論 文
53D-10-7

TCN을 통한 메시지 전송 능력 향상에 관한 연구

Performance Improvement of Message Transmission over TCN(Train Communication Network)

崔 明 浩* · 文 鍾 千** · 朴 宰 賢†
(Myung-ho Cho · Chong-chun Moon · Jaehyun Park)

Abstract - The data transmission over MVB(Multifunction Vehicle Bus) of TCN(Train Communication Network) is divided into the periodic transmission phase and the sporadic transmission phase. TCN standard recommends the event-polling method as the message transfer in the sporadic phase. However, since the event-polling method does not use pre-scheduling to the priority of the messages, it is inefficient for the real-time systems. To schedule message transmission, a master node should know the priority of message to be transmitted by a slave node prior to the sporadic phase, but the existing TCN standard does not support any protocol for this. This paper proposes the slave frame bit-stuffing algorithm, with which a master node gets the necessary information for scheduling and includes the simulation results of the event-polling method and the proposed algorithm.

Key Words : Train Communication Network, Multifunction Vehicle Bus, Message Transmission, Event-Polling Algorithm, Slave Frame Bit-Stuffing Algorithm

1. 서 론

전동차 시스템은 현대에 이르러 그 사용이 대중화되고 있으며 교통 및 화물 시스템에서 차지하는 비중이 점점 커지고 있다. 이러한 추세에 따라 개발된 고속전철을 제어하기 위해서 분산 제어 시스템이 널리 사용되고 있다. 분산 제어 시스템에 사용되는 네트워크 시스템은 전동차 시스템의 특성상 다음과 같은 특징들을 가지게 된다. 차량내의 하나의 데이터 버스를 모든 지능형 스테이션 및 I/O 기기들이 공유하며, 데이터 서비스에 있어서는 제어신호와 같이 전송시간이 한계치를 넘지 않아야하는 시간 제약성(Time Critical) 데이터와 시간제약성은 작지만 상당한 크기의 일반 데이터를 전송할 수 있어야 있다. 또한 전체 전동차 네트워크 시스템 차원에서는 차량간의 네트워크 연결과 차량 구성의 변화에 대한 지능성도 필요하게 된다. 국제 표준화 기구인 IEC (International Electrotechnical Commission)에서는 이러한 특징들에 부합하는 전동차용 통신 프로토콜 TCN (Train Communication Network)을 제안하여 IEC 61375-1 International Standard로 표준화하였다. 이 TCN은 분산된 제어기기들 사이의 빠르고 정확한 데이터 교환과 차량들과 플러그인 장비들 간의 상호운용성 및 유연성을 기본적인 목적으로 두고 있다[1][2][3].

† 교신저자, 正會員 : 仁荷大學 情報通信工學科 副教授
E-mail : jhyun@inha.ac.kr
* 學生會員 : 仁荷大學 情報通信大學院 碩士
** 正 會 員 : 仁荷大學 情報通信工學科 碩士課程
接受日字 : 2004年 3月 10日
最終完了 : 2004年 5月 19日

TCN은 차량내 통신 버스인 MVB(Multifunction Vehicle Bus)와 차량간 통신 버스인 WTB(Wired Train Bus)의 이중 계층 구조로 구성되며 데이터 전송은 프로세스 데이터(Process data), 메시지 데이터(Message data), 관리용 데이터(Supervisory data)의 세 가지 데이터 서비스로 구분되어 전송되며 전송구간은 그림 1에서 보는 바와 같이 전송방식에 의해 주기적 전송구간과 비주기적 전송구간으로 나누어진다[1][2].

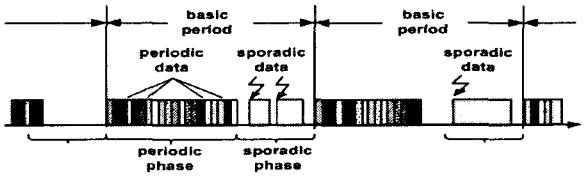

그림 1 TCN의 주기적 전송구간과 비주기적 전송구간
Fig. 1 Periodic data and sporadic data transmission phases of TCN

MVB에서는 제한된 전송 대역폭을 효율적으로 사용하기 위해 접근권한을 스케줄링(Periodic List)에 근거해 할당하게 된다. 주기적 전송 구간에서는 데이터가 포트 단위로 전송되기 때문에 각 포트의 고유 전송 주기인 개별주기(Individual Period)를 근거해 스케줄링하게 되며, 그 결과는 Periodic List에 기록되어 사용한다. 반면 비주기적 전송구간에서는 메시지 데이터 서비스 및 관리용 데이터 서비스가 이뤄지게 된다. 메시지 데이터 서비스의 마스터 프레임은 프로세스 데이터 서비스와는 달리 논리적 어드레스가 아닌 디바이스 고유의 물리적 어드레스를 사용하고, 호출자(Caller)/응답자

