X ≤ b-m)
L	이 비트는 리더와 태그에서 N_{p_1}과 N_{p_2}를 구별하기 위한 비트이다. L_1(L_1 ∈ {0, 1})
F	리더가 태그로부터 전송받은 R 비트열에서 첫 번째 충돌 비트의 인덱스이다.
Q	이 비트열은 N_p에서 L_1을 제외한 프리픽스이다. N_{p_1}에서 [I + D]로 구성되고, N_{p_2}에서 [S +P +R]로 구성된다. $Q_1 \cdots Q_X$(Q ∈ {0, 1}, 1 ≤ X ≤ b-m)

이상의 충돌이 발생한 경우에 N_{p_2}를 질의한다. 이 때 리더는 m의 값에 따라 설정된 S 비트열 $S_1S_2S_3$과 N_{p_1}에서 전송된 비트열($P_1 \cdots P_X$)과 전송받은 R 비트열에서 충돌이 없는 비트열($R_1 \cdots R_{F-1}$)로 구성된다. N_{p_1}만으로 질의를 구성하지 않는 이유는 존재하는 태그 ID에 대한 정보 없이 재질의 할 경우, 태그의 무응답 상태(S_I)를 초래하고, 인식 시간을 증가시키기 때문이다.

3.2 확장된 비트 변형 방식

선행 연구[10]에서, 비트 변형 방식(Bit change

Mode, BCM)을 제안하였다. 이 방식은 연속 2 비트 충돌 발생 시, 태그들은 연속 2비트와 매칭되는 변환 비트열을 리더로 전송한다. 연속 2비트에 대해 2^2 = 4비트가 필요하고, 연속 3비트(2^3 = 8), 4비트(2^4 = 16) 등 그 이상도 가능하다. 이 방식의 단점은 변환될 비트열 모든 태그들이 값을 저장하고 있어야 매칭되는 비트열을 리더로 전송할 수 있다. 따라서 성능 향상이 하드웨어에 매우 제약적인 단점을 갖고 있다. 이 단점을 해결하기 위해, 본 연구에서는 데이터를 저장하지 않고, 비트열을 변환하여 전송하는 확장된 비트 변형 방식(Extended Bit Change Mode, EBCM)을 고안하였다.

〈표 3〉 m의 값에 따른 S 비트열과 변환 비트열

m	$S_1S_2S_3$	$E_1E_2 \cdots E_X$
2	010	$E_1E_2 \cdots E_4$
3	011	$E_1E_2 \cdots E_8$
4	100	$E_1E_2 \cdots E_{16}$
5	101	$E_1E_2 \cdots E_{32}$
6	110	$E_1E_2 \cdots E_{64}$

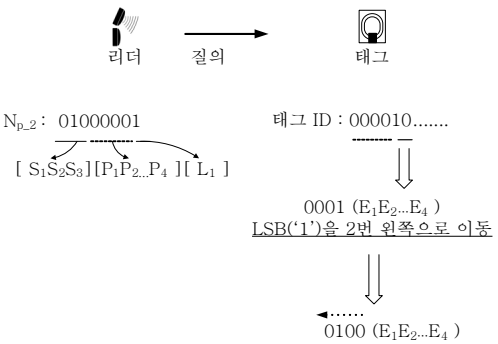
EBCM의 변환 비트열은 $E_1E_2 \cdots E_X$($E_X \in \{0, 1\}$, $1 \le x \le 2^m$, 초기의 최하위 비트는 '1' 설정)로 나타내고, $E_1E_2 \cdots E_X$ 중에서 하나의 비트만 비트 '0' 또는 비트 '1'로 생성한다. EBCM의 최대 비트 길이는 2^6 = 64이다. 최대 비트 길이가 6비트로 설정한 이유는 EPC global의 표준 태그 ID가 96비트이므로, 이 비트 내의 길이를 갖도록 m의 최대 값을 6까지로 하였다. m의 설정 값에 따른 S 비트열 $S_1S_2S_3$과 태그의 변환 비트열 $E_1E_2 \cdots E_X$은 <표 3>과 같다[6].

