ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

4비트 패턴에 따른 슬롯 할당 기법을 이용한 RFID 태그 충돌 방지 알고리즘

A RFID Tag Anti-Collision Algorithm Using 4-Bit Pattern Slot Allocation Method

김 영 백[1*] 김 성 수[2] 정 경 호[3] 안 광 선[3]
Young Back Kim Sung Soo Kim Kyung Ho Chung Kwang Seon Ahn

요 약

RFID 시스템에서는 다중 태그가 동일한 주파수로 동시에 리더의 요청에 응답하기 때문에 발생하는 태그 충돌을 중재하는 절차가 필수적이다. 이 절차를 충돌 방지 알고리즘이라 하며 RFID 시스템에서 가장 핵심적인 기술이다. 본 논문에서는 다중 태그의 고속 식별을 위한 4-BPSA(4-Bit Pattern Slot Allocation) 알고리즘을 제안한다. 제안한 알고리즘은 슬롯을 사용하는 트리 기반의 알고리즘으로서 4비트 패턴에 따른 슬롯 할당 기법을 이용하여 정확한 예측을 통해 빠르고 효율적으로 태그를 식별한다. 알고리즘에 대한 수학적 성능 분석을 통해 worst-case에서 4-BPSA의 시간 복잡도가 O(*n*)이며 기존의 알고리즘에 비해 성능이 개선되었음을 보인다. 또한 MATLAB을 이용한 시뮬레이션 실험을 통한 알고리즘의 성능 평가 결과에 의거해 4-BPSA 알고리즘이 태그당 평균 0.7회의 질의를 수행하며 태그의 개수와 상관없이 안정적인 성능을 보이는 것을 검증하였다.

주제어 : 충돌 방지, 충돌 추적, 슬롯 쿼리 트리, 4 비트 패턴

ABSTRACT

The procedure of the arbitration which is the tag collision is essential because the multiple tags response simultaneously in the same frequency to the request of the Reader. This procedure is known as Anti-collision and it is a key technology in the RFID system. In this paper, we propose the 4-Bit Pattern Slot Allocation(4-BPSA) algorithm for the high-speed identification of the multiple tags. The proposed algorithm is based on the tree algorithm using the time slot and identify the tag quickly and efficiently through accurate prediction using the a slot as a 4-bit pattern according to the slot allocation scheme. Through mathematical performance analysis, We proved that the 4-BPSA is an O(n) algorithm by analyzing the worst-case time complexity and the performance of the 4-BPSA is improved compared to existing algorithms. In addition, we verified that the 4-BPSA is performed the average 0.7 times the query per the Tag through MATLAB simulation experiments with performance evaluation of the algorithm and the 4-BPSA ensure stable performance regardless of the number of the tags.

☞ keyword : Anti-collision, Collision Tracking, Slotted Query Tree, 4-Bit Pattern

1. 서 론

RFID는 제품 식별과 추적, 재고 관리에서 기존의 바코드를 대체하여 물류 및 유통 분야를 혁신할 것으로 기대되는 유망한 기술이다[1]. RFID 시스템은 질의기라고도 불리는 리더와 라벨이라고도 불리는 태그로 구성된 비접촉 자동 인식 시스템이다[2]. RFID 리더의 가장 중요한 기능은 자신의 인식 범위 내에 있는 개별 제품들에 부착된 태그의 고유한 EPC(Electronic Product Code) 코드를 식별하는 것이다. 식별된 제품의 EPC 코드는 해당하는 제품의 생산 정보나 유통 이력 등의 상세 정보를 제공하며, 인터넷을 통해 사용자에게 즉시 제공된다[3].

수동형 RFID 태그(EPC Global Class I or Class II tags)는 대량으로 물류 및 유통되는 개별 제품에 부착되며 저비용, 저전력, 초소형의 특징을 가진다. 이러한 제약 조건으로 인해 태그는 자체 전원이 없고 간단한 연산 능력만

1 Automotive IT Platform Research Team, Daegu-Gyeongbuk Research Center, ETRI, Daegu, 711-883, Korea
2 Dept. of Mobile Engineering, Kyungwoon University, Gumi, 730-739, Korea
3 Dept. of Computer Engineering, Kyungpook National University, Daegu, 702-701, Korea
* Corresponding author (realtech@daegu.ac.kr)
[Received 12 June 2013, Reviewed 26 June 2013, Accepted 1 August 2013]

가지며, 반송파 감지(carrier sensing) 능력도 없으므로 다른 태그들과 통신할 수 없고 단지 리더의 요청에 대해 응답만 할 수 있다[4].

RFID 태그와 리더 간의 통신과 관련된 문제점은 기존 무선 통신의 채널 다중 접근의 문제로 정의 할 수 있다. 하지만 태그의 여러 제약 조건들로 인해서 기존의 무선 통신에서 연구된 다양한 기법들을 대부분 적용할 수 없다[5]. RFID 리더는 태그 식별을 위한 요청 메시지를 무선 채널을 통하여 브로드캐스팅 하며, 리더의 인식 영역 내에 다수의 태그는 동일한 주파수로 동시에 응답하게 된다. 이때 태그 충돌(tag collision)이 발생하게 되어 각 태그를 정확하게 식별할 수 없는 문제점이 발생한다[6]. 따라서 리더가 다중 태그를 정확히 식별하기 위해서는 태그 충돌(tag collision)을 효과적으로 중재하는 절차가 필요하다. 이 절차를 충돌방지(anti-collision) 알고리즘이라 하며, RFID 시스템에서 가장 핵심적이며 필수적인 기술이다[6].