(Replier)의 형태로 메시지를 주고받는다. MVB에서 메시지 데이터는 그 특성상 이벤트가 발생될 때 비주기적으로 데이터 갱신이 이루어지며 각 노드는 데이터 갱신이 일어나면 마스터 노드에게 메시지 데이터 전송요구를 하게 된다. 이를 위해 TCN에서는 마스터 노드가 노드들에게 메시지 데이터 전송의 유무를 확인하는 collision-based의 Event-polling 방식을 정의하고 있다. 이 알고리즘은 각 스테이션을 leaf로 하는 이진 탐색 트리(Binary Search Tree)를 이용한 것으로 차량내 버스의 비주기적 전송구간에서 발생할 수 있는 충돌(Collision)을 해결하기 위한 매체 엑세스 제어 프로토콜(Media Access Protocol)로 사용되고 있다. 대칭형 프로토콜에 사용되는 트리 알고리즘은 각 스테이션이 충돌해소주기(Collision Resolution Period) 동안 모든 피드백을 검사해야 하는 단점이 있지만 비대칭형 프로토콜에서의 적용은 마스터만이 피드백을 검사하기 때문에 구현 및 수정이 쉽다. 마스터는 전송제어 정보를 브로드캐스팅 및 그룹 멀티캐스팅하고 이에 응답하는 슬레이브 디바이스를 검사함으로써 소스 디바이스를 결정하게 되는데 할당된 비주기적 전송 구간에서 소스 디바이스를 결정하지 못했다면 다음 비주기적 전송구간에서 검색과정을 계속하여 소스 디바이스를 결정하게 된다. 소스 디바이스를 결정하게 되면 메시지를 비주기적 전송 구간에서 전송하게 된다. 이러한 Event-polling 방식은 트리 구조를 감안한다면 충돌이 발생했을 경우 메시지를 전송하려는 노드 개수의 증가와 트리 레벨에 따라 그 시간지연이 매우 증가하게 된다. 또한, 물리적 디바이스 어드레스 공간이 TCN에서는 12 bits로 할당되어 있기 때문에 각 노드들의 물리적 어드레스가 연속적으로 할당되어 있지 않다면 최악의 경우, 14개의 텔레그램(1 General Event, 11 Group Event, 1 Single Event, 1 Event Read)이 필요하게 되고 텔레그램 외에도 충돌과 무응답까지 고려한다면 이는 엄청난 시간 지연을 의미하는 것이다.

기존 알고리즘의 또 다른 단점은 디바이스의 중요성이 고려되지 않는다는 것이다. TCN 표준에서는 메시지 우선순위로 높은 우선순위와 낮은 우선순위의 두 가지 우선순위를 정의하고 있는데 이는 단순히 각 디바이스 내에서의 메시지 우선순위를 의미한다. 우선순위가 혼재되어 있는 경우 메시지를 전송하게 되는 장치는 우선순위가 아닌 물리적 어드레스에 의해서 결정되게 된다.

기존 TCN 표준의 메시지 전송 방법에서의 문제점을 해결하기 위해서 다음과 같은 방법들을 생각해볼 수 있다. 첫 번째 방법은 시간 지연이 텔레그램 수에 의해서 좌우되므로 가능한 밸런스형 트리형식을 구축하여 필요한 텔레그램 수의 최대값을 낮추는 물리적 어드레스 정적 할당 방법이다. 앞서도 언급했듯이 12 bits의 어드레스가 순서 없이 임의대로 할당되었을 경우에는 한 번의 이벤트를 읽어 들이는 경우에 최대 14개의 텔레그램이 소요된다. 따라서 밸런스형 트리를 구성할 때 고려할 점은 특히 자주 이벤트가 발생하는 두 디바이스는 최하위 비트만 다른 어드레싱을 할당하여 충돌을 최소화 시켜줘야 한다는 점이다. 이렇게 할당함으로써 빈번히 발생하는 충돌을 제거하고 시간 지연을 줄일 수 있게 된다. 두 번째 방법은 MVB 네트워크 트래픽 분석에 의한 동적 하위 트리 레벨 검색 방안이다. 만약 n개의 디바이스에서 산발적인 데이터 전송 이벤트의 발생이 있다고 가정하면 레벨 0에서가 아니라 트래픽에 맞는 레벨에서의 이벤트 제안의 시작이 레벨 0에서의 시작으로 인한 필연적인 몇 번의 충돌을 피할 수 있게 되므로 시간 지연을 줄일 수 있다. 세 번째 방법은 프로세스 데이터 패킷 해석에 의한 소프트웨어적 방안이다. 이 방법은 디바이스에 할당된 각 포트 중 가장 주기가 빠른 포트를 선택하여 메시지 데이터 전송에 대한 정보(디바이스의 물리적 어드레스를 포함)를 포함하게끔 데이터 셋을 정의하는 것이다. 그러나 기본적으로 데이터에 포함된 정보를 처리해야 하는 소프트웨어적인 방식을 사용함으로써 네트워크 호스트에 프로세싱 오버헤드를 가중시키고, 프로세스 데이터 공간을 많이 차지함으로써 동적으로 메시지 정보를 할당하여 전송하는 데에는 많은 문제가 발생할 수 있다.

위와 같은 세 가지 방법들은 TCN 표준의 Event-polling 알고리즘을 그대로 사용한 경우에서 약간의 성능 향상을 기대할 수 있는 방법이지만, 위에서 언급한 세 가지 문제점들을 해결하는 데 있어서 Event-polling 알고리즘의 문제점을 그대로 안게 되어 결국 기대만큼의 성능 향상을 볼 수 없다. 따라서 이러한 문제점들을 해결하기 위해서는 기존의 Event-polling방식에서 제공하는 collision-based arbitration scheme을 사용하지 않고 새로운 방식을 정의하여 미리 메시지 데이터 전송 순위를 스케쥴링할 수 있는 방안이 필요하다. 마스터 노드가 비주기적 전송 구간 이전에 필요한 정보를 모든 취득한 경우에는 메시지 데이터의 선스케쥴링이 가능하며 기존의 방법에서 제기되었던 물리적 어드레스 할당 문제도 쉽게 해결할 수 있고 디바이스 중요성도 고려될 수 있다. 또한, 메시지 전송 정보를 polling을 하지 않기 때문에 슬레이브 노드의 브로드캐스팅에 의해서 발생하는 collision이나 무응답에 의한 시간 지연도 제거할 수 있다. 본 논문에서는 위와 같이 기존의 Event-polling의 문제점들을 해결하기 위해 마스터 노드가 비주기적 전송 구간 이전에 메시지 데이터 정보들을 취득하여 메시지 데이터 전송을 선스케쥴링할 수 있는 Slave Frame Bit-stuffing 알고리즘을 제안하고자 한다.

