韓國 컴퓨터情報學會 論文誌
第13卷 第5號, 2008. 9.

2008-13-5-4-6

RFID 시스템에서 추가 비트를 이용한 빠른 태그 예측 알고리즘

백 덕 화*, 김 성 수**, 안 광 선***

A Fast Tag Prediction Algorithm using Extra Bit in RFID System

Baek Deuk Hwa *, Kim Sung Soo **, Ahn Kwang Seon ***

요 약

RFID(무선 주파수 인식기술)은 무선 주파수를 사용하여 전자 태그를 자동으로 인식하는 기술이며, 인식영역 내의 모든 태그를 빠르게 인식하기 위하여 RFID 리더는 충돌 방지 알고리즘이 필요하다. 본 연구는 태그 충돌을 중재 하는 트리 기반의 TPAE(Tag Prediction Algorithm using Extra bit) 알고리즘을 제안한다. 제안한 알고리즘은 태그 아이디에서 모든 아이디 비트를 하나씩 식별하지 않아도 전체 태그를 식별할 수 있다. 리더는 태그 아이디에 추가 비트 정보를 이용한다. 만약 두 개나 다수 개의 비트에 충돌이 발생한다면, 추가 비트를 검사하여 '1'의 개수를 파악한다. 실험에서, 제안된 알고리즘은 태그 아이디 길이와 태그 개수에 상관없이 쿼리 트리 알고리즘과 이진 검색 알고리즘 보다 약 50 %의 질의 반복 횟수를 감소한다.

Abstract

RFID(Radio Frequency IDentification) is a technology that automatically identifies objects containing the electronic tags by using radio frequency. In RFID system, the reader needs the anti collision algorithm for fast identifying all of the tags in the interrogation zone. This paper proposes the tree based TPAE(Tag Prediction Algorithm using Extra bit) algorithm to arbitrate the tag collision. The proposed algorithm can identify tags without identifying all the bits in the tag ID. The reader uses the extra bit which is added to the tag ID and if there are two collided bits or multiple collided bits, it checks the extra bit and grasps the tag IDs concurrently. In the experiment, the proposed algorithm had about 50% less query iterations than query tree algorithm and binary search algorithm regardless of the number of tags and tag ID lengths.

▸ Keyword : RFID system(RFID 시스템), 충돌 방지(Anti collision), 태그 충돌(tag collision), 패리티 비트(parity bit)

• 제1저자 : 백덕화 교신저자 : 안광선
• 접수일 : 2008. 6. 27, 심사일 : 2008. 8. 21, 심사완료일: 2008. 9. 25.
* 창원전문대학 인터넷정보과 부교수 ** 경북대학교 컴퓨터공학과 박사과정 *** 경북대학교 컴퓨터공학과 교수

I. 서 론

RFID(Radio Frequency IDentification)는 RF 신호를 사용하여 상품에 부착된 전자 태그를 식별하여 상품 정보를 얻을 수 있는 비 접촉 식별기술이다. 또한 네트워크 및 센싱 기술과 함께 유비쿼터스 사회를 실현하기 위한 핵심 기술이다. RFID는 유통/물류뿐만 아니라 식품, 의료/약품, 도로/교통, 우정, 문화, 생산자동화, 소방/방재, 금융, 환경, 정보유통 등 다양한 분야에 적용이 가능하다[1][2].

RFID 시스템은 크게 리더와 태그로 구성된다. 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[3][4]. 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 식별 할 수 없게 한다. 따라서 최대한 충돌이 적게 발생하고, 충돌을 적절히 중재할 수 있는 충돌 방지 알고리즘이 필요하다.

본 논문에서는 하나의 리더가 다수의 태그를 식별 하는 다중 태그 환경에서 더 빠른 식별을 위해 TPAE(Tag Prediction Algorithm using Extra bit) 알고리즘을 제안한다. 기존의 알고리즘은 태그들 간 충돌이 발생하지 않도록 중재하여 특정시점에 하나의 태그만이 응답하도록 하여 모든 태그를 식별하는 반면, 추가 비트를 이용한 알고리즘은 식별 과정에서 발생하는 충돌 비트의 개수와 패리티(parity) 비트, 태그 아이디 내의 '1'의 개수를 사용하여 태그 아이디를 식별한다. 태그 아이디의 모든 비트 값을 확인하지 않고도 태그를 식별 할 수 있다.

