IEEE802.15.4 WPAN에서의 Cyclic Contention Free Access 기법

준회원 곽 원 근*, 종신회원 이 재 용*

Cyclic Contention Free Access Scheme for IEEE802.15.4 WPAN

Woon-geun Kwak* *Associate Member*, Jai-yong Lee* *Lifelong Member*

요 약

IEEE802.15.4의 비경쟁기반 다중 접속 기법으로 적용된 GTS(Guaranteed Time Slot)의 성능 분석과 문제점 분석을 통해, 개선된 형태의 Cyclic Contention Free Access(Cyclic-CFA) 기법을 제안한다. Cyclic-CFA 기법은 Polling 기법을 개선한 스케줄링 방식으로, 장치(device)들은 PAN 코디네이터(coordinator)로부터 부여받은 Sequence Number를 기반으로 순차적으로 데이터를 전송한다. 별도의 Polling 패킷(packet)이 없고 단일방향성(unidirectional)으로 동작하므로, 고전적인 Polling 기법의 오버헤드를 줄일 수 있어 보다 효율적으로 채널을 사용할 수 있다. 또한 GTS보다 많은 수의 장치가 비경쟁 데이터 전송 서비스를 받을 수 있으며, 서비스 지연 시간을 줄일 수 있다.

Key Words : IEEE802.15.4, WPAN, Guaranteed Time Slot, Contention Free Access

ABSTRACT

The GTS(Guaranteed Time Slot) of IEEE802.15.4 standard, which is the contention free access mechanism, has some problems such as the limited number of deployed devices, the low channel utilization and the service confirm delay. The proposed Cyclic-CFA(Contention Free Access) scheme is a modified polling algorithm that allows a large number of devices to be served Contention Free Access without polling packets. The Cyclic-CFA scheme improves the channel utilization dramatically and also reduces service delay time.

Ⅰ. 서 론

IEEE802.15.4의 수퍼프레임(Superframe)은 경쟁 기반의 CSMA/CA 와 비경쟁 기반의 GTS가 공존한다. 비경쟁 기반의 GTS는 제어, 모니터링, 보안등과 같은 timeliness 특성을 갖는 응용분야에 적합하도록 배타적인 채널 할당 방법을 사용한다. 지만 GTS를 사용할 수 있는 장치의 수가 제한적이고, 높은 BO(BeaconOrder)에서 대역의 낭비가 심하다.

제안한 Cyclic-CFA 기법은 비경쟁 기반 서비스에서의 서비스 시작 지연과 대역 낭비 문제를 개선하고 보다 많은 장치가 비경쟁 기반의 데이터 전송 서비스를 받을 수 있도록 하였다.

Ⅱ. 관련 연구

수퍼프레임은 PAN 코디네이터의 비콘(beacon)과 함께 시작되고, active 구간과 inactive 구간으로 나뉜다[1]. Active 구간(=SD, Superframe Duration)은 동일한 크기의 16개 수퍼프레임슬롯(Superframe Slot, 이하 '슬롯'이라 함)으로 나뉘며, CSMA/CA 기반의 CAP(Contention Access Period)와 GTS 기

※ 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(IITA-2006-C1090-0603-0038)
* 연세대학교 전기전자공학과 Ubinet LAB(wongeun@yonsei.ac.kr, jyl@yonsei.ac.kr)
논문번호 : KICS2007-05-207, 접수일자 : 2007년 5월 7일, 최종논문접수일자 : 2007년 7월 9일

반의 CFP (Contention Free Period)로 구성 된다.

Duty cycle은 BO와 SO(Superframe Order)에 의해서 2^{SO-BO}값을 갖는다.

BO와 SO의 값에 따라 전력 소모는 문론, 서비스 지연, 지터(jitter)등의 부가적인 문제가 있을 수 있으므로 각 응용분야에 맞는 최적의 BO와 SO를 찾아야 한다.

GTS를 할당 받기 위해서는 CSMA/CA과정을 통해 GTS 할당요청을 PAN 코디네이터에게 해야 하며, PAN 코디네이터는 슬롯 단위로 GTS를 할당한다. GTS할당요청에 대한 확인(confirm)은 비콘을 통해 이루어진다. GTS 할당은 7개까지 가능하다. 장치는 슬롯을 배타적으로 할당 받아 사용한다. 단, 일정 횟수 이상의 비콘 주기 동안 전송할 데이터가 발생하지 않으면, 해당 GTS는 해제된다.

GTS를 할당 받기 위한 일련의 과정은 CSMA의 성능문제와 밀접한 관계가 있다.