2. Slave Frame Bit-stuffing Algorithm

Event-polling 방식에서의 문제점을 근본적으로 해결하려면 마스터 노드는 비주기적 전송구간 이전에 미리 메시지 데이터 전송에 대한 정보들을 파악하여 선행 스케쥴링을 통한 메시지 전송 순위를 미리 결정함으로써 Event-polling 방법에 의해 발생하는 시간 지연을 피할 수 있게 되며, 미리 디바이스의 중요도와 메시지의 우선순위를 고려할 수 있게 된다. 이를 위해 Event-polling 방식에서 시간 지연을 크게 줄이고 메시지 전송을 향상시키기 위해서는 물리적 디바이스 어드레스 할당에 관한 방법이 필요하거나 새로운 방식의 메시지 전송 요청 방법이 필요하다.

마스터 노드가 비주기적 전송구간 이전에 미리 메시지 데이터 전송에 대한 정보들을 가지고 있다면 Event-polling 방법에 의한 중재 지연을 피할 수 있고, 미리 우선순위를 고려할 수 있게 된다. 따라서 주기적 전송구간에서 노드들은 마스터 노드에게 메시지 데이터 전송 요청 정보를 전달할 수

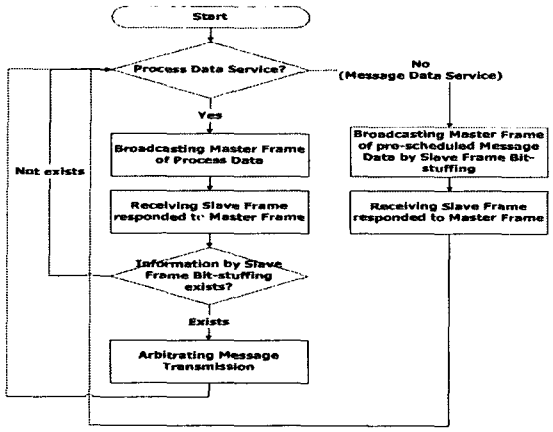

그림 2 마스터 노드에서의 동작 순서
Fig. 2 The flowchart of operation of the master node

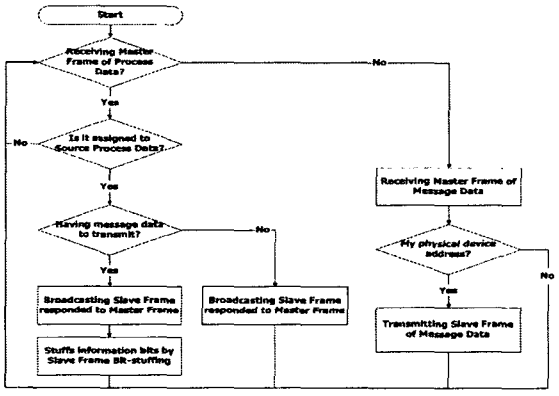

그림 3 슬레이브 노드에서의 동작 순서
Fig. 3 The flowchart of operation of the slave node

있는 방법이 필요하다. 이것을 위하여 슬레이브 프레임에 메시지 데이터 전송에 필요한 정보를 bit 단위로 stuffing하는 Slave Frame Bit-stuffing algorithm(이하 SFB algorithm)을 제안한다. SFB algorithm은 기존의 Event-polling 방식에서 사용하는 충돌에 기초한 중재 방식을 완전히 배제하는 방식을 사용하며 이를 위해 메시지를 전송하려는 노드에 관한 정보를 마스터 노드가 미리 습득함을 그 목적으로 한다.

SFB algorithm은 다음과 같은 순서를 따르게 된다. 마스터 노드는 프로세스 데이터 서비스 구간에서 미리 정의된 스케쥴링에 의거해 마스터 프레임을 브로드캐스팅하게 되고 이를 수신한 슬레이브 노드들은 마스터 프레임을 디코딩하여 마스터 프레임이 자신에게 소스로 할당된 프로세스 데이터를 요구하는가를 판단하게 된다. 이 때, 자신에게 소스로 할당된 프로세스 데이터이면 해당 슬레이브 노드는 슬레이브 프레임을 전송하게 되고 그 경우에 해당 노드가 전송해야 할 메시지 데이터를 가지고 있다면 슬레이브 프레임 뒤의 slave frame-master frame idle 구간에서 메시지 전송 정보를 stuffing 하게 된다. 이 과정을 통하여 해당 프로세스 데이터가 싱크로 할당된 모든 노드들은 프로세스 데이터를 수신하게 되고, 마스터 노드는 해당 프로세스 데이터가 소스로 할당된 슬레이브 노드의 메시지 전송 요청 정보를 취득하게 된다. 그림 2와 3은 마스터 노드와 슬레이브 노드에서 일어나는 동작에 관한 플로우차트를 나타낸다.

