이기종 필드버스 통합을 위한 필드버스 게이트웨이 설계 및 구현

이영민, 김명균

울산대학교 컴퓨터정보통신공학부

The Design & Implementation of Fieldbus Bridge for Integration of different Fieldbus networks

Yeong-min, Lee · Myung-kyun, Kim

Department of Computer Engineering and Information Technology, University of Ulsan

E-mail : imleemin@nate.com, mkkim@ulsan.ac.kr

요 약

본 논문에서는 CAN네트워크와 Profibus 네트워크를 통합하기 위한 통합 프로토콜 게이트웨이를 설계, 구현하였다. CAN/Profibus 네트워크를 통합하기 위하여 스위칭이더넷을 백본 네트워크로 사용하였고, 각각의 필드버스 네트워크는 CAN/Ethernet 및 Profibus/Ethernet 게이트웨이를 이용하여 이더넷 스위치에 연결되며, 각 게이트웨이는 필드버스 프레임과 이더넷 프레임 사이의 변환을 수행한다. 또, 스위칭 이더넷 환경에서 게이트웨이 사이에 분산 경성 실시간 스케줄링 알고리즘을 적용하여 메시지의 실시간성을 만족할 수 있게 하였다. CAN/Ethernet 및 Profibus/Ethernet 게이트웨이 구현을 위해 실시간 메시지 전송을 지원하는 리눅스 2.6.31.12 Real-Time Patch 버전을 사용하였고, 실제 구현을 통해 메시지의 성공적인 변환과 실시간성을 확인할 수 있었다.

ABSTRACT

In this paper, we have designed and implemented the integrated protocol gateway for the integration of CAN and Profibus networks. To do that, we used the Switched Ethernet as Backbone network, and each fieldbus network is connected by CAN/Ethernet or Profibus/Ethernet gateway, and each gateway perform the translation between fieldbus and Ethernet frames. Futhermore, we realized the real-time features in the environment of the Switched Ethernet by applying the distributed hard real-time scheduling algorithm among each gateways. To implement tht CAN/Ethernet and Profibus/Ethernet gateways, we used the Linux of kernel 2.6.31.12 real-time patched version(PREEMTED_RT), and we could verify successful message translation and real-time features through real implementation.

키워드

통합 필드버스, 스위칭이더넷, 실시간 메시지 스케줄링

Ⅰ. 서 론

오늘날 대부분의 산업 현장에서는 다양한 필드버스를 이용해 공장 자동화 네트워크를 구축하고 있다. 필드버스 네트워크는 1980년대 말 유럽에서부터 활발히 진행된 표준화 연구에 의하여 각 업체들의 공동체별로 다양한 표준이 나오게 되었다.[1]

각각의 표준을 따르는 제어기들은 동일한 통신 규격을 따르므로 하나의 산업용 네트워크로 구성되어 작동될 수 있다. 그러나 실제 산업 현장에서는 하나의 표준을 따르는 장비들로 구성되기보다 각각의 표준을 따르는 제어기들이 서로 독립된 네트워크를 형성하여 전체 네트워크로 구성되어지는 경우가 많다. 이러한 경우, 서로 다른 필드버스 네트워크는 접근 방식 및 프로토콜이 다르므로 하나의 네트워크로 통합하여 관리하기 힘들다는 문제점을 지니고 있다.

본 논문에서는 이러한 통합에 따른 문제점을 해결하기 위해 스위칭 이더넷(Switched Ethernet)을 이용한 통합 프로토콜 게이트웨이를 설계하였다.

각 필드버스는 게이트웨이를 통해 이더넷으로 연결이 되어지고, 게이트웨이는 필드버스와 이더넷 프레임 사이의 변환을 수행한다. 이더넷에서 메시지의 실시간성을 보장하기 위해 분산 경성 실시간 알고리즘을 적용하였고, 실험을 위해 필드버스 프로토콜 중에서 가장 널리 쓰이는 Profibus와 CAN을 통합하는 작업을 수행하였다.