충돌방지 알고리즘은 태그의 응답을 중재하는 방법에 따라 크게 확률적(probability) 알고리즘과 결정적(deterministic) 알고리즘으로 분류된다[4-6]. 확률적 알고리즘은 알로하(aloha) 프로토콜에 기반을 두며 슬롯 간의 시간차를 이용하여 태그 식별을 수행한다[2]. 하지만 응답 슬롯을 태그가 임의(random)로 선택하기 때문에 태그 수가 많아지면 성능이 저하될 수 있으며 소위 태그 기아 문제(tag starvation problem)로 인해 일정한 시간 안에 모든 태그를 인식하지 못 할 수도 있다[4,7].

결정적 알고리즘은 트리(tree) 프로토콜에 기반을 두며 태그의 고유한 ID를 이용하여 응답 그룹을 점진적으로 나누어 가며 충돌을 피한다[8,9]. 이 과정에서 질의 트리를 형성하고 그 트리의 노드를 순회하며 태그 식별을 수행한다. 이 알고리즘은 모든 태그를 인식할 수 있고 식별 과정을 예측할 수 있다는 장점이 있다[10].

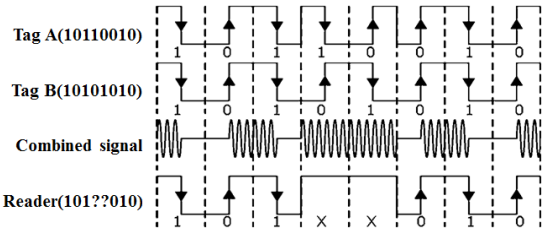
본 논문에서는 다중 태그의 고속 식별을 위한 4-BPSA (4-Bit Pattern Slot Allocation) 알고리즘을 제안한다. 제안한 알고리즘은 슬롯을 사용하는 트리(slotted tree) 기반의 결정적 알고리즘으로서 4비트 패턴에 따른 슬롯 할당 기법을 이용하여 정확한 예측을 통해 빠르고 효율적으로 태그를 식별한다. 또한 worst-case의 시간 복잡도 분석과 MATLAB을 이용한 시뮬레이션을 통해 제안하는 4-BPSA의 알고리즘 성능을 평가한다.

본 논문은 다음과 같이 기술한다. 2장에서는 제안한 4-BPSA 알고리즘과 관련된 연구를 기술하고, 3장에서는 4-BPSA 알고리즘을 상세히 설명한다. 4장에서는 worst-case의 시간 복잡도를 분석하고, 5장에서는 MATLAB을 이용한 시뮬레이션을 통해 4-BPSA의 알고리즘 성능을 평가한다. 마지막으로 6장에서 결론을 기술한다.

2. 관련연구

트리 기반의 프로토콜에서는 리더의 질의에 대한 태그 응답의 충돌 발생 여부에 따라 필요한 제어가 이루어진다. 태그 응답에서 충돌을 판단하는 방법에는 충돌 여부만을 판단하는 충돌 감지(collision detecting) 기법과 개별 비트 단위로 충돌을 검출하는 충돌 추적(collision tracking) 기법이 있다[11]. 충돌 추적 기법은 맨체스터 코딩(Manchester code) 사용하면 구현이 가능하다[6,11].

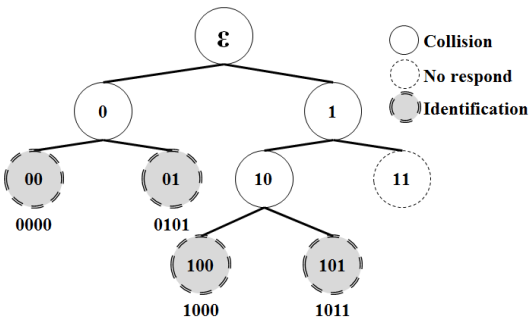
맨체스터 코드(Manchester code)는 각 비트 구간 내에서 위상 변이에 따라 비트 값을 결정한다[6,11]. 만일 어떤 비트 구간 내에서 위상 변이가 없으면 그 비트 위치에서 충돌이 발생한 것으로 인식한다. (그림 1)은 맨체스터 코드(Manchester code)를 사용하여 개별 비트 단위로 충돌을 검출하는 예를 보여준다. tag A의 ID는 10110010이고 tag B의 ID는 10101010일 때, 두 태그가 각각의 ID를 동시에 전송하면 bit 4와 bit 5 구간 내에서 위상 변이가 서로 반대로 중첩되어 결국 각각의 비트 구간 동안에 위상이 변하지 않게 된다. 따라서 bit 4와 bit 5에서 충돌이 발생한 것으로 검출된다.