SFB algorithm 방식에서 사용하는 중재 방식은 프로세스 데이터 구간에서 브로드캐스팅되는 슬레이브 프레임에 메시지 전송 정보를 stuffing 하는 방식이며, 이 때 stuffing되는 정보는 해당 슬레이브 프레임을 브로드캐스팅한 노드의 메시지 전송 정보가 된다. 즉 임의의 노드가 전송할 메시지를 가지고 있다고 한다면, 해당 노드가 메시지에 대한 정보를 마스터에게 전송하기 위해서는 주기적 전송구간 중 자신에게 소스로 할당된 포트의 데이터를 마스터 노드가 요청한 경우에만 가능하게 된다. 만약, 주기적 전송 구간에서 슬레이브 프레임을 전송하는 기회가 주어지지 않으면 모든 노드들은 메시지 전송 요구를 할 수 없으며 슬레이브 프레임을 전송할 때까지 대기해야 하므로, 버스 관리자는 모든 노드가 동등한 메시지 전송 기회를 가지기 위해 미리 프로세스 데이터의 스케쥴링 리스트를 메시지 전송 요구를 동등하게 할 수 있게 작성함을 전제 조건으로 한다. 따라서 모든 노드는 슬레이브 프레임을 전송할 때 자신에게 전송하려고 하는 메시지가 존재하는지를 검사하여 메시지가 존재한다면 해당 정보를 슬레이브 프레임에 stuffing하여 마스터 노드에게 알리게 된다.

TCN 표준에서는 주기적 전송 구간에서 슬레이브 프레임의 전송이 끝나고 다음 마스터 프레임이 전송되기 전까지의 시간적 간격을 최소 3 μs(충돌이나 무응답일 경우를 제외한)로 규정하고 있다[1][2]. 슬레이브 프레임의 전송이 끝난 후 버스가 idle 상태가 되는데 걸리는 시간은 MVB의 세 가지 물리적 계층(Electrical Short Distance, Electrical Middle Distance, Optical Glass Fiber)에서 최대 300ns이다. 즉 bit-stuffing은 최소 300ns 이상, 최대 3 μs 이내에 시작되어야 한다. bit-stuffing에 의해서 더해지는 정보는 메시지 데이터 전송 요구 및 그 우선순위가 된다. 각 디바이스는 자신이 전송할 메시지를 가지고 있음을 표시해야 하므로 자신의 물리적 어드레스를 bit-stuffing 정보에 포함시켜야 하며 이 물리적 어드레스는 12 bits이다. 또한, 전송하고자 하는 메시지의 우선순위 정보를 추가시켜야 하는데 기본적으로 TCN에서 정의한 메시지 우선순위는 높은 우선순위와 낮은 우선순위의 두 가지이므로 우선순위에 필요한 bit는 1 bit이다. 여기서, 물리적 어드레스를 12 bits로 하고 우선순위를 기존의 방법과 같이 두 가지로 한정한 것은 TCN 표준에 의해 구축된 기존 MVB 네트워크와 호환성을 유지하기 위함이다. 위와 같이 메시지 전송에 필요한 정보는 13 bits이지만 슬레이브 프레임에 stuffing되는 메시지 정보를 MVB 컨트롤러가 인식하기 위해서는 최초 bit가 1이어야 마스터 노드에서 인식할 수 있으므로 start bit로 1 bit가 필요하다. 따라서 SFB에 필요한 bit는 총 14 bits가 된다. SFB에 사용되는 프레임은 아래와 그림 4와 같은 형식을 취하며, 각 bit와 시간 변수의 의미는 다음과 같다.

- bit 1: 전송할 메시지 데이터가 있으면 '1'로 세팅한다.(Start bit)
- bit 2: 전송할 메시지 데이터가 High-priority인 경우는 '1'로 세팅하고, Low-priority인 경우는 '0'으로 세팅한다.
- bit 3~14: 자기 자신의 물리적 고유 어드레스를 세팅한다.
- t_{mr}: 슬레이브 프레임과 SFB에 의한 프레임 사이의 시간 간격으로 2 μs를 권장한다.

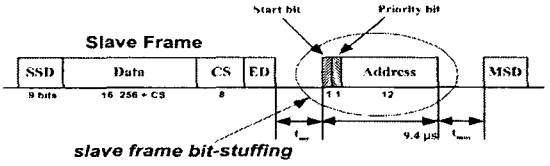

그림 4 Slave Frame Bit-stuffing 프레임 포맷
Fig. 4 The frame of Slave Frame Bit-stuffing

● t_{mm}: SFB에 의한 프레임과 마스터 프레임 사이의 시간 간격으로 2 μs를 권장한다.

마스터 노드는 주기적 전송구간 내에 받은 SFB에 의한 프레임들을 디코딩한 후 스케쥴링하여 중재 없이 메시지 데이터 서비스를 시작한다. 이를 위하여 마스터 노드에는 Event-polling에서 사용되었던 이진 트리 탐색 알고리즘을 기본 스케쥴링 정책으로 사용할 수 있다. 또한, SFB 방법은 선스케쥴링이 가능하므로 미리 각 디바이스의 중요성을 고려하여 디바이스 우선순위를 MVB 네트워크에 미리 설정하여 놓는 방법을 이용하면 같은 우선순위를 갖는 메시지 전송 요청 정보를 이진 탐색 트리를 이용하지 않고 디바이스 우선순위를 이용하여 메시지 전송을 미리 스케쥴링할 수 있다. 즉, 위와 같은 정책들을 사용한다면 SFB 방식에 의해서 받은 메시지 전송 정보 중 메시지 우선순위가 높게 설정된 디바이스와 기본적으로 그 중요성이 높은 디바이스만 먼저 검사하게 되고 그 디바이스들 간에서는 메시지 arbitration을 이진 트리 탐색 알고리즘을 이용하여 메시지 전송 슬레이브 노드를 결정하게 된다. 물론, 위와 같은 경우에서 메시지 우선순위와 디바이스 중요성이 모두 높은 디바이스는 먼저 검색되어 제일 먼저 메시지를 전송할 수 있다. 이것은 어디까지나 MVB 네트워크가 어떤 디바이스들로 그리고 어떤 방식으로 구성되어 있는가에 의해 호스트 관리자가 가장 효율적으로 메시지를 전송할 수 있는 정책을 적용하는 것이 가장 좋다고 할 수 있다.

