대역폭 할당 기법에 의한 필드버스 네트워크의 트래픽 관리 및 제어

Management and Control of Fieldbus Network Traffic by Bandwidth Allocation Scheme

홍 승 호
(Seung Ho Hong)

Abstract : Fieldbus is the lowest level communication network in factory automation and process control systems. Performance of factory automation and process control systems is directly affected by the data delay induced by network traffic. Data generated from several distributed field devices can be largely divided into three categories: sporadic real-time, periodic real-time and non real-time data. Since these data share one fieldbus network medium, the limited bandwidth of a fieldbus network must be appropriately allocated to the sporadic real-time, periodic real-time and non real-time traffic. This paper introduces a new fieldbus design scheme which allocates the limited bandwidth of fieldbus network to several different kinds of traffic. The design scheme introduced in this study not only satisfies the performance requirements of application systems interconnected into the fieldbus but also fully utilizes the network resources. The design scheme introduced in this study can be applicable to cyclic service protocols operated under single-service discipline. The bandwidth allocation scheme introduced in this study is verified using a discrete-event/continuous-time simulation experiment.

Keywords: fieldbus, sporadic real-time, periodic real-time, non real-time, traffic, bandwidth, allocation

I. 서론

최근 20여년에 걸쳐 컴퓨터 및 통신 기술의 발전과 더불어 자동화 기술은 급속하게 발전하여 왔다. 미국, 유럽, 일본 등의 기술 선진국에서 개발되고 있는 최근의 첨단 생산 자동화 시스템에서는 컴퓨터를 이용하여 단위 공정의 자동화를 이루고, 이들을 다시 수평적, 수직적으로 통합하여 전체 공정을 일관되게 관리함으로써 생산성 향상과 생산비용의 절감을 비롯하여 생산 공정의 설계, 구축 및 유지 관리에 유연성을 제공하는 등의 효과를 극대화하고 있다[1,2]. 이러한 자동화 시스템을 구축하기 위하여서는 각각의 공정에서 생성되는 정보들을 수집, 분석, 가공 및 저장하고, 또한 각 공정에서 필요한 정보를 적시에 제공하기 위한 정보의 통합화를 구현하는 것이 중요한 문제로 대두된다[3-8]. 정보의 통합화를 구현하기 위하여서는 생산 현장 내에 모든 정보의 흐름을 파악하여 각각의 자동화 요소에 필요한 정보를 적시에 제공할 수 있는 기능이 부과되어야 한다. 네트워크는 자동화 요소들 간에 정보 교환을 가능케 함으로써 첨단 생산자동화 시스템에서 중추 및 신경 기능을 담당하고 있다. 따라서, 궁극적으로 모든 공정의 통합화를 추구하는 미래의 생산자동화 시스템에 있어서 네트워킹 기술은 가장 핵심이 되는 기술 가운데 하나라고 볼 수 있다.

1980년대 초반부터 생산자동화 환경에서 이기종의 자동화 장비들 간의 통신을 위한 표준화된 네트워킹 시스템으로 MAP(Manufacturing Automation Protocol)[9,10]이 개발되었다. 그러나 MAP은 구조상 OSI(Open Systems Interconnection) reference model에서 제시하고 있는 7계층을 모두 가지고 있어 생산 현장의 필드에 설치된 각종 필드 장비들 간의 실시간 통신을 지원하기에는 적합하지 않은 시스템으로 인식되고 있다. 1980년대 후반부터는 생산 현장의 필드에 설치된 각종 제어 및 자동화 관련 장비들에서 생성되는 데이터들의 실시간 통신을 지원하며 가격이 저렴한 네트워크 시스템의 필요성이 제기되었으며, 이러한 목적을 위하여 개발된 네트워크 시스템이 필드버스이다. 필드버스는 필드에 설치된 각종 센서, 단일루프제어기, 소형 PLC, 모터, 밸브, 로봇, CNC 등의 공작 기계를 비롯하여 이러한 장비들을 제어하는 다중루프 제어기, 중대형 PLC 등의 자동화 기기에서 생성되는 데이터를 실시간으로 처리하며, 따라서 첨단 생산시스템의 네트워크 구조상 가장 기본이 되는 네트워크이다.

필드버스에 접속되는 각종 필드 장비들로부터 생성되는 데이터는 크게 산발적 실시간, 주기적 실시간 및 비실시간 데이터의 세 가지 종류로 구분된다. 필드버스 시스템에서는 특성이 다른 이러한 데이터들이 하나의 네트워크 미디엄을 공유하게 된다. 따라서 필드버스 네트워크에서 이러한 데이터들의 트래픽이 제대로 관리 및 제어되지 못하는 경우에는 산발적 및 주기적 실시간 데이터의 지연 시간이 미리 정해진 한계치를 초과할 수 있다. 즉, 데이터 지연 시간의 초과로 인하여 필드버스에 접속된 각종 응용 시스템들의 성능 요구 사항을 만족시키지 못하는 경우가 발생한다. 이러한 경우에 대비하여 필드버스 네트워크의 대역폭은 산발적 및 주기적 실시간과 비실시간 데이터들에게 적절히 할당되어 필드버스에 접속되는 각종 응용 시스템들의 성능 요구 사항을 만족시키도록 설계되어야 한다.

필드버스는 최근에 개발된 시스템이며, 따라서 아직은 필드버스 시스템의 설계에 관한 논문이 많이 발표되고 있지 않다. Cavalieri et. al.[11]은 IEC/ISA 필드버스[12]에서 우선 순위 기능을 사용하는 경우에 각각의 우선 순위 데이터들에 대한 대역폭 할당 기법을 제시하였다. 또한, Shin과 Chou[22]의 논문에서는 여러 개의 필드버스 링크들이 브리지를 통하여 연결되는 경우에 실시간 데이터 처리 기법에 대하여 기술하였다. Hong[13]은 제어 시스템의 센서와 콘트롤러 등과 같이 주기적으로 데이터를 생성하는 장비들이 필드버스에 접속되는 경우에 필드버스의 트래픽 관리 및 제어

접수일자 : 1996. 2. 15., 수정완료 : 1997. 1. 23.
홍승호 : 한양대학교 제어계측공학과
※ 이 연구는 한국과학재단 연구비 지원에 의한 결과임

기법을 제시하였다. 그밖에 필드버스에서 실시간 데이터 처리 기법을 다룬 논문으로는 [23]이 있다. Hong[13]에서 제시된 기법의 기본 개념은 주기적 데이터의 생성 주기(또는 제어 시스템의 관점에서 보면 제어 루프의 샘플링 주기)를 적절히 선정함으로써 제어 시스템의 성능 요구 사항을 만족시키는 동시에 네트워크 자원의 이용도를 최대화하는 것이다. 본 논문은 Hong[13]의 연구 결과를 일반적인 필드버스 시스템으로 확장하여, 자동화 및 분산제어 시스템에서 서로 다른 특성을 갖는 산발적 실시간, 주기적 실시간 및 비실시간 데이터가 공존하는 경우에 각각의 데이터의 트래픽 상태에 따라 필드버스 네트워크에서 제공하는 대역폭을 적절히 할당함으로써 필드버스에 접속된 응용 시스템들의 성능 요구 사항을 만족시키는 동시에 대역폭 이용도를 최대화할 수 있는 새로운 기법을 제시한다.

대부분의 필드버스 시스템에서 채택하고 있는 미디엄 접속 제어 프로토콜은 크게 (i) 중앙 제어 방식의 마스터-슬레이브 시스템(예, MIL-STD-1553B[14], FIP[15], Bitbus [17])과 (ii) 분산 제어 방식의 토큰-패싱 시스템(예, Profibus[16], Arcnet[17], SAE HSDB[18])으로 나뉘어질 수 있다. 그 밖에 마스터-슬레이브와 토큰-패싱의 혼합 시스템에 스케듈링 방식을 도입한 IEC/ISA 필드버스[12]가 있으나 이는 아직 규격이 완성되지 않은 상태에 있다. 이러한 프로토콜들은 상호간에 장단점은 있으나, 필드버스의 주기적 실시간, 산발적 실시간 및 비실시간 데이터 트래픽의 요구 조건을 완벽히 만족시키지 못하고 있다. IEC/ISA 필드버스가 주기적 실시간, 산발적 실시간 및 비실시간 데이터 트래픽의 요구 조건을 만족시킬 수 있도록 규격이 작성되고 있으나, 이를 실제로 구현하기에는 프로토콜이 너무 복잡하다는 단점을 가지고 있다. 본 논문에서는 필드버스에서 대역폭 할당 기법을 통하여 주기적, 산발적 실시간 및 비실시간 데이터 트래픽을 처리할 수 있는 트래픽 관리 및 제어의 원칙적인 방법론을 제시한다. 본 논문에서 제시하는 대역폭 할당 기법은 단일-순환 서비스를 채택하는 모든 미디엄 접속 제어 프로토콜에 적용될 수 있다. 여기서 단일-순환 서비스란 서버가 미리 정해진 순서에 따라 필드버스에 접속된 각각의 노드를 차례로 한 번씩 방문하며, 각각의 노드의 전송큐에서는 서버가 도착하는 순간에 대기하고 있는 데이터를 한 번에 한 개씩 FIFO(First-In-First-Out)방식으로 전송하는 방식을 말한다. 본 논문에서 제시하는 트래픽 관리 및 제어 기법은 이러한 기법이 적용되는 새로운 프로토콜을 개발할 수도 있고, 기존의 마스터-슬레이브 및 토큰-패싱 방식으로 동작되는 필드버스의 프로토콜을 적절히 변형하여 적용할 수도 있다.

본 논문의 II장에서는 필드버스 네트워크를 통하여 전송되는 데이터 트래픽의 특성에 관하여 기술한다. III장에서는 필드버스 네트워크의 트래픽을 관리 및 제어하기 위한 대역폭 할당 기법이 기술되고 이와 더불어 이를 실제로 구현하기 위한 알고리듬이 제시된다. 대역폭 할당 기법을 이용한 필드버스 네트워크 시스템 설계의 예와 더불어 본 논문에서 제시하는 대역폭 할당 기법의 타당성 검증이 IV장에서 제시되고, 마지막으로 본 논문의 결론 및 추후 연구 사항이 V장에 기술된다.

II. 필드버스 네트워크의 트래픽 특성

공장자동화 및 분산제어 시스템의 필드에 설치된 각종 장비에서 생성되는 데이터는 크게 주기적 실시간, 산발적 실시간 및 비실시간 데이터로 구분할 수 있으며, 이들의 특징은 다음과 같다.

