논문 2007-44SC-4-13

네트워크 기반 제어를 위한 LonWorks/IP 라우터의 설계 및 구현

(Design and Implementation of LonWorks/IP Router for Network-based Control)

현 진 욱*, 최 기 상**, 최 기 흥***

(Jin Wook Hyun, Gi Sang Choi, and Gi Heung Choi)

요 약

산업계에서 디바이스 제어 네트워크에 접속을 위한 기술과 인터넷을 통한 빌딩 자동화 시스템에 접속을 위한 기술의 수요가 커지고 있다. 이러한 기술에서는 디바이스 제어 네트워크와 인터넷 같은 데이터 네트워크의 통합과 광역 분산제어 시스템으로 조직화하는 것이 필요하며, 이는 VDN(virtual device network) 이라는 구조 하에서 실현될 수 있다 [1,2]. 디바이스 제어 네트워크와 데이터 네트워크의 내역은 그 응용의 차이 때문에 아주 다르다. 따라서 VDN의 구현을 위하여는 디바이스 제어 네트워크와 데이터 네트워크 간에 통신 프로토콜을 번역하고, 목적지로 정보를 효과적으로 송신할 수 있는 라우터가 필요하다. 이 논문은 VDN에 기반한 NCS(networked control system)의 개념을 제안하고 임베디드 시스템[3]을 이용하는 라우팅 알고리듬을 제안한다.

Abstract

Demand for the technology for access to device control network in industry and for access to building automation system via internet is on the increase. In such technology integration of a device control network with a data network such as internet and organizing wide-ranging DCS(distributed control system) is needed, and it can be realized in the framework of VDN(virtual device network)[1,2]. Specifications for device control network and data network are quite different because of the differences in application. So a router that translates the communication protocol between device control network and data network, and efficiently transmits information to destination is needed for implementation of the VDN, This paper proposes the concept of NCS(networked control system) based on VDN(virtual device network) and suggests the routing algorithm that uses embedded system.[3]

Keywords : VDN(virtual device network), DCS(distributed control system), NCS(networked control system)

I. 서 론

언제든지 장소와 관계없이 네트워크에 접속할 수 있는 유비쿼터스 시대가 도래하면서 산업현장과 빌딩의 전자기계장비뿐만 아니라 가정의 가전제품까지 네트워크에 연결되어 사용자가 원격으로 제어 및 관리, 감시할 수 있는 기술에 대한 수요가 생성되고 있다.[4] 그래서 디바이스 제어 네트워크는 기존의 폐쇄적인 기술 기반에서 보다 지능적이고 개방적인 기술 기반으로 즉, 기존 IT 인프라와 통합되는 방향으로 진화하고 있다. 이를 위해서는 산업현장 기기들의 정보교환에 사용되는 디바이스 제어 네트워크와 인터넷에서 사용되는 데이터 네트워크의 통합이 필요하고 이렇게 두 네트워크가 통합된 형태를 가상 디바이스 네트워크(virtual device network)

* 정회원, LG이노텍 연구원
(Research & Develpment Engineer, LG Innotek)
** 정회원, 서울시립대학교 전자전기컴퓨터공학부
(Dept. of Electrical & Computer Engg.., Univ. of Seoul)
*** 정회원, 한성대학교 기계시스템공학과
(Dept. of Mechanical Systems Engg., Hansung University)
접수일자: 2007년4월27일, 수정완료일: 2007년6월4일

라 한다.[1] 가상 디바이스 네트워크로의 통합에 있어 중요한 문제는 디바이스 제어 네트워크와 데이터 네트워크에서 요구되는 사양이 상당부분 다르다는 것이다. 그래서 두 네트워크 간의 데이터의 프로토콜을 변환하고 통신 패킷을 원격지까지 전송할 수 있는 라우터가 필요하다.

본 연구에서는 가상 디바이스 네트워크를 구성하기 위해 반드시 필요한 라우터의 개념과 틀을 제안한다.