2절에서는 통합을 위한 측면에서 CAN과 Profibus에 대해 기술하고, 3절에서는 통합 게이트웨이의 설계에 대해서 계층별로 기술하며, 4절에서 구현에 관하여 기술한 후, 마지막 5절에서 결론 및 향후 연구 방향에 대해서 기술한다.

Ⅱ. CAN / Profibus

본 절에서는 통합을 위해 CAN과 Profibus의 데이터 링크 계층에 대해서 간략하게 살펴본다.

1. CAN(Control Area Network)

CAN은 차량내부 네트워크 제어를 위해 독일 Bosch사에 의해서 개발된 프로토콜로, 우수한 실시간 처리능력으로 인해서 많은 산업용 장비에 사용되고 있다. 버스 방식의 연결 구조를 가지고 최대 1Mbit/s의 속도를 제공할 수 있다. CSMA/CD AMP(Arbitration with Message Priority) 접근 방식을 사용하며 메시지 기반의 통신을 채택하여 주소가 아닌 메시지 식별자(Identifier)에 의해 구별되어지는 것이 특징이다.

CAN이 하나의 프레임을 통하여 전송할 수 있는 데이터의 크기는 0 ~ 8 바이트로 제한되며, 식별자를 통한 경쟁에서 우선 순위가 낮은 메시지는 계속해서 경쟁에서 뒤지게 되는 문제점이 발생할 수 있다. [2]

2. Profibus

Profibus(Process Fieldbus)는 토큰 패싱 방식을 이용하는 필드버스로써 하나 이상의 마스터와 슬레이브로 구성된다. 마스터 스테이션은 다수의 슬레이브를 가질 수 있으며, 버스 및 트리 구조의 토폴로지로 구성될 수 있다. 토큰 순회는 오직 마스터 스테이션 사이에서만 이루어지고, 토큰을 소유한 마스터만이 자신에게 속한 슬레이브나 다른 마스터에게 메시지를 전송할 수 있다. 슬레이브 스테이션은 마스터 스테이션의 요청에 대한 응답만이 가능한 수동적인 통신을 수행한다.

Profibus의 전송 가능한 데이터 크기는 8 바이트(고정길이 데이터 프레임)에서 최대 246바이트(가변길이 데이터 프레임)까지 가능하다. 프레임의 각 필드는 1바이트씩 구성되며, Profibus 프레임의 모든 바이트는 UART 문자로 전송된다. [3]

Ⅲ. 통합 프로토콜 게이트웨이 설계

본 논문에서 제안하는 통합 프로토콜 게이트웨이의 인터페이스는 CAN-Ethernet, Profibus-Ethernet 두 가지의 형태이다. [그림 1]은 통합 게이트웨이를 이용한 통합 필드버스 네트워크 구조를 나타내고, [그림 2]는 필드버스 통합을 위한 프로토콜 계층 구조를 나타낸다.

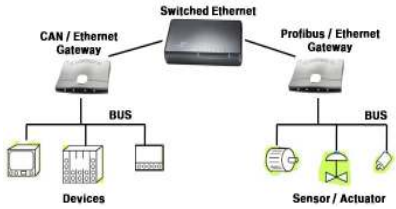


그림 1. 통합 필드버스 네트워크

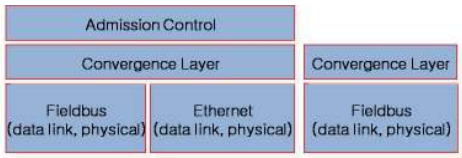


그림 2. (a) 게이트웨이 (b) 필드버스 노드

1. 논리 주소와 물리 주소

서로 다른 필드버스 프로토콜을 통합하여 데이터를 효과적으로 전달하기 위해서는 공통의 주소 지정 방식이 필요하다. 통합 환경에서 각 필드버스 노드는 논리 주소와 물리 주소를 가지게 되는데, 논리주소는 게이트웨이와 이더넷으로 연결된 전체 네트워크에서 각 노드가 가지는 유일한 주소를 나타내며, 물리 주소는 각 필드버스의 프로토콜에 따라 사용되는 기존 주소이다. 논리 주소는 16비트(2바이트)로 이루어져 있으며, 최상위 비트인 0번째 비트는 멀티캐스트 여부를 나타내고, 1~7비트까지 7비트는 통합 필드버스 게이트웨이의 주소를 뜻한다. 하위 8~15비트는 한 필드버스의 각 노드의 식별자로 사용된다.