Slotted CAMA/CA 알고리즘[1]은 NB(Number of Backoff), BE (Backoff Exponent), CW(Contention Window)의 3가지 변수를 갖는다. 장치는 random[0, $(2^{BE}-1)$]×aUnitBackoffPeriod[1]동안 기다린 다음 (backoff 단계), CW 만큼의 CCA (Clear Channel Assessment)를 수행한다(CCA 단계). 2회 CCA를 모두 성공하면 데이터를 전송하고 실패하면 NB만큼의 backoff 단계와 CCA 단계를 반복한다.

Ⅲ. GTS 성능 분석

IEEE802.15.4 표준에서 GTS Descriptor는 7개로 제한되는데, 첫째는 비콘의 payload가 커지는 것을 방지하기 위해서고, 둘째는 GTS가 16개로 한정된 슬롯을 7개 이상 사용할 경우, 표준에서 정한 최소한의 CAP 길이(*aMinCAPLength*[1])를 PAN 코디

네이터가 확보하는데 어려움이 있기 때문이다. 이는GTS가 슬롯 단위로 할당되는 구조적 문제이다.

GTS가 슬롯 단위로 할당되면서 발생하는 또 다른 문제는 GTS utilization 문제이다.

장치의 data rate보다 큰 대역을 갖는 슬롯을 할당 받으면, 대역의 낭비가 심해진다.

GTS는 CAP에서 할당 요청을 하며, 다음 비콘에서 할당 승인(confirm)을 하고, CFP구간에서 서비스를 받는다. 따라서 BI에 비례하여 서비스 지연이 급격히 증가한다.

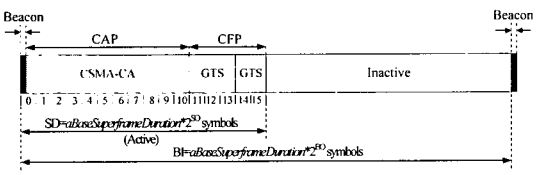


그림 1. 수퍼프레임 구조

3.1 GTS Utilization 분석

장치는 burst 크기, b와 data rate, r에 의해 시간 t동안 $\gamma_{b,r}(t)$ 길이의 데이터를 생성할 수 있다[3].

$$\gamma_{r,b}(t)=\begin{cases} rt+b & if\ t>0 \\ 0 & otherwise \end{cases} \quad (1)$$

WPAN 장치의 특성상 $\gamma_{b,r}(t)$은 상대적으로 작은 값을 갖게 되며, 사용하지 않는 대역, t_{unused}가 발생할 수 있다.

비콘의 GTS Descriptor List에서 i 번째 노드의 burst 크기는 b_i, data rate는 r_i이고, k_i개의 타임 슬롯(T_{slot}=60×2^{SO}symbols)을 GTS로 할당 받았다고 하자. t_{unused}가 발생하지 않는 경우($\gamma_{r,b}(t)\geq k_i T_{slot}$), 최대 데이터프레임(Data Frame)의 개수 n은 식 (2)와 같다.

$$n=\left\lceil \frac{k_i T_{slot}}{l_{frame}+IFS+l_{ack}+t_{ack}} \right\rceil$$
$$where\ IFS=\begin{cases} SIFS & l_{frame}<\text{aMaxSIFSFrameSize} \\ LIFS & otherwise \end{cases} \quad (2)$$

IFS(Inter Frame Space)[1]는 프레임길이에 따라 SIFS(12symbols)와 LIFS(40symbols)로 나뉜다.

따라서 전체 데이터 프레임의 길이, $L_{allFrames}$은 전체 타임슬롯에서 오버헤드를 뺀 값과 같다.

$$L_{allframes}=k_i T_{slot}-nt_{IFS}-nt_{ACK} \quad (3)$$

본 논문에서 프레임의 길이를 나타내는 대문자 L은 ack의 turnaround 시간(12~32symbols), t_{ack}와 IFS(t_{IFS})를 포함하는 단위이고, 소문자 l은 t_{ack}와 t_{IFS}를 뺀 순수한 데이터 프레임의 길이를 나타낸다.

t_{unused}가 발생하는 경우($\gamma_{r,b}(t)<k_i T_{slot}$), 최대로 생성할 수 있는 데이터 프레임의 길이는 $\gamma_{r,b}(k_i T_{slot})$로 bound 되므로, t_{unused}가 발생하지 않는 경우와 발생하는 경우를 고려한 각 GTS에서의 최대 데이터 프레임길이 T_{data}^{i}는 식 (4)와 같다.

$$T_{data}^{i} = \min\begin{pmatrix} b_i + rk_i T_{slot} \\ L_{frame} \end{pmatrix} \quad (4)$$

이며, CFP구간 전체의 utilization은 모든 장치가 생성하는 최대 데이터 프레임 길이와 CFP의 비로 나타낼 수 있다.

$$U_{CFP} = \sum_{i=1}^{N} \frac{T_{data}^{i}}{k_i * T_{slot}} \quad (5)$$

여기서 N은 수퍼프레임에서의 GTS Descriptor의 개수를 나타낸다. T_{slot}은 SO값에 따라 Binary Exponent하게 증가하므로 U_{CFP}는 SO값이 커질수록, r과 b값이 작을수록 급격히 감소한다.