마지막으로, 이런 새로운 SFB 방식을 기존 MVB 네트워크에 사용한다 하더라도 문제없이 호환될 수 있는데, 그 이유는 SFB 방식에서의 Bit-stuffing은 기존 MVB 네트워크에서의 idle 상태에서 이루어지게 되므로 기존 MVB 네트워크 상에서의 데이터 충돌이나 시간제약적인 문제들이 발생하지 않는다. 또한, Bit-stuffing에 대한 정보는 SFB 방식이 적용된 마스터 노드만이 그 정보를 받아들여 사용할 수 있으므로 그 정보를 받은 마스터 노드는 미리 세팅된 arbitration 정책에 의해 메시지 전송을 arbitration한다. 만약 마스터 노드가 SFB 방식이 적용되지 않은 기존 MVB 네트워크용 마스터 노드라면 당연히 이 정보는 필요 없는 데이터가 되어 무시되게 되며 마스터 노드는 그 정보에 상관없이 기존의 Event-polling 방식을 사용하여 메시지 전송을 arbitration하게 된다. 위에서 언급한 것처럼 메시지 전송은 기본적으로 마스터 노드에 의해 주도되고 슬레이브 노드는Bit-stuffing 이외에는 마스터 노드가 전송하는 마스터 프레임에 의해 응답하게 되어 있으므로 SFB-stuffing 방식은 기존의 MVB 네트워크와도 문제없이 호환될 수 있다는 장점을 가진다.

3. 성능 분석

3.1. 해석적 방법에 의한 Bit-stuffing 알고리즘 성능 분석

해석적 방법에 의해서 메시지 데이터가 전송될 때 생기는 시간 지연에 의한 성능 분석을 하였다. 발생하는 시간 지연은 TCN 표준에 나와 있는 기본적인 시간 변수를 이용하였고, Event-polling 방법과 SFB 방법 모두 이진 트리 탐색 알고리즘을 사용하였고 MVB 네트워크를 구성하고 있는 디바이스들의 중요도는 모두 동일하게 적용하였다[1].

전체 노드 수를 변화시키면서 Event-polling 방법과 SFB 방법에서 메시지 패킷이 전송되는데 발생하는 시간 지연을 비교하였다. 노드 수는 4개에서 32개까지 2배씩 증가시키고, 전송대기 중인 메시지는 2개로 가정하였다. 또한, 노드의 물리적 어드레스는 12 bits를 모두 사용하지 않는 것으로 가정하였는데 이는 12 bits를 모두 사용하게 되면 트리 구조의 레벨이 일정해 지므로 노드수와 상관없이 시간 지연값의 최대값이 변하지 않게 되기 때문이다. 따라서 물리적 어드레스 할당은 Event-polling에서 제일 좋은 성능을 보일 수 있는 정적 어드레스 할당 방법을 사용한다. 아래 표 1은 그 결과를 표시한다. 시간 지연은 최소 시간 지연과 최대 시간지연을 표시하였다.

결과에서 알 수 있듯이 SFB 방식은 메시지 전송을 위한 부가 정보를 슬레이브 프레임에 덧붙이는데 드는 시간을 제외한 어떠한 시간 지연도 발생하지 않는다. 그러나 Event-polling 방식은 정적 어드레스 할당 방식을 사용했음에도 불구하고 그 시간 지연이 최소인 경우에도 굉장히 크다는 것을 알 수 있다.

TCN 표준에서 권장하는 메시지 전송 구간은 기본 주기의 40%인데 기본 주기를 1 ms, 메시지 데이터를 256 bits, 시간 지연은 최소라고 가정하였을 경우 실제 메시지 데이터를 전송하는데 필요한 텔레그램의 전송 시간은 220 μs이다. 따라서 시간 지연을 최대라고 가정하면 노드수가 4개인 경우를 제외한 모든 경우는 기본 주기 1 ms안에 메시지를 전송할 수가 없으며, 물리적 어드레스를 12 bits 모두 사용하는 경우에는 그 시간 지연이 기본 주기의 2배인 2 ms안에도 메시지 데이터를 전송할 수가 없는 치명적인 문제점을 가지고 있다. 게다가 시간 지연이 최대라고 가정할 경우 그 문제점은 더 심각해진다. 하지만, SFB 방식은 어떠한 경우에도 기본 주기 안에는 무조건적으로 메시지 데이터 전송을 마칠 수 있는 장점을 가지게 된다. 또한, 제안된 SFB 방식은 위와 같은 장점 이외에도 현재 MVB 네트워크 시스템을 구축하는 데 사용되고 있는 Adtranz사의 상용 MVB Controller와 호환성을 유지할 수 있다[9].

