무선통신 프로토콜의 충돌방지 알고리즘 개선

*임정현 * 이권익 *양두영 *김흥수 **고재권

*제주대학교 **미래정보통신

*he2guy@cheju.ac.kr

Improvement of Anti-collision Algorithm for RF Air-interface Protocol

*Lim, Jung-Hyun * Lee, Kwoun-Ig *Yang, Doo-Yeong *Kim, Heung-Soo **Ko, Jae-Kweon

*Cheju National University **Mirae Telecom

요약

본 논문은 유비쿼터스 센서 네트워크와 텔레매틱스 서비스에 사용되는 무선 환경 프로토콜 표준 중 UHF 대역 프로토콜인 EPCglobal의 Class 1 Gen 1, Class 1 Gen 2의 에어인터페이스를 분석하고 각 프로토콜에서 규정된 충돌방지 알고리즘을 구현하였다. 그리고 각 프로토콜에서 규정하는 충돌방지 알고리즘의 성능을 개선한 알고리즘을 제안하고 성능을 비교하였다.

개선된 알고리즘과 표준 알고리즘의 성능 시뮬레이션 결과 개선된 Class 1 Gen 1 알고리즘은 태그 수 100개 일 때 54.5%, 태그 수 1000개 일 때 63.4% 감소하였다. 개선된 Class 1 Gen 2 알고리즘은 태그 수 100개 일 때 7.9%, 태그 수 1000개 일 때 11.7% 감소하였다. 개선된 충돌방지 알고리즘은 표준 알고리즘보다 태그 인식 성능이 상당히 개선됨을 알 수 있다. 따라서 개선된 충돌방지 알고리즘은 유비쿼터스 센서 네트워크와 텔레매틱스 서비스 등에 사용되는 무선인식 시스템의 성능 개선 방안으로 적용될 수 있다.

1. 서론

무선인식 시스템은 무선주파수 신호를 이용하여 태그에 부착된 IC칩에 저장되어 있는 고유 정보를 비접촉식으로 판독하는 식별장치로 기존의 바코드나 자기인식 장치의 단점을 보완하고 사용의 편리성을 향상시켜 줄 차세대의 유비쿼터스 센서 네트워크(USN)와 텔레매틱스 서비스 구현에 있어 가장 핵심적인 기술이다. 뿐만 아니라 보안 솔루션과 관련하여 무선 LAN과 무선인식 기술을 통합하는 과정도 고려해야 한다. 이러한 무선인식 시스템은 태그, 리더, 미들웨어 및 응용 서비스 플랫폼으로 구성된다[1].

무선인식 시스템에서 리더는 인식영역 내의 태그에 요청 메시지를 전송하고, 요청 메시지를 수신한 태그는 자신의 정보를 리더로 전송한다. 그러나 인식영역 내에 다수의 태그가 존재하여 동시에 요청 메시지에 대한 응답을 하게 되면 RF 통신 채널 상에 충돌(collision)이 발생하게 되고, 결과적으로 리더는 태그의 정보를 정확하게 수신하지 못하게 된다. 이러한 문제를 해결하기 위해 충돌방지 알고리즘(anti-collision algorithm)이 사용되며, 이는 다중접속 방식 시스템의 성능을 결정짓는 중요한 요소가 된다. 무선인식 시스템의 성능은 태그를 인식하는데 필요한 시간과 태그가 소모하는 전력으로 결정되어진다. 다중접속 방식을 갖는 무선인식 시스템에서 태그 데이터 간의 충돌은 인식시간과 소모 전력의 증가를 가져오는 직접적인 원인이 되므로 무선인식 시스템의 효율성을 높이기 위해서는 태그 충돌을 최소화 하는 다중접속 기술이 필요하다.