3.2 GTS 서비스 지연 시간

GTS를 요청하는 과정에서의 CSMA/CA성능으로 인한 지연의 증가나, 자원의 제약으로 인한 GTS 할당 지연, 노드가 m개의 비콘 주기 동안 Sleep 모드로 들어갔을 때의 지연문제 등, 현실적인 측면에서의 서비스 지연 분석이 이루어져야 한다.

본 논문에서는 CSMA/CA로 인해 발생하는 GTS 할당요청 및 확인(confirm), 최종 데이터 전송 서비스를 받을 때까지의 지연 시간을 함께 분석하였다.

3.2.1 CSMA/CA Access 지연 시간

CSMA/CA를 통해 채널에 접근하기 위해서는 2 CCA 시간 동안 채널이 clear해야 한다. 2 CCA시간 동안 데이터 장치가 데이터를 전송할 확률을 τ로 정의하면, j 노드로 이루어진 PAN에서, 채널이 clear할 확률은 $p=(1-\tau)^{j-1}$이다.

CSMA/CA의 최대 backoff 횟수를 maxNB라고 하자. k번째 backoff 후에 채널이 clear할 확률은 $p(1-p)^{k-1}$ 이고, 따라서 평균 backoff 횟수, N_b는

$$N_b = \sum_{k=1}^{\max NB} kp(1-p)^{k-1} \quad (6)$$

의 값을 갖는다.

각 backoff 단계에서 발생하는 평균 CCA횟수는 $\tilde{c} = \sum_{c=1}^{c=2} cp^{c/2}$ 이고, 평균적으로 backoff하는 시간은 BE의 함수로 식 (7)과 같이 표현할 수 있다.

$$\tilde{b}_{(BE)} = E[U(0, 2^{BE}-1)] = \frac{2^{BE}-1}{2} \quad (7)$$

따라서 CSMA/CA에서 CCA 시간을 제외한 backoff으로 인해 발생하는 평균 지연 시간은 식 (8)과 같다. 첫째 항은 BE가 증가할 때의 평균 backoff 시간을 더한 값, 두 번째 항은 N_b 의 소수점 이하의 값에 의한 backoff 시간, 세 번째 항은 *macMaxBE*[1]에 도달했을 때의 평균 bckoff시간을 나타낸다.

$$T_{backoff} = \sum_{BE=macMinBE}^{n} \tilde{b}_{(BE)} + (N_b - \lfloor N_b \rfloor)\tilde{b}_{(k)} + \tilde{b}_{(macMaxBE)} \times (macMinBE + \lfloor N_b \rfloor - 1 - n)^{+}$$

where,

$$n = \min(macMinBE + \lfloor N_b \rfloor, macMaxBE),$$
$$k = \min(n+1, macMaxBE),$$
$$(a)^{+} = \max(0, a) \quad (8)$$

식 (9)는 $T_{backoff}$와 평균 CCA 지연 시간을 더한 값으로, CSMA/CA에서의 채널 접근 지연 시간을 나타낸다.

$$D_{CSMA} = (N_b \tilde{c} + T_{backoff}) aUnitBackoffPeriod \quad (9)$$

3.2.2 GTS 할당 확인(CTS Confirmation) 지연 시간

GTS 할당요청에 대한 확인은 비콘을 통해서 이루어진다. 현재 CAP 구간에서의 GTS 할당요청이 이루어진 경우와, 남아있는 CAP구간이 짧아서 다음 CAP 구간에 GTS를 요청해야 하는 경우(CCA difference)로 나누어 그림 2의 분석모델을 이용해 확인 지연 시간을 얻을 수 있다.

GTS 할당요청 패킷(packet)의 도착은 구간 [0,BI]에서 uniform하다고 가정한다.

첫째, 현재의 CAP구간에서 GTS 요청이 성공할 확률, 즉 CCA difference가 발생하지 않을 확률 P_{ncd}는 식 (10)과 같다.

$$P_{ncd} = P[(D_{CSMA} + L_{req} + t) \le T_{CAP}] = \frac{T_{CAP} - D_{CSMA} - L_{req}}{BI} \quad (10)$$

여기서 랜덤변수 t는 BI 구간에서의 GTS 요청 패킷의 도착 시간, L_{req}은 패킷의 길이(IFS와 ack 시간포함), T_{CAP}는 CAP의 길이(FinalCAPslot×T_{slot}),를 나타낸다. 이때 다음 비콘에서 GTS 할당 확인이 되므로 평균 기다림 시간 W_1은 식 (11)과 같다.