3.2. 시뮬레이션에 의한 Bit-stuffing algorithm성능 분석

Event-polling 방식에 의한 MVB 시스템과 SFB 방식에 의한 MVB 시스템을 상용 시뮬레이션 툴인 Arena로 모델링하여 시뮬레이션을 수행하였다. 시뮬레이션 조건은 다음과 같이 가정하였다.

표 1 Event-polling과 Slave Frame Bit-stuffing의 시간 지연 비교

Table 1 Comparison of the delay of Event-polling and the delay of Slave Frame Bit-stuffing

Number of nodesMin(μs)	Slave Frame Bit-stuffing algorithm		Event-polling algorithmMax(μs)	
	Min(μs)	Max(μs)	Min(μs)	Max(μs)
4	19		108.7	129.4
8				194.1
16				258.8
32				323.5

● SFB 방식에서 마스터 노드는 메시지 데이터 스케쥴링 알고리즘으로 Event-polling 방식에서 사용되는 이진트리 탐색(Binary Search Tree)을 사용하고 MVB 네트워크를 구성하고있는 디바이스들의 중요도는 모두 동일하게 적용하였다.

● 마스터권 권양 이양(Mastership Transfer)는 발생하지 않는다.

● 프로세스 데이터가 전체 네트워크의 점유율이 60%를 넘지 않게 하기 위해서 각 basic period내에서는 프로세스 데이터 전송을 최대 10개로 한정하고 각 프로세스 데이터는 F-code를 1(16bit)로 한다.

● MVB 통신의 데이터 전송 시간은 TCN 표준에 준거하여 두 시나리오에 적용한다.[1]

● Event-polling 방식에서 각 노드의 물리적 어드레스는 메시지 전송의 시간 지연을 최대한 줄이기 위해 연속되게 할당하였다.

● 메시지의 우선순위는 높은 우선순위와 낮은 우선순위로 구성되며, 각 디바이스의 중요도는 반영하지 않는다.

● 기타 통신상의 장애는 발생하지 않는다.

노드를 32개까지 한정한 것은 TCN 표준에서 정의한 MVB 네트워크의 최대 디바이스 개수이기 때문이다. 이 중에서 노드가 16개로 구성된 MVB 네트워크는 현재 우리나라에서 개발중인 HSR-350 고속전철의 차량 내 최대 디바이스 개수를 기준으로 한 것이다. HSR-350은 7개의 차량으로 구성되어 있으며, 자체개발한 TCN이 적용된 네트워크 카드를 사용하고 있다. HSR-350의 차량 중 최대의 디바이스 개수를 갖는 차량은 Power Car로서 Power Car1과 Power Car2, 모두 이중화 디바이스(Gateway, SCU, ATC)까지 총 16개의 MVB 네트워크 디바이스를 탑재하고 있다. 두 알고리즘 모두 메시지 개수를 10000 개로 한정, 발생시켜 시뮬레이션 하였으며 프로세스 데이터 구간은 제외시키고 비주기적 전송 구간에서 관리형 데이터 서비스는 일어나지 않는다는 가정 하에 순수 메시지 처리에 드는 시간 비용만 측정한 것이다. 프로세스 데이터 구간을 제외한 이유는 메시지 전송 구간은 프로세스 데이터 구간의 조정에 의해서나 또는 기본 주기의 조정에 의해서 얼마든지 그 길이를 조절할 수 있기 때문이다.

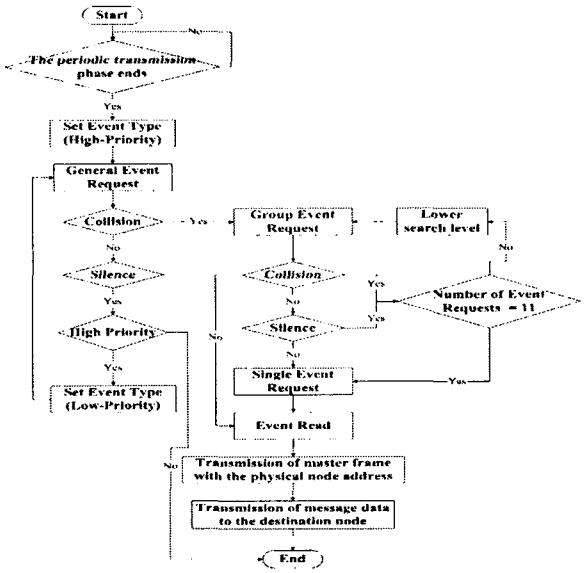

그림 5 Event-polling algorithm의 모델

Fig. 5 Modeling of Event-polling algorithm

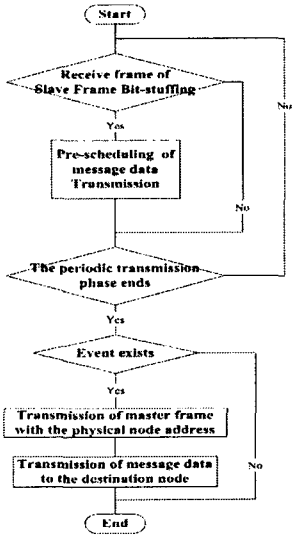

그림 6 Slave Frame Bit-stuffing algorithm의 모델

Fig. 6 Modeling of Slave Frame Bit-stuffing

단, SFB algorithm은 프로세스 데이터 구간에서 bit-stuffing이 일어나기 때문에 이 부분의 시간 지연은 비주기적 전송 구간에서 일어나는 것으로 가정하여 시뮬레이션 하였다. 따라서 아래 결과 그래프의 SFB 알고리즘이 적용된 메시지 전송에서의 시간 지연은 프로세스 구간에서 bit-stuffing에 의해 발생하는 시간 지연을 의미한다.