충돌방지 알고리즘은 크게 결정적 알고리즘(deterministic algorithm)과 확률적 알고리즘(probability algorithm)으로 구분된다. 결정적 알고리즘은 이진탐색(binary search)을 기반으로 한 것이고, 확률적 알고리즘은 알로하(ALOHA)를 기반으로 한다. 이러한 리더와 태그간의 통신 방식은 여러 프로토콜로 정의되어 있다. 무선인식에 관한 하드웨어 규정과 리더와 태그간의 통신방식 등을 제정하는 표준은 ISO 18000, EPCglobal, U-센터 등 여러 단체에서 진행되고 있다.

본 논문에서는 충돌방지 알고리즘의 성능을 개선하기 위하여 UHF 대역 프로토콜로 규정된 EPCglobal의 Class 1 Gen 1, Class 1 Gen 2의 리더와 태그사이의 통신 방식과 태그 인식 과정을 분석한다. 그리고 각 프로토콜에서 규정된 충돌방지 알고리즘인 빈슬롯, 슬롯알로하 알고리즘의 성능을 C언어로 구현하여 확인한다. 그리고 분석된 결과를 바탕으로 리더와 태그 간의 송수신되는 명령어를 줄이거나 태그 응답 타이밍을 줄이는 등의 방식을 사용하여 각 알고리즘의 성능을 개선하고 그 성능을 고찰한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 EPC Class-1 에어인터페이스에 대해 설명하고, 3장에서는 각 프로토콜에서 규정한 충돌방지 알고리즘의 개선 방안을 제시하고, 4장에서는 시뮬레이션 결과와 리더 시스템에서의 충돌방지 알고리즘 구현에 대하여 논의하며, 5장에서는 결론을 내린다.

2. 클래스-1 충돌방지 알고리즘

가. C1 G1

EPC Class 1 Gen 1 에서 리더와 태그의 통신은 패킷 단위로 이루어진다[2]. 리더에서 태그로 보내는 패킷은 완전한 명령을, 태그에서 리더로 보내는 패킷은 완전한 응답을 포함한다. 리더가 태그로 전송하는 명령어는 8개의 필드와 5개의 패리티비트 필드로 구성되고, 그 형식은 [PREAMBLE][CLKSYNC][SOF][CMD][P1][PTR][P2][LEN][P3][VALUE] [P4][P5][EOF] 이다.

태그는 리더 명령어를 수신하면 응답을 하거나, 상태를 변화 시킨다. 태그가 리더 동작 영역으로 들어가게 되어 적절한 파워의 리더 신호를 수신하면 태그는 모든 상태로 변화 가능한 Awake 상태가 된다. 태그는 ScrollID, ScrollAllID, VerifyID, PingID 명령어를 수신하면 Reply 상태가 되어 리더에게 응답 신호를 전송한다. 완전히 식별된 태그의 신호는 다른 태그를 인식하는 동안 방해가 되기 때문에 리더는 식별된 태그에 Quiet 명령어를 전송하고 수신한 태그는 잠시 응답하지 않는 Asleep 상태가 되며, 그 반대 과정은 Talk 명령어를 사용한다. 완전히 인식된 태그 중에 프라이버시 등의 문제로 제거되어야 할 때 리더는 그 태그에게 Kill 명령어를 보내고 태그는 암호와 비교하여 일치하면 Dead 상태가 된다.