$$W_1 = BI - E[U(0, T_{CAP} - D_{CSMA} - L_{req})] = BI - \frac{T_{CAP} - D_{CSMA} - L_{req}}{2} \quad (11)$$

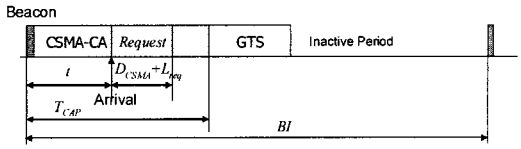


그림 2. GTS 서비스 지연 시간 분석 모델

둘째, t가 패킷을 보내기에 충분치 않은 경우(CCA difference가 일어나는 경우)를 보면, 최대 maxDiffer 만큼의 비콘 주기 중에 m번째 비콘 주기에 성공적으로 패킷을 보낼 확률을 구한 다음 식 (12)와 같이 평균 m값을 구할 수 있다.

$$\tilde{m}_g = \sum_{m=1}^{\max Differ} m p_{ncd}(1-p_{ncd})^{m-1} \tag{12}$$

평균 기다림 시간 W_2는 식 (13)다음과 같다.

$$W_2 = BI - T_{CAP} + (\tilde{m}_g - 1)BI + W_1 \tag{13}$$

따라서 GTS를 요청해서 확인받을 때까지의 총 지연 시간은 $D_{confirm}$은 식 (14)와 같다.

$$D_{Confirm} = P_{ncd}W_1 + P_{cd}W_2 \tag{14}$$

3.2.3 전체 GTS 서비스 지연 시간

비콘을 통해 GTS할당이 확인되면, 장치는 Beacon GTS Fields 에서 자신의 GTS Descriptor에 명시된 GTS Starting Slot으로부터 GTS length 만큼의 타임 슬롯을 독점적으로 사용한다. GTS Reallocation[1]이 발생하지 않는 한 GTS Starting Slot은 항상 고정된다.

GTS Starting Slot이 수퍼프레임에서 n번 슬롯이라고 한다면, 길이는 nT_{slot} 이므로 GTS 할당 요청해서 L_{pkt} 길이의 패킷이 서비스 받기까지의 전체 지연시간 D_{GTS}는 다음과 같이 근사할 수 있다. 여기서 GTS Expiration[1]과 sleep 모드는 고려하지 않았다.

$$D_{GTS} = D_{Confirm} + nT_{slot} + L_{pkt} \tag{15}$$

Ⅳ. Cyclic-CFA(Contention Fee Access) 기법

제안하는 Cyclic-CFA 기법은 타임 슬롯을 독점적으로 사용하지 않고, CFP구간을 여러 장치가 공유하게 되며, 비경쟁 서비스를 요청하는 장치들은 CFP구간이 시작할 때 바로 확인(Service Request Confirm) 동작이 일어난다.

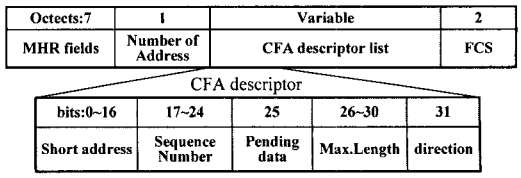


그림 3. CFA_TIM frame format

제안 기법은 스타 토폴로지를 근간으로 하며, PAN 코디네이터와 장치 간에는 히든 노드 문제가 없다고 가정한다. 따라서 장치간의 히든 노드 문제는 PAN 코디네이터를 통해 해결할 수 있다.

비콘 구간의 CAP에서 CFA 서비스를 요청하면 CFA_TIM(CFA_Traffic Information Message, 이하 TIM이라함)메시지에 의해 확인(confirm)이 이루어지고, CFAP 구간에서 서비스를 받는다.

TIM 메시지는 그림 3의 형식으로 갖는다. CFA descriptors list필드에 서비스를 요청한 장치들의 정보가 저장되어 있다. Short address는 장치의 16bit 주소이고, Sequence Number(이하 SN이라함)는 데이터를 보낼 순서를 말하며, 요청한 순서대로 0번부터 순차적으로 부여한다. Pending data는 코디네이터가 해당 장치에 보낼 데이터가 있음을 나타내며, Max. length는 장치의 최대 data rate, direction은 송/수신의 방향을 설정하기 위한 필드이다. CFA descriptors list에 등록되면, 장치가 서비스 해제 요청을 할 때까지 계속 서비스를 받게 된다.