(1) 주기적 실시간 데이터 : 주기적 실시간 데이터는 주로 피드백 제어 루프들로부터 생성된다. 제어 루프는 콘트롤러와 플랜트로 구성되며, 이러한 장비들에서는 주기적으로 데이터가 샘플링된다. 샘플링된 데이터는 해당 노드의 전송큐에 삽입되어 서버(즉, 또는 폴링 신호)에 의하여 데이터 전송 권한이 주어질 때까지 기다린다. 이러한 전송큐에서의 대기 시간과 데이터를 전송하는데 소요되는 시간이 네트워크에서 데이터 전송 지연 시간을 야기한다. Ray et. al.[19]에 의한 시뮬레이션 연구 결과를 통하여 이미 보고된 바와 같이 이러한 데이터 지연 시간은 매번 데이터를 전송할 때마다 불규칙하게 변하는 특성을 가지고 있다. 만일에 이러한 데이터 지연 시간이 샘플링 주기보다 커지는 경우에는 전송큐에서의 메시지 제거 및 공허 샘플링 등의 바람직하지 못한 현상들로 인하여 제어 루프의 성능이 크게 저하되는 동시에 제어 신호의 고주파 잡음으로 인하여 액튜에이터의 마모가 가속화될 수 있다[20]. 따라서 필드버스에 접속되는 제어 루프에서 주기적으로 발생되는 제어 데이터는 (i) 네트워크의 데이터 지연 시간이 제어 시스템의 샘플링 주기를 초과하지 않도록 하는 동시에, (ii) 센서에서 데이터를 샘플링한 시점에서 시작하여, 센서 데이터를 통하여 콘트롤러에서 생성된 제어 신호가 액튜에이터에 도달하는 시점까지의 경과 시간으로 정의되는 루프 지연 시간이 미리 정해진 한계치를 초과하지 않도록 설계되어야 한다[13]. 주기적 데이터의 발생 빈도는 다른 데이터들에 비하여 매우 높고, 데이터 패킷의 길이는 짧다.

(2) 산발적 실시간 데이터 : PLC를 이용한 순차 제어를 비롯하여 경고 신호 등과 같이 각종 사건의 발생을 알려주기 위한 데이터들은 산발적으로 발생한다. 이러한 데이터들은 데이터 발생 빈도가 상대적으로 낮고, 데이터 길이가 짧으며, 불규칙하게 생성되는 반면에 매우 제한된 시간 이내에 전송이 완료되어야 하는 특성을 가지고 있다. 이러한 데이터들은 가장 높은 우선 순위를 가지고 전송되어야 한다.

(3) 비실시간 데이터 : 제어 및 자동화 관련 프로그램과 데이터 파일의 전송을 비롯하여 데이터 베이스 관리를 위한 메시지 전송 등과 같은 데이터들은 전송 지연 시간에 크게 구속을 받지 않는다. 이러한 데이터들은 필드버스가 제공하는 네트워크 대역폭 가운데 산발적 실시간 데이터와 주기적 실시간 데이터에 할당되고 남은 대역폭을 사용하여 전송되어야 한다. 비실시간 데이터는 데이터 발생 빈도가 낮고, 비주기적으로 발생하며, 데이터의 크기도 상대적으로 매우 크다. 길이가 긴 비실시간 데이터는 여러 개의 패킷으로 분리되어 전송된 후, 수신단에서 재조립된다.

필드버스에서 서로 다른 특성을 가지고 있는 산발적 실시간, 주기적 실시간 및 비실시간 데이터들의 전송을 동시에 처리하기 위한 대역폭 할당 기법을 제시하기 위하여 N개

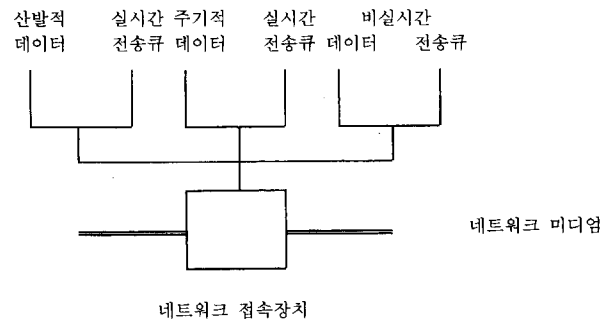


그림 1. 노드에서의 데이터 전송큐.
Fig. 1. Transmitter queues in a node.

의 노드로 구성된 필드버스 네트워크를 생각하자. 그림 1에 나타난 바와 같이 만일에 한 노드에서 여러 형태의 데이터들이 동시에 생성된다면, 이들에 대한 데이터 전송큐는 각각 따로 설치되어야 한다. 서버가 노드에 도착하면 이는 산발적 실시간, 주기적 실시간 및 비실시간 데이터 전송큐의 순서로 각각의 전송큐를 방문하여 대기하고 있던 데이터들을 전송한다. 이 때 서버는 해당 전송큐에서 FIFO(First-In-First-Out) 방식으로 한 번에 한 개씩만의 데이터를 전송하는 단일 서비스(single service discipline)를 제공하는 것으로 가정한다.

필드버스 네트워크에는 M개의 제어 루프들이 접속된 것으로 가정하자. 각각의 제어 루프는 플랜트의 센서 노드와 콘트롤러 노드의 두 개의 데이터 전송 노드를 갖는다(플랜트의 액튜에이터 노드는 일반적으로 네트워크를 통하여 데이터를 전송하지 않는다). 따라서 필드버스 네트워크 내에 주기적 실시간 데이터를 생성하는 노드의 수는 $N_p=2M$이다. 제어 루프 i에 속한 센서와 콘트롤러 노드는 T_i의 동일한 샘플링 주기를 갖는다. 공정 제어에 관련된 주기적 실시간 데이터는 일반적으로 짧은 길이를 갖는다. 따라서, 본 논문에서 주기적 실시간 데이터는 $\overline{L}_p$ 비트의 일정한 길이로 패킷화되는 것으로 가정한다. 만일에 네트워크의 데이터 전송 속도가 B 비트/초라면, 패킷화된 주기적 실시간 데이터의 전송 시간은 $L_p=\overline{L}_p/B$ 초이다. Ray et. al.[19]의 시뮬레이션 결과에서 나타난 바와 같이 제어 루프에서 콘트롤러는 가장 최근에 생성된 센서 데이터를 이용하여 제어 신호를 발생시켜야 하며, 따라서 주기적 실시간 데이터 전송큐의 큐 용량은 하나로 제한한다.

본 연구에서 필드버스에 접속되는 산발적 실시간 및 비실시간 데이터 전송큐의 개수는 각각 N_c 및 N_a개로 구성되는 것으로 가정한다. 산발적 실시간 데이터는 매우 짧은 길이를 갖는다. 따라서 산발적 실시간 데이터 역시 $\overline{L}_c$ 비트의 일정한 길이로 패킷화되는 것으로 가정하며, 산발적 실시간 데이터의 전송 시간은 $L_c=\overline{L}_c/B$ 초이다. 산발적 실시간 데이터는 노드 j에서 λ_c^j의 빈도로 Poisson 분포를 가지며 도착하는 것으로 가정한다. 비실시간 데이터의 길이는 상대적으로 매우 길다. 이러한 비실시간 데이터는 전송단에서 여러 개의 패킷으로 분리되어 한 번에 한 개씩의 패킷이 전송되며, 이러한 패킷들은 수신단에서 재조립된다. 비실시간 데이터 패킷의 길이는 $\overline{L}_a$ 비트이고, 패킷의 전송 시간은 $L_a=\overline{L}_a/B$ 초이다. 만일에 노드 j에서 생성되는 비실시간 데이터의 전송시간이 $L_a^j(>L_a)$이면, 이는 $p_j=\lceil L_a^j/L_a \rceil$개의 패킷으로 분리된다. 각각의 노드 j에서 비실시간 데이터는 Λ_a^j의 빈도로 Poisson 분포를 가지며 도착하는 것으로 가정하며, 따라서 노드 j에서 비실시간 데이터 패킷의 단위시간 당 평균 도착 개수 λ_a^j는 $p_j\Lambda_a^j$이다. 산발적 실시간 및 비실시간 데이터 전송큐의 용량은 데이터 패킷이 전송큐 용량의 포화 상태로 인한 메시지 제거 현상이 발생하지 않도록 충분히 크다고 가정한다.

III. 필드버스의 대역폭 할당 기법

본 장에서는 II 장에서 기술된 필드버스 네트워크 시스템의 대역폭 할당 기법을 제시한다. 먼저 본 장의 1 절에서는 비교적 단순한 경우인 주기적 실시간 데이터와 비실시간 데이터 트래픽으로 구성된 필드버스 네트워크 시스템의 대역폭 할당 기법에 대하여 기술하며, 2 절에서는 1 절에서 제시된 개념을 확장하여 산발적 실시간, 주기적 실시간 및 비실시간 데이터 트래픽으로 구성된 일반적인 경우의 필드버스 네트워크의 대역폭 할당 기법을 제시한다.

1. 주기적 실시간/비실시간 데이터 트래픽

주기적인 데이터 트래픽만으로 구성된 필드버스 네트워크 시스템의 대역폭 할당 기법은 Hong[13]에 의하여 제시되었다. 본 절에서는 Hong이 제시한 기법을 주기적 실시간 및 비실시간 데이터 트래픽으로 구성된 필드버스 네트워크 시스템으로 확장하고자 한다. 본 논문에서 제시하는 대역폭 할당 기법의 기본 개념은 필드버스에서 제공하는 네트워크 대역폭을 그림 2에 나타난 바와 같이 주기적 구간(τ_p)과 비실시간 구간(τ_a)으로 구분하는 것이다. 각각의 구간에서 주기적 실시간 및 비실시간 데이터는 시분할(time-division multiplexing) 방식으로 해당 대역폭에 할당되어, 주기적 대역폭 구간에서는 주기적 실시간 데이터만이 전송되며, 비실시간 대역폭 구간에서는 비실시간 데이터만이 전송되도록 한다.

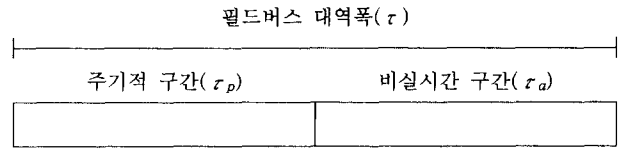


그림 2. 필드버스 대역폭 할당: 주기적 실시간/비실시간 트래픽.

Fig. 2. Fieldbus bandwidth allocation: periodic/non real-time traffic.

제어 루프의 센서 및 콘트롤러에서 생성되는 주기적 실시간 데이터는 Hong[13]에서 제시된 바와 같이 다음의 두 가지 조건이 만족되도록 설계되어야 한다.

(C1) 데이터 제거 및 공허 샘플링 현상이 발생하지 않아야 한다. 즉,

$$\theta_i^{sc} < T_i,\ \theta_i^{ca} < T_i, \quad \forall\ i=1 \text{ 에서 } M. \tag{1}$$

여기서, θ_i^{sc}는 센서 노드에서 센서 데이터가 전송큐에 삽입된 순간부터 콘트롤러 노드가 새로운 센서 데이터를 완전히 수신할 때까지의 데이터 전송 지연 시간이며, θ_i^{ca}는 콘트롤러 노드에서 콘트롤러 데이터가 전송큐에 삽입된 순간부터 액튜에이터 노드가 새로운 제어 신호를 완전히 수신할 때까지의 데이터 전송 지연 시간이고, T_i 는 i번째 제어 루프에서 샘플링 주기이다.