II. 관련연구

2.1 쿼리 트리 알고리즘

쿼리 트리 알고리즘(Query Tree Algorithm)은 트리 기반의 충돌 방지 알고리즘으로 그 동작 방식이 간단하여 쉽게 구현할 수 있다는 장점이 있다. 쿼리 트리 알고리즘은 리더의 질의와 태그의 응답을 한 라운드(round)로 하여 라운드를 반복(iteration)하여 태그를 식별 한다[5]. 각 라운드에서 리더는 태그들의 아이디에 특정 프리픽스(prefix)를 포함하는지를 요청한다. 이때 프리픽스가 일치하는 태그들은 리더에게 응답을 하게 된다. 이때 하나 이상의 태그가 응답하면 리더는 현재 프리픽스를 가지는 태그가 두 개 이상인 것을 알게 된다. 그러면 리더는 '0' 또는 '1'을 현재 프리픽스에 추가하고 더 길어진 프리픽스를 다시 태그들에게 질의한다. 하나의 태그가 유일하게 응답하면, 하나의 태그를 식별한다. 즉 하나의 태그가 응답 할 때까지 프리픽스를 한 비트씩 추가하면서 영역 내의 모든 태그를 식별할 때까지 반복하게 된다. 그림 1은 쿼리 트리 알고리즘의 동작 과정을 나타낸다.

단계	1	2	3	4	5	6	7	8	9
리더→ 태그	질의 (ε)	질의 (0)	질의 (1)	질의 (00)	질의 (01)	질의 (10)	질의 (11)	질의 (000)	질의 (001)
태그 →리더	충돌	충돌	충돌	충돌	무응답	101	110	000	001
태그 1(000)	000	000		000				000	
태그 2(001)	001	001		001					001
태그 3(101)	101		101			101			
태그 4(110)	110		110				110		

그림 1. 쿼리 트리 알고리즘의 동작 과정
Fig 1. The operating process of query tree algorithm

그림 1에서는 4개의 태그가 리더의 질의에 따라서 응답하는 모습을 보여준다. 단계 1,2,3,4는 충돌이 발생하여 식별할 수 없다. 단계 6,7,8,9는 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.