그림 4는 Cyclic-CFA 기법을 응용한 구현 예이며, 표준과의 호환성을 위해 GTS도 같이 동작하도록 설계했다. 코디네이터가 CFA descriptor의 정보와 GTS의 정보를 근간으로 CFP 구간을 설정하고 비콘 프레임의 Final CAP Slot 필드에 CAP 구간의 마지막 슬롯 번호를 써서 모든 장치에 전송한다. CFA 서비스를 요청할 때는 GTS처럼 슬롯 단위로 요청하지 않고, *aUnitBackoffPeriod*(20symbol)[1]단위로 요청하므로 대역의 낭비를 줄일 수 있다.

n번째 비콘 B_n의 Final CAP Slot이 0xA이므로 10번 슬롯까지가 CAP구간이고, n번째 TIM프레임 M_n의 Final CFA Slot이 0xD이므로 11~13번 슬롯이 Cyclic-CFA를 위한 CFAP로 설정되어 있음을 알 수 있다. 14,15번 슬롯은 B_n의 GTS list에 의한 GTS구간이다.

M_n 프레임이 끝나면, FA descriptor list의 SN(#)

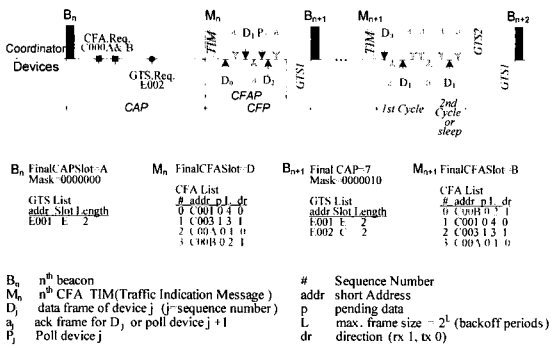


그림 4. Cyclic-CFA 구현 예

이 0번인 장치부터 데이터 D_0를 전송한다. 코디네이터는 ack의 SN을 0으로 설정하여 a_0를 broadcast 한다. SN #1번 장치는 direction이 1이므로 코디네이터가 보내는 데이터 D_1을 수신해야한다.

D_1을 수신하면, 장치는 a_1을 broadcast해서 다음 SN 값을 갖는 장치가 데이터를 전송하도록 한다. 이 때 장치 SN #1번 장치가 전송하는 ack를 받지 못한 SN #2번 장치는 데이터 전송을 시작할 수 없기 때문에 LIFS(long inter-frame space)동안 데이터 전송이 없으면, 코디네이터가 P_1 프레임을 보내 강제로 Polling 한다.

자신의 데이터 전송차례가 되었음에도 전송할 데이터가 없을 때는 자신의 SN을 포함한 ack를 보내 다음 SN 값을 갖는 장치가 데이터를 전송할 수 있도록 한다.

형평성 문제로 SN값은 매 TIM마다 달라질 수 있다. M_n에서 CFA descriptor list의 SN #2번까지 데이터를 보냈으므로, M_{n+1}에서는 M_n에서의 SN #3번 장치가 SN #0번을 부여받는다.

모든 장치가 데이터 전송을 완료해도, CFAP 구간에 여유가 있으면, sleep 상태로 들어가거나 SN #0번 장치부터 다시 데이터 전송을 시작한다.

B_{n+1} 수퍼프레임을 보면, GTS는 4개의 타임슬롯으로 2개의 장치만 비경쟁 접근 서비스를 받을 수 있는 반면, Cyclic-CFA의 경우 5개의 슬롯으로 4개의 장치가 서비스를 받을 수 있다. SO값이 커질수록 하나의 타임슬롯에서 Cyclic-CFA를 받을 수 있는 장치의 수는 급격히 커진다.

4.1 Cyclic-CFA Utilization 분석

CFA_TIM의 CFA list에 N개의 SN이 있다면, 첫 번째 사이클(cycle)에서 queue에 전송할 데이터가 있는 장치의 수 k는 이항분포를 따르므로 평균 $N\rho$개의 장치가 평균 l_{frame} 크기를 갖는 데이터를 전송한다. 여기서 ρ는 inter-arrival rate λ와 service rate μ의 비, λ/μ로 표현되는 utilization factor이다. 첫 번째 cycle에서의 utilization 비율, U_{1cycle} 과 2번째 cycle의 utilization 비율, $U_{n*cycle}$은 식 (16)과 같다. 여기서 l_{TIM}은 CFA_TIM의 길이, l_{frame}은 데이터 프레임 길이, l_{ack} 는 ack 프레임 길이(22symbols)이다. 2번째 cycle의 경우 l_{TIM}이 없으므로 N과 무관한 값을 갖는다.

$$U_{1cycle} = \frac{N\rho l_{frame}}{l_{TIM} + t_{IFS} + N\rho(l_{frame} + t_{ack}) + N(t_{IFS} + l_{ack})}$$

$$U_{n*cycle} = \frac{\rho l_{frame}}{\rho(l_{frame} + t_{ack}) + t_{IFS} + l_{ack}}, \quad \text{where n = 2,3,...} \quad (16)$$

Cycle 횟수 n은 식 (17)의 부등식으로 구할 수 있다.