(C2) i번째 제어 루프에서의 루프지연시간 D_i가 미리 정하여진 최대 한계값 Φ_i를 초과하지 않아야 한다. 즉,

$$D_i \le \Phi_i, \quad \forall\ i=1 \text{ 에서 } M. \tag{2}$$

루프 지연 시간 D_i는 센서 노드에서 데이터를 샘플링한 시점에서 시작하여, 새로운 센서 데이터를 통하여 콘트롤러 노드에서 생성된 제어 신호가 액튜에이터에 완전히 도달하는 시점까지의 경과 시간이며, 이는 Hong[13]으로부터 다음과 같이 나타내어질 수 있다.

$$D_i=\left\lceil \frac{\theta_i^{sc}}{T_i} \right\rceil T_i+\theta_i^{ca} \tag{3}$$

(3)에서 θ_i^{sc}와 θ_i^{ca}가 시간에 대하여 변하는 변수이기 때문에 D_i 역시 시간에 대하여 가변한다. 시간에 대하여 불규칙하게 가변하는 지연 시간을 갖는 제어 시스템에 대한 해석은 지금까지 매우 어려운 문제로 남아 있다[20]. 따라

서, 주어진 제어루프에 대하여 최대 허용 루프 지연 시간 Φ_i를 구하는 문제는 간단하지가 않다. 그러나 Hong[13]에서 제시된 바와 같이 가변하는 루프 지연 시간 D_i를 아래의 (4)와 같이 시간에 대하여 일정한 루프 지연 시간 D_i'로 대체한다면 제어 시스템의 주어진 성능 요구 사항을 만족시키는 최대 허용 루프 지연 시간 Φ_i는 기존의 Bode 및 Nyquist 등의 주파수 영역 해석 기법을 통하여 쉽게 얻을 수 있다.

$$D_i' = 2T_i + (\sup \theta_i^{ca} - \min \theta_i^{ca}) \tag{4}$$

위 (4)에 주어진 일정한 루프 지연 시간은 가변하는 루프 지연 시간이 가변 범위 내에서 제어 시스템의 안정도에 미치는 영향보다 제어 시스템을 더욱 안정되도록 한다[13]. (4)로부터 주어진 제어 시스템에 대한 최대 허용 루프 지연 시간 Φ_i($i=1$ 에서 M)은 미리 구하여질 수 있으며, 주기적 구간 τ_p 내에서 필드버스의 대역폭 할당 문제는 제어 시스템의 성능에 대한 조건 (C1)과 (C2)를 만족할 수 있는 주기적 실시간 데이터의 도착 주기(또는 제어 시스템의 관점에서 보면 데이터 샘플링 주기) T_i($i=1$ 에서 M)를 구하는 문제로 변환된다.

주기적 구간에서 주기적 실시간 데이터들에 대한 네트워크 대역폭 할당 문제는 Hong[13]에서 제시된 윈도우 스케듈링 알고리듬을 변형하여 적용할 수 있다. 윈도우 스케듈링 알고리듬에서는 주기적 구간 τ_p를 r개의 윈도우로 나누어 각각의 윈도우에 주기적 실시간 데이터를 실어 보낸다. 따라서 각 윈도우의 길이는 주기적 실시간 데이터의 전송 시간 L_p와 동일하다. 윈도우 스케듈링 알고리듬의 기본 개념은 N_p개의 주기적 실시간 데이터를 생성하는 노드가 r ($r \le N_p$)개의 윈도우를 서로 다이내믹하게 공유하도록 하여 τ_p의 주기적 구간 동안 샘플링 되는 데이터의 개수가 r개를 초과하지 않도록 각 제어 루프에서의 샘플링 주기 T_i($i=1$ 에서 M)를 스케듈링 하는 것이다. 이 경우에 τ_p의 시간 동안에 생성된 모든 데이터들은 같은 시간 내에 모두 전송이 완료될 것이며, 따라서 네트워크에 접속된 모든 제어 루프에서 데이터의 전송 지연 시간은 τ_p를 초과하지 않는다.

Hong[13]의 경우에서와 같이 필드버스의 트래픽이 주기적 실시간 데이터만으로 구성되는 경우에는 먼저 M개의 제어 루프들 가운데 샘플링 주기가 가장 작은 제어 루프의 샘플링 주기 T_1을 결정한 후 주기적 구간 τ_p를 T_1과 동일하게 하여 T_1 내의 윈도우 개수 r을 미리 구할 수 있었다. 그러나 주기적 실시간 트래픽과 비실시간 트래픽이 공존하는 경우에는 윈도우의 개수 r(또는 주기적 구간 τ_p)를 미리 결정할 수 없다. 왜냐하면 대역폭 할당을 위한 필드버스 설계의 초기 단계에서는 필드버스에서 제공하는 대역폭 가운데 얼마만큼을 주기적 실시간 트래픽에 할당하여야 할 것인가를 결정할 수 없기 때문이다. 따라서 Hong[13]에서 제시된 윈도우 스케듈링 알고리듬은 다음과 같이 변형되어야 한다.

먼저 $\boldsymbol{\Phi}$를 다음과 같이 M개의 제어루프의 최대 허용 루프 지연 시간 Φ_i로 구성되는 벡터라 하자.

$$\boldsymbol{\Phi} = [\Phi_1, \Phi_2, \ldots, \Phi_M], \ \Phi_i \le \Phi_{i+1} \tag{5}$$

여기서 Φ_i는 크기가 같거나 증가하는 순서로 배열된다. 또한 벡터 $\boldsymbol{T}$와 $\boldsymbol{K}$를 다음과 같이 정의하기로 한다.

$$\boldsymbol{T} = [T_1, T_2, \ldots, T_M], \ T_i \le T_{i+1}, \ \forall i \tag{6}$$

$$\boldsymbol{K} = [k_1, k_2, \ldots, k_M], \ k_i = \frac{T_i}{T_1}, \ k_i \le k_{i+1}, \ \forall i \tag{7}$$

벡터 $\boldsymbol{T}$의 성분은 M개의 제어 루프에서의 샘플링 주기로 구성되고, 벡터 $\boldsymbol{K}$의 성분은 최소 샘플링 주기 T_1에 대한 각각의 제어 루프의 샘플링 주기 T_i의 비를 나타내며, 이러한 성분들 역시 크기가 같거나 증가하는 순서로 배열된다.

Hong[13]으로부터 주기적 구간 내에서 제어 루프 1에서의 샘플링 주기 T_1과 제어 루프 i($i=2$에서 M)의 샘플링 주기 T_i는 다음과 같이 결정된다.

$$T_1 = \frac{(\Phi_1 + L_p)}{3} \tag{8}$$

$$T_i = k_i T_1, \ k_i = \left\langle \frac{\Phi_i - (T_1 - L_p)}{2T_1} \right\rangle, \ \forall i = 2 \text{ 에서 } M. \tag{9}$$

여기서 $y = \langle x \rangle$는 y 가 x를 초과하지는 않으나 x와 가장 가까운 2의 멱(즉, 2^{n_i}, $n_i \in \{0, 1, 2, \ldots\}$)으로 정의된다. 또한 노드 j($j=1$ 에서 N_p)에서 첫 번째 데이터를 생성하는 순간 t_j는 다음과 같이 결정된다.

$$t_j = \inf [A_l \ge A_{l-1} : u^j(A_l) \le r], \quad j=1 \text{ 에서 } N_p, \ l=1 \text{ 에서 } k_M. \tag{10}$$

여기서 A_l은 T_M의 구간에서 l-번째 T_1 슬롯이 시작되는 순간이며, T_M은 (6)에 나타난 바와 같이 M개의 제어 루프들 가운데 최대 샘플링 주기를 갖는 제어루프의 샘플링 주기이다. 또한 $u^j(A_l)$은 A_l의 순간에 노드 1부터 노드 j까지 샘플링된 데이터의 개수이다. t_j가 (10)에서와 같이 주어지면 모든 주기적 구간 τ_p에서 생성되는 데이터의 개수는 r을 초과하지 않는다. 즉,

$$u^{N_p}(A_l) \le r, \quad \forall \ l=1 \text{ 에서 } k_M. \tag{11}$$

α_K를 최소 샘플링 주기 T_1의 시간 동안에 M개의 제어 루프에서 생성되는 데이터 개수의 평균값이라 하자. 즉,

$$\alpha_K = 2 \sum_{i=1}^{M} \frac{1}{k_i}. \tag{12}$$

(12)로부터 주기적 구간 τ_p 내에 윈도우의 개수 r은 다음과 같이 결정된다.

$$r = \lceil \alpha_K \rceil \tag{13}$$

필드버스의 대역폭은 그림 3에 나타난 바와 같이 T_1 길이의 시분할 대역폭으로 분할되며, T_1의 시분할 대역폭 내에서 τ_p의 주기적 구간은 (13)으로 주어진 r개의 윈도우로 구성된다. 윈도우의 크기는 주기적 실시간 데이터 전송 시간과 동일한 L_p의 길이를 가진다. 그림 3에서 σ는 각 노드에서 서버 처리 시간과 서버 최대 전송 시간을 포함하는 서버 오버헤드이다($\sigma \ll L_p$). 그림 3에서 주기적 구간의 최댓값은 $rL_p + R$이며, 여기서 $R = N\sigma$는 서버가 네트워크 내의 모든 노드를 한 번씩 방문하는데 소요되는 서버 순환

시간이다. 만일 $rL_p+R>T_1$인 경우에는 T_1의 주기 동안에 생성된 주기적 실시간 데이터 가운데 일부는 같은 기간 동안에 전송이 되지 못하고 전송큐에서 제거되는 현상이 발생할 수 있다. 따라서 주기적 실시간 트래픽이 안정화되기 위한 조건은 다음과 같이 주어진다.

$$rL_p+R \le T_1 \tag{14}$$

만일에 주어진 필드버스 네트워크가 (14)의 조건을 만족하지 못하는 경우에는 네트워크 설계자는 필드버스에 접속되는 제어 루프의 개수 M을 제한하여야 한다.

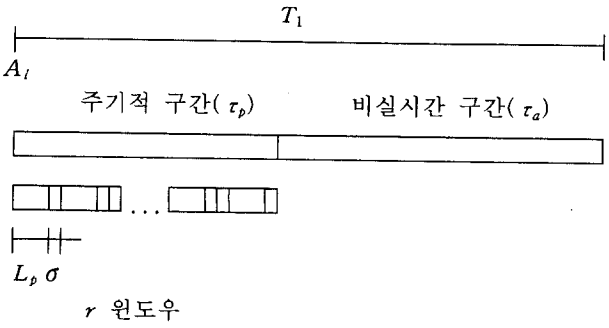


그림 3. 주기적 구간에서 대역폭 할당.
Fig. 3. Bandwidth allocation in the periodic interval.

