Profibus 토큰 패싱 프로토콜 성능 모델에서의 전송 지연 특성

Communication Delay Properties in Performance Model of Profibus Token Passing Protocol

이 경 창, 김 현 희, 이 석*
(Kyung Chang Lee, Hyun Hee Kim, and Suk Lee)

Abstract : In many automated systems, such as manufacturing systems and process plants, a fieldbus is a very important component for the exchange of various and sometimes crucial information. Some of the information has a tendency to rapidly lose its value as time elapses after its creation. Such information or data is called real-time data that includes sensor values and control commands. In order to deliver these data in time, the fieldbus network should be tailored to have short delay with respect to the individual time limit of various data. Fine-tuning the network for a given traffic requires the knowledge on the relationship between the protocol parameters such as timer values and the performance measure such as network delay. This paper presents a mathematical performance model to calculate communication delays of the Profibus-FMS network when the timer value and the traffic characteristics are given.

Keywords : token passing protocol, Profibus-FMS, fieldbus, communication delay, performance model, real-time requirements, target rotation time

I. 서론

최근 들어, 산업용 네트워크는 자동화 시스템을 구성하는데 있어 필수적인 요소가 되고 있다. 특히, 산업용 네트워크는 자동화 시스템에서 사용되는 센서나 구동기, 제어기와 같은 필드 장치들 간의 신뢰성 있는 데이터 교환을 보장함으로써, 시스템이 대형화되고 지능화되는데 있어 핵심적인 역할을 하고 있다. 산업용 네트워크는 필드장치를 일대일로 연결하는 전통적인 점대점(point-to-point) 방식에 비하여, 케이블의 절약이나, 유지 보수의 용이, 확장성의 증대, 설치 비용의 감소 등과 같은 장점을 가지고 있다. 이러한 특징으로 인하여, 산업용 네트워크는 전통적인 공정 제어 시스템 뿐만 아니라, 조립 생산 시스템이나 물류 자동화 시스템 등과 같은 다양한 분야로 확산되어 가는 추세이다[1][2].

일반적으로, 산업용 네트워크에서 교환되는 데이터는 실시간 데이터와 비실시간 데이터로 구분되며, 공유된 전송 매체를 통하여 동시에 전송될 수 있어야 한다. 만약, 이러한 데이터들이 제대로 관리되지 못하면 즉, 실시간 데이터가 정해진 시간 한계치 내에 전송되어야 한다는 실시간 요구 조건(real-time requirements)이 만족되지 못하면, 시스템의 성능이 저하될 뿐만 아니라, 시스템에 치명적인 오류가 발생될 수 있다.

이러한 문제를 해결하기 위하여, 1980년대 말부터 자동화 시스템에서 실시간 요구 조건을 만족시킬 수 있는 산업용 네트워크 즉, 필드버스가 여러 지역 표준기관들에 의하여 개발되었으며, 1999년 말 Profibus, Fieldbus Foundation, WorldFIP 등과 같은 6개의 프로토콜을 포함하는 IEC 61158 필드버스가 국제 표준으로 제정되었다[3].

* 책임저자(Corresponding Author)
논문접수 : 2003. 3. 21., 채택확정 : 2003. 8. 26.
이경창, 김현희, 이석 : 부산대학교 기계공학부
(gclee@pnu.edu/cjssus48@pnu.edu/slee@pnu.edu)
※ 이 논문은 2002년도 한국 학술진흥재단의 지원에 의하여 연구되었음.(KRF-2002-002-D00024)

현재까지 네트워크 성능 변수의 선정에 관한 문제는 많은 필드버스 연구자들의 관심 분야이며, 네트워크 시스템의 성능 향상을 위하여 여러 방향에서 접근 방향이 모색되어져 왔다. 대표적인 연구로서, Tovar[5][6]와 Vitturi[7]는 Profibus-FMS [8]의 데이터 링크 계층에서의 토큰 사이클 타임에 관한 분석을 바탕으로 실시간 요구 조건을 보장할 수 있는 T_{TR} 선정 방법에 대하여 제시하였으며, Hong[9]은 Profibus-FMS의 응용 계층에서 T_{TR} 의 변화에 따른 전송 지연(end-to-end communication delay)의 특성에 대하여 분석하였다. 또한, Hong[10]은 Profibus에서 주기적 메시지를 일정한 시간 한계 이내에전송함으로써 실시간 요구 조건을 보장할 수 있는 대역폭 할당 기법(bandwidth allocation scheme)을 제시하였으며, Lee[11]는성능 관리에 의한 실시간 요구 조건의 보장 문제에 대하여 다루었다.

이러한 연구에도 불구하고 아직까지 토큰 패싱 프로토콜에서 성능 변수의 선정은 매우 어려운 문제 영역에 속하며, 대다수의 네트워크 관리자는 이론적인 방법보다는 숙련된 경험을 바탕으로 한 시행 착오적 방법을 택하고 있다. 특히, 네트워크 관리자는 선정된 초기 성능 변수를 실제 시스템에 적용하여 시운전을 수행하면서, 네트워크의 성능이 향상되도록 성능 변수를 조절하는 방식을 사용하고 있다. 그러나 이러한 방법은 실제 네트워크 시스템에 적용하여야 한다는 문제점으로 인하여 성능 변수의 선정에 오랜 시간이 소요될 뿐만 아니라, 네트워크 시스템의 오동작과 같은 문제를 일으킬 수 있기 때문에 효율적인 방법이라 볼 수 없다.

따라서, 본 논문에서는 Profibus-FMS 토큰 패싱 프로토콜의 성능을 확인하기 위한 하나의 방법으로써, 성능 변수에 따라 전송 지연을 수학적으로 계산할 수 있는 성능 모델을 제시하고, 도식적 방법과 시뮬레이션 방법을 통하여 전송 지연을 계산, 비교 하였다. 또한, 실험 모델의 결과와 비교하여 성능 모델의 유효성을 검토하였다. 특히, 본 논문에서 제안된 성능 모델은 실제 시스템에서 사용되는 트래픽 조건을

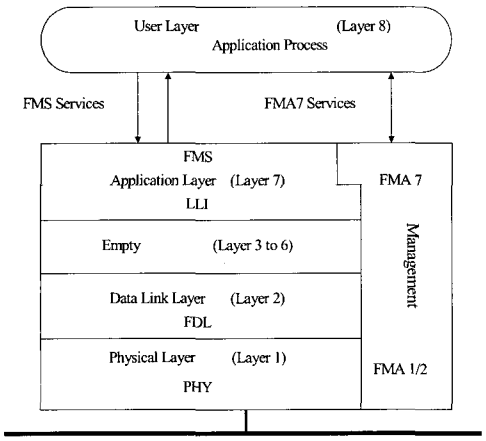


그림 1. Profibus-FMS 프로토콜 구조.
Fig. 1. Protocol architecture of Profibus-FMS.

입력으로 하여 전송 지연을 빠른 시간 내에 계산 할 수 있기 때문에, 실제 시스템의 시운전 없이도 적절한 성능 변수를 선정 할 수 있는 유효한 도구가 될 것이다.

본 논문은 6장으로 구성되어 있다. 2장에서는 Profibus-FMS에 대하여 간략히 서술하고, 3장에서는 Profibus-FMS의 성능 모델을 제시한다. 4장에서는 성능 모델을 도식적 방법과 시뮬레이션 방법을 이용하여 표현하고, 전송 지연 알고리즘을 제시한다. 5장에서는 성능 모델과 실험 모델의 전송 지연을 비교함으로써 성능 모델의 유효성을 검증한다. 마지막으로, 6장에서는 요약과 향후 과제에 대해서 서술한다.

II. 토큰 패싱 프로토콜 : Profibus-FMS

Profibus-FMS[8]는 필드 장비들의 실시간 통신 특성을 보장하기 위하여, 그림 1과 같이 OSI(open system interface) 7계층 모델 중 물리계층, 데이터링크계층, 응용계층만을 가지고 있다. 또한, 사용자와의 인터페이스를 위하여 8계층과 네트워크 관리를 위하여 관리계층을 가지고 있다. Profibus-FMS의 물리계층으로는 RS485가 사용되며, 버스 토폴로지(bus topology)를 사용한다. Profibus의 경우 전송속도는 9.6Kbps에서 500 Kbps까지 지원한다.

데이터링크계층에는 매체접속제어(medium access control, MAC) 기능과 논리링크제어(logical link control, LLC) 기능이 정의되어 있다. Profibus-FMS는 토큰 전달(token passing) 방식을 사용하며, 버스에 접속된 스테이션들을 마스터(master)와 슬레이브(slave)로 구분한다. 마스터는 논리적 링(list of active station, LAS)을 따라 회전하는 토큰을 소유할 수 있으며, 토큰을 소유한 경우 다른 스테이션과의 통신을 수행할 수 있다. 그러나, 슬레이브는 토큰을 소유할 수 없으며, 마스터의 요청에 의해서만 통신에 참여할 수 있다.