$$l_{TIM} + t_{IFS} + N\rho(l_{frame} + t_{ack}) + N(t_{IFS} + l_{ack}) + n(\rho(l_{frame} + t_{ack}) + t_{IFS} + l_{ack}) \le \max CFP \cdot T_{slot} \quad (17)$$

따라서 Cyclic-CFA에 의한 CFAP구간의 utilization, U_{CFA}는 식 (18)다음과 같다.

$$U_{CFA} = \frac{N\rho l_{frame} + n\rho l_{frame}}{T_{1Cycle} + T_{n*Cyclel}}$$

$$\text{Where, } T_{1Cycle} = l_{TIM} + t_{IFS} + N\rho(t_{frame} + t_{ack}) + N(t_{IFS} + l_{ack})$$

$$T_{n*Cyclel} = n(\rho(l_{frame} + t_{ack}) + t_{IFS} + l_{ack}) \quad (18)$$

4.2 Cyclic-CFA 서비스 지연 시간

4.2.1 CFA 서비스 요청 확인 지연시간

첫째, CCA difference가 발생하지 않을 확률, P_{NCD}는 그림 5에서 $D_{CSMA}+L_{req}$가 T_{CAP} 구간 안에 들어와야 하므로 식 (19)와 같고, 현재 수퍼프레임의 CFA_TIM에서 요청 확인이 이루어 지므로 평균 기다림 시간 W_1'은 식 (20)과 같다.

$$P_{NCD} = P[(D_{CSMA} + L_{req} + t) \le T_{CAP}] = \frac{T_{CAP} - D_{CSMA} - L_{req}}{BI} \quad (19)$$

$$W_1' = T_{CAP} + L_{TIM} - E[U(0, T_{CAP} - D_{CSMA} - L_{req})] = L_{TIM} + \frac{T_{CAP} + D_{CSMA} + L_{req}}{2} \quad (20)$$

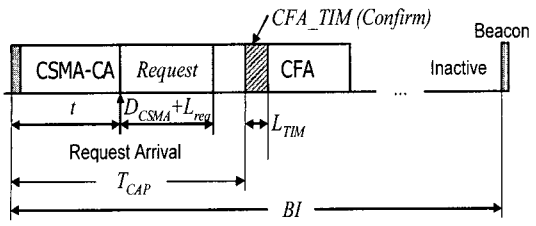


그림 5. Cyclic-CFA 서비스 지연 분석 모델

둘째, CCA difference가 일어날 확률은 P_{CD}=1-P_{NCD}이고, 평균 발생 횟수, m_c는 식 (21)과 같다. 평균 기다림 시간 W_2'는 식 (22)와 같다.

$$\tilde{m}_c = \sum_{m=1}^{\max Differ} mp(1-p_{NCD})^{m-1} \tag{21}$$

$$W_2^{'} = BI - T_{CAP} + \tilde{m}_c BI + W_1^{'} \tag{22}$$

장치가 CFA 서비스를 요청해서 확인(confirm) 받을 때까지의 지연시간은 식 (23)과 같다.

$$D_{Confirm}^{'} = W_1^{'} P_{NCD} + W_2^{'} P_{CD} \tag{23}$$

4.2.2 CFA 서비스 시작 지연 시간

CFA Descriptor List에서 i번째 데이터 전송 순서를 갖는 장치는 SN이 자신보다 빠른 (i-1)개의 장치가 데이터 전송을 완료해야 서비스를 받을 수 있다. i-1개의 장치 중 현재 CFAP구간에서 전송할 데이터가 있는 장치의 개수를 j라고 한다. 데이터가 있든 없든, ack 패킷은 항상 발생한다. 따라서 i번째 장치가 CFA 서비스를 받기까지 기다려야 할 시간은 T_{CAP}+ L_{TIM}+(i-1)L_{ack}+jL_{pkt}이므로, 구간 [0, T_{CAP}+ L_{TIM}+(i-1)L_{ack}+jL_{pkt}]에 도착한 패킷은 현재 CFAP 구간에서 서비스 받을 수 있다. 여기서 L_{TIM}은 CFA_TIM(그림 3)의 길이로, 헤더(10octets)와 N개의 CFA descriptor (N×4 octects), 그리고 IFS를 더한 값(10octets+N×4octects)×2symbos/octet+t_{IFS})이고, L_{ack} (=l_{ack}+t_{IFS})은 ack의 길이, L_{pkt}은 전송하고자 하는 데이터 패킷의 길이를 나타낸다. 이 구간에 데이터 패킷이 도착할 확률 P_{c1}은 식 (23)과 같고 기다림 시간 X_1은 식 (24)와 같다.