Class 1 Gen 1의 충돌방지 알고리즘은 이진트리에 기반을 둔다. 이진트리는 원래 1비트 단위로 트리를 확장하는 형태이지만, Class 1 Gen 1의 빈 슬롯(bin slot) 충돌방지 알고리즘은 3비트 단위로 확장하는 형태이다. Class 1 Gen 1 에서 태그 인식은 리더가 'VALUE=0, LEN=0'을 포함하는 PingID 명령어를 인식영역 내의 임의의 태그에 전송함으로써 시작된다. PingID 명령어를 전송한 후 리더는 Bin Modulation 작업을 하고 태그의 비트열이 Bin 값과 일치하는 태그가 각 Bin에서 응답을 한다. Bin을 검사한 후 각 Bin에서 충돌 여부를 판단하고 충돌이 일어나지 않으면, 리더는 Bin 값을 [VALUE]에 포함한 후 ScrollID 명령어를 태그에 보내게 된다. [VALUE]에 맞는 태그는 자신의 전체 ID를 ScrollID 명령어에 대한 응답으로 리더에 보내게 되고, 리더는 CRC체크를 하여 오류 없이 전송 되었다면 태그 식별 코드를 저장하고 과정을 처음으로 되돌린다. 만약, Bin에서 충돌이 일어나면 리더는 Bin 값을 [VALUE]에 저장한 후 다시 PingID 명령어를 리더 인식영역 내에 전송한다. 여기서, Bin 검사는 Bin의 처음 '000'부터 시작한다.

나. C1 G2

EPC Class 1 Gen 2의 태그 인식은 슬롯 알로하(slotted aloha) 방식을 사용한다[3]. 인식영역 내의 태그를 인식하기 위해서 리더는 기본적으로 SELECT, INVENTORY 그리고, ACCESS의 세 동작과정을 이용한다. SELECT 과정에서는 INVENTORY 과정에 앞서, 통신할 특정 태그를 리더가 선택하는 과정, INVENTORY 과정은 슬롯을 발생시켜 하나 이상의 태그가 응답하게 하고 응답한 태그 중에서 하나의 태그 인식에 필요한 PC, EPC, CRC-16을 요청하는 과정, ACCESS 과정은 INVENTORY 과정을 성공적으로 마친 태그와 리더가 일-대-일 통신을 하는 과정이다. INVENTORY 과정은 Query, QueryAdjust, QueryRep, ACK, 그리고 NAK 등의 명령어들로 구성된다. Query는 인벤토리 프레임의 슬롯 개수를 초기화하고 태그가 프레임에 참여하게끔 한다. 프레임에 참여한 태그는 Query 명령어에 포함되어 있는 슬롯 파라미터 Q를 참조하여 랜덤넘버(random number)를 선택하고 이를 슬롯카운터(slot_counter)에 로드한다. 여기서 '0'을 선택한 태그들은 Reply 상태로 전환되고, 전환되는 즉시 리더로 응답 메시지를 전송한다. 그 밖의 '0'이 아닌 랜덤넘버를 선택한 태그들은 Arbitrate 상태로 전환되고 QueryAdjust 또는 QueryRep 명령어를 기다린다.

하나의 슬롯에 대해 단일 태그가 응답으로 RN16(random number of 16-bit)을 리더로 전송했다면, 충돌이 발생하지 않는다. 이때 리더는 수신한 RN16을 포함하는 ACK 명령어를 태그로 전송한다. 이 과정을 통해 리더에 인식된 태그는 Acknowledge 상태로 전환되고, PC, EPC, 그리고 CRC-16을 리더에 전송한다. 리더는 QueryAdjust 또는 QueryRep 명령어를 전송하여 태그를 대기상태로 전환하고 태그 데이터를 메모리에 저장한다. 만약 일정시간동안 태그가 ACK 수신에 실패하거나 잘못된 ACK를 수신하면 태그는 다시 Arbitrate 상태로 전환되어 다음 프레임까지 대기하게 된다.

프레임을 초기화시키는 Query가 발생한 후에, 리더는 하나 이상의 QueryAdjust 또는 QueryRep 명령어를 전송한다. QueryAdjust 명령어는 이전 Query를 반복하고 Q 값을 증감시킨다. QueryRep 명령어는 Q와 같은 파라미터들을 변화시키지 않고 Query를 반복한다. 이때 새로운 태그를 프레임에 추가시키지는 않는다. 어떤 시점에서 리더는 새로운 Query 명령어를 전송하는데, 그로인해 새로운 프레임이 시작된다. Arbitrate나 Reply 상태에 있던 태그는 처음으로 QueryAdjust 명령어를 수신하고 Q를 조정한다. 그래서 랜덤넘버를 선택하고 랜덤넘버는 그들의 슬롯카운터로 로드된다. 여기서 Arbitrate 상태에 있는 태그들은 그들의 슬롯카운터를 QueryRep 명령어를 수신할 때마다 감소시키고 슬롯카운터가 [0000h]에 도달했을 때 Reply 상태로 전환되어 RN16을 리더에 전송한다. 이런 과정을 반복적으로 수행함으로써 태그를 인식할 수 있다.