EBCM은 다음과 같은 장점을 가진다. 첫째는 BCM보다 더 많은 연속/비연속 충돌 비트열("X0XX00X …")을 재질의 없이 해결함으로써, 리더의 인식 성능을 향상시킨다. 둘째로, 초기에 m의 값이 고정되고, 태그는 변환 과정에 필요한 변환 비트 수를 미리 계산할 수 있다. 셋째로, BCM에 비해 변환 비트의 수가 많음에도 불구하고 간단한 동작으로 변환이 가능하다. 넷째로, BCM과 같이, 변환 비트열을 저장할 필요가 없다.

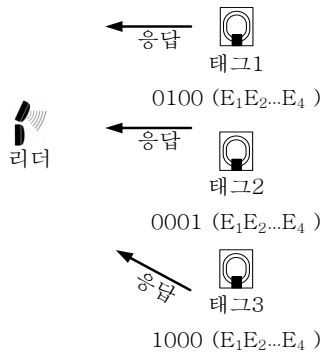
[그림 5]는 태그 ID의 비트 변환을 위한 과정이다. 충돌이 2비트 이상 발생할 경우, 리더는 설정된 m의 값에 따라 동작한다. 아래 그림에서, m = 2이고, S 비트열은 "010"로 설정되었다. 따라서 리더는 인식된 비트열 "0000"($P_1P_2 \cdots P_4$)을 포함한 N_{p_2} 프리픽스를 태그로 질의한다.

태그는 인식된 비트열 "0000"($P_1P_2 \cdots P_4$)과 태그 ID의 일부분인 프리픽스와 일치여부를 검사하고, S 비트열은 "010"이므로, 이후 2비트를 변환한다. 이 비트열 "10"의 값은 10진수로 2이다. 그리고 변환 비트열 "0001"($E_1E_2 \cdots E_4$)은 최하위 비트(LSB)를 10진수 값('2')에 따라, 2번 왼쪽으로 이동하여 변환 비트열 "0100"($E_1E_2 \cdots E_4$)을 생성한다.

[그림 5]의 과정 이후, [그림 6]과 같이, 태그들은 변환된 서로 다른 비트 패턴을 리더로 전송한다. 이



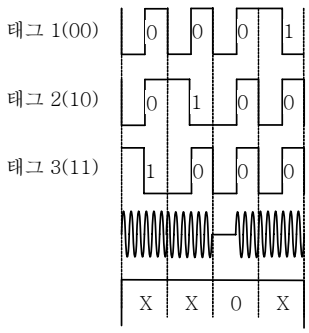
[그림 5] 태그 ID의 비트 변환 과정



[그림 6] 변환된 비트열을 리더로 전송

때 각 태그들의 데이터는 동기화를 맞추어 동시에 받는다고 가정된다. 동기화가 이루지지 않을 경우는 재질의 한다. 리더는 m = 2으로 설정된 것을 알고 있으므로, 변환된 비트열 $E_1E_2 \cdots E_4$이 모두 전송될 때까지 데이터를 받아들인다.

[그림 7]과 같이, 리더는 합성된 신호를 Manchester code를 이용해 이진 데이터로 변환한다. 변환된 비트열은 "XX0X"이며, 전송받은 비트열을 분석해서 "00", "10", "11"의 ID를 추출할 수 있다.



[그림 7] 리더의 태그 ID 인식

3.3 리더와 태그의 동작

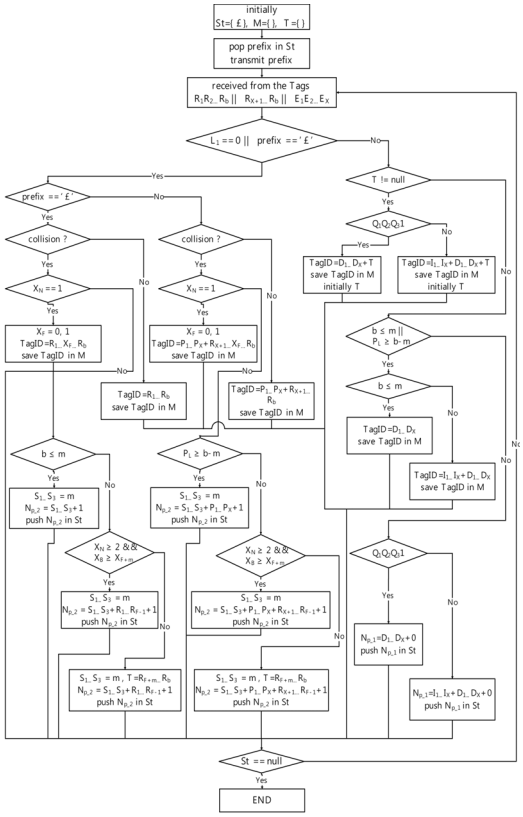
리더는 Stack(St), Memory(M), Temp Memory(T)를 초기화한 후 '£'을 St에 저장한다. 초기 '£'을 태그들에게 전송한다. 다음부터는 제안 알고리즘에 의해 생성된 프리픽스(N_P)를 St에서 pop하여 전송한다. 리더는 L_1('0'||'1')에 따라 인식방법을 달리한다. <표 2>에서, L_1 비트는 N_{p_1}('0')과 N_{p_2}('1')를 구별하기 위한 비트이다. 이 비트 값에 따라 리더는 인식 과정을 달리한다. 그리고 초기의 prefix가 '£'인 경우도 있으므로, 이 경우를 고려하여 흐름도를 구성하였다.