그림 3의 T_1의 대역폭 구간에서 주기적 구간 τ_p를 제외한 나머지 대역폭은 비실시간 데이터의 전송을 위하여 할당된다. 비실시간 구간 τ_a에서는 비실시간 데이터들만이 전송되며, 이러한 데이터는 전송 지연 시간에 크게 구애를 받지 않는다. 어느 노드에서든지 주기적 실시간 데이터가 생성되면 비실시간 구간 τ_a는 중지되고 네트워크 대역폭은 다시 주기적 실시간 데이터의 전송을 위한 주기적 구간 τ_p로 전환된다. 이를 위하여 서버의 FC(Frame Control) 영역에 1 바이트 길이의 PB(Periodic Byte)를 추가한다. PB의 첫 번째 비트 PSB(Periodic Service Bit)는 네트워크 내의 모든 노드들에게 비실시간 구간이 종료되고 새로운 주기적 구간이 시작되었음을 알리는데 사용되며, 나머지 7비트는 어느 노드로부터 주기적 구간이 시작되었는가를 기록하는데 사용된다.

서버(토큰 또는 마스터 노드의 폴링 신호)는 미리 지정된 순서에 따라 필드버스 내의 모든 노드들을 차례로 방문한다. 서버가 도착하면 노드는 먼저 주기적 실시간 데이터 전송큐에 전송될 메시지가 대기하고 있는가를 검사한다. 만일에 서버가 도착한 순간에 주기적 실시간 데이터 전송큐에 데이터가 대기하고 있고 서버의 PSB가 0의 값을 가지고 있으면, 노드는 PSB를 1로 변환하고 PB의 나머지 7비트 영역에 해당 노드의 주소를 기록한다. 해당 노드는 비실시간 데이터 전송큐에 데이터가 대기하고 있는가에 관계없이 주기적 실시간 데이터만을 전송한 후 서버를 다음 노드(토큰-패싱 시스템의 경우) 또는 마스터 노드(마스터-슬레이브 시스템의 경우)에게 넘긴다. 서버는 네트워크 미디엄을 통하여 전파되기 때문에 네트워크 내의 모든 노드들은 서버의 PSB를 통하여 주기적 구간이 시작되었음을 인지한다. 이때부터 주기적 구간 τ_p가 시작되며, 네트워크 내의 모든 노드들은 서버가 도착하면 PSB가 다시 0으로 변환될 때까지 주기적 실시간 데이터만을 전송하고 비실시간 데이터는 전송하지 않는다. 주기적 구간 τ_p 동안 N_p개의 주기적 실시간 데이터 전송큐 가운데 r개 이하의 데이터만이 생성되도록 스케듈링 되어 있으므로 서버가 네트워크 내의 모든 노드들을 순환하는 동안에 이러한 주기적 실시간 데이터들은 전송이 완료된다. 네트워크 내의 각 노드는 서버가 도착할 때마다 PB에 기록된 주소와 자기 노드 주소를 비교하며, 서버가 PSB를 1로 변환한 노드로 돌아오게 되면 해당 노드는 PB의 모든 비트를 0으로 변환한다. 이 때부터 네트워크 내의 모든 노드들은 비실시간 구간이 시작되었음을 인지하게 되어 서버가 노드에 도착하면 대기하고 있던 비실시간 데이터 패킷을 전송한다. 이러한 동작은 서버의 FC 영역에 1 바이트 길이의 PB 만을 추가하면 되므로 매우 간단히 구현될 수 있다.

그림 3에서 τ_p는 반드시 T_1 주기 내에 존재하여야 하나 τ_p의 시작 시점이 반드시 A_1과 일치할 필요는 없다. 왜냐하면 대부분의 경우에 주기적 데이터가 생성된 시점 A_1에 서버는 다른 노드에 위치하고 있을 것이고 서버가 현 위치에서 주기적 데이터가 생성된 노드로 전달되기까지의 시간이 소요되기 때문에 주기적 데이터가 생성된 시점 A_1과 주기적 구간 τ_p가 시작되는 시점이 일치하지 않기 때문이다. 따라서 주기적 구간 τ_p는 반드시 T_1 주기 내에 존재하여야 하나, τ_p가 일정한 주기로 반복될 필요는 없다.

그림 3에서 비실시간 구간 τ_a의 길이는 $T_1-\tau_p$이다. T_1의 구간에서 생성된 최대 r개의 주기적 실시간 데이터의 전송 지연 시간이 T_1을 초과하지 않도록 하기 위하여서는 비실시간 데이터의 패킷 전송 시간 L_a를 제한하여야 한다. 그림 4에는 A_1의 순간에 생성된 주기적 실시간 데이터의 전송 지연 시간이 T_1을 초과하지 않는 한계 상황이 나타나 있다. 그림 4에서 A_1의 순간에 최대 r개의 주기적 실시간 데이터가 생성된다. 이러한 주기적 실시간 데이터의 전송 지연 시간이 T_1을 초과하지 않도록 하기 위하여서는 같은 시간 내에 필드버스 내의 모든 노드들이 최소한 한 번의 데이터 전송 기회를 가질 수 있어야 한다. 즉, 서버 순환 시간이 T_1을 초과하지 않도록 하여야 한다. C_b와 C_e를 각각 주기적 실시간 데이터가 도착하는 시점을 기준으로 하여 한 번의 서버 순환 시간이 시작되는 시점과 종료되는 시점이라 하자. C_b와 C_e의 시점 사이에 서버는 네트워크 내의 N개의 노드를 모두 방문한다. 각각의 노드에서 주기적 실시간 및 비실시간 데이터 전송큐는 서버가 도착하면 한 번에 한 개씩의 데이터 패킷을 전송할 수 있으므로 C_b와 C_e의 시점 사이에서 서버가 최대한으로 방문할 수 있는 주기적 실시간 및 비실시간 데이터 전송큐의 개수는 각각 r과 $(N-r)$이다. 그러나 A_1의 순간에 한 노드에서 주기적 실시간 데이터가 생성되었을 때 서버가 같은 노드의 비실시간 데이터 전송큐에서 데이터 패킷을 전송하고 있는 중이라면, 서버는 이미 전송이 시작된 비실시간 데이터 패킷의 전송을 완료하여야 할 것이다. 하나의 노드에서 서버는 주기적 실시간 데이터 전송큐와 비실시간 데이터 전송큐의 순서로 방문하므로, 이후 서버는 일단 다음 노드로 전달되며 네트워크 내의 모든 노드를 순환한 후 해당 노드로 되돌아와 주기적 실시간 데이터를 전송하여야 할 것이다. 따라서 T_1의 시간 동안에 실제로 전송될 수 있는 비실시간 데이터의 최대 개수는 $(N-r+1)$이 된다. T_1의 구간에서 생성된 r개의 주기적 실시간 데이터의 전송 지연 시간이 T_1을 초과하지 않도록 하기 위하여서는 여러개의 패킷으로 분리되어 전송되는 비실시간 데이터의 패킷 전송 시간은 다음의 조건을

만족하여야 한다.

$$L_a \le \frac{T_1 - rL_p - R}{N - r + 1} \tag{15}$$

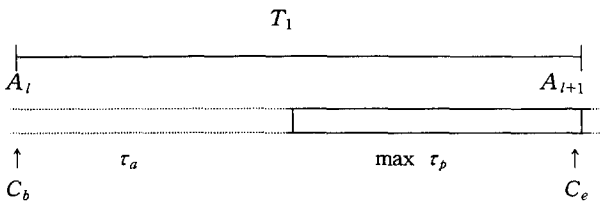


그림 4. 비실시간 데이터의 패킷 전송 시간 제한.
Fig. 4. Limitation of the packet transmission time of the non real-time data.

필드버스에서 주기적 실시간 및 비실시간 데이터의 트래픽이 안정화되기 위하여서는 비실시간 데이터의 도착 빈도를 네트워크의 용량이 초과되지 않는 범위로 제한하여야 한다. 주기적 실시간 및 비실시간 데이터가 공존하는 필드버스 네트워크의 안정화 조건은 Ibe와 Cheng[21]이 제시한 순환 서비스 큐잉 시스템에서의 station backlog 기법을 이용하여 구할 수 있다. t 라는 매우 긴 시간 동안에 노드 i의 비실시간 데이터 전송큐에 도착하는 데이터 패킷의 평균 개수를 생각하자. 비실시간 데이터 전송큐가 안정화되기 위하여서는 도착된 모든 데이터 패킷을 전송하는데 소요되는 시간의 평균값이 t를 초과하지 않아야 한다. t의 시간 동안에 도착하는 비실시간 데이터 패킷 개수의 평균값은 $\lambda_a^i t$이며, 서버는 한 번에 한 개씩의 데이터만을 전송하므로 도착된 데이터 패킷이 모두 전송되기 위하여서는 같은 기간 동안에 서버가 노드 i를 $\lambda_a^i t$번 방문하여야 한다. 따라서, t의 시간 동안에 노드 i에서 비실시간 데이터 패킷을 전송하는데 소요되는 시간은 $(\lambda_a^i t)L_a$이다. 또한, 서버가 한 번 순환하는데 R만큼의 시간이 소요되므로, 노드 i의 비실시간 데이터 패킷을 모두 전송하기 위하여 서버가 순환하는데 소요되는 시간은 $(\lambda_a^i t)R$이다. 다음에는 t의 시간 동안에 네트워크 내의 다른 노드들에서 전송되는 비실시간 데이터 패킷의 전송 시간을 고려하자. 노드 $j(\neq i)$에 대하여 만일에 $\lambda_a^j < \lambda_a^i$라면 t의 시간 동안에 노드 j에서 전송되는 비실시간 데이터 패킷 개수의 평균값은 $\lambda_a^j t$ 개이며, 그렇지 않은 경우에는 노드 j에 도착된 비실시간 데이터패킷들 가운데 $\lambda_a^i t$개의 데이터 패킷만이 평균적으로 전송될 것이다. 따라서, t의 시간 동안에 다른 노드들에서 전송되는 비실시간 데이터 패킷의 전송 시간은 $L_a \sum_{j=1, j\neq i}^{N_a} \min[\lambda_a^i t, \lambda_a^j t]$가 된다. 다음에는 t의 시간 동안에 전송되는 주기적 실시간 데이터의 전송 시간을 고려하자. t의 시간 내에 존재할 수 있는 T_1 구간의 개수는 (t/T_1)이다. 각각의 T_1 구간에서 주기적 실시간 데이터를 전송하는데 소요되는 시간의 평균값은 서버 순환 시간을 포함하여 $(\alpha_K L_p + R)$이므로, t의 시간 동안에 전송되는 주기적 실시간 데이터의 전송 시간은 $(t/T_1)(\alpha_K L_c + R)$이다. 노드 i에서 비실시간 데이터 전송큐가 안정화되기 위하여서는 t의 시간 동안에 도착한 데이터 패킷을 전송하는데 소요되는 시간이 t보다 작아야 한다. 즉,

$$(\lambda_a^i t)L_a + (\lambda_a^i t)R + (t/T_1)(\alpha_K L_p + R) + L_a \sum_{j=1, j\neq i}^{N_a} \min[\lambda_a^i t, \lambda_a^j t] < t. \tag{16}$$