3. 개선된 클래스-1 충돌방지 알고리즘

가. C1 G1

본 논문에서는 기존의 Class 1 충돌방지 알고리즘에서 스택 개념을 도입하고 PingID 명령어에 대한 태그의 응답이 8비트인 점을 착안하여, 충돌이 일어나지 않은 Bin을 우선 인식하고, 충돌이 일어난 Bin은 응답으로 수신된 8비트열들을 서로 비교하여 일치하는 비트열을 스택에 저장하는 방법이다. 하나의 태그를 완벽히 인식한 후, 다시 'VALUE=0, LEN=0'으로 하여 PingID 명령어를 보내는 것이 아니라 우선 스택을 검사하여 저장된 데이터가 있으면 그 값을 [VALUE]와 [LEN]에 저장한 후 PingID 명령어를 보내고, 스택에 저장된 데이터가 없으면 다시 'VALUE=0, LEN=0'으로 하여 PingID 명령어를 보내는 형태이다.

그림 1은 개선된 EPC Class 1 Gen 1 알고리즘의 태그 인식 과정을 나타낸 것이다. Class 1 Gen 1 에서 태그 인식은 리더가 'VALUE=0, LEN=0'을 포함하는 PingID 명령어를 인식영역 내의 임의의 태그에 전송함으로써 시작된다. PingID 명령어를 전송한 후 리더는 Bin Modulation 작업을 하고 태그는 자신의 비트열과 Bin이 일치하는 각 Bin에서 자신의 ID중 8비트를 응답으로 전송한다. 이때 리더는 각 Bin에서의 태그의 응답을 임시 메모리에 저장한다. 리더는 첫 Bin(000)부터 마지막 Bin(111) 까지 순서대로 Bin 검사를 한다. 프로토콜에서는 앞의 Bin에서 태그의 응답이 있으면 그 이후의 Bin에 대해서는 처리를 하지 않는 반면, 개선된 알고리즘에서는 임시메모리를 사용하여 각 Bin에서의 응답을 저장했기 때문에 8개의 Bin에서의 태그 응답을 처리할 수 있다. 각 Bin에서 충돌 여부를 판단하고 충돌이 일어나지 않으면, 리더는 Bin 값을 [VALUE]에 포함한 후 ScrollID 명령어를 태그에 보내게 된다. [VALUE] 값과 자신의 비트열이 일치하는 태그는 자신의 전체 ID를 ScrollID 명령어에 대한 응답으로 리더에 보내게 되고, 리더는 CRC체크를 하여 오류 없이 전송 되었다면 태그 식별 코드를 저장한다. 충돌이 발생하면, 리더는 응답으로 들어온 8비트열들을 비교하고, 일치하는 비트열까지 스택에 저장한다. 그리고 난 후 다음 충돌이 없는 Bin에서의 태그를 먼저 인식한다. 8개의 Bin 검사가

끝난 후, 리더는 스택을 검사하고, 스택에 저장된 데이터가 있으면 그 데이터를 [VALUE]에 포함한 PingID 명령어를 전송한다. 만약 스택에 저장된 데이터가 없다면, 리더는 'VALUE=0, LEN=0' 인 PingID 명령어를 전송한다.