2.2 이진 검색 알고리즘

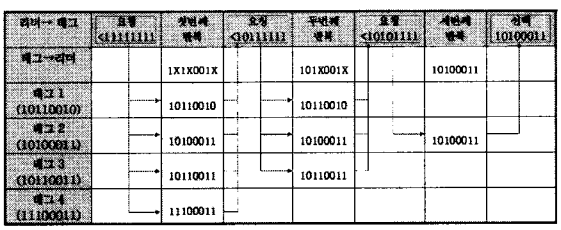


그림 2. 이진 검색 알고리즘의 동작 과정
Fig 2. The operating process of binary search algorithm

그림 2는 이진 검색 알고리즘을 사용하여 8 비트 태그들을 식별하는 과정을 보여준다. 이진 검색 알고리즘(Binary

Search Algorithm)은 쿼리 트리 알고리즘과 더불어 대표적인 트리 기반 알고리즘 중 하나이다[1]. 이 둘의 차이점은 쿼리 트리 알고리즘이 태그들 간의 충돌 발생 여부를 검사 하는데 반하여 이진 검색 알고리즘은 비트 간의 충돌 발생 여부를 감지 한다는 점이다. 그렇게 함으로써 충돌이 발생한 비트를 기준으로 태그를 반으로 줄여 식별 과정을 진행하게 된다[6]. 이진 검색 알고리즘의 시작은 리더가 요청(≦11111111)명령을 전송하면서 시작한다. 일련번호 '11111111'은 8 비트 태그가 가질 수 있는 가장 큰 값이다. 리더의 식별 영역 내의 태그들은 요청 명령에 인수로 받은 일련번호와 자신의 아이디를 비교하여 작거나 같은 아이디를 가지지는 태그는 리더에게 응답한다. 따라서 처음의 요청 명령에는 모든 태그가 응답하게 되고 이때 충돌이 발생하게 된다. 태그를 분할하기 위해서 충돌이 발생한 첫 번째 비트를 '0'으로 바꾼 후 다시 태그들에게 요청 명령을 보내게 되고, 이러한 과정이 하나의 태그만 응답할 때까지 반복된다. 하나의 태그가 응답하여 이를 식별한 후에는 식별된 태그는 비활성 상태로 전환하여 다음의 요청 명령에 더 이상 응답하지 않는다. 이와 같은 방법으로 많은 태그가 리더의 식별 영역 내에서 대기하고 있는 경우 개별 태그를 선택하기 위한 반복 횟수가 점차적으로 감소된다. 이진 검색 알고리즘의 신뢰성 있는 동작을 위해서는 모든 태그가 정확한 동기를 이루어 정확한 시각에 자신들의 일련번호 전송을 시작하여야 한다. 그래야만 충돌 시 정확한 비트 위치의 결정이 가능하다.

III. 추가 비트를 이용한 태그 예측 알고리즘

본 장에서는 인식영역 내의 모든 태그를 빠르게 식별할 수 있는 추가 비트(extra bit)를 이용한 빠른 태그 예측 알고리즘(TPAE)을 제안한다. 기존의 알고리즘이 태그를 식별하기 위하여 특정 시점에 하나의 태그만이 응답하도록 하는데 반하여, TPAE 알고리즘은 태그 간에 충돌이 발생하는 비트의 위치를 가지고 태그들을 식별한다. TPAE 알고리즘은 충돌이 발생한 비트의 수와 태그 아이디의 '1'의 개수를 분석하여 식별한다. 이렇게 함으로 TPAE 알고리즘은 태그 아이디의 모든 비트 값을 확인하지 않고도 예측을 통하여 태그 아이디를 식별할 수 있다.

3.1 맨체스터 부호화와 비트 간의 충돌 검출

리더는 태그 간에 충돌이 발생한 비트의 위치와 태그의 충돌 개수를 알기 위하여 맨체스터 부호화(manchester coding)를 사용한다. 맨체스터 부호화는 각 비트 구간에서 위상이 변하는 모양으로 비트 값을 결정한다[1][7].

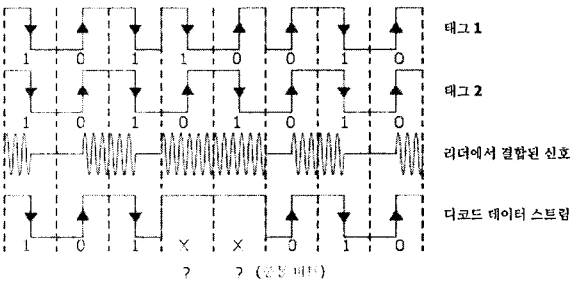


그림 3. 맨체스터 부호화를 통한 개별 비트의 충돌 검출
Fig 3. Collision behaviour for Manchester code

그림 3에서 태그1의 식별 코드는 "10110010"이고 태그2의 식별 코드가 "10101010"일 때, 리더의 요청 신호에 두 태그는 자신의 아이디를 리더에 전송한다. 두 태그 간의 위상이 다른 비트들(비트4와 비트5)은 양의 변화와 음의 변화가 서로 중첩되어 위상이 변하지 않는 상태가 한 비트 구간 동안 지속된다. 이러한 상태는 맨체스터 부호화 시스템에서는 허용되지 않으므로 오류로 판단하고, 충돌로 인식한다. 이처럼 맨체스터 부호화를 적용하면 각 비트 별로 충돌 여부를 검출 할 수 있다.

3.2 추가 비트

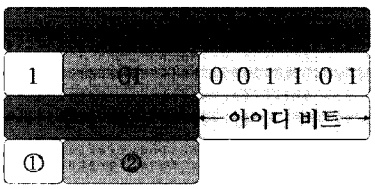


그림 4. 추가 비트를 가진 태그 아이디의 구조
Fig 4. Structure of tag ID with extra bit

제안한 알고리즘은 태그 아이디 내의 추가 비트와 아이디 비트가 포함한 '1'의 개수를 사용하여 식별한다.