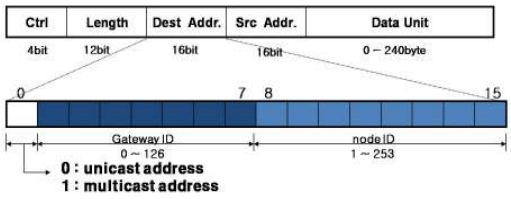


그림 3. 논리 필드 버스 주소 및 프레임 구조

2. Convergence 계층

Convergence 계층은 필드버스 데이터 링크 계층 상위에 존재하면서 데이터링크 프레임과 Convergence 계층 프레임 간의 상호 변환 및 프레임 분할, 재조립의 역할을 수행한다.

3. Admission Control 계층

Admission Control 계층은 Convergence 계층의 상위에서 Convergence 계층 프레임의 스케줄링을 관리하고 주소 사상 테이블 및 브리지 테이블을 관리하는 역할을 한다. 주소 사상 테이블은 논리주소와 그에 해당하는 물리주소의 1:1 맵핑 정보를 저장하는 테이블로 데이터링크 프레임에서

Convergence 계층 프레임으로 변환될 때 사용되어지고, 브리지 테이블은 다른 게이트웨이의 MAC Address와 해당하는 논리 주소의 1:1 맵핑 정보를 저장하는 테이블로 각 게이트웨이에서 다른 게이트웨이로 프레임을 전송할 때 사용되어진다.

4. 프로토콜 변환

각 노드에서 다른 필드버스 네트워크로 데이터를 전송하고자 할 때, 자신의 논리주소를 송신 주소로 하고, 목적지 노드의 논리 주소를 수신 주소로 하는 Convergence 계층 프레임을 구성한 후, 해당 필드버스 프레임의 데이터 필드에 넣어 전송한다. 이 프레임은 게이트웨이로 전달되어지고, 게이트웨이에서는 목적지 논리 주소에 해당하는 필드버스 게이트웨이로 메시지를 전송하기 위해 브리지 테이블 정보를 이용하여 이더넷 프레임을 생성한 후 전송한다. 목적지 필드버스의 게이트웨이는 이 메시지를 다시 자신의 프로토콜에 맞게 변환한 후 목적지 논리 주소에 해당하는 물리 주소로 메시지를 전송한다.

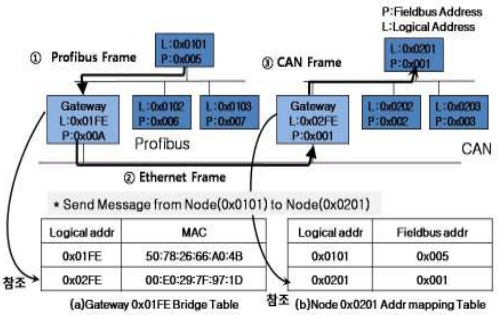


그림 4. 프로토콜 변환 예

5. 스케줄링 알고리즘

Admission Control 계층에서는 이더넷 스위치 환경에서 발생할 수 있는 출력 큐의 큐잉지연을 효과적으로 제어하여 메시지의 실시간성을 만족시키기 위해 스케줄링을 수행한다. 본 논문에서는 스위칭이더넷에서 경성 실시간 메시지 전송을 위해 본 연구실에서 개발한 메시지 스케줄링 알고리즘을 적용하였다.[4] 분산 환경에서 이루어지므로, 스위치에 다른 기능추가 없이 구현이 가능하다.