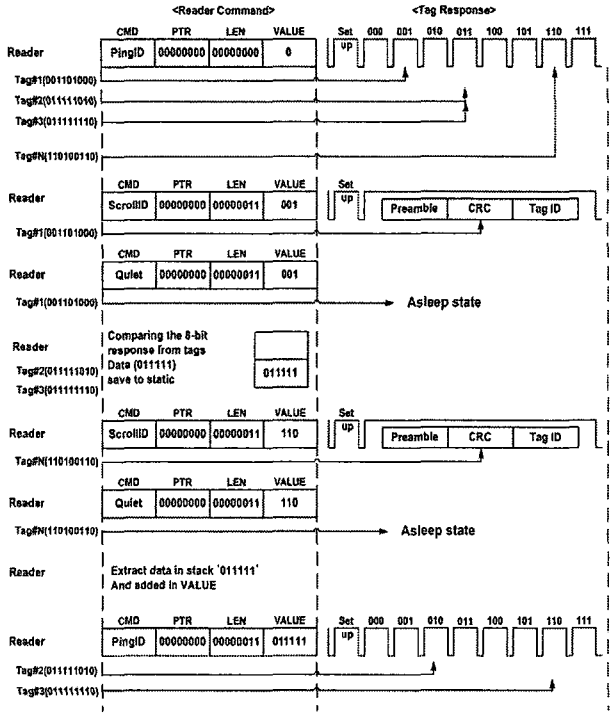


그림 1. 개선된 C1 G1 알고리즘의 태그 인식 과정 예시

나. C1 G2

기존의 Class 1 Gen 2 충돌방지 알고리즘에서는 태그의 수가 슬롯 수보다 많아 충돌이 많이 발생하거나, 태그의 수가 슬롯 수보다 적어 슬롯의 낭비가 있을 경우 리더는 Query_Adjust 명령어를 사용하여 Q 값을 1씩 증감하여 슬롯의 개수를 조절한다. 본 논문에서는 Q 값을 1씩 증감시키지 않고, 태그 수를 추정하여 그에 맞는 Q 값을 사용한다. 그리고 Class 1 Gen 2에서 충돌이 발생한 슬롯의 시간이 응답이 없는 idle 슬롯의 시간보다 길기 때문에 충돌이 발생하는 슬롯을 줄이기 위하여 예측한 태그 값보다 큰 Q 값을 사용하였다. 예를 들어 예측한 태그 수가 72일 경우 Q 값을 6과 7중에 7을 선택 하여 슬롯 개수를 128개로 하여 충돌을 최소화하는 방법을 사용하였다.

그림 2는 Class 1 Gen 2의 동작 과정과 본 논문에서 사용된 Q-선택 과정을 나타낸 그림이다. Query나 Query_Adjust 명령어에 대한 한 라운드가 종료된 후에 다음 라운드의 사이즈를 결정하게 된다. 이때 충돌이 발생한 슬롯 수를 세어 현재 남아 있는 태그 수를 예측 한다. i 번째 라운드에서 프레임 사이즈가 $N(i)$, 태그의 개수가 $n(i)$, 충돌이 발생한 슬롯 수 C_{slot}, 성공한 슬롯 수 S_{slot}, 빈 슬롯 수는 E_{slot}라고 한다. 충돌 슬롯이 발생할 확률, 성공 슬롯이 발생할 확률, 빈 슬롯이 발생할 확률은 모두 이항분포를 따른다[4].

$$P_q(i)=\binom{n(i)}{q}\left(\frac{1}{N(i)}\right)^q\left(\frac{N(i)-1}{N(i)}\right)^{n(i)-q} \quad (1)$$

여기서, $q=0$ 일 때 는 태그의 응답이 없는 빈 슬롯의 확률, $q=1$ 일 때 은 성공적으로 전송될 확률, 나머지($q>1$)는 충돌이 발생할 확률이다. 현재 i번째 라운드에서 예측된 태그 수는 성공한 슬롯 수와 충돌이 발생한 슬롯 수를 사용하여 다음과 같이 쓸 수 있다.