그림 4는 제안한 알고리즘을 위한 태그 아이디의 구조를 보여준다. 제안한 알고리즘에서 태그 아이디는 추가 비트와 아이디 비트로 구성된다. 아이디 비트는 태그에 할당되는 고유한 아이디 번호를 저장하는 부분이다. 추가 비트에서 ②는 아이디 비트의 '1'의 개수를 2로 나눈 몫이며 ①은 나머지 값

이다. 그림 7의 예에서 아이디 비트 값은 001101이므로 ②는 몫인 '01'이며 ①은 '1'이다. 나머지가 되는 ①은 아이디 비트에 대한 짝수 패리티 값이 된다. 이 몫과 나머지를 이용하여 아이디 비트의 '1'의 개수를 알 수 있다.

추가 비트는 아이디 비트의 모든 비트가 '1'인 경우를 표현할 수 있어야 하므로, 추가 비트의 길이는 다음의 수식으로 구할 수 있다.

$$L_{extra\ bit} = \lfloor \log_2(L_{tagID} - L_{extra\_bit}) \rfloor + 1 \cdots\cdots\cdots (1)$$

L_{tagID}는 태그 아이디의 전체 길이를 나타내며, L_{extra_bit}는 추가 비트의 길이를 나타낸다. 예를 들어 태그 아이디의 총 길이가 9비트 일 때 수식 (1)을 만족하는 추가 비트의 길이(L_{extra_bit})는 3이며, 아이디 비트의 길이는 6이 된다. 일반적으로 다음과 같은 특성을 얻을 수 있다.

- 리더는 태그 아이디의 최상위 비트에서 최하위 비트로 인식을 진행하므로 패리티 비트(① 위치)를 제일 먼저 식별한다.
- 패리티 비트가 '0'인 그룹과 '1'인 그룹으로 나뉘어 태그의 인식 과정이 진행된다.

3.3 리더의 요청과 태그의 응답

제안한 알고리즘은 리더의 요청(request)과 태그의 응답(respond)을 반복적으로 수행함으로써 모든 태그를 식별한다. 리더는 태그들에게 프리픽스를 인수로 가지는 요청 신호를 전송한다. 태그는 수신한 프리픽스와 자신의 아이디 값과 일치하는 태그만이 리더에게 응답한다.

그림 5는 제안한 알고리즘에서 리더의 요청과 태그의 응답이 어떻게 진행 되는지를 보여준다. 첫 번째 반복에서 리더는 첫 번째 프리픽스 값인 엡실론(ε)을 태그들에게 보낸다. 엡실론(ε)은 모든 태그가 응답하게 된다. 리더는 두 번째 반복에서 프리픽스 값인 '0'을 전송하면 프리픽스가 일치하는 태그1과 태그2가 응답한다. 리더는 제안된 알고리즘을 이용하여 태그1과 태그2를 즉시 식별하게 된다. 세 번째 반복에서 프리픽스 값 '1'을 전송하면 프리픽스가 일치하는 태그3, 태그4, 태그5가 응답한다. 이 경우도 제안된 알고리즘을 이용하여 세 개의 태그를 한 번에 인식한다.

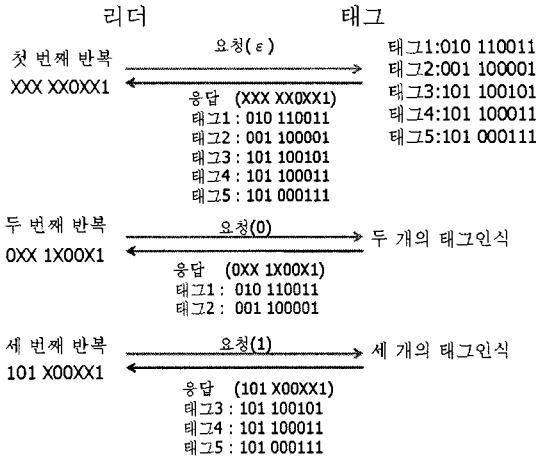


그림 5. 리더의 요청과 태그의 응답
Fig 5. Reader's request command and tag's response

3.4 식별방법

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

- N_{id} = 아이디 비트의 '1'의 개수
- N_p = 패리티 비트
- N_1 = 아이디 비트에서 확인된 '1'의 수
- N_c = 아이디 비트에서 충돌된 비트의 개수