마스터는 전송 권한의 설정을 위하여 T_{TR} (target rotation time), T_{RR} (real rotation time), T_{TH} (token holding time)와 같은 3가지 종류의 타이머를 가진다. 여기에서, T_{TR} 은 토큰이 버스를 1회전하는데 예상되는 시간으로 정의되며, T_{RR} 은 실제

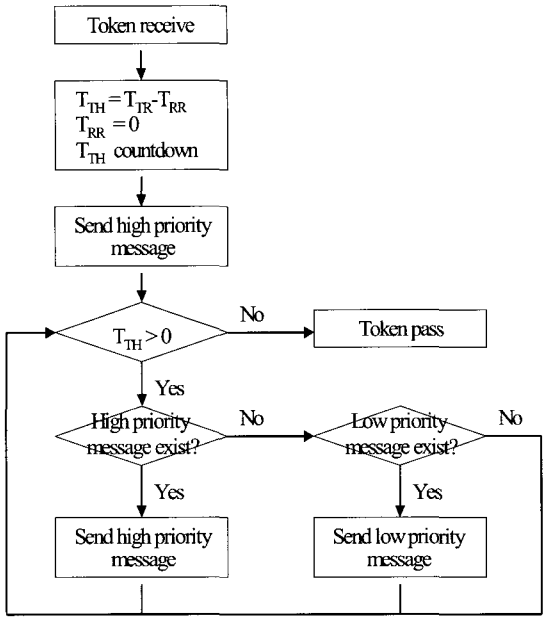


그림 2. 우선순위도구의 플로우 차트.
Fig. 2. Flowchart of the priority mechanism.

토큰이 버스를 1회전하는 데 걸린 시간으로 정의된다. 또한, T_{TH} 는 토큰을 가진 스테이션에서 통신이 가능한 시간으로 T_{TR} 과 T_{RR} 의 차이(T_{TR} - T_{RR})로 정의된다. 그림 2는 Profibus-FMS에서의 우선순위 도구에 의한 전송 절차를 나타내고 있다. 그림에서, 마스터는 토큰을 받은 후, T_{TH} 를 계산하고 나서 하나의 높은 우선순위 메시지 사이클(high priority message cycle)을 수행한다. 여기에서, 메시지 사이클이란 통신 서비스에 따라 해당 메시지를 송신하고 나서부터 그에 상응하는 응답 메시지를 수신하는 일련의 과정으로 정의하며, 메시지 사이클의 메시지 교환 절차는 통신 서비스의 특징에 따라 결정된다. 하나의 메시지 사이클이 끝나고 나면, 마스터는 T_{TH} 를 다시 계산하고, T_{TH} 가 종료되었는가, 즉, T_{TH} 가 0 보다 작은지 검사한다. 만약, T_{TH} 가 종료되지 않은 경우에는 높은 우선순위 메시지 사이클에서 낮은 우선순위 메시지 사이클 순으로 하나의 메시지 사이클을 추가적으로 수행한다. 그리고 나서, 마스터는 T_{TH} 를 재계산한다. 만약, T_{TH} 가 종료되거나, 전송할 메시지가 없는 경우, 마스터는 논리적 링에 있는 다음 스테이션에게 토큰을 송신한다.

III. Profibus-FMS의 성능 모델

1. Notation

본 절에서는 성능모델을 구현하는데 있어서 필요한 기본 수식들을 정리하였다.

i	iteration(순회), token rotation number, i= 1 ~ m
j	station number, j= 1 ~ n
k	message number, k= 1 ~ p
h	high priority message
l	low priority message

T_T 토큰 전송에 소요되는 시간

$T_{E_i}^j$ 토큰 송신 시간

$T_{S_i}^j$ 토큰 수신 시간

Th_C 높은 우선 순위 메시지의 송신에 소요되는 시간

Th_{Gk}^l k번째 높은 우선 순위 메시지의 생성 시간

Th_{Fk}^l k번째 높은 우선 순위 메시지의 전송 완료 시간

$N_{A_i}^j$ 토큰은 수신한 시점에 계산된 T_{TH} 동안 전송 가능한 메시지의 수

$Nh_{Q_i}^j$ 토큰을 수신한 시점에서 송신 큐에 저장되어 있는 높은 우선순위 메시지의 수

$Nh_{T_i}^j$ 전송된 높은 우선 순위 메시지의 개수

2. 성능 모델

본 장에서 제안된 Profibus-FMS 토큰 패싱 프로토콜의 성능 모델은 다음과 같은 가정을 바탕으로 하고 있다. 첫째, 네트워크 인터페이스 카드(network interface card, NIC)의 송신 큐를 충분히 크게 함으로써, 각 스테이션에서 생성되는 메시지는 취소(overflow)되지 않고 모두 송신 큐(queue)에 저장될 수 있다고 가정한다. 둘째, 전송 중 에러는 발생하지 않는다고 가정한다. 셋째, GAP 갱신 시간(gap update time)을 충분히 크게 설정함으로써, GAP 갱신을 위한 통신과 스테이션의 출입은 발생하지 않는다고 가정한다.

이러한 가정을 바탕으로 하여, j번째 스테이션의 i번째 순회 (iteration)에서 토큰을 수신한 시점에서 송신 큐에 저장되어 있는 높은 우선 순위 메시지의 수($Nh_{Q_i}^j$)와 낮은 우선순위 메시지의 수($Nl_{Q_i}^j$)는 (1)에 나타낸 바와 같이 i번째 순회의 토큰 송신시간($T_{E_i}^j$)까지 생성된 메시지의 총 수에서 i-1번째 순회까지 전송된 메시지($Nh_{T_i}^j$)의 총 수를 뺀 값과 같다.

$$\begin{aligned} Nh_{Q_i}^j &= Nh_{G_i}^j(0, T_{E_i}^j) - \sum_{l=1}^{i-1} Nh_{T_l}^j \\ Nl_{Q_i}^j &= Nl_{G_i}^j(0, T_{E_i}^j) - \sum_{l=1}^{i-1} Nl_{T_l}^j \end{aligned} \quad (1)$$

식에서, $Nh_{G_i}^j(0,t)$는 j번째 스테이션의 전송이 시작되고 난 후 부터 t시점까지 생성된 높은 우선순위 메시지의 개수이며, $\sum_{l=1}^{i-1} Nh_{T_l}^j$는 $i-1$번째 순회까지 전송된 높은 우선순위 메시지의 합이다. 또한, $Nl_{G_i}^j(0,t)$는 j번째 스테이션의 전송이 시작되고 난 후부터 t시점까지 생성된 낮은 우선순위 메시지의 개수이며, $\sum_{l=1}^{i-1} Nl_{T_l}^j$는 $i-1$번째 순회까지 전송된 낮은 우선순위 메시지의 합이다.

그리고, j번째 스테이션의 i번째 순회에서 토큰을 수신한 시점에 계산된 T_{TH} 동안 전송 가능한 메시지의 수($N_{A_i}^j$)는 다음과 같다.

$$N_{A_i}^j = \begin{cases} 1 & \text{if } G\left(\frac{T_{TH_i}^j}{Th_C^j}\right) \le 0 \\ G\left(\frac{T_{TH_i}^j}{Th_C^j}\right) & \text{if } 0 < G\left(\frac{T_{TH_i}^j}{Th_C^j}\right) \le Nh_{Q_i}^j \\ Nh_{Q_i}^j + G\left(\frac{T_{TH_i}^j - Nh_{Q_i}^j \times Th_C^j}{Tl_C^j}\right) & \text{otherwise} \end{cases} \quad (2)$$

$$\text{where } T_{TH_i}^j = T_{TR}^j - (T_{S_i}^j - T_{S_{i-1}}^j)$$

식에서, G(x)의 경우 x가 1보다 작은 경우에는 1이며, x가 1보다 클 때는 소수 첫째자리에서 올림한 정수 값으로 정의된다. Th_C^j는 j번째 스테이션에서 높은 우선순위 메시지의 송신에 소요되는 시간이고 Tl_C^j는 낮은 우선순위 메시지의 송신에 소요되는 시간으로써, 메시지 송신 시간은 스테이션의 성능과 메시지 전송에 사용되는 통신 서비스에 따라 결정된다. 마지막으로, T_{TR}^j은 j번째 스테이션의 T_{TR} 이며, $T_{S_i}^j$는 토큰 수신시간이다. $T_{TH_i}^j$는 i번째 순회에서 계산된 T_{TH}로써 T_{TR}^j에서 $i-1$번째 순회에서 토큰을 받은 후 부터 다시 받는 데까지 걸리는 시간을 뺀 값(T_{RR}^j)으로 정의된다. 여기에서, $T_{TH_i}^j$를 Th_C^j로 나눈 개수($G(T_{TH_i}^j / Th_C^j)$)가 0보다 작을 경우에는 1개의 높은 우선순위 메시지만이 전송이 될 수 있으며, $Nh_{Q_i}^j$보다 작을 경우에는 $T_{TH_i}^j$ 동안 $G(T_{TH_i}^j / Th_C^j)$ 개의 높은 우선순위 메시지가 전송될 수 있다. 그러나, $G(T_{TH_i}^j / Th_C^j)$가 $Nh_{Q_i}^j$보다 클 경우에는 $T_{TH_i}^j$ 동안 $Nh_{Q_i}^j$개의 높은 우선순위 메시지뿐만 아니라, (2)에 나타낸 바와 같이 낮은 우선순위 메시지를 추가적으로 전송할 수 있다.