Ⅱ. 가상 디바이스 네트워크

가상 디바이스 네트워크(virtual device network)는 산업 현장이나 빌딩 자동화 시스템(BAS, building automation system) 등에서 기기들 간의 정보교환에 사용되는 필드 버스 네트워크, 혹은 디바이스 제어 네트워크와 비즈니스 네트워크를 위한 인터넷에서 사용되는 데이터 네트워크가 통합된 네트워크이다.[1] 디바이스 제어 네트워크의 사양은 데이터 네트워크의 사양과는 차이가 있다. 데이터 네트워크는 대량의 정보를 고속 전송하는 데 초점이 맞추어져 있지만 디바이스 네트워크는 분산 제어, 신뢰성 있는 통신, 데이터의 실시간 전송, 간편한 시스템 구성과 설치 등과 같은 산업 현장에서 적합한 응용에 초점이 맞추어져 있다.[5] 이러한 차이점 때문에 현재 인터넷의 발달에도 불구하고 디바이스 제어 네트워크가 존재한다. 하지만 디바이스 제어 네트워크의 특성상 거리의 제한이 있고 사용자가 쉽게 접근할 수 없기 때문에 서로 다른 네트워크들로 이루어진 광범위한 분산 제어 시스템 ,즉 가상 디바이스 네트워크를 구현하기 위해서는 거리상의 제약이 없고 어디서나 접근이 가능한 데이터 네트워크와 통합해야 한다.[6]

TCP/IP를 포함하는 인터넷 프로토콜 집합인 IP 네트워크는 기업을 위한 통합 네트워크이다. IP 네트워크를 통하여 서로 떨어져 있는 디바이스 제어 네트워크를 연결함으로써 많은 지역에 퍼져 있는 디바이스 제어 네트워크들이 이음새 없는 가상 디바이스 네트워크로 간단히 통합될 수 있다. 다시 말해 분리되어 있는 디바이스 제어 네트워크들이 데이터 네트워크를 매개로 하여 정보를 교환함으로써 가상적으로 하나의 네트워크처럼 구성되어 마치 바로 옆에 있는 것처럼 관리되는 네트워크(peer-to-peer network)를 말한다.

그림 1에 가상 디바이스 네트워크의 일반적인 구조를 나타내었다.[1] 가상 디바이스 네트워크의 핵심은 피어 투 피어(peer-to-peer) 네트워크라는 것이다.[7] 이러한

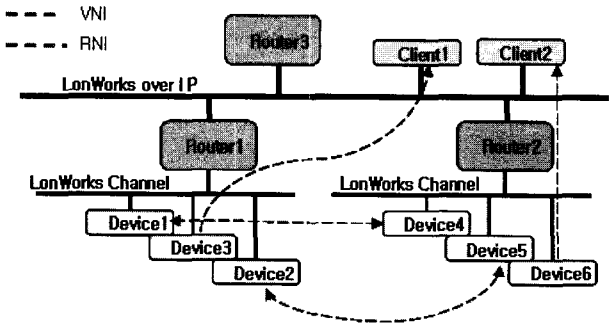


그림 1. 가상 디바이스 네트워크의 일반적인 구조
Fig. 1. General structure of virtual device network.

가상 디바이스 네트워크에서의 정보의 전달은 다음과 같은 세 가지 형태로 분류될 수 있다.

첫째, 서로 분리된 LonWorks 디바이스 제어 네트워크의 디바이스들이 라우터를 통해 서로 정보를 주고받는 것이다. 예를 들면 그림 1에서 LonWorks 디바이스 제어 네트워크의 '디바이스 1'의 정보가 '라우터 1'을 지나서 IP 네트워크를 통해서 '라우터 2'에 전달되고, '라우터 2'에 연결된 LonWorks 디바이스 제어 네트워크의 '디바이스 4'로 전달되는 것이다. 반대 방향의 정보전달도 같은 과정으로 수행되며 이러한 인터페이스 개념을 VNI(virtual network interface)라 정의한다.

둘째, LonWorks 디바이스 제어 네트워크의 임의의 디바이스와 데이터 네트워크의 임의의 클라이언트사이의 라우터를 경유한 정보의 전달이다. 이때 클라이언트는 웹기반 ActiveX Control이나 PC 응용 프로그램이 될 수 있다. 이러한 인터페이스 개념을 RNI(remote network interface)라 정의한다.