L_1==0 또는 '£'일 때, 인식 과정의 프리픽스는 m에 따라 증가하고, 이 경우 전송받은 비트열($R_1R_2 \cdots R_b \,||\, R_{X+1} \cdots R_b$)에서 충돌수를 검사하여, 없거나 1개이면 태그는 인식하게 된다. 이것은 Manchester code를 사용하여 충돌 위치와 개수를 파악 할 수 있어 가능하다. 충돌 비트 수가 2개 이상이면 프리픽스의 증가(불필요한 질의의 생성) 없이 다음의 조건에서, N_{P_2}를 생성한다. 첫째로, $b \leq m$는 할당된 태그 ID의 비트 길이(b)가 설정된 m의 값 내에서만 충돌이 존재하는 경우이고, 둘째로, prefix == '£'이면서, $X_N \geq 2$ && $X_B \geq X_{F+m}$는 초기 '£'을 질의한 후, 충돌 비트 수(X_N)가 2 이상이면서, 충돌 비트 위치(X_B)가 첫 번째 충돌 비트(X_F)부터 m값(X_{F+m}) 이후에도 존재하는 경우이다. 셋째로, $P_L \geq b-m$ 는 프리픽스의 길이(P_L) 값이 b-m의 값보다 크거나 같은 경우이고, 넷째로, prefix != '£'이면서, $X_N \geq 2$ && $X_B \geq X_{F+m}$인 경우이다. 이런 세부적인 조건들은 충돌 비트의 위치와 개수를 이용하여, 불필요한 질의를 생성하지 않고, 최적의 성능을 도출한다.

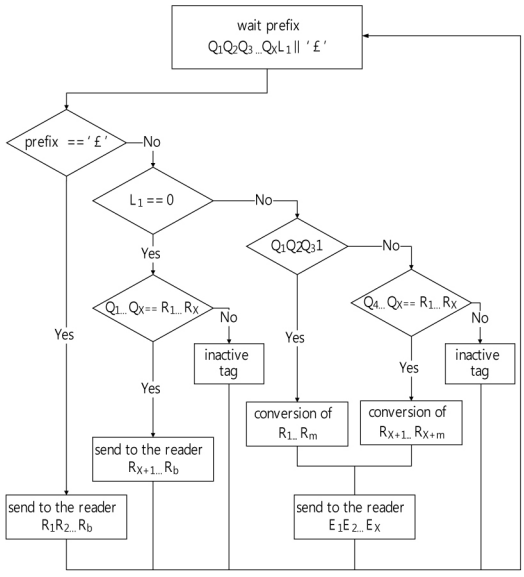
L_1==1인 경우는 EBCM에 의해 변환된 비트열 $E_1E_2 \cdots E_X$을 태그로부터 전송 받고, 태그의 원본 ID로 디코드($D_1D_2 \cdots D_X$)한다. 이 때, Temp Memory(T)는 충돌 비트들이 $X_N \geq 2$ && $X_B \geq X_{F+m}$의 조건을 만족하지 않는 경우로, 충돌 비트들이 X_{F+m-1} 이내에만 존재한다. 즉, $R_{F+m} \cdots R_b$비트열은 충돌 비트가 없으므로, 이 비트열을 T에 저장하고, 바로 다음 질의에 디코드된 비트열과 T에 저장된 비트열을 결합하면, 전체 태그 ID를 추출할 수 있다. 반대로 T가 비었다면 추출할 수 없는 경우로 N_{P_1}를 생성하여 St에 저장하고, 재질의 한다. 따라서 이 경우에 프리픽스의 길이가 증가한다.