그럼 7의 그래프에서 알 수 있듯이 노드 개수가 증가함에 따라 Event-polling 방식은 시간 지연이 많이 증가하는 것을 알 수 있다. 이는 트리 레벨의 증가로 인해 Group Request 및 충돌, 무응답이 늘어나기 때문이다. 그에 반해 SFB algorithm의 경우에는 시간 지연이 비교적 일정하다고 할 수 있다. Event-polling 방식에서는 노드수가 32개인 경우에는

시간 지연이 다른 노드 수들의 결과보다 많이 증가하는 것을 알 수 있는데 이 또한 메시지 전송 요청의 증가로 인한 중재 시간 지연이 크게 발생하기 때문이다. 그러나 SFB algorithm의 경우에는 노드 개수가 32개인 경우에도 그 시간 지연이 별로 증가하지 않는다. Event-polling 방식에서는 메시지 전송 요청을 하고자 하는 디바이스의 개수 이외에도 디바이스의 어드레스 할당에 의해서 시간 지연이 영향을 받는데 비하여 SFB algorithm은 단순히 메시지 전송 요청을 하고자 하는 디바이스의 개수에 의해 시간 지연이 영향을 받기 때문이다.

시뮬레이션 결과에 의하면 각 경우에 Event-polling 방식이 SFB algorithm보다 적게는 700 ms에서 많게는 1000 ms 이상 시간 지연을 더 발생하게 되는데 메시지 데이터가 실제 전송되는 시간은 약 220 μs이므로 메시지 데이터를 3000개에서 4500개 정도를 더 전송할 수 있는 시간에 해당하며, 성능 향상 입장에서 보면 약 30% 정도의 성능 향상이 있다고 볼 수 있다. 실제로는 Slave Frame Bit-stuffing algorithm에서도 약간의 시간 지연이 발생하기 때문에 위에서 언급한 개수의 메시지 전송은 가능하지 않더라도 상당한 숫자의 메시지 데이터를 전송할 수 있음에는 틀림없다. 또한, 위 시뮬레이션에서 Event-polling에 사용한 어드레싱 할당 방법은 앞에서 제시한 정적 어드레스 할당 방법을 사용한 것으로 MVB의 각 네트워크 디바이스는 밸런스형 트리로 구성된 경우이기 때문에 만약, 12 bits의 모든 물리적 어드레스를 사용하고 디바이스의 중요도를 고려하게 된다면 당연히 그 시간 지연은 훨씬 더 크게 일어날 것으로 예측할 수 있다.

다음 시뮬레이션은 HSR-350에 해당하는 노드 16개의 구성을 기준으로 각 노드에서 메시지 발생 주기를 2, 4, 8 ms로 변화시켜 수행하였다. 주기 1 ms의 경우는 Event-polling 방식에서 이진 트리 탐색 시 일정한 검색 과정을 수행하기 때문에 시간 지연값이 일정하게 되므로 시뮬레이션에서 제외하였다.

그림 8 에서와 같이 메시지 발생 빈도를 변화시키면 두 알고리즘 모두 시간 지연이 늘어나게 되지만, 어떠한 발생 빈도에서도 SFB 방식이 Event-polling 방식보다 시간 지연이 적음을 알 수 있다. Event-polling 방식에서는 메시지 발생 주기가 짧아지면 비주기적 전송 구간에서 충돌이 많이 발생하게 되어 시간 지연이 크게 증가하는 특성을 가지게 되고 그 주기가 기본 주기와 같아지면 탐색 시간 지연은 최대값으로 일정해지게 된다. 이는 이진 트리 탐색 순서가 항상 일정하기 때문에 메시지 전송 노드 선택 또한 일정한 노드가 선택될 수 있고 노드의 개수가 16개이므로 트리 레벨이 4로 일정하기 때문이다. 이것은 메시지 발생 주기가 짧은 경우에는 일정 노드들의 메시지 전송 구간 점유라는 심각한 문제가 발생할 수 있다는 것을 말해준다. SFB 방식은 메시지 발생 주기가 짧아짐에 따라 그 시간 지연도 거의 비례적으로 늘어나게 되지만, 한번의 기본 주기에서 최대로 전송 가능한 프로세스 데이터는 한정되어 있으므로 Event-polling 방식보다 시간 지연은 줄어들게 되고 일정 이상 증가하지 않게 되며 선스케줄링을 이용하여 특정 노드가 메시지 전송을 점유하는 문제점을 방지할 수 있다.

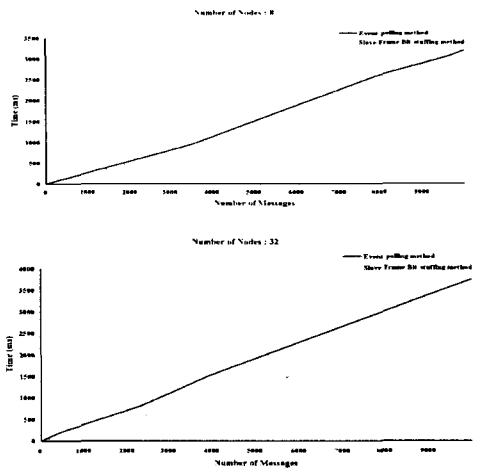

그림 7 MVB 네트워크에서의 두방식에 의한 시간지연비교
Fig. 7 The delay comparison of two algorithms in MVB network