(그림 1) 맨체스터 코드를 이용한 충돌 추적
(Figure 1) Collision tracking of Manchester code

쿼리 트리(QT : Query Tree)는 대표적인 결정적 알고리즘이다[12]. 태그의 응답은 리더의 질의에 의해서만 결정되므로 메모리레스 방식이라고도 한다. 리더는 태그 ID의 일부(*prefix*)를 사용하여 질의하면 ID가 *prefix*와 일치하는 태그들만 응답한다. 이때 충돌이 발생하면 *prefix*의 길이를 1비트씩 늘려가면서 충돌을 일으킨 태그들을 두 부분 집합(subsets)으로 나누고 각각의 부분 집합에 대해 다시 쿼리를 보내서 식별하게 된다. (그림 2)는 QT가

4개의 태그(0000, 0101, 1000, 1011)를 식별하는 예를 보여준다. 초기에 리더가 ‘ε(empty string)을 질의하면 특정 *prefix*에 상관없이 모든 태그들이 응답한다. 이때 4개의 태그가 동시에 응답하여 충돌이 발생하므로 리더는 *prefix*의 길이를 1비트 늘려 ‘0’과 ‘1’을 큐에 추가하고, 큐에서 ‘0’을 pop하여 다시 질의한다. 이 과정을 반복하여 리더가 ‘00’, ‘01’, ‘100’, ‘101’의 *prefix*를 각각 질의할 때마다 각각의 태그 ID가 식별된다. 한편 QT는 무응답 사이클(idle cycles)로 인한 성능 저하를 피할 수가 없다. 이는 충돌 감지(collision detecting) 기법을 사용하기 때문인데, 충돌 감지 기법에서 리더는 태그의 응답에서 단순히 충돌 발생 여부만을 판단하며, 충돌이 발생한 경우에 수신된 비트 정보를 전혀 이용하지 않고 폐기한다[11].



(그림 2) 쿼리 트리 알고리즘
(Figure 2) Query tree algorithm

최근에 제안된 충돌 트리(CT : Collision Tree) 알고리즘은 충돌 추적 트리(CTTA : Collision Tracking Tree) 알고리즘 및 개선된 쿼리 트리(IQT : Improved Query Tree) 알고리즘과 그 기법이 동일한 것으로서, 맨체스터 코드(Manchester code)의 특성을 활용한 충돌 추적(collision tracking) 기법을 사용하여 QT 알고리즘의 성능을 개선하였다[7,11,13]. 리더는 충돌 추적(collision tracking) 기법으로 태그의 응답에서 충돌이 발생한 비트의 위치를 정확히 추적한 후 충돌이 발생하기 이전의 비트 내용을 *prefix* 확장 시에 포함한다. 결국 리더는 충돌이 발생한 경우에만 *prefix*를 확장하여 재질의 하는데, 이때 확장된 *prefix*는 ‘현재의 *prefix*’+‘정상적으로 수신된 bits’+‘추가된 1 bit’로 구성된다. 일반적으로 충돌 추적(collision tracking) 기법을 적용하면 충돌 감지(collision detecting) 기법을 적용하는 것에 비해 *prefix*의 확장이 빠르며 무응답 *prefix*는 생성되지 않는다. 따라서 CT는 무응답 사이클(idle cycles)을 제거하여 태그 식별에 필요한 질의 횟수를 줄임으로서 빠른 성능을 보인다.

BSQTA(Bi-Slotted Query Tree Algorithm)와 BSCTTA (Bi-Slotted Collision Tracking Tree Algorithm)는 기존의 트리 기반의 알고리즘에서 충돌이 발생할 경우 마지막 1비트만 다른 한 쌍의 *prefix*를 생성하고 재질의 한다는 특성을 이용하여 QT 및 CT 알고리즘의 성능을 개선하였다[14]. 즉, 생성된 *prefix* 쌍은 ‘공통되는 비트들’과 ‘추가된 1비트’(‘0’ 또는 ‘1’)로 구성되는데 ‘공통되는 비트들’과 일치하는 응답 그룹은 ‘추가된 1비트’를 기준으로 다시 두 부분 집합(subsets)으로 나뉘게 된다. 이때 나뉘는 한 쌍의 응답 그룹 각각을 2개의 타임 슬롯으로 나누어 한 번에 응답 받는 원리이다. 이를 통해 기존의 2회에 해당하는 질의 횟수가 1회로 줄고 슬롯을 나누는 데 사용된 1비트는 전송에서 생략된다. 결과적으로 기존의 알고리즘에 비해 오버헤드를 *‘half − 1bit’*로 줄인 것이다.

ESA-IA(Efficient Slot Allocation-Inference Algorithm)는 *prefix*와 일치하는 응답 그룹을 패리티 메커니즘을 이용하여 두 부분 집합(subsets)으로 나누고 각각을 2개의 타임 슬롯으로 나누어 한 번에 응답 받는다[15]. 그리고 리더는 각 슬롯에서 충돌 비트의 개수가 2개 이하인 경우 충돌 비트 예측을 통해 태그를 인식한다. 그러나 실험 결과 ESA-IA는 무응답 사이클(idle cycles)로 인한 성능 저하를 보였다. 그 원인은 태그 ID에서 각 비트의 절대 위치와 상관없이 단순히 ‘1’의 개수를 기준으로 그룹을 나누기 때문으로 분석할 수 있다.