리더는 위의 과정에 따라 동작한 후, St를 체크한다. St에 저장된 프리픽스가 존재하는 동안 반복적으로 동작하고, St가 비었다면, 인식 과정은 끝난다.

태그는 리더로부터 프리픽스 '£' 또는 L_1==0 또는 L_1==1인 비트열을 전송 받는다. 프리픽스 '£'를 전송 받으면, 전체 태그 ID($R_1R_2 \cdots R_b$)를 리더로 전송한다. 이 후, 질의 비트열의 L_1('0'||'1')에 따라 전송받은 프리픽스($Q_1 \cdots Q_X$)와 태그 ID($R_1 \cdots R_X$)의 일치 여부를 검사하고, 불일치하면 inactive상태로 대기한다. 일치한다면, L_1==0이면, $R_{X+1} \cdots R_b$를 리더로 전송한다. 그리고 L_1==0이면, $S_1S_2S_3$에 따라 EBCM의 비트열 $E_1E_2 \cdots E_X$를 리더로 전송한다.



[그림 8] 리더의 흐름도



[그림 9] 태그의 흐름도

4. 알고리즘 분석

이 절에서는 제안 알고리즘을 분석하여, 다중 태그를 인식하는데 걸리는 지연시간을 계산한다. 초기에, m = 6로 설정 되었다고 가정한다. 인식 과정에서, 태그로부터 전송 받은 비트 열에서 충돌 비트 수가 항상 2 이상이고, 태그 수가 $64^2 \le n < 64^{b-6}$이면, 질의 수(프리픽스의 수, P_N)의 worst case는 다음과 같이 나타낼 수 있다.

$$P_N \le \frac{128n}{63} - \frac{128}{63} \qquad (1)$$

증명 : 태그는 유일한 비트열 $R_1R_2R_3...R_b(R_b \in \{0, 1\}$, b는 태그 ID의 비트길이)를 가지며, N개가 존재할 수 있다. 이 때 인식 트리는 각 높이 (K, $0 < K \le \log_{64} n - 1$)를 기준으로, 최대 64^K개의 프리픽스를 처리할 수 있다. 최악의 경우, 태그 ID 비트 길이는 정의 1에 의해 6K씩 증가한다. 따라서 최악의 경우는 태그 ID의 길이에 따라 최대 태그 수가 존재하는 경우이다. 이때 인식된 프리픽스의 비트열은 마지막 6비트를 제외한 $I_1I_2...I_{b-6}$("$I_1I_2...I_{b-6}$ XXXXXX")를 가진다. 이 때 모든 태그를 인식하기 위해서 높이(K)는 $\log_{64} n - 1$까지 증가한다.

태그 ID의 비트 길이에 따라, 최대 태그 수가 존재하지 않을 경우는 최소 태그 수를 이용해 P_N 을 구한다. 최소 태그 수가 2개 이면, 충돌 비트 수는 2 이상 발생할 수 있다. 제안 알고리즘은 충돌 비트 수가 2 이상이면, 질의를 생성하므로, $P_N = n/2$ 이 되고, 충돌 노드가 증가하는 경우는 충돌 비트가 존재하는 경우이며, 충돌 비트가 존재할 수 있는 태그 수는 $\log_{64} n + 1$이다. 제안 알고리즘은 N_{p_1}과 N_{p_2}을 반복하면서 전송하기 때문에 P_N은 2배가 된다. 따라서 $\log_{64} n - 1$까지는 $2 \times (64^K)$이고, $\log_{64} n$까지는 $2 \times (n/2)$을 생성한다. 마지막으로 $\log_{64} n + 1$은 위의 높이에서 존재하는 태그의 수만큼의 질의가 생성되었으므로, n개 질의를 생성한다. P_N(n, K)은 다음과 같다.

$$P_N(n,K)= \quad (2)$$
$$\begin{cases} 2(64^K) & \text{if } (0< K\le \lfloor \log_{64} n-1 \rfloor) \\ n & \text{if } (\lfloor \log_{64} n-1 \rfloor < K\le \lfloor \log_{64} n+1 \rfloor) \end{cases}$$