결과적으로, (1)과 (2)로부터 j번째 스테이션의 i번째 순회에서 전송되는 높은 우선순위의 메시지 수($Nh_{T_i}^j$)와 낮은 우선순위의 메시지 수($Nl_{T_i}^j$)는 다음과 같이 계산된다.

$$(Nh_{T_i}^j, Nl_{T_i}^j) = \begin{cases} (N_{A_i}^j, 0) & \text{if } N_{A_i}^j \le Nh_{Q_i}^j \\ (Nh_{Q_i}^j, \min(Nl_{Q_i}^j, N_{A_i}^j - Nh_{Q_i}^j)) & \text{otherwise} \end{cases} \quad (3)$$

즉, j번째 스테이션은 토큰을 수신한 후 하나의 높은 우선순위 메시지를 전송하고 나서, $T_{TH_i}^j$가 종료되지 않은 경우에 높은 우선순위, 낮은 우선순위 메시지의 순서로 전송한다. 여기에서, $N_{A_i}^j$가 $Nh_{Q_i}^j$보다 작을 경우에는 $N_{A_i}^j$개의 높은 우선 순위 메시지만이 전송될 수 있다. 반면, $N_{A_i}^j$가 $Nh_{Q_i}^j$보다 크다면, $Nh_{Q_i}^j$개의 높은 우선순위 메시지를 전송하고 나서, $Nl_{Q_i}^j$와 $N_{A_i}^j - Nh_{Q_i}^j$ 중에서 작은 값만큼의 낮은 우선순위 메시지가 전송될 수 있다.

일반적으로, 높은 우선순위 메시지는 주어진 시간 한계 내에 전송되지 못하면 시스템의 성능에 악영향을 주기 때문에, 반드시 요구 전송 지연 이내에 전송될 수 있어야 한다. 반면, 낮은 우선순위 메시지는 전체 네트워크에 여유가 있을 때에 전송이 이루어지면 된다. 여기에서, 만약 특정 스테이션에서 T_{TR} 의 값이 너무 크다면, 다수의 낮은 우선순위 메시지가 보내지게 됨으로써, 다른 스테이션들이 사용할 수 있는 T_{TH} 가 작아지게 될 것이다. 이러한 현상은 다른 스테이션들이 높은 우선순위 메시지를 전송하는 데 충분한 T_{TH} 를 가지지 못하게 만듦으로써, 높은 우선순위 메시지의 전송 지연을 증가시키는 원인이 된다. 반대로, 어떤 스테이션에서 T_{TR} 의 값이 너무 작다면, 송신 큐에 저장되어 있는 높은 우선순위 메시지를 충분히 보내지 못한 상태에서 다음 스테이션으로 토큰을 전송할 것이다. 또한, 다른 스테이션들은 T_{TH} 가 충분하기 때문에, 다수의 낮은 우선순위 메시지를 전송할 수 있을 것이다. 이러한 문제는 그 스테이션에서 높은 우선순위 메시지의 전송 지연을 증가시키는 원인이 된다. 따라서, 네트워크의 초기 설계 단계에서부터 높은 우선순위 메시지의 생성량에 따라 적절한 T_{TR} 을 설정함으로써, 높은 우선순위 메시지의 전송 지연을 최소화 시켜야 할 것이다.

다음으로, j번째 스테이션의 i번째 순회에서 토큰 수신 시간($T_{S_i}^{j}$)은 이전 스테이션에서의 토큰 송신 시간($T_{E_i}^{j-1}$)에 토큰 전송에 소요되는 시간(T_T)을 더한 값으로서 다음과 같다.

$$T_{S_i}^{j} = \begin{cases} T_{E_i}^{j-1} + T_T & \text{if } j \neq 1 \\ T_{E_{i-1}}^{n} + T_T & \text{if } j = 1 \end{cases} \tag{4}$$

여기에서, j-1번째 스테이션의 i번째 순회에서 토큰 송신 시간($T_{E_i}^{j-1}$)은 $T_{S_i}^{j-1}$ 와 높은 우선순위 및 낮은 우선순위 메시지의 전송에 소요된 시간을 더한 값으로써, 다음과 같이 계산된다.

$$T_{E_i}^{j-1} = T_{S_i}^{j-1} + (Nh_{T_i}^{j-1} \times Th_C^{j-1} + Nl_{T_i}^{j-1} \times Tl_C^{j-1}) \tag{5}$$

또한, (3)과 (5)를 이용하여, j번째 스테이션에서의 k번째 메시지의 도착 확률 분포 값을 fh_k^j 라 한다면, k번째 높은 우선순위 메시지의 생성 시간($Th_{G_k}^{j}$)과 전송 완료 시간($Th_{F_k}^{j}$)은 다음과 같이 계산된다.

$$\begin{aligned} Th_{G_k}^{j} &= Th_{G_{k-1}}^{j} + fh_k^j \\ Th_{F_k}^{j} &= T_{S_i}^{j} + (k - \textstyle\sum_{l=1}^{i-1} Nh_{T_l}^{j}) Th_C^{j} \\ &\text{where } \textstyle\sum_{l=1}^{i-1} Nh_{T_l}^{j} < k \le \sum_{l=1}^{i} Nh_{T_l}^{j} \end{aligned} \tag{6}$$

식에서, i번째 순회에서 송신 큐에 남아 있는 첫번째 높은 우선순위 메시지는 $\sum_{l=1}^{i-1} Nh_{T_l}^{j} + 1$ 번째에 생성된 메시지로서, 토큰이 수신되자 마자 바로 전송되기 때문에, $T_{S_i}^{j} + Th_C^{j}$ 시간에 전송이 완료된다. 또한, i번째 순회에서는 $Nh_{T_i}^{j}$ 개의 높은 우선순위 메시지가 전송될 수 있으므로, $\sum_{l=1}^{i} Nh_{T_l}^{j}$ 번째 생성된 높은 우선순위 메시지는 $T_{S_i}^{j} + (Nh_{T_l}^{j} \times Th_C^{j})$ 시간에 전송이 완료된다. 따라서, 이를 모든 순회로 확장하면, k번째 높은 우선순위 메시지의 전송 완료 시간은 (6)과 같이 나타낼 수 있다.

그리고 나서, k번째 낮은 우선순위 메시지의 도착 확률 분포를 fl_k^j 라 한다면, $Tl_{G_k}^{j}$ 와 $Tl_{F_k}^{j}$ 는 다음과 같이 계산된다.

$$\begin{aligned} Tl_{G_k}^{j} &= Tl_{G_{k-1}}^{j} + fl_k^j \\ Tl_{F_k}^{j} &= T_{S_i}^{j} + (Nh_{T_l}^{j} \times Th_C^{j} + (k - \textstyle\sum_{l=1}^{i-1} Nl_{T_l}^{j}) \times Tl_C^{j}) \\ &\text{where } \textstyle\sum_{l=1}^{i-1} Nl_{T_l}^{j} < k \le \sum_{l=1}^{i} Nl_{T_l}^{j} \end{aligned} \tag{7}$$