스케줄링 알고리즘에서 각 송수신 링크는 Macro Cycle(MC)의 연속으로 구성되어 있다. MC는 또 기본적인 전송 단위인 Elementary Cycle(EC)로 나누어지고, 각 EC는 Periodic Cycle(PC)와 Aperiodic Cycle(AC)로 구분되는데, 이 때 PC는 주기적 메시지의 전송을 위해 사용되고, AC는 비주기적 메시지의 전송을 위해 사용된다.

각 노드의 네트워크 초기화 과정에서 미리 정해진 주기와 마감시간을 가지고 메시지가 생성되면, 이 정보는 각 게이트웨이로 전송되어 지고, 스케줄링 알고리즘을 통해 해당 메시지가 마감시간 내 전송 가능한지를 검사하는 진입제어 프로세스를 AC 구간에서 수행한 후, 해당 메시지의 주기에 해당하는 EC의 PC 구간 스케줄링 정보를 업데이트하게 된다. 전송 동기화를 위해 마스터 게이트웨이를 사용하여 일정 주기로 TM(Trigger Message)를 전송하고, 마스터는 TM 전송외에 스케줄링에 관련된 다른 동작은 하지 않는다.

Ⅳ. 통합 프로토콜 게이트웨이 구현

설계 내용을 바탕으로 통합 프로토콜 게이트웨이를 구현하였다. 개발 환경은 실시간 전송을 위해 선점이 가능하고 고정밀 POSIX 타이머 기능을 가지는 리눅스 커널 2.6.31.12 리얼타임 패치를 적용한 버전을 사용하였고, 스위치는 NETGEAR 사의 Fast Ethernet Switch FS608 v2를 사용하였다. FS608 v2는 100Mbps로 동작 시 20us 이상의 스위칭 지연을 지니고 있으며, Store and Forwarding 방식으로 동작한다. Profibus와 CAN은 Softing사의 Profiboard-PCI, CAN-ACx-PCI를 각각 사용하였고, 본 구현에서는 기존 연구를 토대로 Convergence 계층 개발에 이어 Admission Control 계층에서 스케줄링 알고리즘을 적용하였다.

CAN에서 최대 데이터 사이즈는 8 바이트 이므로 Profibus에서 CAN으로 8 바이트 보다 큰 데이터가 전송될 때는 분할 과정을 거치게 된다. 그러므로, 스케줄링의 실시간성을 검증하기 위해, 프레임 변환 과정과 스케줄링 과정을 분리해서 실시하였다. 스케줄링을 위한 실험 구성은 총 5개의 노드와 하나의 스위치를 사용해 각 노드와 스위치를 전이중 방식으로 연결하였고, 1 개 노드는 각 EC의 시작을 알리기 위해 TM을 전송하는 마스터 노드로 동작하게 하여 전송시 동기화를 수행하였다.

TM은 64 바이트의 크기를 가지고, MC(한 MC 내의 EC의 수), EC, PC, AC 구간에 대한 정보를 가지고 있다. 본 실험에서는 M=6, EC=2ms, PC=1.2ms, AC=0.8ms의 값을 지닌다. 본 실험에서 각 노드는 메시지 길이가 [70, 120]us 사이의 값을 갖고 주기는 {1, 2, 3} 중 하나의 주기를 갖는 30개의 메시지를 임의로 생성하여 이들을 메시지 버퍼에 저장 하고 있다.

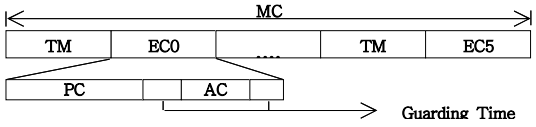


그림 5. 실험에 사용된 전송 모델

게이트웨이 구현에 사용된 스케줄링 알고리즘은 경성 실시간 스케줄링 알고리즘이므로, 각 사이클 구간에서 엄격한 시간 동기화를 요구한다. 그러므로 모든 메시지들은 해당하는 PC와 AC에서 반드시 전송을 마칠 수 있도록 해야 된다. 하지만 실제 시스템에서는 제한된 Time Resolution

과 같은 시스템적인 한계로 인해 이러한 동기화를 수행하는데 어려움이 따른다.