3. 제안하는 4-BPSA 알고리즘

본 논문에서 제안하는 4-BPSA(4-Bit Pattern Slot Allocation) 알고리즘은 4비트 패턴에 따른 슬롯 할당 기법을 이용하여 정확한 예측을 통해 빠르고 효율적으로 태그를 식별한다. 4-BPSA에서 리더는 *k*-bit 길이의 *prefix*를 질의한다. 리더가 질의한 *prefix*와 ID의 일부가 일치하는 태그들은 각각 자신의 타임 슬롯에 자신 ID의 *k* 번째 비트부터 마지막 비트까지를 순차적으로 전송한다. 4-BPSA에서는 태그가 응답하는 비트열에서 첫 4-bit에 대해 다음과 같이 정의한다.

a) 태그가 응답하는 비트열의 첫 4-bit를 $b_1b_2b_3b_4$으로 정의하고 첫 비트를 b_1으로 정의한다. 이때 b_1은 0이거나 1인 정수이다.

c) b_1을 제외한 $b_2b_3b_4$에서 '1'의 개수를 e로 정의한다. 이때 e는 0과 3 사이의 정수이다.

d) 태그가 응답하는 타임 슬롯의 번호를 S_i로 정의한다. 이때 i은 0과 5 사이의 정수이다.

4-BPSA에서 태그는 식(1)을 이용하여 자신이 응답할 타임 슬롯을 결정한다. b_1은 0이거나 1인 정수 값이며 e를 3으로 모듈러 연산(%)한 값은 0과 2사이의 정수 값이다.

$$S_i = (b_1 \times 3) + (e \,\%\, 3) \tag{1}$$

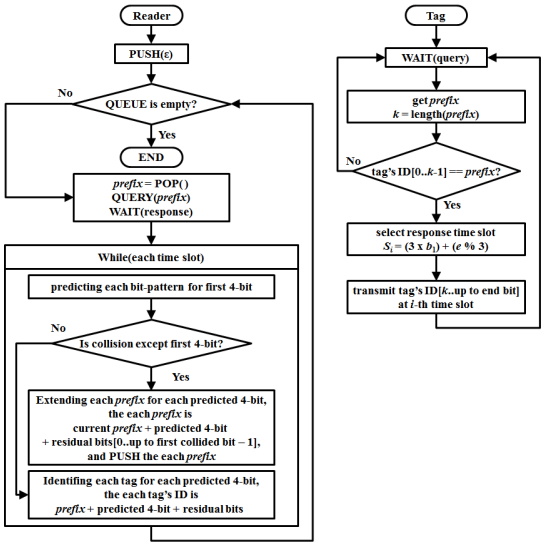
(표 1)은 $b_1b_2b_3b_4$의 비트 패턴에 따른 타임 슬롯 할당 정보를 보여준다. 각 타임 슬롯에 할당된 비트 패턴은 2개 이상의 태그가 응답하면 2개 이상의 비트에서 충돌이 발생하도록 설계되어 있다. 4-BPSA는 비트 패턴에 따라 6개의 타임 슬롯을 사용한다. S_0와 S_3에는 2개의 비트 패턴이 할당되어 있으므로 각 타임 슬롯 내에서 2개의 태그가 동시에 응답할 경우 쉽게 예측이 가능하다. S_1과 S_2와 S_4와 S_5에는 각각 3개의 비트 패턴이 할당되어 있다. 이들 타임 슬롯에서는 응답한 태그 개수만큼 충돌이 발생한다. 즉 동시에 응답한 태그의 비트 패턴에 따라 충돌 비트 개수와 충돌 비트 위치가 결정된다. 따라서 각 타임 슬롯 내에서 2개 이상의 태그가 동시에 응답할 경우 쉽게 예측이 가능하다.

(표 1) 4비트 패턴에 따른 슬롯 할당
(Table 1) Slot allocation according to 4-bit pattern

$b_1b_2b_3b_4$	b_1	(e % 3)	S_i
0000	0	0	0
0111			
0001	0	1	1
0010			
0100			
0011	0	2	2
0101			
0110			
1000	1	0	3
1111			
1001	1	1	4
1010			
1100			
1011	1	2	5
1101			
1110			

예를 들어 S_1에서 0XXX(X는 비트 충돌)가 수신되면, S_1은 표 1에 따라 b_1이 0이고 e가 1이므로 0001, 0010, 0100인 세 개의 비트 패턴을 가진 태그를 예측할 수 있다. 만약 S_1에서 00XX가 수신되면, 충돌이 발생한 비트의 위치에 따라 0001과 0010인 두 개의 질의를 생성한다. 이때 응답하지 않은 0100에 대한 질의는 생성하지 않는다. 비트 패턴에 따른 슬롯 할당에 의해 각 슬롯 내에서 충돌의 정확한 예측이 가능하다. 이에 따라 불필요한 질의를 생성하지 않는다. 이는 결국 태그가 무응답 하는 사이클을 생략하게 되므로 전체 태그 ID를 인식하는 시간이 줄어든다.

(그림 3)은 4-BPSA에서 리더와 태그의 동작 알고리즘을 흐름도로 나타낸 것이다.