식에서, i번째 순회에서 송신 큐에 남아 있는 첫번째 낮은 우선순위 메시지는 $\sum_{l=1}^{i-1} Nl_{T_l}^{j} + 1$ 번째에 생성된 메시지로서, $Nl_{T_i}^{j}$ 이 0보다 큰 경우에 $Nh_{T_i}^{j}$ 개의 높은 우선순위 메시지가 먼저 전송되고 난 이후에 전송될 수 있다. 따라서, $\sum_{l=1}^{i-1} Nl_{T_l}^{j} + 1$ 번째 생성된 낮은 우선순위 메시지는 $T_{S_i}^{j} + (Nh_{T_l}^{j} \times Th_C^{j}) + Tl_C^{j}$ 시간에 전송이 완료된다. 또한, i번째 순회에서는 $Nl_{T_i}^{j}$ 개의 낮은 우선순위 메시지의 전송이 가능하므로, $\sum_{l=1}^{i} Nl_{T_l}^{j}$ 번째 생성된 낮은 우선순위 메시지는 $T_{S_i}^{j} + (Nh_{T_l}^{j} \times Th_C^{j}) + (Nl_{T_l}^{j} \times Tl_C^{j})$ 시간에 전송이 완료된다. 따라서, 이를 확장하면, k번째 낮은 우선순위 메시지의 전송 완료 시간은 (7)과 같이 나타낼 수 있다. 예로, $Nh_{T_i}^{j}$ 의 값이 각 순회별로 1, 3, 2의 값을 가지고, $Nl_{T_i}^{j}$ 의 값이 각 순회별로 0, 1, 2의 값을 가진다고 한다면, 3번째 순회에서 전송되는 높은 우선순위 메시지는 5, 6번째 생성된 메시지이고 낮은 우선순위 메시지는 2, 3번째 생성된 메시지이다. 즉, 2번째 순회까지 총 4개의 높은 우선순위 메시지가 전송되었으므로, 3번째 순회에서 가장 먼저 전송되는 메시지는 5번째 생성된 높은 우선순위 메시지이다. 또한, 3번째 순회에서는 2개의 높은 우선순위 메시지가 전송이 가능하고, 추가로 2개의 낮은 우선순위 메시지의 전송이 가능하다. 따라서, 6번째 생성된 높은 우선순위 메시지의 전송 완료 시간($Th_{F_6}^{j}$)은 (6)으로부터 k가 6이고, $\sum_{l=1}^{i-1} Nh_{T_l}^{j}$ 가 4이므로, $T_{S_3}^{j} + 2Th_C^{j}$ 가 된다. 또한, 3번째 생성된 낮은 우선순위 메시지의 전송 완료 시간($Th_{l_3}^{j}$)은 (7)로부터 $Nh_{T_i}^{j}$ 가 2이고, k가 3, $\sum_{l=1}^{i-1} Nl_{T_l}^{j}$ 가 1이므로, $T_{S_3}^{j} + 2Th_C^{j} + 2Tl_C^{j}$ 이 된다.

여기에서, Profibus-FMS는 하나의 낮은 우선순위 메시지를 전송하고 나서, T_{TH} 가 종료되지 않았다면, 높은 우선순위 메시지가 생성되어 있는 지를 검사하고, 높은 우선순위 메시지가 존재하면 이를 먼저 전송한다는 특징을 가지고 있다. 예로, 토큰을 받은 시점에 3번째 순회에서 전송되어야 할 2개의 높은 우선순위 메시지 중에 1개는 송신 큐에 저장되어

있고, 나머지 1개는 2번째 낮은 우선순위 메시지 사이클의 수행 도중에 생성된다고 가정하자. 이러한 경우, 전송 순서는 송신 큐에 있는 5번째 높은 우선순위 메시지, 송신 큐에 있는 3번째 낮은 우선순위 메시지, 전송 도중에 생긴 6번째 높은 우선순위 메시지 그리고, 전송도중에 생긴 4번째 낮은 우선순위 메시지 순이 된다. 이때, 6번째 높은 우선순위 메시지의 전송 완료 시간은 $T_{S3}^{j}+2Th_{C}^{j}+Tl_{C}^{j}$ 가 되고, 4번째 낮은 우선순위 메시지의 전송 완료 시간은 $T_{S3}^{j}+2Th_{C}^{j}+2Tl_{C}^{j}$ 가 된다. 그러나, 이러한 현상은 성능 모델로 기술하는 것이 매우 어렵기 때문에, 본 성능 모델에서는 고려하지 않았으며, 시뮬레이션을 이용한 전송 지연 계산에서 (6)과 (7)의 계산 절차를 수정함으로써 고려하였다.

결과적으로, (6)과 (7)을 이용하여, j번째 스테이션에서 k번째 높은 우선순위 메시지의 전송 지연(Dh_{k}^{j})과 낮은 우선순위 메시지의 전송 지연(Dl_{k}^{j})을 계산하면 다음과 같다.

$$\begin{aligned} Dh_{k}^{j} &= Th_{F_k}^{j} - Th_{G_k}^{j} + Th_{D}^{j} \\ Dl_{k}^{j} &= Tl_{F_k}^{j} - Tl_{G_k}^{j} + Tl_{D}^{j} \end{aligned} \tag{8}$$

식에서, Th_{D}^{j} 와 Tl_{D}^{j} 는 수신 스테이션이 메시지를 수신하여 응용 계층까지 보내는 데 걸리는 수신 지연(delivery delay)으로써, 통신 서비스와 수신 스테이션의 성능에 따라 결정된다. 일반적으로, Profibus-FMS에서는 높은 우선순위 메시지의 전송에는 Information Report 와 같은 미확인 서비스(unconfirmed service)가 주로 사용된다. 그림 4는 Information Report 서비스의 타이밍도(timing diagram)를 나타내고 있다. 그림에서, Information Report 서비스는 FDL로 SDA(send data with acknowledgement)를 사용하기 때문에, 송신 스테이션이 요청 프레임(request frame)을 송신하면, 수신 스테이션은 요청 프레임을 성공적으로 수신하였다는 SC 프레임을 전송하고 나서, 지시(indication) 메시지를 응용 계층으로 전달한다. 그리고 나서, 수신 스테이션은 다음 차례에서 토큰을 수신하게 되면, 해당되는 ACK 프레임(ACK frame)을 송신 스테이션에게 전송한다. 마지막으로, 송신 스테이션은 ACK 프레임을 수신하고 나서, 이에 대한 SC 프레임을 수신 스테이션에게 전송한다.

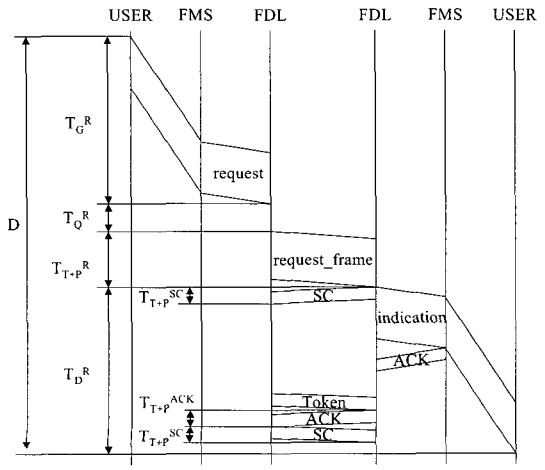


그림 3. Information Report 서비스의 타이밍도.
Fig. 3. Timing diagram of Information Report service.

그림에서, 메시지의 전송 지연은 송신 스테이션의 응용 계층에서 메시지(request message)가 생성되어 물리 계층의 송신 큐에 저장되는 데까지 소요되는 메시지 생성 지연(generation delay, T_G^R), 전송되기 전까지 송신 큐에서 기다려야 하는 대기 지연(queuing delay, T_Q^R), 요청 프레임(request frame)이 네트워크를 통하여 수신 스테이션의 물리 계층까지 전송되는 데까지 소요되는 프레임 전송 지연(frame transmission delay, T_T^R)과 전파 지연(propagation delay, T_P^R), 그리고, 수신 스테이션의 물리 계층에서 요청 프레임을 수신하고 나서, 응용 계층까지 지시 메시지를 전달되는 데 걸리는 수신 지연(delivery delay, T_D^R)으로 구성되어 있다. 여기에서, 메시지가 생성되어 송신 큐에 대기하지 않고 바로 전송될 경우에는 $T_G^R+T_{T+P}^R+T_{T+P}^{SC}$ 의 시간으로서, 이를 메시지의 송신에 소요되는 시간(Th_C^j)으로 간주할 수 있다.

IV. 성능 모델에서의 전송 지연 계산

1. 도식적 방법을 이용한 전송 지연 계산

본 절에서는 3장에서 제안된 성능 모델을 이용하여 전송 지연을 계산할 수 있는 방법의 하나로서, 도식적인 방법을 제안한다. 도식적인 방법은 Stalling의 문헌[13]에서 볼 수 있는 방법으로서, Stalling은 802.4 토큰 버스 모델에서 토큰을 수신하게 되면 전송 가능한 시간동안 메시지를 충분히 전송할 수 있다고 가정하고, 각 스테이션별로 메시지 전송량을 계산하였다. 본 논문에서는 이를 확장하여, 메시지가 생성되는 시간을 고려하여, 제한된 메시지가 존재하는 상황에서 전송 지연을 계산할 수 있는 도식적인 방법을 개발하였다.