식 (2)을 이용하여 최대 P_N을 계산하면

$$\begin{aligned} P_N &\le \sum_{K=1}^{\lfloor \log_{64} n+1 \rfloor} P_N(n,K) \quad (3) \\ &\le 2\sum_{K=1}^{\lfloor \log_{64} n-1 \rfloor} 64^k + \sum_{\lfloor \log_{64} n \rfloor}^{\lfloor \log_{64} n+1 \rfloor} n \\ &\le \frac{128n}{63} - \frac{128}{63} \end{aligned}$$

m = 2로 설정된 경우, 충돌 조건에 따른 프리픽스 생성은 위와 동일하고, 태그 ID 비트 길이는 정리 1에 의해 2K씩 증가한다. 이 때, $P_N(n, K)$의 worst case는 다음과 같이 나타낼 수 있다.

$$P_N(n,K)= \quad (4)$$
$$\begin{cases} 2(4^K) & \text{if } (0< K\le \lfloor \log_{4} n-1 \rfloor) \\ n & \text{if } (\lfloor \log_{4} n-1 \rfloor < K\le \lfloor \log_{4} n+1 \rfloor) \end{cases}$$

제안된 알고리즘에서, m의 설정 범위는 $2 \le m \le 6$이다. 이 때, 각 설정 값에 따라 태그 수가 $(2^m)^2 \le n < (2^m)^{b-m}$ 이면, P_N는 아래와 같다.

$$P_N(n,K)= \quad (5)$$
$$\begin{cases} 2((2^m)^K) & \text{if } (0< K\le \lfloor \log_{2^m} n-1 \rfloor) \\ n & \text{if } (\lfloor \log_{2^m} n-1 \rfloor < K\le \lfloor \log_{2^m} n+1 \rfloor) \end{cases}$$

5. 성능평가

이 절에서는 제안 알고리즘의 성능을 트리 기반의 Query Tree Algorithm, 4-ary Query Tree Algorithm, Improved Query-Tree을 비교하여 다중태그를 인식하는데 걸리는 지연 시간을 평가한다. 성능 평가는 분석에 의해 산출된 질의 수와 시뮬레이션 프로그램을 구현하여 실험한 질의 수를 Matlab을 활용하여 그래프로 나타낸다. 트리 기반 알고리즘의 질의-응답 수는 한 번 질의에 한 번 응답이므로, 질의 수만은 산출하여도, 각 알고리즘의 성능 비는 동일하다.

실제 무선 통신에서 존재하는 데이터 손실, 데이터 송·수신시의 전파 반사, 전파의 노이즈, 통신 하드웨어 등의 제약 사항은 배제하고, 리더와 태그 간의 질의-응답 시 데이터 간섭으로 인한 데이터 추출방식에 대해서만 실험 결과를 도출하였다.

본 실험에서 사용하는 태그 표준은 EPC Tag Data Standards Version 1.6을 따른다[6]. 태그 규격은 헤더(Header) 8비트, 업체코드(General Manager Number) 28비트, 상품코드(Object Class) 24비트, 일련번호(Serial Number) 36비트의 4개 필드로 구성된, General Identifier(GID-96)이다.

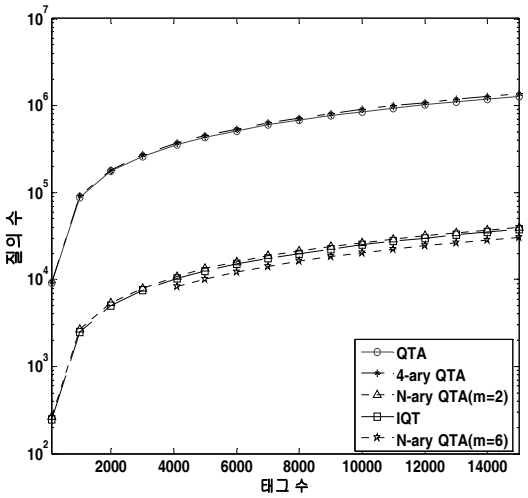
〈표 4〉 실험 환경 조건

태그	태그 ID(96 비트)
태그표준	EPC Tag V1.6
태그 ID	GID-96
충돌 방식	다중 태그 충돌
시뮬레이션 프로그램	C#으로 구현
태그 ID의 할당	랜덤 할당
생성된 태그의 수	100~15,000