(그림 3) 4-BPSA 알고리즘의 흐름도
(Figure 3) Flowchart of 4-BPSA algorithm

4-BPSA에서 한 사이클은 리더의 한 번의 질의와 매치되는 태그당 한 번의 응답으로 구성된다. 한 사이클에서 리더가 먼저 *prefix*를 질의한다. 리더가 질의한 *prefix*에 매치되는 각각의 태그는 식(1)에 따라 해당 타임 슬롯을 선택하고 응답한다. 리더는 각각의 타임 슬롯에서 수신된 $b_1b_2b_3b_4$에 대해 (표 1)에 따라 비트 패턴을 예측한다. 그리고 각각의 타임 슬롯에서 $b_1b_2b_3b_4$ 외에서 충돌이 있는지 검사한다. 만일 $b_1b_2b_3b_4$ 외에서 충돌이 없다면 태그를 인식한다. 이때 인식된 태그 아이디는 각각 '현재의 *prefix*' + 'predicted 4-bit' + 'residual bits'이다. 만일 $b_1b_2b_3b_4$

외에서 충돌이 있다면 해당 타임 슬롯에서 예측된 4-bit 패턴 각각에 대해 *prefix*를 확장한다. 이때 확장된 *prefix*는 각각 '현재의 *prefix*'+'predicted 4-bit'+'residual bits [0..up to first collided bit - 1]'이다.

기존 트리 기반의 알고리즘은 리더가 수신한 신호에서 1 비트라도 충돌이 있으면 *prefix*를 확장하여 재질의 한다. 충돌 없이 유일하게 1개의 태그가 응답할 때에만 하나의 태그 ID를 인식한다. 반면 4-BPSA 알고리즘은 4 비트 패턴에 따라 할당되는 타임 슬롯을 사용하기 때문에 각각의 타임 슬롯에서 $b_1b_2b_3b_4$에서만 충돌이 있고 $b_1b_2b_3b_4$ 외에서 충돌이 없다면 비트 패턴에 따른 정확한 예측에 따라 추가적인 재질의 없이 최대 16개의 태그 ID를 인식할 수 있다. 그리고 각각의 슬롯에서 $b_1b_2b_3b_4$ 외에서 충돌이 발생하여 충돌이 발생한 첫 비트 전까지 *prefix*를 확장하는 경우라도 $b_1b_2b_3b_4$에 대한 비트 패턴의 정확한 예측에 따라 불필요한 *prefix*는 생성하지 않으면서 *prefix*를 빠르게 확장한다. 이를 통해 4-BPSA 알고리즘은 불필요한 *prefix*의 확장을 줄이고 질의 횟수를 최소화함으로써 전체 태그 ID를 인식하는 시간을 줄인다.

4. 성능분석

본 절에서는 4-BPSA 알고리즘의 성능 분석을 위해서 worst-case의 시간 복잡도를 분석한다. 기존 트리 기반의 알고리즘은 충돌이 발생하면 *prefix*를 생성하여 재질의 한다. 4-BPSA 알고리즘은 $b_1b_2b_3b_4$ 외에서 충돌이 발생하면 *prefix*를 생성하여 재질의 한다. 따라서 전체 태그 ID를 인식하는데 걸리는 시간은 *prefix*의 생성 횟수에 비례한다. 4-BPSA 알고리즘에서 리더와 태그 간의 질의-응답은 *prefix* tree로 명명된 tree로 표현할 수 있다. 결국 4-BPSA의 worst-case는 *prefix* tree에서 최대 개수의 *prefix* node가 생성되는 경우라고 할 수 있다. 4-BPSA는 실제 존재하는 태그에 대해서만 질의를 한다. 따라서 4-BPSA가 전체 태그 ID를 인식하는데 필요한 *prefix* node의 개수는 태그 ID의 비트 길이와 상관없이 실재 존재하는 태그의 개수에 따라 결정된다.

전체 태그의 개수를 n이라 하고 높이를 h라고 하면 *prefix* tree에서 최대 개수의 *prefix* node가 생성되는 경우는 식(2)와 같다. 즉 h가 $\lfloor \log_{16}n-1 \rfloor$까지는 전 16진 트리(full 16-nary tree)를 형성하며 h가 $\lfloor \log_{16}n+1 \rfloor$까지 서브 트리(sub-tree)를 확장하는 경우이다. 4-BPSA는 한 번의 질의로 최대 16개의 *prefix* 노드를 생성한다. 따라서 트리의 h번째 높이에서 생성될 수 있는 최대 노드 개수는 16^h 개이다. 또한, 4-BPSA는 최소 2개의 태그가 충돌하는 경우 *prefix* node를 생성한다. 따라서 h번째 높이의 서브 트리(sub-tree)에서 생성될 수 있는 최대 노드 개수는 $\frac{n}{2}$ 개이다. 그러므로 4-BPSA가 n개 태그를 인식할 때 만들어진 *prefix* tree의 h번째 높이에서 *prefix* node의 최대 개수 $N_{prefix}(n, h)$는 다음과 같다.

$$N_{prefix}(n,h)=\begin{cases}16^h\,(0<h\le\lfloor\log_{16}n-1\rfloor)\\ \frac{n}{2}\;(\lfloor\log_{16}n-1\rfloor<h\le\lfloor\log_{16}n+1\rfloor)\end{cases} \quad (2)$$

식(2)을 이용하여 *prefix* tree에서 생성되는 *prefix* node의 총 개수 N_{prefix}를 계산하면 아래와 같다.

$$\begin{aligned}N_{prefix} &\le \sum_{h=1}^{\lfloor\log_{16}n+1\rfloor} N_{prefix}(n,h)\\ &= \sum_{h=1}^{\lfloor\log_{16}n-1\rfloor} 16^h + \sum_{h=\lfloor\log_{16}n\rfloor}^{\lfloor\log_{16}n+1\rfloor}\frac{n}{2}\\ &\le \frac{16}{15}(n-1)\end{aligned} \quad (3)$$

QT 알고리즘에서 태그 ID 길이가 k이고 n개의 태그가 존재하는 경우 만들어지는 *prefix* node의 개수 N_{QT}는 아래와 같다[16].

$$N_{QT}=n\times(k+2-\log_2 n)-3 \quad (4)$$

CT 알고리즘에서 n개의 태그를 인식하기 위한 *prefix* node의 개수 N_{CT}는 아래와 같다[7].