1.1 도식적 방법

본 절에서는 도식적 방법을 개발하기 위하여, 몇 가지 가정을 추가하여 성능 모델을 다음과 같이 단순화하였다. 먼저, 각 스테이션에서 높은 우선순위 및 낮은 우선순위 메시지가 일정한 주기(Th_M^j 및 Tl_M^j)로 생성된다고 가정한다면, (1)은 다음과 같이 단순화될 수 있다.

$$\begin{aligned} Nh_{Q_i}^{j} &= \mathrm{int}(T_{E_i}^{j}/Th_{M}^{j}) - \textstyle\sum_{l=1}^{i-1} Nh_{T_l}^{j} \\ Nl_{Q_i}^{j} &= \mathrm{int}(T_{E_i}^{j}/Tl_{M}^{j}) - \textstyle\sum_{l=1}^{i-1} Nl_{T_l}^{j} \end{aligned} \tag{9}$$

식에서, $\mathrm{int}(T_{E_i}^{j}/Th_{M}^{j})$ 는 통신이 시작되고 나서부터 i번째 순회에서 토큰이 송신된 시간인 $T_{E_i}^{j}$ 시간까지 생성된 메시지의 수로서, $(T_{E_i}^{j})$ 를 주기 (Th_M^j) 로 나눈 값으로 계산된다. 여기에서, int(x) 는 정수화 함수로서, x에서 소수점 이하를 버리는 함수이다.

둘째, 각 스테이션에서 토큰을 수신하게 되면, 토큰을 수신하기 직전까지 생성되어 있는 메시지만 현재 순회(iteration)에서 전송하고, 토큰이 수신된 이후에 생성된 메시지는 다음

순회에 전송된다고 가정한다면, (9)는 다음과 같이 단순화될 수 있다.

$$\begin{aligned} Nh_{Q_i}^{\ j} &= int(T_{S_i}^{\ j} / Th_M^{\ j}) - \sum_{l=1}^{i-1} Nh_{T_l}^{\ j} \\ Nl_{Q_i}^{\ j} &= int(T_{S_i}^{\ j} / Tl_M^{\ j}) - \sum_{l=1}^{i-1} Nl_{T_l}^{\ j} \end{aligned} \quad (10)$$

셋째, 높은 우선순위와 낮은 우선순위 메시지의 전송 시간이 동일하다고 가정한다면 즉, 동일한 통신 서비스를 사용하고 메시지 길이가 동일하다고 가정한다면, (2)는 다음과 같이 단순화될 수 있다.

$$N_{A_i}^{\ j} = G(\frac{T_{HT_i}^{\ j}}{T_C^{\ j}}) \quad \text{where } T_C^{\ j} = Th_C^{\ j} = Tl_C^{\ j} \quad (11)$$

마지막으로, (5)의 토큰 송신 시간도 다음과 같이 단순화될 수 있다.

$$T_{E_i}^{\ j} = T_{S_i}^{\ j} + (Nh_{T_i}^{\ j} + Nl_{T_i}^{\ j}) \times T_C^{\ j} \quad (12)$$

이상과 같이 단순화된 성능 모델을 바탕으로 하여, 그림 4는 j번째 스테이션의 i번째 순회에서 토큰 수신 시간과 토큰 송신 시간, 전송되는 메시지의 수를 계산하는 도식적 방법을 나타내고 있다. 그림에서, 해당 스테이션이 토큰을 수신하게 되면, (4)를 이용하여 $T_{S_i}^{\ j}$를 계산 한다(①). 그리고 나서, (10)을 이용하여 $Nh_{Q_i}^{\ j}$와 $Nl_{Q_i}^{\ j}$를, (11)을 이용하여 $N_{A_i}^{\ j}$를 계산한다(②). 세 번째로, $Nh_{Q_i}^{\ j}$와 $N_{A_i}^{\ j}$의 관계에 따라, (3)을 이용하여 $Nh_{T_i}^{\ j}$와 $Nl_{T_i}^{\ j}$를 구한다(③). 마지막으로, (12)로부터 $Nh_{T_i}^{\ j}$와 $Nl_{T_i}^{\ j}$를 이용하여 $T_{E_i}^{\ j}$를 구한다(④).

1.2 도식적 방법에 의한 전송 지연의 결과

도식적 방법을 이용하여 네트워크 시스템에서의 전송 지연을 구하면 다음과 같다. 먼저, 500Kbps의 전송 속도를 가지는 PROFIBUS-FMS 네트워크에 4대의 스테이션이 접속되어 있고, 각 스테이션에서는 10,000비트 타임 주기로 높은 우선순위 메시지가 생성된다고 하자. 이러한 설정은 각 스테이션에서 상태 정보나 감지 정보가 주기적으로 중앙 제어기로 전송되는 원격 감시 시스템으로 간주할 수 있다. 또한, 각 스테이션에서 일정기간동안 25,000비트 타임 주기로 낮은 우선순위 메시지도 생성된다고 하자. 이는 원격 감시 시스템에서 각 스테이션에서 공정 데이터나 응용 프로그램이 전송되는 경우라고 볼 수 있다. 여기에서, 비트 타임은 전송 속도의 역수이며, 전송 속도가 500Kbps인 경우, 1비트 타임은 2μ sec로 계산된다. 또한, $T_C^{\ j}$는 1,750비트 타임, T_T는 500비트 타임, $Th_D^{\ j}$와 $Tl_D^{\ j}$는 4,000비트 타임이라고 하자. 이는 Pentium III (500MHz)급의 PC에서 40바이트의 메시지를 Information Report 서비스로 송신한 경우에 측정된 값이다.

이러한 조건에서, T_{TR}을 30,000비트 타임으로 설정하고 나서, 도식적 방법으로 표현하면 그림 5와 같다. 그림에서 스테이션 1의 첫번째 순회는 통신을 시작하는 단계이므로, 토큰 수신 시간은 0 비트 타임으로 설정되었다(①). 다음으로, 현재 스테이션 1의 큐에 저장되어 있는 높은 우선 순위 메시지와 낮은 우선 순위 메시지의 개수는 도식적 표현의 초기 조건이 통신의 시작과 동시에 메시지가 생성된다고 가정하였기 때문에, 각각 1로(1+1) 설정되었다. 여기에서, (11)을 이용하여 N_A를 계산하면 18이 된다(②). 세 번째로 Nh_T와 Nl_T는 이번 순회에서 전송에 성공한 메시지의 개수이기 때문에, 높은 우선 순위 메시지와 낮은 우선 순위 메시지가 하나씩 전송 되었으므로, 1+1 이 됨을 알 수 있다(③). 마지막으로, 스테이션 1에서 모든 통신을 마치고, 다음 스테이션으로 토큰을 송신한 시간이 기재되어 있다(④). 첫번째 순회에서는 높은 우선 순위 메시지와 낮은 우선 순위 메시지가 하나씩 전송되었으므로, 각각의 메시지 전송에 1,750 비트 타임이 소요되어, 3,500 비트 타임 이후에 토큰은 다음 스테이션으로 전송 된다.

이상과 동일한 방법으로, 스테이션 2 의 첫번째 순회에서 토큰을 수신한 시간은, 스테이션 1의 토큰 송신 시간에 토큰 전송 시간인 500 비트 타임을 더한 4,000 비트 타임이 된다. 또한, 현 스테이션에서 생성되어 있는 메시지는 높은 우선 순위 메시지 1개와 낮은 우선 순위 메시지 1개이며(1+1), N_A는 15개가 된다. 따라서, 2개의 메시지가 전송되고, 특히, 7,500 비트 타임이 지난 토큰은 다음 스테이션으로 전송 된다.

순회 1에서는 N_A가 Nh_Q와 Nl_Q의 합보다 크기 때문에, 생성된 메시지는 해당 순회에서 큰 전송 지연 없이 전송 될 수 있다. 그러나, 순회가 거듭될수록 Nh_Q와 Nl_Q의 합이 N_A보다 크게 됨에 따라, T_{TH}의 여유가 없어지게 되고, 큐에 저장된 메시지가 다 전송되지 못하는 현상이 발생한다. 이로 인하여, 전송 지연이 증가하게 된다.

그림에서, 각 순회별로 높은 우선순위 메시지와 낮은 우선순위 메시지의 전송 개수를 계산할 수 있다. 이러한 결과와 (6) 및 (8)을 이용하여, 각 스테이션에서 높은 우선 순위 메시지의 전송 지연을 계산하면 그림 6과 같다. 그림에서, 모든 스테이션에서 높은 우선순위 메시지의 전송 지연이 35,000 비트 타임 이하로 유지됨을 알 수 있었다. 또한, 각 스테이션들의 평균 전송 지연이 9,500 비트 타임으로서 모든 스테이션이 비슷한 경향의 전송 지연을 보이고 있음을 알 수 있었다. 여기에서, 평균 전송 지연은 각 전송 지연의 합을 전송에 성공한 메시지의 개수로 나눈 값으로 정의하였다. 이러한 결과는 네트워크의 전송 능력이 각 스테이션으로 균등하게 분산할 수 있는 장점을 가진 토큰 패싱 프로토콜의 능력에 의한 것으로 평가되었다.