본 논문에서 성능 평가를 위한 실험 환경 조건은 <표 4>와 같다. 제안 알고리즘은 하나의 리더에 다수 개의 태그가 존재할 경우를 가정한다. 태그 ID의 할당은 랜덤 할당을 사용하였다. 랜덤 할당 방식은 시간 변수를 종자(seed) 값으로 하여, 태그 ID 값의 범위를 균일 분포(uniform distribution)을 갖도록 생성하였다. 랜덤 할당 방식을 사용한 태그 ID 값은 중복 되지 않는다. 태그 ID의 수는 100~15,000개를 생성하였다. 그리고 성능 평가는 C#으로 구현된 시뮬레이션 프로그램을 사용하여, 질의 수를 도출하였다. 이 때 태그의 수(N), 태그 ID 비트 길이(b), m의 값을 조절할 수 있다.

5.1 분석에 의한 질의 수

리더의 질의 수(P_N)은 인식 시간을 계산하는데 중요한 요소이다. 질의 수가 적을수록 리더는 태그들을 빨리 인식한다. 각 알고리즘은 질의 수를 줄임으로써 성능을 향상시킨다. [그림 10]은 트리 기반의 알고리즘들과 비교한 질의 수를 나타낸다. 이 질의 수는 각 알고리즘들의 최악의 경우를 분석한 식에 적용하여, 질의 수를 비교하였다. 태그의 수를 점진적으로 증가 시키면서, 각 알고리즘에 대한 질의 수의 변화를 Matlab를 사용하여 계산하였다. QTA와 4-ary QTA 충돌 감지 방식이고, IQT와 N-ary QTA는 충돌 검출 방식을 사용한다. 따라서 충돌 검출 방식이 충돌 감지 방식에 비해 우수한 성능을 나타낸다. [그림 10]의 결과에서, m = 6일 때, 태그 수가 64^2(4,096)보다 작은 경우는 조건에 맞지 않으므로, 결과를 나타내지 않는다.



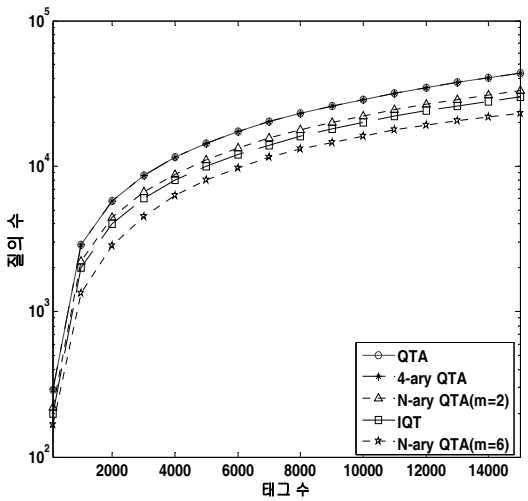
[그림 10] 분석에 의한 질의 수

N-ary QTA의 질의 수는 QTA에 비해 97%(m = 2)~98%(m = 6), 4-ary QTA에 비해 97%(m = 2)~98%(m = 6)이 개선되었고, IQT에 비해 m이 2로 설정되었을 경우는 7% 성능의 저하를 나타낸다. 그리고 m이 6으로 설정되었을 경우는 19%의 성능이 개선되었다.

5.2 실험에 의한 질의 수

[그림 11]은 QTA, 4-ary QTA, IQT, N-ary QTA들을 C#으로 구현하여, 성능을 나타낸 결과이다. 성능 평가에 사용된 태그 ID는 96비트의 비트 길이를 가지며(EPC global), 태그 수에 따라 랜덤으로 할당하였다. 실험 결과는 16종류의 태그 수를 변경하면서, 각 태그 수를 5회 평균적으로 도출하였다. 실험 결과는 분석에 의한 실험 결과와 비교해 모든 알고리즘들이 낮은 수치를 나타낸다.