$$P_{C1} = P[t \le (T_{CAP} + L_{TIM} + (i-1)L_{ack} + jL_{pkt})] = \frac{T_{CAP} + L_{TIM} + (i-1)L_{ack} + jL_{pkt}}{BI} \tag{23}$$

$$X_1 = \frac{T_{CAP} + L_{TIM} + (i-1)L_{ack} + jL_{pkt}}{2} \tag{24}$$

[T_{CAP}+ L_{TIM}+(i-1)L_{ack}+jL_{pkt}, BI]구간에 도착하는 경우의 확률은 1-P_{c1}이고, 기다림 시간은 i-1개의 장치가 모두 서비스를 받을 때까지이다. 장치 또는 코디네이터의 queue가 비어있지 않을 확률이 ρ이므로, i-1개중 k개의 queue에 데이터가 있을 확률은 식 (25)와 같이 이항 분포를 따른다.

$$P[X_W = x] = \begin{cases} \binom{i-1}{k} \rho^k (1-\rho)^{i-k-1} \\ \qquad , x = T_{CAP} + L_{TIM} + (i-1)L_{ack} + kL_{pkt} \\ 0 \qquad , \text{otherwise} \end{cases} \tag{25}$$

식 (25)의 평균은 (i-1)ρ이므로 다음 수퍼프레임의 CFAP구간에서 i-1번째의 ack가 올 때까지의 평균 기다림 시간은 식 (26)과 같다.

$$E[X_W] = T_{CAP} + L_{TIM} + (i-1)L_{ack} + \rho(i-1)L_{pkt} \tag{26}$$

따라서 총 기다림 시간은 현재 수퍼프레임에서의 평균 기다림 시간의 합, X_2로 표현된다.

$$X_2 = \frac{BI + T_{CAP} + L_{TIM} + (i-1)L_{ack} + (\rho - j)L_{pkt}}{2} \tag{27}$$

따라서 CFA_TIM에서 CFA 서비스 확인을 받고 데이터를 전송할 때까지 기다려야 하는 평균 시간은 식 (28)과 같다.

$$D_{CFP} = X_1 P_{c1} + X_2 (1 - P_{c1}) + L_{pkt} \tag{28}$$

4.2.3 CFA 서비스의 전체 지연 시간

CFA 서비스의 전체 지연 시간, 즉 서비스를 요청해서 서비스를 받을 때까지의 지연시간은 서비스 확인 지연시간과 서비스 시작 지연 시간의 합으로 나타낼 수 있다.

$$D_{R2S}^{'} = D_{confirm}^{'} + D_{CFA} \tag{29}$$

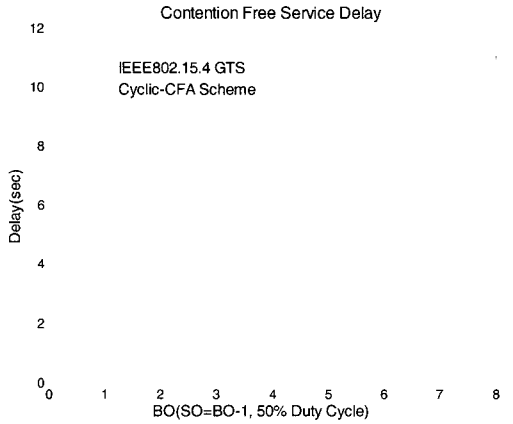


그림 6. Contention Free Access 서비스 지연 비교

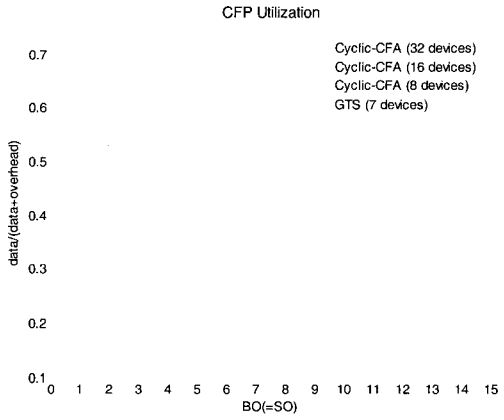


그림 7. Contention Free Access 서비스의 utilization 비교

V. 결 과

모의실험을 위해 ns2의 IEEE802.15.4 MAC에 MLME.CFA primitive를 추가하고, 코디네이터의 기능 일부를 수정했다. 1개의 PAN 코디네이터를 중심으로 200개의 장치를 스타토폴로지로 구성했다. 각 장치는 최대 3번의 CSMA/CA를 재시도 할 수 있고, CCA difference는 최대 10개의 수퍼프레임 동안 일어날 수 있으며, 최대 3개 수퍼프레임 구간 동안 sleep 모드로 들어갈 수 있다.