2. 시뮬레이션 방법을 이용한 전송 지연 계산

(1) T_S	(4) T_E
(2) Nh_Q+Nl_Q, N_A	(3) Nh_T+Nl_T

그림 4. 전송 지연 계산을 위한 도식적 방법.

Fig. 4. Diagrammatic method for calculation of communication delay.

본 절에서는 3장에서 제안된 성능 모델을 이용하여 전송 지연을 계산할 수 있는 시뮬레이션 방법을 제안한다. 앞 절에서 제안된 도식적 방법에서는 랜덤하게 생성되는 메시지에 대한 전송 지연을 구하는 것은 매우 복잡한 계산으로 인하여, 한계를 가지고 있다. 따라서, 실제 네트워크 환경과 유사한 성능 모델을 만들기 위해서는 랜덤하게 생성되는 메시지의 전송 지연을 계산할 수 있는 알고리즘이 필요하다. 따라서, 본 논문에서는 전송 지연을 컴퓨터 프로그램을 이용하여 계산할 수 있는 알고리즘을 개발하였다.

2.1 시뮬레이션 방법

본 절에서는 제안된 성능 모델을 이용하여 전송 지연을 계산할 수 있는 시뮬레이션 방법을 제안한다. 그림 7은 시뮬레이션을 이용한 j번째 스테이션에서 i번째 순회에서의 전송 지연을 계산하기 위한 흐름도(flowchart)를 나타내고 있으며, 이를 단계별로 설명하면 다음과 같다.

단계 1. (4)를 이용하여 j번째 스테이션에서 i번째 순회에서의 토큰 수신 시간($T_{S_i}^j$)을 계산한다.

단계 2. $T_{E_i}^j$에 $T_{S_i}^j$를, $Nh_{T_i}^j$와 $Nl_{T_i}^j$에 0을 하여 초기화 시킨다.

단계 3. (1)에서 $Nh_{Q_i}^j$와 $Nl_{Q_i}^j$를, (2)에서 $N_{A_i}^j$를 계산한다.

단계 4. $Nh_{Q_i}^j$에 저장된 메시지의 유, 무를 검사한다. 메시지가 있는 경우는 단계 5로 넘어간다. 만약, 메시지가 없을 경우, 현재 스테이션의 통신을 종료하고 나서, 토큰을 다음 스테이션으로 전송한다.

단계 5. 하나의 높은 우선순위 메시지를 전송하고 나서, $T_{E_i}^j$와 $Th_{F_k}^j$를 계산한다. $Th_{F_k}^j$는 (6)을 수정하여 생성된 (13)을 이용하여 구한다. 그리고 (8)을 이용하여 Dh_k^j와 Dl_k^j을 구한다.

$$Th_{G_k}^j = Th_{G_{k-1}}^j + fh_k^j$$
$$Th_{F_k}^j = T_{S_i}^j + (Nl_{T_i}^j \times Tl_C^j + (k - \sum_{l=1}^{i-1} Nh_{T_l}^j))Th_C^j \quad (13)$$
$$\text{where } \sum_{l=1}^{i-1} Nh_{T_l}^j < k \le \sum_{l=1}^{i} Nh_{T_l}^j$$

단계 6. 재계산된 $T_{E_i}^j$를 이용하여 $Nh_{Q_i}^j$와 $Nl_{Q_i}^j$, $N_{A_i}^j$를 다시 계산한다.

단계 7. $N_{A_i}^j - Nh_{T_i}^j - Nl_{T_i}^j$의 값이 0보다 큰 가를 검사한다. 만약 0보다 작다면, 전송할 수 있는 타이머($T_{TH_i}^j$)의 여유가 없는 상태이다. 따라서, 현재 스테이션의 통신을 종료하고 나서, 토큰을 다음 스테이션으로 전송한다.

단계 8. 만약 $N_{A_i}^j - Nh_{T_i}^j - Nl_{T_i}^j$의 값이 0보다 크다면, $Nh_{Q_i}^j$의 값이 0보다 큰 가를 검사한다. 만약 0보다 작다면, 전송 큐에 높은 우선순위 메시지가 존재하지 않는 경우이다. 이러한 경우, 낮은 우선순위 메시지를 전송하기 위하여 단계 10으로 간다.

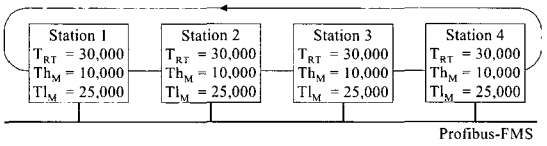
단계 9. 만약 $Nh_{Q_i}^j$의 값이 0보다 크다면, 높은 우선순위 메시지를 전송하기 위하여 단계 5로 되돌아 간다.

단계 10. $Nl_{Q_i}^j$의 값이 0보다 작은 가를 검사한다. 만약 0보다 작다면, 전송 큐에 낮은 우선순위 메시지가 존재하지 않는 경우이다. 따라서, 현재 스테이션의 통신을 종료하고 나서, 토큰을 다음 스테이션으로 전송한다.

단계 11. 만약 $Nl_{Q_i}^j$의 값이 0보다 크다면, 하나의 낮은 우선순위 메시지를 전송하고 나서, $T_{E_i}^j$와 $Tl_{F_k}^j$를 계산한다. 여기에서 $Tl_{F_k}^j$는 해당 메시지의 전송이 시작되는 시점에 Tl_C^j를 더하여 계산하고, $Nl_{T_i}^j$의 값을 1증가시킨다. 그리고 단계 6으로 돌아간다.

2.2 시뮬레이션 방법에 의한 전송 지연의 결과

본 절에서는 계산 지연을 계산해 낼 수 있는 시뮬레이션 방법을 프로그래밍하기 위하여 Visual C++을 사용하였다. 본 논문에서 컴퓨터 시뮬레이션은 그림 7의 흐름도에 나타난



1	0 1+1,18	3,500 1+1	4,000 1+1,15	7,500 1+1	8,000 1+1,13	11,500 1+1	12,000 2+1,11	17,250 2+1
2	17,750 1+0,8	19,500 1+0	20,000 2+0,9	23,500 2+0	24,000 2+0,9	27,500 2+0	28,000 1+1,9	31,500 1+1
3	32,000 2+1,10	37,250 2+1	37,750 1+1,8	41,250 1+1	41,750 2+1,8	47,000 2+1	47,500 2+0,7	51,000 2+0
4	51,500 2+1,7	56,750 2+1	57,250 2+1,7	62,500 2+1	63,000 2+1,6	68,250 2+1	68,750 2+1,6	74,000 2+1
5	74,500 2+0,5	78,000 2+0	78,500 2+1,6	83,750 2+1	84,250 2+1,6	89,500 2+1	90,000 3+1,6	97,000 3+1
6	97,500 2+1,5	102,750 2+1	103,250 3+1,4	110,250 3+1	110,750 3+1,3	116,000 3+0	116,500 2+1,3	121,750 2+1
7	122,250 3+1,4	129,250 3+1	129,750 2+1,3	135,000 2+1	135,500 2+2,4	142,500 2+2	143,000 3+1,3	148,250 3+0
8	148,750 2+1,3	154,000 2+1	145,500 3+1,4	161,500 3+1	162,000 3+1,3	167,250 3+0	167,750 2+2,4	174,750 2+2
9	175,250 3+2,3	180,500 3+0	181,000 3+1,3	186,250 3+0	186,750 2+2,4	193,750 2+2	194,250 3+1,3	199,500 3+0
10	200,000 2+3,4	207,000 2+2	207,500 2+2,3	212,750 2+1	213,250 3+1,3	218,500 3+0	219,000 2+2,4	226,000 2+2

그림 5. 도식적 방법을 이용한 전송 지연 계산.

Fig. 5. Calculation of communication delay using diagram-matic method.

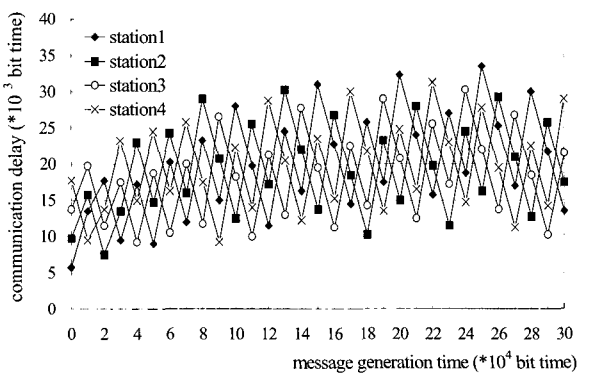


그림 6. 도식적 방법에 의한 전송 지연.

Fig. 6. Communication delay calculated by diagrammatic method.