[그림 11]에서, N-ary QTA의 질의 수는 QTA에 비해 23%(m = 2)~45%(m = 6), 4-ary QTA에 비해 24%(m = 2)~45%(m = 6)이 개선되었고, IQT에 비해 m이 2로 설정되었을 경우는 11% 성능의 저하를 나타낸다. 또한 m이 6으로 설정되었을 경우는 21%의 성능이 개선되었다.



[그림 11] 실험에 의한 질의 수

<표 5>는 [그림 11]의 데이터를 사용하여, 하나의 태그를 인식하는데 필요한 질의 수를 나타낸다. 이 때, x는 태그 수이고, y는 x에 대한 각 알고리즘의 5회 평균 질의 수이며, 비례 상수 C는 y/x이다.

〈표 5〉 질의에 대한 비례상수(C = y/x)

알고리즘 \ 태그 수(x)		1,000	5,000	9,000	15,000
QTA	y	2,899	14,415	25,937	43,395
	C	2.90	2.88	2.88	2.89
4-ary QTA	y	2,891	14,351	25,977	43,399
	C	2.89	2.87	2.89	2.89
IQT	y	1,999	9,999	17,999	29,999
	C	2.00	2.00	2.00	2.00
N-ary QTA (m = 2)	y	2,204	10,995	19,908	33,206
	C	2.20	2.20	2.21	2.21
N-ary QTA (m = 6)	y	1,340	7,980	14,620	23,078
	C	1.34	1.60	1.62	1.54

6. 결 론

RFID에서 리더는 다수의 태그를 식별하기 위한 충돌방지 알고리즘이 필요하다. 본 논문은 다중 태그인식을 위해 리더의 질의문(N_p)과 태그의 비트 전송 방식(EBCM)을 제안하였다.

제안 알고리즘은 트리 기반의 충돌 검출 방식이다. 이 알고리즘은 트리 기반에서 태그 응답 시, N_p와 EBCM을 사용하여, 한 번 질의에 많은 정보를 획득하여, 정확하고 빠르게 태그 ID 인식을 한다. 따라서 슬롯 알로하 기반의 알고리즘의 단점인 타임 슬롯에 의한 지연 시간과 무응답 슬롯을 발생시키지 않는다. 또한 N-ary QTA은 Manchester code를 사용하여 비트 충돌에 대해 충돌 위치와 개수를 추출할 수 있다.

제안 알고리즘은 리더의 질의로부터 태그가 자신의 ID를 전송하는 것과 변형된 ID(EBCM)를 전송하는 방식으로 나뉜다. 태그는 제안 알고리즘의 질의문 구성에 의해, L비트에 따라 리더로 데이터를 전송한다. 두 응답 방식을 교대로 태그로 전달하면서, 한 번 질의에 최대 64개(m = 6)의 태그 ID를 인식함으로써 인식 시간을 줄인다. 또한 제안 알고리즘은 m의 값을 적절하게 조절 할 수 있다. RFID 시스템에서, 리더와 태그 사이의 통신 거리, 주변 환경, 주파수 특성 등에 따라, 이 값을 조정함으로써 인식 성능을 유지하면서, 신뢰성을 높일 수 있다.

그리고 제안 알고리즘의 worst case를 분석하여 알고리즘의 성능을 추출하였다. 분석에서, N-ary QTA의 질의 수는 QTA에 비해 97%(m = 2)~98%(m = 6), 4-ary QTA에 비해 97%(m = 2)~98%(m = 6)이 개선되었고, IQT에 비해 7%(m = 2) 성능저하, 19%(m = 6) 성능이 개선되었다. 실험에 의한 연구 결과에서, N-ary QTA의 질의 수는 QTA에 비해 23%(m = 2)~45%(m = 6), 4-ary QTA에 비해 24%(m = 2)~45%(m = 6)이 개선되었고, IQT에 비해 11%(m = 2) 성능저하, 21%(m = 6) 성능이 개선되었다.

향후 연구과제는 데이터 송·수신 시, $E_1E_2 \cdots E_X$ 문자열을 개선하여 보안 기능과 데이터 추출 기능을 동시에 할 수 있도록 할 것이다.

참 고 문 헌

[1] 김성수, “RFID 다중 태그 인식을 위한 슬롯 할당 예측 알고리즘”, 경북대학교 대학원 공학박사학위논문, 2012.