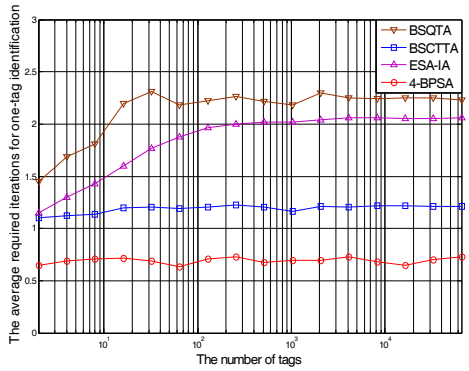
$$N_{CT}=2n-1 \quad (5)$$

4-BPSA가 n개의 태그를 각각 인식하기 위해 생성하는 *prefix* node의 개수는 worst-case의 경우 식(3)과 같다. 즉 4-BPSA가 worst-case에서 time complexity of O(n)임을 나타낸다. 이를 기존의 대표적 트리 기반 알고리즘인 식(4)의 QT 알고리즘과 식(5)의 CT 알고리즘에 비교하면 성능이 개선된 것을 알 수 있다.

5. 성능평가

본 논문에서 제안하는 4-BPSA 알고리즘의 성능 평가를 위해 MATLAB R2012b 버전에서 4-BPSA 알고리즘과 슬롯을 사용하는 트리 기반의 비교 대상 알고리즘들(BSQTA, BSCTTA, ESA-IA)의 핵심 프로토콜을 시뮬레이션 프로그램으로 작성하여 비교 분석하였다. 실험 환경은 하나의 리더와 다수 개의 태그들로 구성되고 태그들은 모두 리더의 인식 범위 내에 산재하며 태그 ID는 무작위로 부여된 것으로 설정하였다. 비교 대상 알고리즘들 간의 상대적인 식별 성능을 비교하기 위해 핵심 프로토콜 외에 나머지 하드웨어(HW) 및 소프트웨어(SW) 기능은 모두 동일한 조건으로 가정하였다. 이에 따라 HW에 별도의 추가 기능이 필요한 전송 중지 명령(ACK)은 시뮬레이션에서 제외 되었다[17]. 태그 ID의 길이는 128비트, 전송 속도는 128Kbps로 설정하였다[18]. 태그 개수는 2^1에서 2^{16}까지 증가시키면서 10회 반복 실험한 결과의 평균값을 가지고 성능을 비교하였다. 실험 결과 수집된 데이터에 대해 하나의 태그당 평균 질의 횟수, 리더와 태그 간의 평균 전송 비트 수, 초당 평균 식별 태그 수를 각 알고리즘 별로 비교하였다.

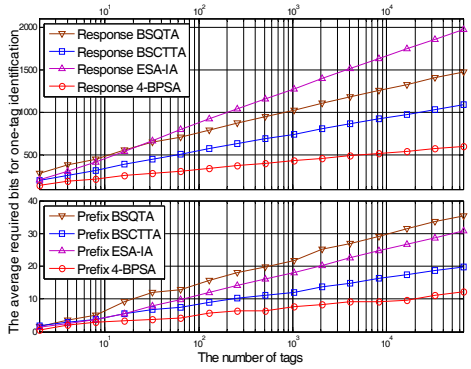
(그림 4)는 하나의 태그를 인식하는 데 필요한 평균 질의 횟수를 그래프로 나타낸 것이다.



(그림 4) 태그당 평균 질의 횟수
(Figure 4) The average required iteration for one-tag identification

(그림 4)를 보면 충돌 감지(collision detecting) 기법을 사용하는 QT 알고리즘을 기반으로 하는 BSQTA는 무응답 사이클(idle cycles)로 인해 태그당 질의 횟수가 가장 많은 것으로 나타났다. ESA-IA는 충돌 추적(collision tracking) 기법을 사용하며 2개 비트 충돌에 대해 예측을 수행함에도 불구하고 무응답 사이클(idle cycles)로 인해 성능 저하를 보였다. 이는 태그 ID 각 비트의 절대 위치와 상관없이 단순히 '1'의 개수에 따라 슬롯을 나누기 때문으로 분석된다. 충돌 추적(collision tracking) 기법을 사용하는 충돌 추적 트리(CTTA) 알고리즘을 기반으로 하는 BSCTTA는 하나의 태그당 평균 1.4회 질의하며 태그의 개수와 상관없이 평균적인 성능을 보이는 것으로 나타났다. 제안하는 4-BPSA는 충돌 추적(collision tracking) 기법을 사용하며 4비트 패턴에 대해 예측을 수행함으로써 하나의 태그당 평균 0.7회의 질의를 수행하며 태그의 개수와 상관없이 안정적인 성능을 보이는 것으로 나타났다.