(16)로부터 노드 i의 비실시간 데이터 전송큐가 안정화되기 위하여서는 데이터 패킷의 도착 빈도 λ_a^i가 다음의 조건을 만족하여야 한다.

$$\lambda_a^i < \frac{1 - L_a \sum_{j=1}^{N_a} \min[\lambda_a^i, \lambda_a^j] - [(\alpha_K L_c + R)/T_1]}{R} \tag{17}$$

필드버스 내의 모든 비실시간 데이터 전송큐가 안정화되기 위하여서는 데이터 패킷의 도착 빈도가 가장 높은 전송큐가 안정화되어야 한다. $\lambda_a^m = \max[\lambda_a^i,\ j=1 \text{ to } N_a]$라 하면, 주기적 실시간 및 비실시간 데이터가 공존하는 필드버스 네트워크에서 비실시간 트래픽의 안정화 조건은 다음과 같다.

$$\lambda_a^m < \frac{1 - L_a \sum_{i=1}^{N_a} \lambda_a^i - [(\alpha_K L_c + R)/T_1]}{R} \tag{18}$$

만일에 필드버스 네트워크가 (18)으로 주어진 안정화 조건을 만족하지 않는 경우에는 필드버스 설계자는 네트워크에 접속되는 비실시간 데이터 전송큐의 개수 N_a를 줄여야 한다. 이 경우에 비실시간 데이터 전송큐만을 가지는 노드를 우선적으로 제거하며, 만일에 한 노드에 비실시간 데이터 전송큐와 주기적 데이터 전송큐가 공존하는 경우에는 이러한 노드를 제거하고 대역폭 할당 알고리듬을 다시 수행한다. 마지막으로 주기적 실시간 및 비실시간 데이터 트래픽이 공존하는 필드버스 네트워크를 설계하기 위한 대역폭 할당 알고리듬을 정리하면 다음과 같다.

ALGORITHM I

Given : N, M, N_a, L_p, L_a^i, σ, $R = N\sigma$, $N_p = 2M$,
$[\Phi_i, i=1 \text{ 에서 } M]$, $[\Lambda_a^j, j=1 \text{ 에서 } N_a]$

Step 1 : 주기적 실시간 데이터의 샘플링 주기 T_i를 구한다.

$\Phi_1 = \min\ [\Phi_i, i=1 \text{ 에서 } M]$

$T_1 = \dfrac{(\Phi_1 + L_p)}{3}$

$k_i = \left\langle \dfrac{\Phi_i - (T_1 - L_p)}{2T_1} \right\rangle,\ \forall i=1 \text{ 에서 } M$

$\alpha_K = 2\sum_{i=1}^{M} \dfrac{1}{k_i}$

$r = \lceil \alpha_K \rceil$

If $rL_p + R > T_1$, then
(주기적 실시간 데이터 트래픽의 과부하
⇒ M 감소: goto Step 1)
Else
$T_i = k_i T_1$

Step 2 : 각 노드에서 첫 번째 주기적 실시간 데이터 샘플링 순간 t_j를 구한다.

$t_1 = A_1 = 0$

For $(j=2,\ l=1;\ j \le N_p,\ l \le k_M;\ j=j+1,\ l=l+1)$

$$t_j = \inf[A_l \ge A_{l-1} : u^j(A_l) \le r]$$

Step 3 : 비실시간 데이터의 데이터 패킷 길이 L_a를 구한다.

$$L_a \le \frac{T_1 - rL_p - R}{N-r+1}$$

$\lambda_a^i = \lceil L_a^i / L_a \rceil \Lambda_a^i$

$\lambda_a^m = \max[\lambda_a^i,\ j=1 \text{ 에서 } N_a]$

$$\text{If } \lambda_a^m \ge \frac{1 - L_a \sum_{i=1}^{N_a} \lambda_a^i - [(\alpha_K L_p + R)/T_1]}{R}, \quad \text{then}$$

(비실시간 데이터 트래픽의 과부하
⇒ N_a 감소: goto Step 3)
(주기적 데이터 전송큐 공존
⇒해당 노드 감소: goto Step 1)

End

2. 산발적 실시간/주기적 실시간/비실시간 데이터 트래픽

본 절에서는 1절에서 제시된 필드버스의 대역폭 할당 기법을 산발적 실시간, 주기적 실시간 및 비실시간 데이터 트래픽으로 구성되는 일반적인 경우의 필드버스로 확장한다. 산발적 실시간 데이터는 데이터 전송 시간의 제약을 가장 중요하게 고려하여야 하며, 따라서 주기적 실시간 및 비실시간 데이터보다 높은 우선 순위를 가지고 전송되어야 한다. 산발적 실시간, 주기적 실시간 및 비실시간 데이터 트래픽으로 구성되는 필드버스 시스템의 대역폭 할당에 대한 기본 개념이 그림 5에 나타나 있다. 필드버스에서 제공하는 대역폭을 주기적 구간과 비실시간 구간으로 나누는 것은 앞 절에서의 경우와 동일하며, 각 구간에서 데이터 전송 방법 역시 동일하다. 그러나 산발적 실시간 데이터는 주기적 구간과 비실시간 구간에 관계없이 해당 노드에 서버가 도착되는 즉시 주기적 실시간 및 비실시간 데이터에 우선하여 전송된다.

필드버스 내에 산발적 실시간 데이터 전송큐의 개수는 N_c이다. 최악의 경우에 N_c개의 전송큐가 모두 전송할 데이터를 가지고 있을 수 있으므로 주기적 구간 τ_p^*의 최댓값은 $rL_p + N_c L_c + R$이다. 따라서 주기적 실시간 데이터의 전송큐에서 포화 상태가 발생하지 않도록 하는 안정화 조건은 다음과 같다.

$$rL_p + N_c L_c + R \le T_1 \tag{19}$$

만일에 이러한 조건이 만족되지 못하는 경우에는 필드버스 설계자는 네트워크에 접속되는 제어 루프의 개수 M 또는 산발적 실시간 데이터 전송큐의 개수 N_c를 줄여야 한다. 일반적인 경우의 필드버스 시스템에서 산발적 실시간 데이터를 전송하는 전송큐의 개수 N_c는 많지 않으며,
산발적 실시간 데이터 패킷의 길이 L_c 역시 매우 짧다. 따라서, 산발적 실시간 데이터의 전송으로 인한 주기적 실시간 데이터 전송 대역폭의 손실은 그리 크지 않다.

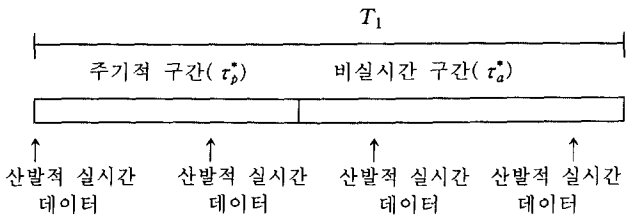


그림 5. 필드버스 대역폭 할당: 산발적 실시간/주기적 실시간/비실시간 트래픽.
Fig. 5. Fieldbus bandwidth allocation: sporadic/periodic/non real-time traffic.

그림 5에서 보는 바와 같이 비실시간 구간 τ_a^*는 $T_1 - \tau_p^*$이다. 주기적 실시간 데이터의 전송 지연 시간이 T_1을 초과하지 않도록 하기 위하여서는 그림 4를 통하여 제시된 것과 유사한 절차를 통하여 비실시간 데이터의 패킷 전송 시간을 다음과 같이 제한하여야 한다.

$$L_a \le \frac{T_1 - (rL_p + N_c L_c + R)}{N-r+1} \tag{20}$$

네트워크 내의 모든 산발적 실시간 데이터 전송큐는 서버가 한 번 순환하는 동안에 한 번의 데이터 전송 기회를 부여받는다. 서버 순환 시간이 T_1을 초과하지 않으므로 산발적 실시간 데이터 전송큐의 끝단에 있는 데이터 패킷의 전송 지연 시간 D_c^m는 T_1을 초과하지 않는다. 즉, $D_c^m \le T_1$. 만일에 산발적 실시간 데이터의 최대 허용 데이터 지연 시간 Φ_c^m가 T_1보다 작게 설정되었다면(즉, $\Phi_c^m < T_1$), 비실시간 데이터의 패킷 전송 시간 L_a는 다음과 같아야 한다.

$$L_a \le \frac{\Phi_c^m - (rL_p + N_c L_c + R)}{N-r+1} \tag{21}$$

패킷화된 비실시간 데이터는 수신단에서 재조립된다. 만일에 산발적 실시간 데이터의 허용 지연 시간이 매우 제한되어 $\Phi_c^m \le (rL_p + N_c L_c + R)$ 하다면, 필드버스의 대역폭은 산발적 및 주기적 실시간과 비실시간 데이터에 의하여 공유될 수 없다. 이 경우에는 주기적 실시간 또는 비실시간 데이터의 트래픽을 줄이거나, 산발적 실시간 데이터 전송을 위한 필드버스를 따로 설치하여야 한다. 그러나 실제적인 경우에 있어서 산발적 실시간 데이터라도 최대 허용 데이터 지연 시간이 수십 msec에서 수백 msec의 범위를 가지고 있어 데이터 허용 지연 시간이 그렇게 엄격하지 않다(즉, Φ_c^m가 일반적으로 주기적 실시간 데이터의 최소 샘플링 주기 T_1보다 작지 않다).