[2] 류지호, 이호진, 석용호, 권태경, 최양희, “RFID 시스템에서 태그 충돌 중재를 위한 하이브리드 기법”, 「한국정보과학회 논문지」, 제34권, 제6호(2007), pp.483-492.

[3] 백덕화, “RFID 시스템에서 추가 비트를 이용한 빠른 태그 예측 알고리즘”, 경북대학교 대학원 공학박사학위논문, 2008.

[4] Cha, J. R. and J. H. Kim, “Dynamic framed slotted ALOHA algorithms using fast tag estimation method for RFID system”, *Consumer Communications and Networking Conference*, Vol.2(2006), pp.768-772.

[5] Eom, J., 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 Communi*

cations Letters, Vol.12, No.11(2008), pp.861-863.

[6] EPCglobal, "EPCglobal Tag Data Standards Version 1.6", GS1(http://www.gs1.org/gsmp/kc/epcglobal/tds/tds_1_6-RatifiedStd-20110922.pdf), 2011.

[7] Finkenzeller, K., *RFID Hand Book : Fundamentals and Applications in Contactless Smart Card and Identification, 2nd Ed.*, Chicester, Sussex, U. K., WILEY, 2003.

[8] ISO/IEC 18000-6 : 2004(E), Information technology-Radio frequency identification for item management Part 6 : Parameters for air interface communications at 860-960 MHz, 2004.

[9] ISO/IEC 18000-7 : 2004(E), Information technology-Radio frequency identification for item management Part 7 : Parameters for active air interface communications at 433 MHz, 2004.

[10] Kim, Y., S. Kim, S. Lee, and K. Ahn, "Improved 4-ary Query Tree Algorithm for Anti-Collision in RFID System", *International Conference on Advanced Information Networking and Applications*, (2009), pp. 699-704.

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

[12] Myung, J. and W. Lee, "Adaptive Binary Splitting : A RFID Tag Collision Arbitration Protocol for Tag Identification", *ACM/Springer Mobile networks and Applications*, Vol. 11, No.59(2006), pp.711-722.

[13] Myung, J. and W. Lee, "Adaptive Binary Splitting : A RFID Tag Collision Arbitration Protocol for Tag Identification", *IEEE International Conference on Broadband Networks*, Vol.1(2005), pp.375-383.

[14] Shih, D., P. L. Sun, D. C. Yen, and S. M. Huang, "Taxonomy and Survey of RFID Anti-collision protocols", *Computer Communications*, Vol.29, Issue.11(2006), pp.2150-2166.

[15] Zhou, F., C. Chen, D. Jin, C. Huang, and M. Hao, "Evaluating and Optimizing Power Consumption of Anti-Collision Protocols for Applications in R FID Systems", *Proceedings of the 2004 International Symposium on Low Power Electronics and Design*, (2004), pp.357-362,

[16] Zhu, L. and T. P. Yum, "A Critical Survey and Analysis of RFID Anti-Collision Mechanisms", *IEEE Communications Magazine*, 2011.

◈ 저 자 소 개 ◈

김 용 환 (hypnus@ikw.ac.kr)

경운대학교 컴퓨터공학과를 졸업하고, 경북대학교 일반대학원 컴퓨터공학과에서 공학석사 및 공학박사를 취득하였다. 현재 경운대학교 컴퓨터공학과 연구교수로 재직하고 있으며, 주요 연구분야는 임베디드 시스템, 디지털 회로 시스템, RFID 충돌방지 프로토콜, 안드로이드 프로그래밍 등이다.

류 명 춘 (mcryoo@ikw.ac.kr)

영남대학교 컴퓨터공학과를 졸업하고, 영남대학교 일반대학원 컴퓨터공학과에서 공학석사 및 공학박사를 취득하였다. 현재 경운대학교 컴퓨터공학과 교수로 재직하고 있으며, 관심분야는 지능정보시스템, 웹기반시스템, 바이오인포매틱스, 빅데이터 등이다.

박 준 호 (jhpark@ikw.ac.kr)

영남대학교 컴퓨터공학과를 졸업 및 동대학원에서 석사 및 박사를 취득하였다. (주)현대전자 멀티미디어연구소에서 HDTV CODEC을 개발하였고, 현재 경운대학교 컴퓨터공학과 교수로 재직하고 있으며, 관심분야는 유비쿼터스, 영상처리시스템 및 임베디드 시스템 등이다.