$$n_{est}=(P_1(i)\times N(i))+\left(\sum_{q=2}^{N} qP_q(i)\right)\times C_{slot}=S_{slot}+K\times C_{slot} \quad (2)$$

$$K=\lim_{N\to\infty}\sum_{q=2}^{N} qP_q(i)=2.39 \quad (3)$$

따라서 다음 라운드에 참여하는 태그 수는 충돌이 발생한 태그이므로, 다음 라운드 크기는 다음과 같다.

$$N(i+1)=H\times(K\times C_{slot}),\ \begin{cases} H=1+\frac{1}{K}, & Static\ frame \\ H=1-\frac{1}{K}, & Dynamic\ frame \end{cases} \quad (4)$$

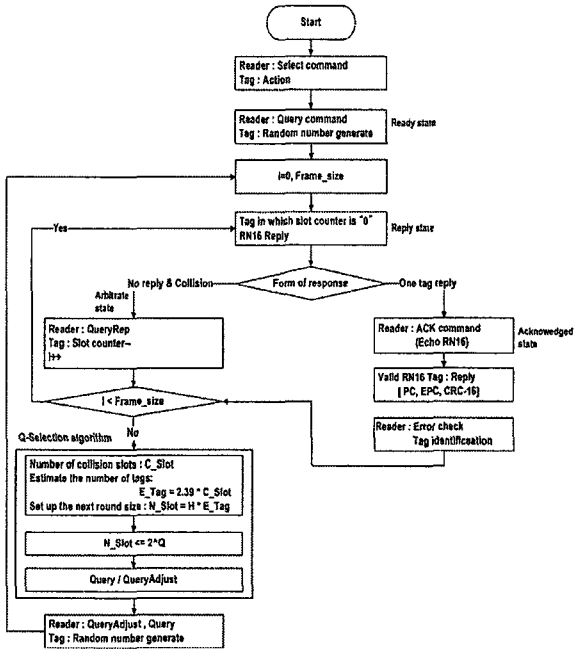


그림 2. 개선된 C1 G2 알고리즘의 흐름도

그림 2는 이를 적용한 개선된 EPC Class 1 Gen 2 알고리즘의 흐름도를 나타낸 것이다. 태그 ID를 인식하는 INVENTORY 과정에 앞서 통신할 특정 태그를 선택하는 Select 명령어를 전송한다. 리더는 프레임의 슬롯 개수를 초기화하고, 태그가 프레임에 참여하게 하는 Query 명령어를 전송한다. 이 명령어를 수신한 태그는 자신의 상태를 Ready 상태로 전환하고, Query 명령어에 포함된 Q-파라미터를 사용하여 ($0\sim 2^Q-1$) 범위에서 자신의 슬롯넘버(RN16)를 선택하고 이를 슬롯카운터로 로드한다. 여기서 '0'을 선택한 태그들은 Reply 상태로 전환되고, 전환되는 즉시 리더로 자신의 RN16을 응답 메시지를 전송한다. 그 밖의 '0'이 아닌 랜덤넘버를 선택한 태그들은 Arbitrate 상태로 전환되고 QueryAdjust 또는 QueryRep 명령어를 기다린다. 만약 그 응답유형이 하나의 태그만 응답 하였다면, 리더는 수신한 RN16을 포함하는 ACK 명령어를 태그로 전송한다. ACK 명령어에 포함된 RN16과 자신의 RN16이 일치하는 태그는 Acknowledge 상태로 전환되고, [PC, EPC, CRC]를 응답으로 리더에 전송한다. 만약 그 응답유형이 충돌이거나 응답이 없다면, 리더는 QueryRep 명령어를 전송하고,

이 명령어를 수신한 태그는 자신의 슬롯카운터 값을 1씩 감소하고, 다시 이 값이 '0'인 태그는 Reply 상태로 전환되어 자신의 RN16을 응답으로 전송한다. 이 과정을 프레임 사이즈만큼 반복하게 되고, 만약 하나의 프레임이 종료되면, 리더는 인식영역 내에 인식되지 않고 남아 있는 태그의 개수를 추정한다. 식 (2)와 식 (4)와 같이 충돌이 발생한 슬롯 수를 사용하여 태그의 개수를 추정하고, 충돌을 최소화하기 위하여 예측한 태그 값보다 큰 Q 값을 설정한다. 재설정한 Q 값을 포함하는 QueryAdjust 명령어를 전송하여 위의 과정을 반복한다.