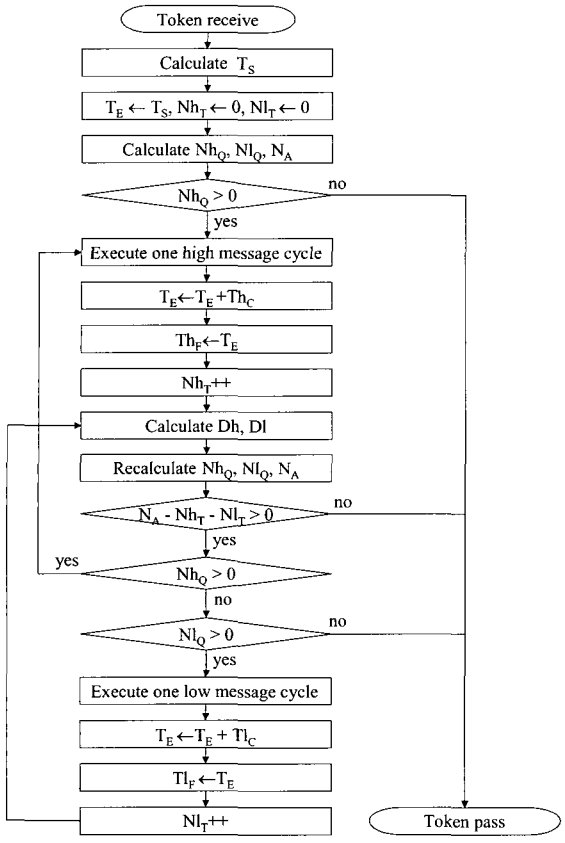


그림 7. 시뮬레이션 방법의 흐름도.

Fig. 7. Flowchart of simulation model for calculating the com munication delay.

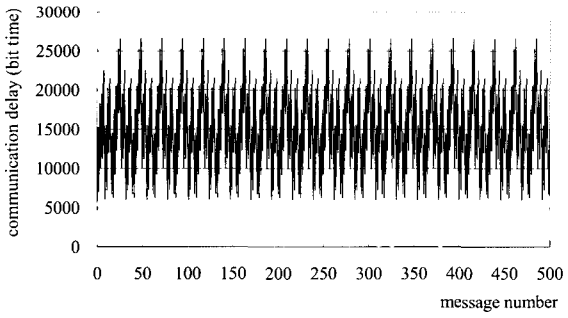


그림 8. 확정적 메시지 생성 주기를 가진 경우의 시뮬레이 방법에 의한 전송지연.

Fig. 8. Communication delay calculated by simulation method with deterministic message generation period.

바와 같은 절차에 따라 각 단계별로 서술된 변수들을 계산할 수 있도록 작성하였다.

또한, 시뮬레이션 방법에서의 도식적인 방법에서 사용한 네트워크 조건과 동일하게 구성하였다. 즉, Profibus-FMS의 전송 속도는 500Kbps, 스테이션의 수는 4대, 높은 우선순위 메시지의 생성 주기는 10,000비트 타임, 낮은 우선 순위 메시지의 생성주기는 25,000비트 타임, T_{TR} 은 30,000 비트 타임으로 설정하였다. 이러한 조건으로 높은 우선순위 메시지가

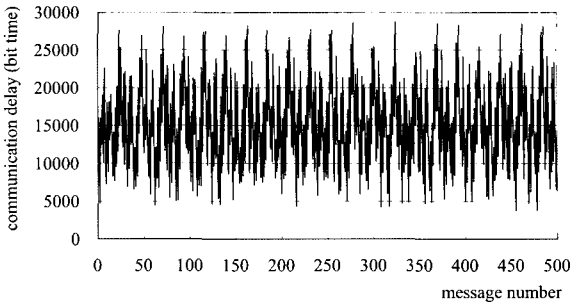


그림 9. 확률적 메시지 생성 주기를 가진 경우의 시뮬레이션 방법에 의한 전송지연.

Fig. 9. Communication delay calculated by simulation method with probabilistic message generation period.

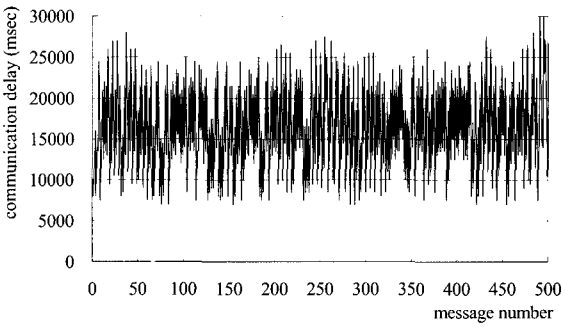


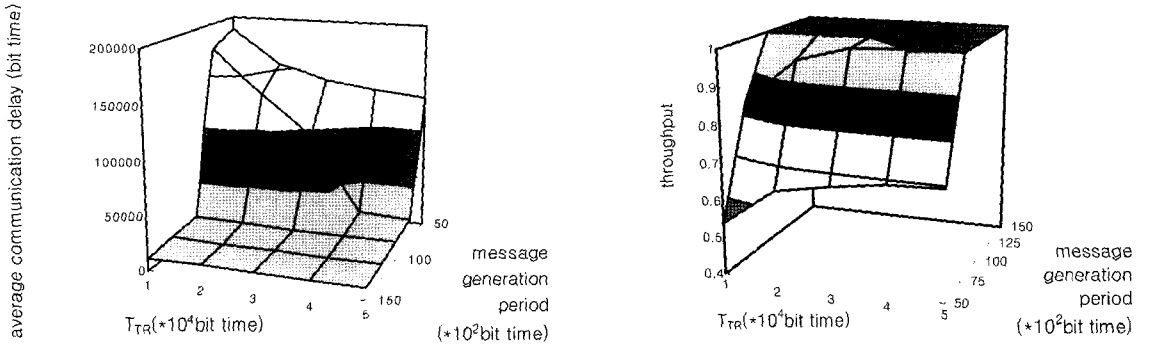
그림 10. 실험 모델에서 측정된 전송 지연.

Fig. 10. Communication delay measured experimental model.

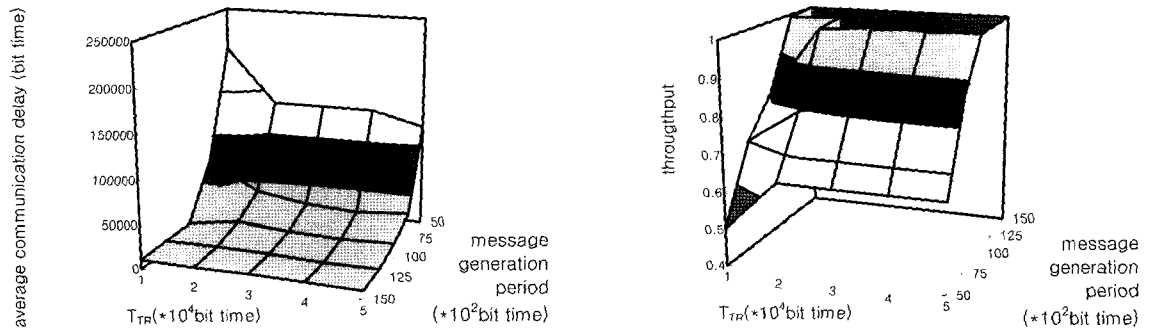
500개 생성되는 동안의 전송 지연을 그림 8에 나타내었다. 그림에서, 높은 우선 순위 메시지의 최대 전송 지연은 26,500 비트 타임, 평균 전송 지연은 15,070비트 타임이며, 전송 지연의 표준 편차는 5,950 비트 타임 임을 관찰할 수 있었다. 또한 전송 지연이 주기적으로 반복됨을 관찰할 수 있었다 이러한 이유는 시뮬레이션 모델의 메시지 생성주기가 확정적이기 때문에 발생하는 것으로 평가되었다. 그림 9는 메시지 생성 주기가 확률적인 경우의 전송 지연을 나타내고 있다. 기본적인 조건은 주기적인 시뮬레이션과 같게 구성하였으나, 높은 우선순위 메시지만 7,500비트 타임에서 12,500비트 타임 사이에서 랜덤 하게 생성되도록 설정하였다. 그 결과, 높은 우선 순위 메시지의 최대 전송 지연은 28,700비트 타임, 평균 전송 지연은15,155비트 타임, 전송 지연의 표준 편차는 6,125비트 타임으로 나타남을 알 수 있었다. 이러한 결과로부터, 확정적인 경우의 메시지 생성주기와 확률적인 경우의 메시지 생성주기의 전송지연이 크게 다르지 않음을 알 수 있었다.

V. 시뮬레이션과 실험 모델의 비교

본 절에서는 Profibus 토큰 패싱 프로토콜의 성능 모델의 유효성을 검증하기 위하여, 실험 모델을 구성하고, 실험 모델의 결과와 성능 모델의 결과를 비교하였다. 실험 모델에는 4대의 트래픽 생성기(traffic generator)와 1대의 성능 관리기 (performance manager)가 접속되어 있다. 여기에서, 트래픽 생성



(a) Average communication delay and throughput by simulation model.