새로운 변수 N_{1r}은 N_c에 포함되어야 하는 '1'의 개수(짝수 또는 홀수)를 의미한다. 리더는 위의 4가지 식별 정보를 가지고 N_{1r}을 구한다. N_c가 2인 경우는 N_p를 이용하여 충돌된 두 개의 태그를 식별한다. 또한 N_c 가 다수 개인 경우에 N_{id}를 이용하여 남아있는 '1'의 개수를 추측하며 다수 개의 태그가 식별된다. 리더가 태그를 식별 할 수 있는 경우는 다음의 두 가지로 구분된다.

1) 추가 비트에서 패리티 비트만 인식된 경우

- N_c = 0: 아이디 비트 내의 모든 비트들의 충돌이 없는 경우로서, 리더의 요청에 하나의 태그가 응답한 경우이므로 리더는 하나의 태그를 식별할 수 있다.
- N_c = 2: 아이디 비트 내의 충돌된 비트의 수가 2개인 경우로서, 식 (2)을 이용하여 태그를 식별한다.

$(N_1 + N_{1r}) \% 2 = N_p \quad (N_{1r} < 3)$ ····· (2)

N_{1r}이 1 인 경우

- 2개의 충돌 비트에 '0' 과 '1'인 태그 예측
- 2개의 충돌 비트에 '1' 과 '0'인 태그 예측

N_{1r}이 0, 2 인 경우

- 2개의 충돌 비트에 '0' 과 '0'인 태그 예측
- 2개의 충돌 비트에 '1' 과 '1'인 태그 예측

2) 추가 비트 전체를 인식한 경우

추가 비트 전체를 인식한 후에 N_{1r} 값은 식 (3)을 이용하여 구한다.

$$N_{1r} = N_{id} - N_1 \quad \cdots\cdots (3)$$

- $N_{1r} = 1$: 충돌이 발생한 비트에 하나의 '1' 과 $N_c - 1$ 개의 '0'을 포함하는 N_c 개의 태그 예측
- $N_{1r} = Nc - 1$: 충돌이 발생한 비트에 $N_c - 1$ 개의 '1'과 하나의 '0'을 포함하는 N_c 개의 태그 예측

위의 두 가지 경우인 1)과 2) 이외의 경우는 식별이 불가능한 경우이다. 식별이 불가능하면 리더는 첫 번째 충돌이 발생한 비트에 '0', '1'을 현재 프리픽스에 추가하고 더 길어진 프리픽스를 다시 태그들에게 질의한다. 식별이 가능할 때까지 이 과정을 반복한다.

그림 6은 제안한 알고리즘의 슈도코드를 나타낸다. 예를 들어, 추가 비트에서 최상위 한 비트인 패리티 비트만 인식된 경우에는 리더가 태그들의 응답으로 0X0101X0101X 신호를 받았다면, 이 경우 패리티 비트는 0이고($N_p = 0$), 충돌(X)이 발생한 비트의 수가 2($N_c = 2$)가 되며, 아이디 비트 내의 확인된 '1'의 개수는 3개($N_1 = 3$)가 된다. 지금까지 확인되지 않은 '1'의 개수($N_{1r} = 1$)는 1이다($N_1 + N_{1r}$) % 2 = N_p ($N_{1r} < 3$) → ($1 + N_{1r}$) % 2 = 0 ($N_{1r} < 3$).

리더는 N_c 에 하나의 '1' 만이 포함된 2개의 태그를 인식한다(000101001011, 000101101010).

추가 비트 전체를 인식한 경우에 리더가 태그들의 응답으로 10101X0X0X1X 신호를 받았다면, $N_{id} = 5$, $N_1 = 2$, $N_c = 4$, $N_{1r} = 3 = N_c - 1$이다($N_{1r} = N_{id} - N_1$ → 5 - 2). N_{1r}의 값은 $N_c - 1 = 3$ 이 된다.