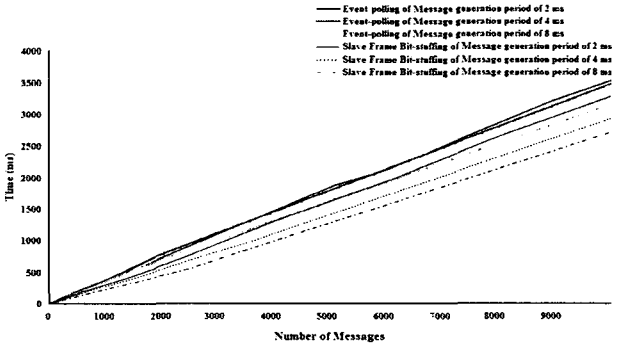

그림 8 메시지 발생 빈도에 따른 지연시간
Fig. 8 The delay comparison to the message frequency

4. 결 론

본 논문에서는 TCN의 MVB에서 메시지 데이터 전송을 향상시키기 위하여 SFB 방식을 제안하였으며, 이를 통하여 다음과 같은 이점을 가질 수 있다.

첫째, 메시지 전송 정보를 취득할 때 필요로 하는 프레임의 개수와 그 처리 시간이 Event-polling에 비해서 현저하게 줄어든다. 둘째, 기존 방법에서는 우선순위별 폴링을 따로 실행하여 낮은 우선순위의 메시지 전송의 경우에는 불필요한 프레임에 의한 시간 지연이 발생했지만, SFB 방식에서는 메시지 데이터 정보를 전송 구간 이전에 미리 취득함으로써 메시지 우선순위에 기초한 선스케줄링이 가능하기 때문에 불필요한 시간 지연을 줄일 수 있다. 셋째, 선스케줄링이 가능하기 때문에 같은 우선순위를 가지는 메시지에서도 디바이스의 중요성에 따라 전송 순위를 결정할 수 있다.

향후 연구 과제로는 SFB 방식을 사용할 경우 프로세스 데이터 서비스 구간에서 bandwidth 손실이 일어나게 되는데 이 손실이 프로세스 데이터 서비스에 영향을 미치지 않게 하는 프로세스 데이터 동적 스케줄링에 대한 연구가 필요하다.

참 고 문 헌

[1] IEC 61375-1 Standard, Train Communication Network: Part (1) General Architecture (2) Real-time Protocol (3) Multifunction Vehicle Bus (4) Wire Train Bus (5) Train Network Management (6) Train Communication Conformance Testing, 1999.

[2] UIC 556 Standard, Information Transportation on the Train Bus, 1999. 05. 01. v2.0

[3] H. Kirrmann and P. A. Zuber, "IEC/IEEE Train Communication Network", 1996.

[4] G. Fadin and F. Cabaliere, "IEC TC9 WG22 Train Communication Network Dependability and Safety concepts", World Congress on Railway Research 97, 1997.

[5] P. Etter and H. Kirrmann, "Use of standardized integrated communication systems on vehicles and trains", Proceedings of the International Conference on Speedup Technology for Railway and Maglev Vehicles, Vol. 1, 1993.

[6] J. H. Park and S. C. Lee, "Performance Evaluation of the Train Communication Network", Workshop on Distributed Computer Control Systems, 1998.

[7] C. Schifers and G. Hans, "IEC 61375-1 and UIC 556-international standards for train communication", Vehicular Technology Conference Proceedings, VTC 2000-Spring Tokyo.

[8] J. Xu and D. L. Parnas, "On satisfying timing constraints in hard-real-time systems", ACM SIGSOFT, 1991.

[9] ABB Daimler-Bensz Transportation(Switzerland) Ltd, "Multifunction Vehicle Bus Controller", Adtranz, 1997.

[10] A. Chavarria, J. L. de Arroyabe, A. Zuloaga, J. Jimenez, J.L. Martin, G. Aranguren, "Slave node architecture for train communications networks", IEEE Conference, Vol. 4, 2000.

[11] S. Cavalieri, A. Di Stefano and O. Mirabella, "Acyclic Traffic Management in FieldBus Scenarios", IEEE International Workshop, 1992.

[12] S. Cavalieri and O. Mirabella, "Enhancing Performance in FieldBus Communication Systems", IEEE Conference, Vol. 2, 1996.

[13] L. R. H. R Franco, "Transmission scheduling for fieldbus: a strategy to schedule data and messages on the bus with end-to-end constraints", IEEE International Joint Symposia, 1996.

저 자 소 개

최 명 호(崔 明 浩)

1976년 6월 29일생. 2002년 인하대 자동화공학과 졸업(공학사). 2004년 정보 통신 대학원(공학석사). 2004~현재 삼성 전자 근무. 관심분야는 내장형시스템, 실시간 시스템.

Tel : 032-863-0442

E-mail : mhchoi@resl.inha.ac.kr

문 종 천(文 鍾 千)

1979년 9월 2일생. 2003년 인하대 컴퓨터공학과 졸업(공학사). 2003년~현재 인하대 정보통신공학과 석사과정. 관심분야는 내장형시스템, 실시간 시스템.

Tel : 032-863-0442

E-mail : ccmoon@resl.inha.ac.kr

박 재 현(朴 宰 賢)

1963년 10월 8일생. 1986년 서울대 제어계측공학과(공학사). 1988년 동 대학원(공학석사). 1994년 동 대학원(공학박사). 1995년~현재 인하대 정보통신공학부, 부교수. 관심분야는 내장형시스템, 실시간시스템, 컴퓨터네트웍

Tel : 032-860-7713, Fax : 032-873-8970

E-mail : jhyun@inha.ac.kr