본 구현에서는 각 PC와 AC 구간의 끝부분에 보호구간(Guarding Time)을 설정하여, 시스템의 내부 처리에 따른 지연 발생에도 잘 동작할 수 있도록 구현하였다. 그러므로 실제 메시지들은 각 보호구간 이전까지 전송을 시작할 수 있고, 진입 제어 수행시 계산을 위해 사용되는 실제 PC의 구간 값은 보호구간을 뺀 값을 사용하게 된다. PC 보호구간은 PC를 위해 설정된 Timer의 콜백 함수가 호출되는 시간과 실험시 사용된 가장 긴 메시지(120us)길이를 더한 값으로 설정하였고, AC 보호구간은 진입제어를 위해 송신자와 수신자 간에 이루어지는 최대 처리 지연 시간(Max Processing Delay)을 고려해 값을 설정하였다.

AC 보호구간의 설정은 스케줄링에 참여하는 노드의 수에 따라 달라질 수 있다. 각 메시지에 대한 진입 제어는 송수신자 간의 진입제어 요청(REQ) 및 응답(REP) 메시지 전송으로 이루어지는데 [4], 이러한 제어 메시지의 전송이 비동기적으로 발생하므로 동시에 같은 노드로 REQ나 REP를 전송하게 되면 스위치의 출력 포트에서 큐잉 지연이 발생하게 된다. 진입 제어 절차는 한 메시지에 대하여 하나의 트랜잭션으로 수행되어야 하고, 다음 PC의 시작 전에 반드시 완료되어야 한다. [4] 참여하는 노드의 수를 N, 한 링크에서의 REQ 메시지 전송 시간을 Δt, 큐잉 지연 없이 하나의 트랜잭션이 이루어지는데 걸리는 시간을 T, 각 링크의 전송 시간은 같고, 전파 지연과 수신 노드의 처리 지연을 무시한다면, 최대 처리 지연 시간은 T+N*Δt 가 된다.

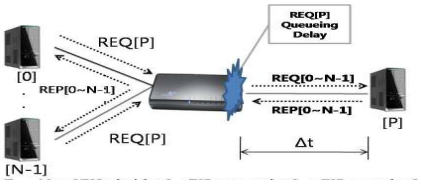


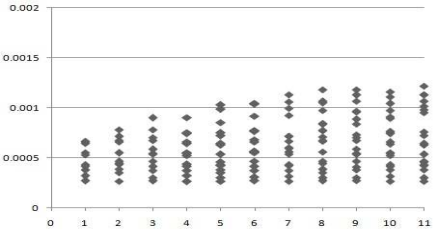
그림 6. AC 구간 제어 메시지 큐잉 지연

보호구간의 시간은 시스템에 따라 다를 수 있다. 그러므로 이론적인 계산 값과 실제 시스템에서 실험적으로 추출된 값을 토대로 각 보호구간을 설정하였다. 구현에 사용된 시스템에서 Timer의 콜백 함수를 호출하기 까지 (50, 80)us가 걸렸으므로, PC 보호구간은 80 + 120 = 200us로 설정하였다. 총 4개의 노드가 실험에 사용되므로, AC 보호구간은 request, reply 프레임의 메시지 길이(64byte)와 스위칭 지연 시간, 각 노드의 네트워크 전송 함수에서 발생하는 오버헤드를 고려하였고, 실험을 통해 최대 발생 지연을 측정하여 230us로 보호구간을 설정하였다.