이 경우는 N_c -1 개의 '1' 과 하나의 '0' 을 포함하는 N_c 개의 태그를 인식할 수 있다(101011010110, 101011010011, 101011000111, 101010010111).

```
Reader
Begin
Initially S = < ε >, M = < >
  Nid ← count_bit(IDbit); //extra_bit=parity+(number of 1's>>1)
  Nc ← collision_count(IDbit); // collision_count of IDbit
  N1 ← bitone_count(IDbit);// number of identified '1's to IDbit
  N1r ← Nc - N1; // number of remaining '1's to IDbit
  Np ← (N1 + N1r) % 2   (N1r < 3) // parity bit of IDbit
  Fcb ← first_collision_bit(receive_RESPONSE()); //first_collision_bit
  while(stack is not empty)
    S ← pop()
    Broadcast q
    If((Fcb< Nid) && (Nc == 2))  // collision_count of ID_bit = 2
      switch(N1r in ID_bit)
        case 1: put 0 and 1 on collision bit position of Idbit
                put 1 and 0 on collision bit position of Idbit
                save the tagIDs in memory M
                break
        case 0: put 0 and 0 on collision bit position of Idbit
                save the tagID in memory M
        case 2: put 1 and 1 on collision bit position of Idbit
                save the tagID in memory M
                break
    else If((Fcb>Nid)     // first_collision_bit > count_bit(IDbit)
      switch(N1r)
        case 1: (collision first bit position =! Collision end bit position)
                 put 1 on collision bit position and
                      puts 0 on remained collision bit position
                collision bit position increment
                save the tagIDs in memory M
                break
        case Nc-1:(collision first bit position =! Collision end bit position)
                  put 0 on collision bit position and
                       puts 1 on remained collision bit position
                 collision bit position increment
                 save the tagIDs in memory M
                 break
    else
      q1= W1W2... Wk-1 0, q2= W1W2... Wk-1 1
      S ← Push(q1, q2)
Tag
Let W= W1W2...Wk be the tagID
Begin
   if (q = ε or q = W1W2...Wlql)
      the tag sends string w to the reader
End
```

그림 6. 제안한 알고리즘의 슈도코드
Fig 6. Pseudo code of proposed algorithm

제안된 알고리즘을 사용하면 2개의 충돌과 다수개의 충돌에 대해 예측하여 태그를 식별할 수 있게 된다.

Ⅳ. 성능평가

본 장에서는 이 논문에서 제안한 알고리즘의 성능을 평가한다. 성능 평가는 시뮬레이션 프로그램을 사용한 실험을 통하여 진행하였고, 실제 무선 통신에서 존재하는 여러 제약이나 환경적 문제를 고려하지 않고 제안한 알고리즘이 얼마나 빠른 시간에 태그들을 식별할 수 있는지를 측정하였다.

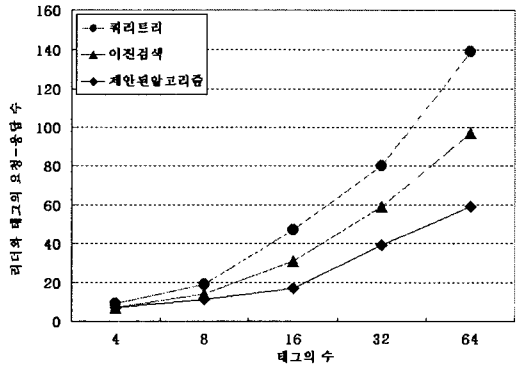
4.1 실험 환경과 성능평가 방법

시뮬레이션 프로그램은 사용자가 태그 아이디의 길이와 태그의 개수 그리고 태그 아이디를 할당하는 방법을 변경할 수 있도록 하였다. 실험에서 사용한 태그 아이디의 길이는 8 비트 일 때와 16 비트로 실험하였다. 각 경우는 다시 랜덤 할당 방법과 순차적인 할당 방법을 사용하였다. 태그의 개수를 변화시키면서 리더와 태그간의 요청과 응답횟수를 측정하였다. 리더의 요청과 태그의 응