필드버스가 적용되는 실제 환경에서 T_1(또는 Φ_c^m)의 시간 이내에 한 노드에서 산발적 실시간 데이터가 여러개 발생할 확률은 매우 적다. 그러나 이러한 상황을 특별히 고려하여야 할 경우에는 본 논문에서 제시하는 대역폭 할당 기법을 다음과 같이 변형하여 적용할 수 있다. T_1(또는 Φ_c^m)의 시간 이내에 한 노드에서 발생하는 산발적 실시간 데이터의 최대 개수를 n_t라 하자. (19)는 필드버스에 접속된 노드의 산발적 실시간 데이터 전송큐가 단일 서비스 방식으로 동작되는 경우로 가정하여 도출되었으나, 여러 개의 데이터를 동시에 처리하여야 하는 경우에는 산발적 실시간 데이터 전송큐를 서버가 도착되는 순간에 대기하고 있는 모든 데이터를 전송하는 소진(exhaustive) 서비스 방식으로 동작 되도록 하고 (19)를 다음과 같이 변형한다.

$$rL_p + n_t N_c L_c + R \le T_1 \tag{22}$$

이러한 경우에 (20)과 (21)도 역시 다음과 같이 변형된다.

$$L_a \le \frac{T_1 - (rL_p + n_t N_c L_c + R)}{N-r+1} \tag{23}$$

$$L_a \le \frac{\Phi_c^m - (rL_p + n_t N_c L_c + R)}{N-r+1} \tag{24}$$

필드버스 네트워크에서 산발적 실시간, 주기적 실시간 및

비실시간 데이터의 트래픽은 네트워크의 용량을 초과하여서는 안된다. 따라서 앞절에서 제시한 station backlog 기법으로부터 노드 i의 산발적 실시간 및 비실시간 데이터 전송큐가 안정화되기 위하여서는 패킷 도착 빈도 λ_c^i와 λ_a^i가 다음과 같이 제한되어야 한다.

$$\lambda_a^i < \frac{1-L_a\sum_{j=1}^{N_a}\min[\lambda_a^i, \lambda_a^j]-L_c\sum_{j=1}^{N_c}\min[\lambda_a^i, \lambda_c^j]-[(\alpha_K L_p+R)/T_1]}{R} \tag{25}$$

$$\lambda_c^i < \frac{1-L_c\sum_{j=1}^{N_c}\min[\lambda_c^i, \lambda_c^j]-L_a\sum_{j=1}^{N_a}\min[\lambda_c^i, \lambda_a^j]-[(\alpha_K L_p+R)/T_1]}{R} \tag{26}$$

필드버스 내의 모든 데이터 전송큐가 안정화되기 위하여서는 가장 높은 데이터 도착 빈도를 갖는 전송큐가 안정화되어야 한다. $\lambda^m=\max[\lambda_c^i, \lambda_a^j,\ i=1$에서 $N_c,\ j=1$에서 $N_a]$라 하면, 필드버스 시스템이 안정화되기 위한 조건은 다음과 같다.

$$\lambda^m < \frac{1-L_c\sum_{i=1}^{N_c}\lambda_c^i-L_a\sum_{j=1}^{N_a}\lambda_a^j-[(\alpha_K L_p+R)/T_1]}{R} \tag{27}$$

만일에 필드버스 시스템이 (27)으로 주어진 조건을 만족하지 못하는 경우에는 네트워크 설계자는 데이터 전송큐의 개수 N_c 또는 N_a를 줄여야 한다. 산발적 실시간, 주기적 실시간 및 비실시간 데이터 트래픽을 모두 수용하는 일반적인 경우의 필드버스 시스템이 단일 서비스 방식으로 동작되는 경우에 대역폭 할당을 통한 설계 기법의 알고리듬이 다음에 제시되었다. 산발적 실시간 데이터가 고갈 서비스 방식으로 동작되어야 하는 경우에는 L_a를 구하는 해당 공식만을 변경하면 된다.

ALGORITHM II

Given : $N,\ M,\ N_c,\ N_a,\ L_c,\ L_p,\ L_a^j,\ \sigma,\ R=N\sigma,$

$N_p=2M,\ [\Phi_i,\ i=1$ 에서 $M],\ \Phi_c^m,$

$[\lambda_c^i,\ i=1$ 에서 $N_c],\ [\Lambda_a^j,\ j=1$ 에서 $N_a]$

Step 1 : 주기적 실시간 데이터의 샘플링 주기 T_i를 구한다.

다음의 조건을 제외하고 ALGORITHM I과 동일

If $rL_p+N_cL_c+R > T_1$, then

(주기적 실시간 데이터 트래픽의 과부하

$\Rightarrow$ M 또는 N_c 감소: goto Step 1)

Step 2 : 각 노드에서 첫 번째 주기적 실시간 데이터 샘플링 순간 t_j를 구한다.

ALGORITHM I과 동일

Step 3 : 비실시간 데이터의 데이터 패킷 길이 L_a를 구한다.

$$L_a = \begin{cases} \dfrac{\Phi_c^m-(rL_p+N_cL_c+R)}{N-r+1}, & \Phi_c^m < T_1 \\ \dfrac{T_1-(rL_p+N_cL_c+R)}{N-r+1}, & \text{otherwise} \end{cases}$$

$\lambda_a^j=\lceil L_a^j/L_a \rceil \Lambda_a^j$

$\lambda^m=\max[\lambda_c^i, \lambda_a^j,\ i=1$ 에서 $N_c,\ j=1$ 에서 $N_a]$

If $\lambda^m \geq \dfrac{1-L_c\sum_{i=1}^{N_c}\lambda_c^i-L_a\sum_{j=1}^{N_a}\lambda_a^j-[(\alpha_K L_p+R)/T_1]}{R}$, then

(산발적 실시간 또는 비실시간 트래픽의 과부하

$\Rightarrow$ N_c 또는 N_a 감소: goto Step 3)

(주기적 데이터 전송큐 공존

$\Rightarrow$해당 노드 감소: goto Step 1)

End

Ⅳ. 대역폭 할당 기법 적용의 예

본 장에서는 시뮬레이션 모델을 이용하여 제 Ⅲ 장에서 기술된 대역폭 할당에 의한 필드버스 설계 기법의 타당성을 검증하고자 한다. 본 모의 실험의 목적은 비실시간 데이터의 트래픽이 크게 증가하여도 산발적 실시간 및 주기적 실시간 데이터의 전송 지연 시간이 요구조건을 만족하는가를 검증하는데 있다. 비실시간 데이터의 트래픽이 낮은 경우에는 당연히 산발적 및 주기적 실시간 데이터의 트래픽에 미치는 영향이 매우 적을 것이고, 따라서 본 예에서는 최악의 경우를 가정하여 네트워크 내의 트래픽이 포화 상태 직전의 매우 높은 트래픽 조건에서 모의 실험을 수행한다. 즉, 이러한 트래픽 조건하에서 산발적 및 주기적 데이터의 전송 지연 시간의 요구 사항이 만족된다면 그 이하의 트래픽 조건하에서는 당연히 전송 지연 시간의 요구 사항이 만족될 것이다. 비실시간 데이터 트래픽을 증가시키기 위하여서는 데이터의 길이 또는 데이터의 도착 빈도를 증가시켜야 하나, 본 논문에서 제시하는 대역폭 할당 기법에서는 비실시간 데이터 패킷의 길이가 L_a로 정해져 있으므로 트래픽을 증가시키기 위하여서는 데이터 도착 빈도를 증가시키도록 한다. 본 예에서는 비실시간 데이터의 전송시간 L_a^j와 도착 빈도 Λ_a^j를 설정하고 이를 p_j개의 패킷으로 분리하는 대신에 비실시간 데이터 패킷의 도착 빈도 λ_a^j를 직접 설정하였다. 이는 본 예를 위하여 개발된 시뮬레이션 모델이 PC 환경에서 SIMAN 언어[24]를 사용하여 개발되었고, 따라서 시뮬레이션 모델에서 동시에 처리되어야 할 entity의 수에 제한이 있기 때문이다. 본 모의 실험에서는 그러나 비실시간 데이터는 단순히 전체 네트워크 트래픽의 부하만을 증가시키는 용도로만 사용하였고, 비실시간 데이터의 데이터 전송 지연 시간은 중요하지 않은 사항이므로 본 논문에서 채택한 비실시간 데이터 패킷의 도착 빈도 λ_a^j를 증가시키는 방법에 대한 모의 실험 결과도 비실시간 데이터의 트래픽이 산발적 실시간 및 주기적 실시간 데이터의 전송 지연 시간에 어떠한 영향을 미치는 가를 알아보기 위한 본 모의 실험의 목적을 달성하는데 있어서 충분한 타당성을 가지고 있다.

본 논문의 네트워크 설계의 예에서 제시하는 필드버스는 10개의 노드($N=10$)와 20개의 데이터 전송큐를 갖는 것으로 한다. 네트워크 내에는 다섯 개의 제어루프가 접속되며, 따라서 네트워크 내의 주기적 실시간, 산발적 실시간 및 비실시간 데이터 전송큐의 개수는 각각 10, 5, 5개씩이다 ($M=5,\ N_p=10,\ N_c=5,\ N_a=5$). 노드 그룹 (1, 2), (3, 4), (5, 6), (7, 8), (9, 10)은 각각 제어 루프 1, 2, 3, 4, 5에 속하여 있으며, 노드 1, 3, 5, 7, 9는 산발적 실시간 데이터 전송큐를 가지고, 나머지 노드 2, 4, 6, 8, 10은 비실시간 데이터 전송큐를 갖는다. 주기적 실시간 데이터 전송큐의 용량은 1이며, 산발적 실시간 및 비실시간 데이터 전송큐의 용량은 전송큐의 포화 상태로 인한 메시지 제거 현상이 발생하지 않도록 충분히 크다고 가정한다. 필드버스의 데이터 전송 속도 B는 500 Kbps이다. 산발적 실시간 데이터와 주기적 실시간 데이터의 길이는 각각 100 비트와 500 비트로 가정하며, 따라서 패킷 전송 시간 L_c와 L_p는 각각 0.2 msec와 1 msec이다. 모든 노드에서 산발적 실시간 및 비실시간 데

이터의 도착 빈도 λ_c^i와 λ_a^i 는 각각 0.01 msec^{-1}와 0.177 msec^{-1}의 Poisson 분포를 갖는 것으로 한다. 각 노드에서의 서버 오버헤드 σ는 0.05 msec이며, 따라서 서버 순환 시간 R은 0.5 msec이다. 산발적 실시간 데이터의 최대 허용 데이터 지연 시간 Φ_c^m는 10 msec이며, 다섯 개의 제어 루프에서 최대 허용 루프 지연 시간은 각각 $[\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5]$ = [25 msec, 60 msec, 100 msec, 200 msec, 400 msec]로 주어지는 것으로 가정한다.

ALGORITHM II의 Step 1으로부터 Φ_1와 T_1은 각각 25 msec와 8.667 msec로 주어지며, 샘플링 주기 비율 벡터 $\boldsymbol{K}$는 $[k_1, k_2, k_3, k_4, k_5]$ = [1, 2, 4, 8, 16]이다. 최소 샘플링 주기 T_1의 시간 동안에 다섯 개의 제어 루프에서 생성되는 데이터 개수의 평균값은 α_K=3.875이며, 따라서 주기적 구간에서 윈도우의 개수는 r=4이다. $rL_p+N_cL_c+R < T_1$이므로, 다섯 개의 제어 루프에서 샘플링 주기 벡터 $\boldsymbol{T}$는 $[T_1, T_2, T_3, T_4, T_5]$ = [8.667 msec, 17.333 msec, 34.667 msec, 69.333 msec, 138.667 msec]로 결정된다. Step 2로부터 노드 1에서 노드 10까지의 첫 번째 주기적 실시간 데이터 생성 순간은 각각 $[t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}]$ = [0 msec, 0 msec, 0 msec, 0 msec, 8.667 msec, 8.667 msec, 25.998 msec, 25.998 msec, 60.662 msec, 60.662 msec]으로 주어진다. 마지막으로 Step 3으로부터 비실시간 데이터의 패킷 전송 시간은 L_a=0.452 msec (패킷 길이는 226 비트)로 주어지며, 네트워크 시스템은 요구되는 트래픽을 모두 처리할 수 있는 안정된 상태임을 알 수 있다.