(b) Average communication delay and throughput by experimental model.

그림 11. 메시지 생성 주기와 T_{TR}의 변화에 따른 평균 전송 지연 처리율.

Fig. 11. Average communication delay and throughput with varying message generation time and T_{TR}.

기는 사전에 정해진 메시지 발생 확률에 따라 메시지를 생성시켜 Information Report 서비스를 이용하여 성능 관리기로 전송하고, 성능 관리기는 수신된 메시지를 분석하여 메시지의 전송 지연을 계산한다. 또한, 실험 모델에서, Profibus 인터페이스 카드로는 Siemens사의 ASPC2 및 SPC3 마이크로프로세서가 탑재된 Softing사의 PROFI-board가 사용 되었으며, 전송 속도는 500Kbps로 설정되었다.

이상과 같은 조건에서, 모든 스테이션에서 높은 우선순위 메시지를 7,500비트 타임에서 12,500 비트 타임의 균일 분포로, 낮은 우선순위 메시지를 25,000 비트 타임 주기로 생성시켰다. 또한, 모든 스테이션의 T_{TR} 을 30,000 비트 타임으로 설정하고 나서, 스테이션 1번에서 500개의 메시지에 대한 전송지연을 측정하면 그림 10과 같다.

그림에서, 높은 우선순위 메시지의 최대 전송 지연은 32,000 비트 타임, 평균 전송 지연은 16,370 비트 타임, 표준편차는 5,740 비트 타임임을 알 수 있었다. 이를 그림 9의 시뮬레이션 모델에서 구한 전송 지연과 비교하면, 최대 및 평균 전송 지연이 거의 동일하고, 전송 지연의 전체적인 경향도 유사함을 알 수 있었다.

또한, 본 절에서는 시뮬레이션 모델과 실험 모델에 의한 전송 지연의 경향이 유사함을 확인하기 위하여, 추가적인 실험을 수행하였다. 먼저, 각 스테이션에서의 높은 우선 순위 메시지의 생성 주기를 5,000, 7,500, 10,000, 12,500, 15,000 비트 타임으로 변화시키고, T_{TR} 을 10,000 비트 타임에서 50,000 비트 타임으로 변화시켜 가면서, 스테이션 1번에서의 높은 우선순위 메시지의 평균 전송 지연과 처리율을 측정하였다. 여기에서, 처리율은 생성된 메시지의 개수를 전송에 성공한 메시지의 개수로 나눈 값으로 정의하였다.

그림 11에서, 시뮬레이션 모델과 실험 모델 모두, 각 스테이션에서 트래픽이 동일한 경우에는 T_{TR} 이 어느 정도 이상 (20,000비트 타임) 크다면, 평균 전송 지연이나 처리율의 차이가 크지 않음을 알 수 있다. 또한, 메시지의 발생 주기가 10,000비트 타임이상이 되는 경우, T_{TR} 에 관계없이 평균 전송 지연과 처리율이 거의 변화가 없음을 알 수 있었다. 뿐만 아니라, 시뮬레이션 모델과 실험 모델의 평균 전송 지연과 처리율의 변화 경향도 거의 유사함을 알 수 있었다. 이러한 실험의 결과로부터, Profibus 토큰 패싱 프로토콜에서의 전송 지연을 성능 모델로부터 쉽게 계산 할 수 있으며, 실제 실험모델에서 측정된 값과 매우 유사하게 나타난다는 것을 확인할 수 있었다.

VI. 결론

본 논문에서는 Profibus-FMS 토큰 패싱 프로토콜에서 전송 지연을 수학적으로 계산할 수 있는 성능 모델을 제안하였으며, 전송 지연을 계산하기 위한 방법으로서 도식적인 방법과 시뮬레이션에 의한 방법을 제시하였다. 또한, 본 논문에서

는 실험 모델을 구축하여 그 결과와 시뮬레이션에 의한 결과를 비교함으로써, 제안된 성능 모델의 유효성을 확인하였다. 특히, 네트워크 관리자가 성능 변수를 설계할 때, 선정된 성능 변수를 Profibus-FMS 성능 모델에 적용하여 네트워크 성능의 확인이 가능하도록 하였다.

따라서, 본 논문에서 제안된 성능 모델은 네트워크 관리자가 경험적 방법에 의하여 성능 변수를 선정하는 데 있어서 유효한 도구가 될 것으로 평가된다. 즉, 네트워크 관리자는 선정한 성능 변수를 직접 실제 네트워크 시스템에 적용하기에 앞서, 성능 모델을 통하여 충분한 성능 평가를 거친 후 적용할 수 있게 됨으로써, 네트워크 관리자는 보다 안정되고 최적화된 네트워크의 성능을 얻을 수 있을 것이다.

향후 과제로 시간에 따라 변하는 네트워크 시스템에서 전체 네트워크의 성능을 최적화할 수 있도록 시간에 따라 성능 변수를 조절하는 온라인 성능 관리 방법에 대한 연구가 이루어져야 할 것이다.

참고문헌

[1] 홍승호, "필드버스 기술 동향," 제어 · 자동화 · 시스템 공학회지, 제 4 권, 제 6 호, pp. 13-18, 1998.

[2] 권욱현, 김영신, "분산 실시간 시스템에서의 네트워크 프로토콜," 제어 · 자동화 · 시스템공학회지, 제 4 권, 제 6호, pp. 27-34, 1998.

[3] IEC 61158-4, *Digital data communications for measurement and control - Fieldbus for use in industrial control systems - Part 4: Data link protocol specification*, IEC, 1999.

[4] S. Lee, S. H. Lee, K. C. Lee, M. H. Lee, and F. Harashima, "Intelligent performance management of networks for advanced manufacturing systems", *IEEE Transactions on Industrial Electronics*, vol. 48, no. 4, pp. 731-741, Aug., 2001.

[5] E. Tovar and F. Vasques, "Cycle time properties of the profibus timed-token protocol", *Computer Communications*, vol. 22, pp. 1206-1216, Aug., 1999.

[6] E. Tovar and F. Vaspues, "Real-time fieldbus communications using profibus networks", *IEEE Transactions on Industrial Electronics*, vol. 46, no. 6, pp. 1241-1251, Dec., 1999.

[7] S. Vitturi, "Some features of two feildbuses of the IEC 61158 standard", *Computer Standards & Interfaces*, vol. 22, pp. 203-215, Aug., 2000.

[8] EN 50170, *Profibus Specification - Normative Parts of Profibus-FMS, -DP, -PA according to the European Standard*, vol 2, Profibus International, 1998.

[9] 이성근, 홍승호, "시뮬레이션 모델을 이용한 IEC/ISA 필드버스 시스템의 데이터 링크 계층 성능 분석", 제어 · 자동화 · 시스템공학회논문지, 제 2 권, 제 3 호, pp. 209-219, 1996.

[10] Z. Zhang, A. Burns, "An optimal synchronous bandwidth allocation scheme for guarnteeing synchronous message deadlines with the timed-token MAC protocol", *IEEE/ACM Transactions on Networking*, vol. 3, no. 6, pp.729-741, Dec., 1995.

[11] S. H. Hong and Y. C. Kim, "Implementation of a bandwidth allocation scheme in a token-passing fieldbus network", *IEEE and Measurement*, vol. 51, no. 2, pp. 246-251, April, 2002.

[12] S. Lee, K. C. Lee, M. C. Han, and J. S. Yoon, "On-line fuzzy performance management of profibus networks", *Computers in Industry*, vol. 46, no. 2, pp. 123-137, Sep., 2001.

[13] W. Stalling, *Local and Metropolitan Area Networks, 4th ed.*, Macmillan Publishing Company, New York, 1993.

이 경 창

1971년 5월 1일 생. 1996년 부산대학교생산기계공학과 졸업. 동대학원 석사(1998년), 동대학원 박사 (2003년). 1998년~2003년 기계공학연구정보센터 전임연구원. 2003년~현재 네트워크 기반 자동화 연구센터 전임 연구원. 관심분야는 필드버스, 산업용 이더넷, 차량용 네트워크, 홈 네트워크.

이 석

1961년 12월 11일 생. 1984년 서울대학교 기계공학과 졸업. 펜실바니아 주립대학교 석사(1985년), 동대학원 박사(1990년). 1990년~1993년 신시내티 대학교 기계공학과 조교수. 1993년~현재 부산대학교 기계공학부 부교수. 관심분야는 필드버스, 차량용 네트워크, 홈 네트워트, DES.

김 현 희

1977년 10월 1일 생. 2000년 동명정보대학교 로봇시스템공학과 졸업. 부산대학교 지능기계공학과 석사(2003년), 2003년~현재 동대학원 박사과정 재학 중. 관심분야는 필드버스, 산업용 이더넷, 홈 네트워크.