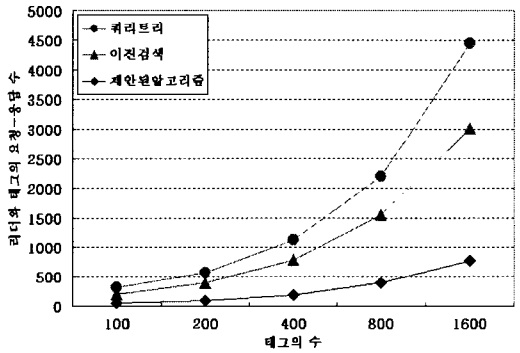
답횟수가 적을수록 식별 영역 내의 태그를 보다 빠른 시간에 식별할 수 있다는 것을 의미하므로 본 성능 평가에서는 시간의 개념과 같다고 간주한다. 실험에서 제안한 알고리즘의 성능을 비교 평가하기 위하여 쿼리 트리 알고리즘과 이진 검색 알고리즘을 비교하였다. 이때 제안한 알고리즘이 전체 아이디 중에서 추가 비트로 사용되는 특징이 있으므로 공정한 비교 평가를 위하여 제안한 알고리즘은 추가 비트만큼 긴 태그 아이디를 가진다. 즉 8 비트 아이디의 경우 4 비트가 추가된 12 비트 태그가, 16 비트의 경우 5 비트가 추가된 21 비트 태그를 사용하여 실험하였다. 즉 고유 식별 번호를 가지는 아이디 비트는 동일하게 사용하지만 제안한 알고리즘의 경우 추가 비트를 추가하여 실험을 진행하였다.

4.2 실험 결과 및 고찰

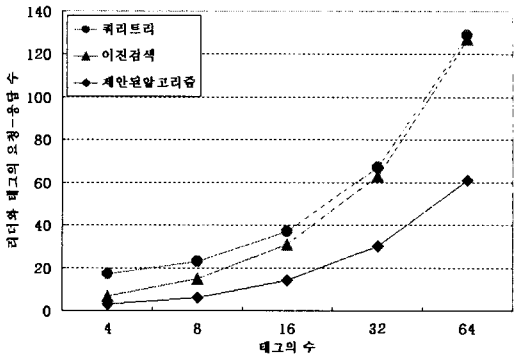
그림 7은 8 비트와 16 비트의 태그 아이디를 랜덤 할당한 경우이다. 그림 7(a)과 그림 7(b)에서 확인할 수 있듯이 태그 아이디의 길이에 상관없이 태그의 개수가 증가할수록 다른 알고리즘보다 우수한 성능을 보임을 확인 할 수 있다.



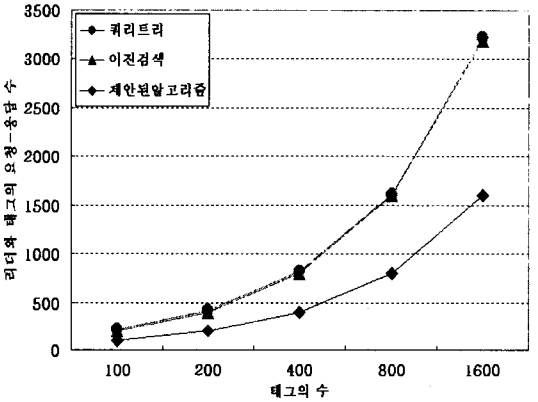
(a) 8 비트 태그 아이디의 랜덤 할당



(b) 16 비트 태그 아이디의 랜덤 할당
그림 7. 태그 아이디의 랜덤 할당
Fig 7. Random assignment of tag IDs



(a) 8 비트 시리얼 넘버의 순차적인 할당



(b) 16 비트 시리얼 넘버의 순차적인 할당
그림 8. 태그 아이디의 순차적인 할당
Fig 8. Sequential assignment of tag IDs

그림 8은 8 비트와 16 비트의 태그 아이디를 순차적인 할당의 경우로 그림 8(a)와 그림 8(b)에서 확인할 수 있듯이, 제안된 알고리즘이 우수한 성능을 보임을 알 수 있다. 즉, 제안된 알고리즘은 태그들의 아이디 길이와 개수가 증가할수록 좋은 성능을 보임을 확인 할 수 있다. 실험을 통하여, 제안된 알고리즘은 랜덤 할당의 경우 쿼리 트리 알고리즘 보다는 약 80%의 질의 반복 횟수를 감소하였고, 이진 검색 알고리즘 보다는 약 70%의 줄였다. 그리고 순차적인 할당 방법의 경우는 쿼리 트리 알고리즘과 이진 검색 알고리즘 보다 약 55 %의 질의 반복 횟수를 감소하였다.