4. 시뮬레이션 결과

표 1은 Class 1 Gen 1 충돌방지 알고리즘에서 리더가 태그로 전송하는 총 명령어 수와 명령어 포맷에 포함되는 [VALUE]필드의 길이를 나타낸 것이다. 태그 인식에 필요한 PingID, ScrollID, Quiet 명령어 수이다. 만약 PingID 명령어, ScrollID 명령어, Quiet 명령어를 여기서 N_P N_S N_Q라 하고, 1개당 평균 [VALUE]길이를 각각 VLP, VLS, $VLQ(=VLS)$, 각 명령어 1개당 전송 시간을 T_R, T_S, T_Q이라하면, 태그의 총 인식시간 T_{total}은 다음과 같이 나타낼 수 있다.

$$
\begin{aligned}
T_{P1} &= 1.25T_0 + 64\mu s + 51T_0 + [VLP]T_0 + 8T_0 + 64T_0 + 2.5T_0 \\
T_{S1} &= 1.25T_0 + 64\mu s + 51T_0 + [VLS]T_0 + 8T_0 + [tagbit]T_0 + 2.5T_0 \\
T_{Q1} &= 1.25T_0 + 64\mu s + 51T_0 + [VLS]T_0 + 8T_0 + 2.5T_0 \\
T_{Total} &= N_P T_{P1} + N_S T_{S1} + N_Q T_{Q1}
\end{aligned} \tag{5}
$$

태그의 총 인식 시간은 프로토콜에서 규정된 링크타이밍을 적용하여 계산하였고, T_0는 북미방식에서 규정된 14.25 μs을 기준으로 계산하였다. 리더에서 태그로 전송하는 신호는 첫 번째 단계 transaction gap 1.25 T_0, 두 번째 단계 CW RF신호는 64 μs, 세 번째 단계 data modulation 부분, 네 번째 단계 TAG Setup 8 T_0, 마지막 단계인 태그 응답 전송부분은 64 T_0, 다음 transaction gap과의 인터벌 2.5 T_0이다. 세 번째 단계인 data modulation부분은 리더 명령어 포맷에서 [VALUE] 길이를 제외한 51비트에 표 1에 주어진 VLP, VLS을 더하여 계산하였다.

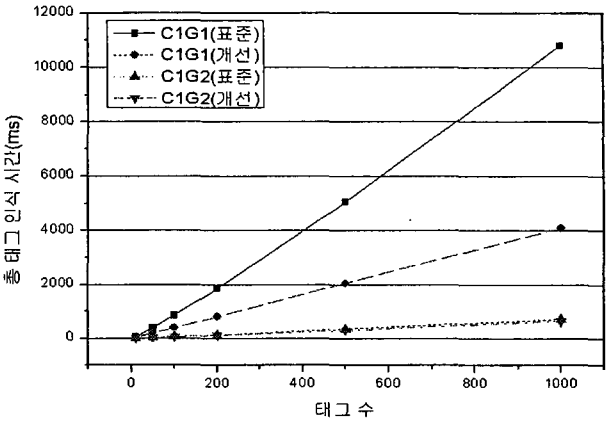
Class 1 Gen 2 알고리즘의 태그 인식 시간은 프로토콜에서 규정된 각 명령어와 태그 응답에 대한 타이밍을 규정한 것이다. 이 링크타이밍을 적용하여 계산하였다. 표 2와 그림 3은 각 알고리즘의 총 태그 인식 시간을 나타낸 것이다.