각 장치 CFA 서비스를 요청하기 위해 발생하는 패킷의 inter-arrival rate는 0.01/BI 이고, CFA 서비스를 받기 시작했을 때의 데이터 패킷의 inter-arrival rate는 0.02/BI이며, 포아송 분포를 따른다. CFA 서비스를 요청해서 확인(confirm)이 되면, 다음 수퍼프레임부터는 별도의 CFA 서비스 요청 없이 CFAP 구간에서 지속적으로 서비스를 받는다. 단, 10개의 비콘 주기 동안 CFAP구간에서 데이터 전송이 일어나지 않으면, PAN 코디네이터가 해당 장치를 CFA descriptor list에서 제외시킨다.

첫 번째 실험은 PAN의 duty cycle을 50%하고 BO를 1~8까지 증가시키면서 CFA를 요청해서 서비스 받을 때까지의 지연시간을 측정하였다. 이때, CAP구간은 *macMinCAPLength*(440symbol)를 만족하는 최소한의 길이로 설정하고, CFAP구간을 최대한으로 하여, 가능한 많은 장치가 CFA 서비스 요청을 해서 서비스를 받도록 하였다.

BO가 커질수록 cyclic CFA의 서비스 지연시간이 GTS에 비해 크게 줄어드는 것을 볼 수 있다. 낮은 BO에서는 GTS가 좋은 결과를 보여주고 있는데, 이는 수퍼프레임의 길이가 짧기 때문에 상대적으로 CFA_TIM이나 ack 프레임과 같은 오버헤드가 커지기 때문이다.

다음은 대역의 utilization을 비교하기 위해 각 장치들의 메모리 크기는 8Kbits~32Kbits, data rate는 32kbps, 최대 GTS 할당 슬롯의 수는 4개, 패킷 크기는 0~154symbols의 임의 값으로 설정하였다. GTS는 낮은 SO와 높은 SO에서 utilization이 현저하게 떨어진다. 낮은 SO에서는 오버헤드가 상대적으로 증가하고, 높은 SO에서는 장치의 data rate가 낮아 사용하지 않는 대역이 증가하기 때문이다.

반면 Cyclic-CFA의 경우 장치의 개수 또는 SO와 무관하게 일정한 비율의 utilization을 갖는다. 이는 cyclic-CFA가 *aUnitBackoffPeriod* 단위로 서비스 구간을 설정하고, 사용하지 않는 대역을 cyclic하게 재사용하기 때문이다.

VI. 결 론

지금까지의 IEEE 802.15.4의 GTS관련 연구가 성능 분석이나, 서비스 가능한 장치의 수를 증가시키는 데에 초점을 두었다면, 제안한 Cyclic-CFA 는 GTS의 장치수의 제한 문제와 더불어 SO가 커지면서 발생하는 대역의 낭비 문제를 해결하고, 서비스 시작 지연 시간을 줄이는데 초점을 두었다.

또한, 비경쟁 서비스의 할당과정에서 발생하는 CSMA/CA의 성능 문제, 요청에 대한 서비스 확인(confirm) 지연 시간, 비경쟁 패킷 전송 서비스 지연시간을 연계하여 GTS와 Cyclic-CFA 기법의 성능을 비교 분석하였다.

결국, Cyclic-CFA를 통해, 비콘의 오버헤드를 줄이면서 비경쟁 서비스 구간의 대역 사용 효율을 획기적으로 개선하였으며, 보다 많은 장치가 비경쟁 데이터 전송 서비스를 받을 수 있다.

참 고 문 헌

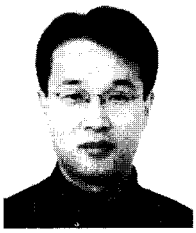
[1] IEEE, "IEEE 802.15.4, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)", *IEEE,* 2006

[2] Anis Koubâa, Mário Alves andEduardo Tovar, "GTS Allocation Analysis in IEEE 802.15.4 for Real-Time Wireless Sensor Networks", *WPDRTS'06, special track on WSNs*, 2006.

[3] Jean-Yves Le Boudec, Patrick Thiran "Network Calculus Parts II and III : A Theory of Deterministic Queuing Systems for the Internet", 2004

[4] Kishor S. Trivedi, "Probability and Statistics with Reliability, Queuing and Computer Science Applications", 2nd Edition, *Willy*, 2002

곽 원 근 (Won-geun Kwak) 준회원



2000년 2월 숭실대학교 전자 공학과

2000년 3월~현재 삼성전자 SW 솔루션개발 선임연구원

2006년 3월~현재 연세대학교 전자공학과 석사과정

<관심분야> WPAN MAC, Zigbee, UWB

이 재 용 (Jai-yong Lee) 종신회원

1977년 2월 연세대학교 전자 공학과

1987년 5월 IOWA State Univ. 공학박사

1987년 7월~1994년 8월 포항 공과대학 교수

1994년 9월~현재 연세대학교 전기전자공학과 교수

<관심분야> Protocol Design for QoS Management, Sensor Network, Wireless Multimedia Protocol.