U를 단위 시간 당 네트워크에서 데이터 전송에 소요되는 시간의 비율로 정의되는 네트워크 이용도라 하자. 산발적 실시간, 주기적 실시간, 비실시간 데이터 트래픽으로 구성되는 필드버스 시스템에서 네트워크의 이용도 U는 다음과 같이 나타내어진다.

$$U=\sum_{i=1}^{N_c}\lambda_c^i L_c+\sum_{j=1}^{N_p}\frac{L_p}{T_j}+\sum_{k=1}^{N_a}\lambda_a^k L_a \qquad (28)$$

주어진 필드버스 시스템의 예에서 산발적 실시간, 주기적 실시간 및 비실시간 데이터의 네트워크 이용도는 각각 0.01%, 44.71% 및 40%이며, 전체 네트워크의 이용도는 U는 84.72 %이다. 이러한 네트워크 이용도는 순수히 데이터를 전송하는데 소요되는 시간만을 고려한 것이며, 서버의 전송에 따르는 필수 불가결한 오버헤드 σ는 제외시켰다. 시뮬레이션 결과로부터 서버의 오버헤드에 의한 대역폭은 전체 네트워크의 대역폭의 15%를 차지하는 것으로 나타났으며, 따라서 주어진 필드버스 설계의 예에서 실제 네트워크의 이용도는 거의 100%에 육박하여 포화 상태 직전의 매우 높은 트래픽 상태이다. 이는 일반적으로 필드버스를 설계하는 경우에 실시간 데이터의 성능 요구 사항을 만족시키기 위하여 네트워크의 이용도를 50% 이하로 설정하는 것에 비하여 매우 높은 수치이다.

다음에는 III 장에서 제시한 대역폭 할당을 통한 필드버스 설계 기법을 통하여 설계된 필드버스 시스템이 산발적 실시간 및 주기적 실시간 데이터의 성능 요구 사항을 만족하는가를 시뮬레이션 기법을 통하여 검증하기로 한다. 시뮬레이션 모델은 네트워크 시스템을 모델링하는 이산 사건 모델과 제어 루프 1의 제어 시스템을 모델링하는 연속 시간 모델을 통합한 이산 사건/연속 시간 시뮬레이션 모델로 구성된다 [19]. 표 I에 산발적 실시간, 주기적 실시간 및 비실시간 데이터의 데이터 전송 지연 시간에 대한 평균값과 최댓값이 비교되어 있다. 여기서 Algorithm은 제 3 장에서 제시된 대역폭 할당 기법을 통하여 설계된 경우에 대한 결과이며, CASE 1과 2는 각각 대역폭 할당 기법을 따르지 않는 경우에 대한 결과이다. CASE 1의 경우는 주기적 실시간 데이터의 샘플링 주기가 적절히 스케듈링되지 않은 경우로 샘플링 주기 벡터 $\boldsymbol{T}$가 $[T_1, T_2, T_3, T_4, T_5]$ = [8.667 msec, 14.632 msec, 47.663 msec, 92.726 msec, 116.991 msec]로 설정된 경우이고, 산발적 실시간 및 비실시간 데이터는 Algorithm의 경우와 동일하다. 또한 CASE 2는 비실시간 데이터의 패킷 전송 시간이 적절히 설정되지 않은 경우로 비실시간 데이터의 패킷 전송 시간이 16 msec(패킷 길이는 8000 비트)인 경우이고, 주기적 실시간 및 비실시간 데이터는 Algorithm의 경우와 동일하다. 네트워크의 이용도 U는 Algorithm, CASE 1 및 CASE 2 의 모든 경우에 대하여 동일하다. 표 I에서 전송 지연 시간은 데이터가 전송큐에 도착한 순간부터 같은 데이터가 수신단의 수신큐에 완전히 수신되는 순간까지의 시간이며, 이의 단위는 msec이다. 또한 실패 데이터 개수 비율은 시뮬레이션 기간 동안 생성된 데이터 개수에 대하여 같은 기간 동안 전송큐의 포화 상태로 인하여 전송되지 못한 데이터의 개수를 나타낸다.

표 1에서 산발적 실시간 데이터의 경우를 보면 Algorithm의 경우에 산발적 실시간 데이터의 최대 전송 지연 시간은 5 msec로 최대 허용 데이터 지연 시간인 10 msec을 초과하지 않으며 따라서 산발적 실시간 데이터에 대한 성능 요구 사항을 만족한다. CASE 1에서도 산발적 실시간 데이터의 성능 요구 사항은 만족되나, 비실시간 데이터의 패킷 길이가 적절히 선정되지 못한 CASE 2의 경우에는 산발적 실시간 데이터의 최대 전송 지연 시간이 26.1 msec으로 10 msec를 크게 초과하여 주어진 성능 요구 사항을 만족시키지 못함을 알 수 있다. 주기적 실시간 데이터를 보면 Algorithm의 경우에는 최대 전송 지연 시간이 6.4 msec로 최소 샘플링 주기 T_1의 8.667 msec를 초과하지 않으며, 따라서 모든 제어 루프에서 최대 루프 지연 시간이 (3)으로부

표 1. Algorithm, CASE 1, CASE 2의 성능 비교.
Table 1. Performance comparison for Algorithm, CASE 1 and CASE 2.

	평균 데이터 전송지연시간 (msec)			최대 데이터 전송지연시간 (msec)			실패 데이터 개수 비율 (실패데이터/생성데이터)		
	Algorithm	CASE 1	CASE 2	Algorithm	CASE 1	CASE 2	Algorithm	CASE 1	CASE 2
산발적 실시간 데이터	1.9	1.8	4.5	5.0	5.9	26.1	0/44	0/44	0/44
주기적 실시간 데이터	3.4	3.1	5.4	6.4	11.0	35.6	0/5368	1/5370	905/5368
비실시간 데이터 패킷	46.1	86.8	34.1	258.4	343.2	142.1	0/10590	0/10590	0/288

터 $D_i \leq T_i+6.4$msec가 되어 모든 제어 루프에서 최대 허용 루프 지연 시간 Φ_i($i=1$ 에서 5)를 초과하지 않는다. 그러나 CASE 1에서는 산발적 실시간 데이터의 최대 전송 지연 시간이 11 msec이 되는 경우에 (3)에서 루프 지연 시간이 $2*8.667+11=28.334$ msec가 되어 제어 루프 1에 대한 루프 지연 시간의 성능 요구 사항인 25 msec를 만족시키지 못하는 경우가 발생한다. CASE 2에서는 최대 데이터 전송 지연 시간이 35.6 msec가 되어 제어 루프 1, 2와 3에서 최대 루프 지연 시간이 각각 $5*8.667+35.6=78.935$msec, $3*17.333+35.6=87.599$msec, $2*34.667+35.6=104.934$msec까지 증가할 수 있으며, 이 경우에 제어 루프 1, 2와 3에 대한 루프 지연 시간의 성능 요구 사항을 만족시키지 못한다. 비실시간 데이터 패킷은 Algorithm의 경우에 데이터 전송 지연 시간이 CASE 1보다는 작으나 CASE 2보다는 큰 것으로 나타났다. 그러나 비실시간 데이터 패킷의 전송 지연 시간은 시스템의 성능에 큰 영향을 미치지 않으므로 중요하지 않다. 산발적 실시간 및 비실시간 데이터의 경우에는 전송큐의 큐용량이 충분히 크다고 가정하였으므로 실패 데이터가 발생하지 않는다. 그러나 주기적 실시간 데이터의 경우에는 전송큐의 용량이 하나로 제한되어 전송 지연 시간이 데이터 샘플링 주기보다 큰 경우에는 실패 데이터가 발생할 수 있다. Algorithm의 경우에는 실패 데이터가 없고, CASE 1의 경우에는 1.2초의 시뮬레이션 기간 동안에 1개의 데이터 전송 실패가 일어났으며, CASE 2에서는 같은 기간 동안에 많은 데이터 전송 실패가 발생하였다.

다음에는 데이터 전송 지연 시간과 데이터 전송 실패가 제어 시스템에 미치는 영향에 대하여 조사하기로 한다. 노드 1과 노드 2로 구성된 제어루프 1에서 플랜트는 노드 2에 접속되며, 플랜트 전달함수 $G_p(s)$는 다음과 같이 주어졌다.

$$G_p(s)=\frac{1}{(0.3s+1)(0.03s+1)} \tag{29}$$

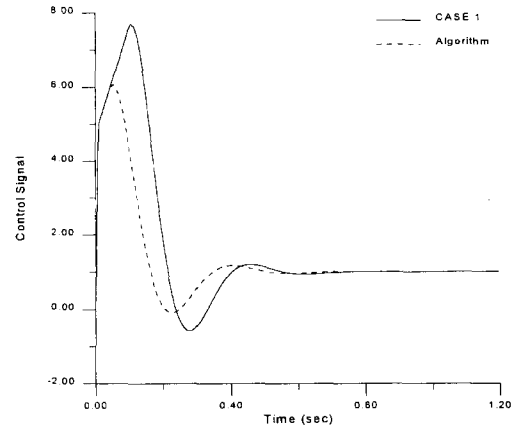
또한 제어루프 1의 디지털 콘트롤러는 노드 1에 접속되며, 콘트롤러의 전달함수를 이에 대응하는 아날로그 값으로 표시하면 다음과 같다.

$$G_c(s)=\frac{7(s+5)}{s} \tag{30}$$

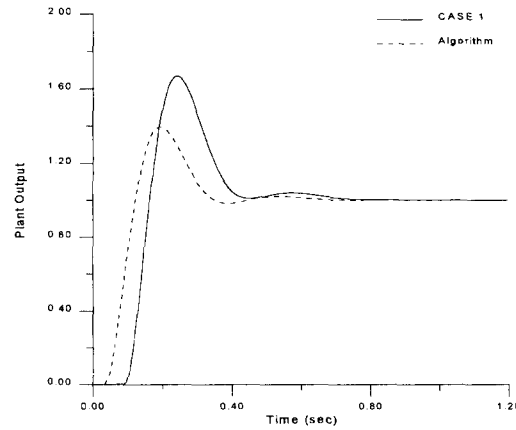
여기서 s는 Laplace 변환 변수이다. 그림 6에는 CASE 1과 Algorithm의 경우에 step 기준 입력에 대하여 제어 루프 1의 제어 신호와 플랜트 출력의 시간에 대한 변화가 비교되었다. 그림 6에서 보는 바와 같이 단 한 번의 주기적 실시간 데이터 전송 실패가 발생하더라도 이것이 제어 입력에 대한 응답의 천이 구간에서 발생하는 경우에는 제어 시스템에 미치는 영향이 매우 커질 수가 있음을 알 수 있다. 그림 7에는 CASE 2와 Algorithm의 경우에 제어 루프 1에 step 기준 입력을 가하였을 때 제어 신호와 플랜트 출력의 시간에 대한 변화가 비교되었다. 그림 7에 나타난 바와 같이 Algorithm의 경우에 제어 신호는 부드럽게 변하고 있으나, CASE 2의 경우에는 주기적 실시간 데이터의 전송 실패로 인하여 제어 신호가 jitter 현상을 일으킴을 알 수 있다. 이러한 고주파 잡음은 액튜에이터의 마모를 초래한다. 또한 플랜트의 제어 성능도 Algorithm의 경우와 비교하여 저하됨을 볼 수 있다.