V. 결론

RFID 시스템에서 다수의 태그를 식별하는 환경에서는 태그 충돌 문제가 발생한다. 충돌을 중재하고 더 빨리 모든 태

그를 식별하기 위해서는 충돌 방지 알고리즘이 필요하다.

본 논문에서는 추가 비트를 이용하여 식별 과정에서 충돌이 발생한 충돌비트의 개수와 태그 아이디 내의 '1'의 개수를 사용하여 태그 아이디를 식별한다. 제안한 알고리즘은 태그 아이디의 모든 비트 값을 확인하지 않고도 태그를 식별할 수 있다. 실험은 8 비트와 16 비트의 태그 아이디를 랜덤 할당 방법을 사용할 때와 순차 할당 방법을 사용한 경우를 나누어 실험 하였다. 랜덤 할당 방법을 사용한 실험에서는 태그 개수가 증가할수록 다른 알고리즘에 비해 제안된 알고리즘이 약 70 %의 질의 반복 횟수를 감소한다. 순차적인 할당 방법을 사용한 실험에서는 좀 더 좋은 성능을 보인다. 태그 아이디 길이와 태그 개수에 상관없이 쿼리 트리 알고리즘과 이진 검색 알고리즘 보다 약 50 %의 질의 반복 횟수를 감소한다.

향후 연구과제로는 제안한 알고리즘을 하드웨어로 구현하여 실제 동작 환경에서 성능을 평가하는 연구가 필요하다. 또한 표준으로 채택된 EPC 코드에 제안한 알고리즘의 태그 구조를 적용하는 연구도 계속 되어야 할 것이다.

참고문헌

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

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

[3] S. Sarma, D. Brock, and D. Engels, "Radio Frequency Identification and the Electronic Product Code," IEEE Micro, vol. 21, no. 6, pp. 50-54, November 2001.

[4] D. W. Engels and S. E. Sarma, "The Reader Collision Problem," In Proceedings of IEEE International Conference on System, Man and Cybernetics, Hammamet, Tunisie, October 2002.

[5] 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.

[6] MIT Auto-ID Center, "Draft protocol specification for a 900MHz Class 0 Radio Frequency Identification Tag," 2003.

[7] L. Liu, Z. Xie, J. Xi, and S. Lai, "An Improved Anticollision Algorithm in RFID System," Mobile Technology, Applications and Systems, 2nd International Conference, 2005.

저 자 소 개

백 덕 화

1988년 2월 : 경북대학교 전자공학과(학사)
1990년 2월 : 경북대학교 대학원전자계산기공학과(석사)
1992년 2월 : 경북대학교 대학원컴퓨터공학과(박사수료)
1991년 ~ 현재 : 창원전문대학 인터넷정보과 부교수
관심분야 : RFID, 정보통신, 모바일 인터넷

김 성 수

2002년 2월 : 금오공과대학교 컴퓨터공학과(학사)
2005년 2월 : 경북대학교 대학원컴퓨터공학과(석사)
2008년 2월 : 경북대학교 대학원컴퓨터공학과(박사수료)
2008년 ~ 현재 : 경북대학교 대학원 컴퓨터공학과 박사과정
관심분야 : RFID, 임베디드 시스템, 센서네트워크

안 광 선

1972년 2월 : 연세대학교 전기공학과(학사)
1980년 2월 : 연세대학교 전자공학과(박사)
1975년 ~ 1976년 : 스페라유니백 근무
1977년 ~ 현재 : 경북대학교 공과대학 컴퓨터공학과 교수
관심분야 : RFID, 임베디드 시스템, 센서네트워크