표 1. C1 G1 충돌방지 알고리즘의 결과

태그 수	Class 1 Gen 1 표준			Class 1 Gen 1 개선		
	Total command	VLP	VLS	Total command	VLP	VLS
10	37.0	1.41	5.10	24.8	3.50	6.90
50	223.0	2.52	7.38	121.2	6.08	9.06
100	482.8	3.07	8.48	242.4	6.94	10.03
200	1034.6	3.58	9.52	487.6	8.14	11.18
300	1611.0	3.87	10.11	729.4	8.83	11.77
400	2197.0	4.04	10.48	977.8	9.25	12.26
500	2807.6	4.21	10.85	1222.0	9.54	12.58
600	3424.0	4.27	11.12	1427.2	9.77	12.84
700	4011.0	4.51	11.40	1715.2	10.00	13.11
800	4676.0	4.54	11.54	1956.4	10.14	13.24
900	4322.4	4.64	11.73	2198.4	10.34	13.43
1000	5950.2	4.68	11.85	2437.2	10.45	13.52

표 2. 각 알고리즘의 총 태그 인식 시간(ms)

태그 수	C1G1(표준)	C1G1(개선)	C1G2(표준)	C1G2(개선)
10	61.9	39.5	5.5	5.6
50	386.4	195.8	28.8	31.4
100	848.3	394.9	91.7	56.8
200	1839.1	803.4	129.4	121.3
500	5057.7	2038.8	342.2	301.4
1000	10817.2	4096.5	722.2	637.5



5. 결론

본 논문은 유비쿼터스 센서 네트워크와 텔레매틱스 서비스에 사용되는 무선인식 표준 중 UHF 대역 프로토콜인 EPCglobal의 Class 1 Gen 1, Class 1 Gen 2의 에어인터페이스를 분석하고 각 프로토콜에서 규정된 충돌방지 알고리즘을 구현하였다. 그리고 각 프로토콜에서 규정하는 충돌방지 알고리즘의 성능을 개선한 알고리즘을 제안하고 성능을 비교하였다.

개선된 Class 1 Gen 1 알고리즘은 태그 수 100개 일 때 54.5%, 태그 수 1000개 일 때 63.4% 감소하였다. 개선된 Class 1 Gen 2 알고리즘은 태그 수 100개 일 때 7.9%, 태그 수 1000개 일 때 11.7% 감소하였다. 개선된 충돌방지 알고리즘은 표준 알고리즘보다 태그 인식 성능이 상당히 개선됨을 알 수 있다. 따라서 개선된 충돌방지 알고리즘은 유비쿼터스 센서 네트워크와 텔레매틱스 서비스 등에 사용되는 무선인식 시스템의 성능 개선 방안으로 적용될 수 있다.

본 논문에서는 충돌방지 알고리즘을 C언어 구현하여 컴퓨터 시뮬레이션을 수행함으로써 그 성능을 확인하였다. 향후에는 프로그램된 충돌방지 알고리즘을 직접 리더시스템에 포팅하여 태그와 리더 간 통신에 적용할 계획이다.

<본 연구는 산업자원부의 지역혁신 인력양성사업의 연구결과로 수행되었음>

참고문헌

[1] K. Finkenzeller, RFID Handbook, Wiley & Sons, 2003.
[2] Auto-ID Center Massachusetts of Technology, Technical Report, "860~930MHz Class 1 radio frequency identification tag radio frequency & logical communication interface specification candidate recommendation, version1.0.1"
[3] EPCglobal, "EPCTM radio-frequency identity protocols Class-1 Generation-2 UHF RFID protocol for communications at 860MHz~960MHz Version 1.0.9", 2004.
[4] B. Zhen, M. Kobayashi, "Framed ALOHA for multiple RFID objects identification", IEICE Trans. Communication, Vol. E88-B, pp. 991-999, March 2005.