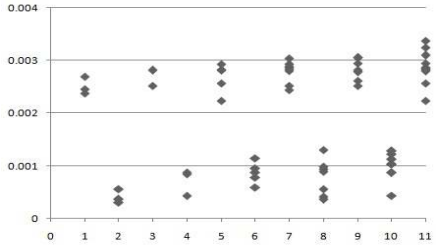
스케줄링 알고리즘은 스케줄링된 메시지들이 전송 되고 있는 가운데에도 AC구간에서 진입제어 과정을 통해 동적으로 새로운 주기적 메시지들을 전송할 수 있도록 설계 되었다. 그러므로 노드의 전송이 시작되면 첫 EC 구간에서는 현재 스케줄링된 메시지가 없기 때문에 아무런 메시지도 전송되지 않는다. 첫 AC 구간에서 진입제어에 성공한 메시지들이 해당 주기에 맞게 다음 EC부터 전송되기 시작한다.

실험에서는 총 두 번의 MC 동안 스케줄링 된 메시지들의 응답시간과 네트워크 이용률(Network Utilization)을 측정하였다. 스케줄된 메시지들의 새로운 인스턴스는 각 메시지들의 주기 구간 시작에서 생성된다고 가정하였다. 응답 시간은 메시지의 인스턴스가 생성된 시점에서 전송이 완료되는 시점까지의 시간을 의미한다.

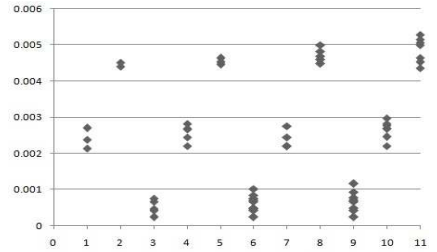
다음 [그림 7]은 스케줄 된 각 메시지들의 응답시간을 보여주고 있다. [그림 7]에서 스케줄링 된 모든 메시지들이 자신의 마감시간 안에 모두 전송되고 있음을 확인 할 수 있다.



(a) 주기 1인 메시지의 응답시간 측정



(b) 주기 2인 메시지의 응답시간 측정



(c) 주기 3인 메시지의 응답시간 측정

그림 7. 메시지 응답시간

스케줄링 알고리즘의 네트워크 이용률은 메시지 진입 제어에 따른 값의 차이를 보이기 위해 2개의 MC 동안 측정된 값을 사용하였다. 본 실험에서 생성한 30개의 모든 메시지는 첫 MC에서 모두 진입 제어를 수행하였다. 이용률은 다음과 같은 식으로 정의 된다. 여기서 N은 노드의 수, M은 매크로 사이클의 수, AM은 비주기사이클의 크기를 나타낸다.

$$U = \frac{\sum \text{time of all the message transmitted except AM}}{M * PC / N}$$

본 실험에서 사용한 메시지들의 네트워크 이용률은 첫 번째 MC에서 0.33, 두 번째 MC에서 0.65로 측정되었다. 첫 번째 MC의 EC0에서는 아직 메시지의 진입 제어가 수행되지 않은 상태이기 때문에 메시지를 전송하지 않았고, 또 진입제어를 통해 점차적으로 각 PC 구간에서 전송되어야 될 메시지들이 정의 되므로, 두 번째 MC에서 보다 좋은 이용률을 보여주었다. 네트워크 이용률은 메시지의 길이와 해당 메시지가 진입 제어를 수행하여 스케줄 되는 순서에 따라 다를 수 있다.

Ⅴ. 결론 및 향후 계획

본 논문에서는 이더넷을 통하여 서로 다른 종류의 필드버스를 통합하는 프로토콜 게이트웨이를 설계, 구현하였다. 통합필드버스 브리지 구현을 위해 실시간 기능을 가지 리눅스를 이용하였고, 실험결과 각 메시지들이 주어진 마감시간내에 전송이 완료됨을 알 수 있었다. 향후 차량 OS를 기반으로 한 차량 내부 네트워크 에 대한 통합 게이트웨이를 구현할 수 있도록 연구를 수행할 것이다.

참고 문헌

[1] 박장환, “필드버스 입문”, 도서출판 동서 2000.
[2] Bosch, "CAN Specification", 1991.
[3] Softing AG, "Profibus API", 2004.
[4] 김명균, 이희찬 “스위칭 이더넷에서 주기적 메시지에 대한 경성 실시간 통신을 위한 메시지 스케줄링 알고리즘”, 2006.9.