(그림 5)는 하나의 태그를 인식하는 데 필요한 평균 전송 비트 수를 평균 질의 비트 수와 평균 응답 비트 수로 나누어 그래프로 나타낸 것이다. 그래프를 보면 전송 비트 수는 질의 횟수에 비례 한다는 것을 알 수 있다. 즉 질의 횟수가 증가할수록 전송 비트 수가 늘어나는 것이다. 이에 따라 비교 대상 알고리즘들에 비해 질의 횟수가 적은 4-BPSA의 평균 전송 비트 수가 가장 적은 것으로 나타났다.

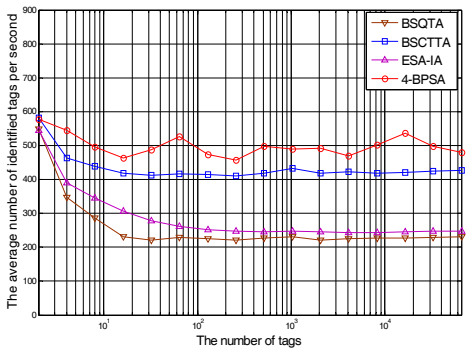


(그림 5) 태그당 평균 전송 비트 수
(Figure 5) The average required bits for one-tag identification

(그림 6)은 초당 평균 식별 태그 수를 그래프로 나타낸 것이다. 그래프를 보면 전송 비트 수는 질의 횟수에 반비례 한다는 것을 알 수 있다. 즉 질의 횟수가 감소할수록 식별 태그 수가 증가하는 것이다. 따라서 비교 대상 알고리즘들에 비해 질의 횟수가 적은 4-BPSA의 초당 평

균 식별 태그 수가 가장 많은 것으로 나타났다.

MATLAB을 이용하여 슬롯을 사용하는 트리 기반 알고리즘들의 핵심 프로토콜을 동일 조건에서 수행한 시뮬레이션 실험을 통해 수집된 데이터를 비교 분석한 결과에 의하면 제안하는 4-BPSA는 평균 0.7회의 질의 횟수를 보인다. 이는 비교 대상 알고리즘들 중에서 가장 좋은 성능을 보이는 BSCTTA의 평균 1.4회의 질의 횟수 보다 약 2배의 성능이 개선된 것이다. 이에 따라 4-BPSA 알고리즘은 비교 대상 알고리즘들에 비해 보다 적은 전송 비트 수를 태그 식별에 사용하며 초당 태그 식별 수에서 가장 좋은 성능을 보였다.



(그림 6) 초당 평균 식별 태그 수

(Figure 6) The average number of identified tags per second

6. 결 론

본 논문에서 제안하는 다중 태그의 고속 식별을 위한 4-BPSA 알고리즘은 충돌 추적(collision tracking) 기법과 4 비트 패턴을 이용한 정확한 예측을 통해 보다 빠르고 효율적으로 태그를 식별한다. 수학적 분석을 통한 알고리즘의 성능 분석 결과 4-BPSA 알고리즘은 worst-case에서 n개의 태그를 식별하기 위한 시간 복잡도가 $\frac{16}{15}(n-1)$ 이하인 것으로 나타났다. 이는 기존의 대표적 트리 기반 알고리즘에서 k비트의 태그 n개를 식별하기 위한 QT 알고리즘의 시간 복잡도 $n\times(k+2-\log_2 n)-3$ 와 CT 알고리즘의 시간 복잡도 $2n-1$에 비해 성능이 개선된 것이다. 또한 MATLAB을 이용한 시뮬레이션 실험을 통한 알고리즘의 성능 평가 결과 4-BPSA 알고리즘은 태그당 평균 0.7회의 질의를 수행하며 태그의 개수와 상관없이 안정적인 성능을 보이는 것으로 나타났다. 이는 비교 대상 알고리즘들 중에서 가장 좋은 성능을 보이는 BSCTTA의 평균 1.4회의 질의 횟수 보다 약 2배의 성능이 개선된 것이다. 이에 따라 4-BPSA 알고리즘은 비교 대상 알고리즘들에 비해 보다 적은 전송 비트 수를 태그 식별에 사용하며 초당 태그 식별 수에서 가장 좋은 성능을 보이는 것으로 나타났다.

향후에는 예측 비트의 개수와 타임 슬롯의 개수를 인식 상황에 따라 적응적으로 변경하는 방법에 대해 연구하고자 한다.

참고문헌(Reference)

[1] S. Sarma, D. Brock and D. Engels "Radio frequency identification and the electronic product code", IEEE Micro, Vol. 21, No. 6, pp.50-54, 2001.

[2] EPCglobal, Radio-Frequency Identity Protocols Class-1 Generation-2 UHF Protocol for Communications at 860 MHz-960MHz Version 1.1.0, Dec. 17, 2005.

[3] MIT Auto-ID Center, Draft protocol specification for a 900 MHz Class 0 Radio Frequency Identification Tag, http://auto-id.mit.edu, Feb. 23, 2003.