V. 결론 및 추후 연구 사항

필드버스는 공장 자동화와 공정의 분산 제어 시스템의 컴퓨터 통신망 계층 구조에서 가장 기본이 되는 네트워크 시스템으로, 필드에 설치된 각종 제어 및 자동화 장비들 간에 데이터 통신을 담당한다. 이러한 장비들에서 생성되는 데이터는 크게 산발적 및 주기적 실시간과 비실시간 데이터로 나뉘어진다.



(a)

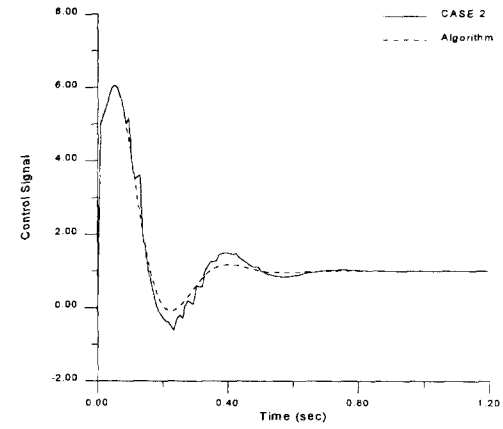


(b)

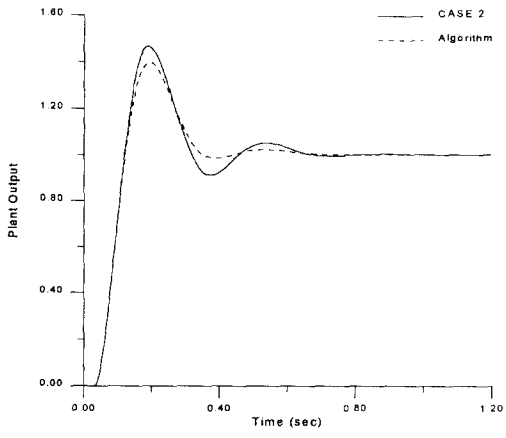
그림 6. 데이터 전송 지연 시간에 대한 제어시스템의 성능: CASE 1 (a) 제어신호, (b) 플랜트 출력.

Fig. 6. Performance of control system with respect to data latency: CASE 1
(a) controller signal, (b) plant output.

필드버스 시스템에서는 이러한 다양한 특성을 갖는 데이터들이 하나의 네트워크 미디엄을 공유하기 때문에 만일에 필드버스 시스템이 부적절하게 설계된다면, 산발적 및 주기적 실시간 데이터의 전송 지연 시간이 미리 지정된 한계치를 초과하여 필드버스에 접속된 응용 시스템의 성능을 저하시키거나, 최악의 경우에는 네트워크 시스템과 네트워크에 접속된 응용 시스템이 불안정한 상태에 도달할 수 있다. 본 논문에서는 단일-순환 서비스로 동작되는 필드버스의 제한된 대역폭을 산발적 실시간, 주기적 실시간 및 비실시간 데이터 트래픽에 적절히 할당하는 대역폭 할당을 통한 필드버스 설계의 원칙적인 방법론을 제시하였다. 여기서 단일-순환 서비스란 서버가 미리 정해진 순서에 따라 필드버스에 접속된 각각의 노드를 차례로 한 번씩 방문하며, 각각의 노드의 전송큐에서는 서버가 도착하는 순간에 대기하고 있는 데이터를 한 번에 한 개씩의 데이터를 전송하는 방식을 말한다. 본 논문에서 제시하는 필드버스 설계 기법은 필드버스에 접속된 응용 시스템의 데이터 전송 지연 시간에 대한 성능 요구 사항을 만족하는 동시에 필드버스에서 제공하는



(a)



(b)

그림 7. 데이터 전송 지연 시간에 대한 제어시스템의 성능: CASE 2 (a) 제어신호, (b) 플랜트 출력.

Fig. 7. Performance of control system with respect to data latency: CASE 2 (a) controller signal, (b) plant output.

대역폭 자원의 이용도를 최대화한다. 본 논문에서는 이산 사건/연속 시간 시뮬레이션 기법을 통하여 본 논문에서 제시하는 대역폭 할당을 통한 필드버스 설계 기법이 필드버스의 대역폭을 충분히 활용하는 동시에 필드버스에 접속되는 응용 시스템의 성능 요구 사항을 만족시켜 줄 수 있음을 입증하였다.

현존하는 필드버스 가운데 어느 프로토콜도 실시간, 주기적 및 비실시간 데이터 트래픽을 동시에 완벽히 처리할 수 있는 프로토콜은 없다. 본 논문을 통하여 기여하고자 하는 핵심적인 사항은 필드버스에서 산발적, 주기적 및 비실시간 데이터를 처리할 수 있는 원칙적인 방법론을 제시하는 것이다. 본 논문에서 제시하는 대역폭 할당 기법은 이러한 기법이 적용되는 새로운 필드버스 프로토콜을 개발할 수도 있고, 기존의 필드버스 프로토콜들을 수정 및 변형함으로써 적용될 수도 있다. 본 연구의 후속 연구에서는 본 논문에서 제시하는 대역폭 할당 기법이 적용되는 새로운 필드버스 프로토콜의 규격 개발을 비롯하여, 본 논문에서 제시하는 대역폭 할당 기법을 Profibus, WorldFIP, Foundation Fieldbus, IEC/ISA 필드버스 등의 기존의 필드버스들에 적용하는 경우에 각각의 필드버스들에 대한 동작원리의 특수성들을 고려하여 네트워크 관련 파라미터의 값을 어떻게 선정하고 프로토콜을 어떻게 변형시켜야 할 것인가에 대한 연구를 수행할 계획이다.

참고문헌

[1] R. U. Ayres, *Computer Integrated Manufacturing*, vol. I-IV, Chapman & Hall, 1992.

[2] D. D. Bedworth, et.al., *Computer-Integrated Design and Manufacturing*, McGraw-Hill, 1991.

[3] J. R. Piementel, *Communication Networks for Manufacturing*, Prentice Hall, 1990.

[4] G. Olsson and G. Piani, *Computer Systems for Automation and Control*, Prentice Hall, 1992.

[5] A. Valenzano, et. al., *MAP and TOP Communications: Standards and Applications*, Addison Wesley, 1992.

[6] P. Pleinevanx and J. D. Decognie, "Time-critical communication networks: field buses," *IEEE Network*, vol. 2, no. 3, pp. 55-63, May, 1988.

[7] 홍승호, 김기암, 김지용, 고성준, "분산제어 및 자동화 시스템과 필드버스," 제어·자동화·시스템공학회지 제 2권, 제 4호, pp. 19-29, 1996, 7.

[8] 홍승호, "MAP : 공장자동화를 위한 네트웍의 표준," 대한기계학회지, 제3권, 제5호, pp. 427-441, 1995. 5.

[9] *MAP 3.0 Specification 1993 Release*, World Federation of MAP/TOP Users Groups, 1993.

[10] A. H. McMillan and C. J. Gardner, *Mini-MAP '93*, Open I. T. Corp., 1994.

[11] S. Cavalieri, A. Di Stefano and O. Mirabella, "Optimization of acyclic bandwidth allocation exploiting the priority mechanism in the fieldbus data link layer," *IEEE Trans. Industrial Electron.*, vol. 40, no. 3, pp. 297-306, June, 1993.

[12] *ISA-dS50.02. Draft Standard: Fieldbus Standard for Use in Industrial Control Systems. Part 4: Data Link Protocol Specification*, 1993.

[13] S. H. Hong, "Scheduling Algorithm of data sampling times in the integrated communication and control systems," *IEEE Trans. on Control Systems Technology*, vol. 3, no. 2, pp. 225-230, June, 1995.

[14] *MIL-STD-1553B Protocol Manual*, United States Government Printing Office, February, 1980.

[15] *FIP Club Functional Specifications*, Club FIP, BP 850, 54011 Nancy, Cedex, France.

[16] *DKE/DIN Trial Use Standard (DIN 19245/V) Profibus*, Version 4.0, August, 1987.

[17] J. R. Jordan, *Serial Networked Field Instrumentation*, Wiley, 1995.

[18] *Aerospace Standard AS4074.1 Linear Token Passing Multiplex Data Bus*, SAE, 1988.

[19] A. Ray, S. H. Hong, S. Lee and P. J. Egubelu, "Discrete-event/continuous-time simulation of distributed data communication and control systems," *Trans. of the Society for Computer Simulation*, vol. 5, no. 1, pp. 71-85, January, 1988.

[20] A. Ray and Y. Halevi, "Integrated communication and control systems: part I analysis and part II-design considerations," *ASME J. Dyn., Meas. &*

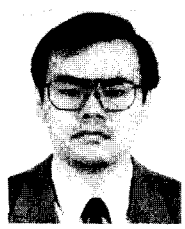
Cont., vol. 110, pp. 367-381, December, 1988.

[21] O. C. Ibe and X. Cheng, "Stability conditions for multiqueue systems with cyclic service," *IEEE Trans. on Automatic Control*, vol. 33, no. 1, pp. 102-103, January, 1988.

[22] K. G. Shin and C. Chou, "Design and evaluation of real-time communication for fieldbus-based manufacturing systems," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 3, pp. 357-367, June, 1996.

[23] F. Vasques and G. Juanole, "Fieldbus MAC mechanisms for hard real time data communication support," *Open Bus System '93*, pp. 225-232, VITA, 1993.

[24] C. D. Pegden, R. E. Shannon and R. P. Sadowski, *Introduction to Simulation Using SIMAN*, McGraw-Hill, 1995.



홍 승 호

1956년 5월 31일생. 1982년 2월 연세대 기계공학과(공학사), 1985년 5월 텍사스공대 기계공학과(공학석사), 1989년 8월 펜실베니아주립대 기계공학과(공학박사), 1989년 ~ 1992년 한국전자통신연구소 선임연구원, 1992년 ~ 현재 한양대 제어계측공학과 조교수. 관심분야는 분산제어 및 자동화 네트워크와 위성 관제 및 제어시스템 등임.