[4] J.H. Ryu, H.J. Lee, Y.H. Seok, T.K. Kwon, Y.H Choi, "A Hybrid Approach to Arbitrate Tag Collisions in RFID systems," Journal of KIISE, Vol. 34, No. 6, pp.483-492, 2007.

[5] C.H. Quan, W.K. Hong, Y.D. Lee, H.C. kim, "Performance Evaluation of Anti-collision Algorithms in the Low-cost RFID System," Journal of KICS, Vol. 30, No. 1B, pp.17-26, 2005.

[6] K. Finkenzeller, "RFID Handbook: Fundamentals and applications in Contactless Smart Cards and Identification," Second Edition, John Wiley & Sons Ltd, pp.195-219, 2003.

[7] X. Jia, Q. Feng, C Ma, "An Efficient Anti-Collision Protocol for RFID Tag Identification," IEEE Commun. Lett., Vol. 14, No. 11, pp.1014-1016, 2010.

[8] J.H. Choi, W.J Lee, "Comparative Evaluation of Probabilistic and Deterministic Tag Anti-collision Protocols for RFID Networks," EUC Workshops 2007, LNCS 4809, pp.538-549, 2007.

[9] S.W. Choi, J.H. Choi, J Yoo, “An Efficient Anti-Collision Protocol for Coping with the Capture Effect in RFID Tag Identification,” Journal of KIISE, Vol. 38, No. 6, pp.476-482, 2011.

[10] P.S. Jeong, W.S. Jung,C.Y. Yun, Y.H. Oh, “A Study on the Performance Improvement of Anti-Collision Algorithm for RFID,” Journal of KICS, Vol. 34, No. 6, pp.149-155, 2009.

[11] C.H. Quan, W.K Hong, Y.D. Lee, H.C. Kim, “A Study on the Tree based Memoryless Anti-Collision Algorithm for RFID Systems,” KIPS trans. Part C, Vol. 11C, No. 6, pp.851-862, 2004.

[12] Ching Law, Kayi Lee, Kai-Yeung Siu, “Efficient Memoryless Protocol for Tag Identification”, Conference of DIALM, ACM, pp.75-84, 2000.

[13] Feng Zhou, Dawei jin, Chenling Huang, Hao Min, “White Paper: optimize the power Consumption of Passive Electronic Tags for Anti-collision Schemes,” Auto-ID center Fudan Univ., October, 2003.

[14] J.H. Choi, D.W. Lee, H.J. Lee, “Bi-slotted tree based anti-collision protocols for fast tag identification in RFID systems,” IEEE Commun. Lett., Vol.10 No.12, pp.861-863, 2006.

[15] S.S KIM, Y.H. KIM, K.S. AHN, “An Inference Algorithm with Efficient Slot Allocation for RFID Tag Identification,” IEICE Trans. on Commun., Vol.E93-B, No.1, pp.170-173, 2010.

[16] J.H. Myung, W.J Lee, “Adaptive splitting protocols for RFID tag collision arbitration,” Conference of MOBIHOC, ACM, pp.202-213, 2006.

[17] H.G. Seo, “Collision Tree Based Anti-collision Algorithm in RFID System,” Journal of KIISE, Vol. 34, No. 5, pp.316-327, 2007.

[18] EPCTM Radio-Frequency Identity Protocols Class- 1 Generation-2 UHF RFID Protocol for Communications at 860MHz-960MHz Version 1.0.9, EPCglobal, January 2005.

◐ 저 자 소 개 ◑

김 영 백(Young Back Kim)

1996년 영남대학교 사회학과 (문학사)
2007년 경북대학교 대학원 컴퓨터공학과(공학석사)
2010년 경북대학교 대학원 컴퓨터공학과(박사수료)
2010년~현재 한국전자통신연구원 선임연구원
관심분야 : RFID, 자동차 IT, Ladar 시스템
E-mail : realtech@daegu.ac.kr

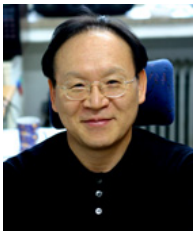
김 성 수(Sung Soo Kim)

2002년 금오공과대학교 컴퓨터공학과(공학사)
2005년 경북대학교 대학원 컴퓨터공학과(공학석사)
2012년 경북대학교 대학원 컴퓨터공학과(공학박사)
2013년~현재 경운대학교 모바일공학과 조교수
관심분야 : 임베디드 시스템, RFID, 센서 네트워크
E-mail : ninny@ikw.ac.kr

◐ 저 자 소 개 ◑

정 경 호(Kyung Ho Chung)

2000년 대구대학교 컴퓨터정보공학과 (공학사)
2002년 경북대학교 대학원 컴퓨터공학과(공학석사)
2011년 경북대학교 대학원 컴퓨터공학과(공학박사)
2013년~현재 경북대학교 컴퓨터학부 외래교수
관심분야 : 임베디드 시스템, RFID, 정보보호
E-mail : mccart@knu.ac.kr



안 광 선(Kwang Seon Ahn)

1972년 연세대학교 전기공학과(공학사)
1975년 연세대학교 대학원 전자공학과(공학석사)
1980년 연세대학교 대학원 전자공학과(공학박사)
1977년~현재 경북대학교 컴퓨터공학과 교수
관심분야 : 임베디드 및 디지털 시스템 설계
E-mail : gsahn@knu.ac.kr