셋째, 데이터 네트워크상의 임의의 클라이언트와 라우터들 간의 정보교환이다. 이것은 서로 분리된 LonWorks 디바이스 제어 네트워크를 라우터와 연결하여 가상 디바이스 네트워크에서 동작하도록 초기화하는데 필요한 정보를 교환할 때 이용한다.

Ⅲ. 가상 디바이스 네트워크 운영과 관리

데이터 네트워크를 통하여 원격지의 LonWorks 디바이스 제어 네트워크를 운영하고 관리하는 도구의 하나로 마이크로소프트사의 윈도우 호환 LNS(lonworks network services) 응용 프로그램을 들 수 있다. 이것은 에셜론사에서 제공하는 LNS 응용 프로그램 개발 도구에 의해 개발되어 인터넷에 연결된 인터넷 클라이언트 PC에 설치된다. LNS 응용 프로그램은 가상 디바이스

네트워크를 구성하고 네트워크를 구성하는 개별 디바이스들 간의 정보 교환을 가능하게 하며 가상 디바이스 네트워크를 운영하고 관리하는데 이용된다. 이를 통하여 근거리는 물론 원격지에서 개별 디바이스를 제어하고 감시할 수 있다. 이와 같은 방법으로 가상 디바이스 네트워크를 운영하고 관리하기 위해서는 디바이스 네트워크와 데이터 네트워크를 연결하고 있는 라우터가 필요하다.

이 라우터는 EIA/CEA-852 표준을 따르며, 터널링(tunneling)기법을 사용한다. 터널링 기법은 IP 프로토콜의 데이터 패킷 내부의 데이터 영역에 LonWorks 디바이스 제어 네트워크의 데이터 패킷을 암호화하여 실어 보내는 것이다. IP 네트워크의 터널을 통해 LonWorks 디바이스 제어 네트워크의 데이터 패킷, 좀 더 엄밀하게 말하자면 주소정보를 포함하고 있는 네트워크 계층의 메시지(explicit 메시지)와 이를 128 비트로 암호화한 정보를 함께 전송하고 수신한 쪽에서는 이것을 해독하여 하위의 디바이스 제어 네트워크의 목적지로 전달하는 것을 말한다. 이러한 방법으로 가상 디바이스 네트워크의 라우터는 안전하게 디바이스 제어 네트워크의 정보를 전송할 수 있는 연결 통로가 된다. EIA/CEA-852표준에 의한 라우터를 이용하여 구현되는 가상 디바이스 네트워크는 LNS 응용 프로그램을 통하여 네트워크의 구성에서부터 운영, 관리 및 감시 제어가 수행된다.

TCP/IP를 통한 웹을 기반으로 한 원격지의 LonWorks 디바이스를 제어하기 위한 가능한 해결책으로 자바[8], ActiveX, CGI, ISAPI, 마이크로소프트 윈도우 기반 응용 프로그램을 꼽을 수가 있는데 그 중 ActiveX 기술을 이용한 LonWorks 디바이스 제어 네트워크에의 접속은 쉽게 구현될 수 있는 가시적 응용시스템을 위한 하나의 해결방안이 될 수 있다. ActiveX 기술을 이용하면 편리하고 인상적인 웹 화면을 통해서 산업현장의 제어 및 감시 관련 정보를 그래프와 같은 직감적으로 인지할 수 있는 방법으로 감시할 수 있다. 이를 위해서 먼저 LNS 응용 프로그램을 이용하여 라우터 하위의 LonWorks 디바이스 제어 네트워크를 미리 구성한다. 이러한 상태에서 라우터에 대한 접근 권한을 획득한 사용자가 웹 브라우저 화면을 통하여 라우터의 웹 서버에 접근하여 다른 라우터 서버와의 연결을 설정해준다. 가상 디바이스 네트워크를 구성하는 라우터의 웹 서버는 자신이 속한 지역의 디바이스 제어 네트워크에 대한 정보를 웹 브라우저 화면을 통해 사용자에게 제공한다. 사용자는 제공된 정보를 토대로 개별 디바이스들끼리의 연결을 지정하거나 해제시킨다. 이러한 설정이 되면 라우터는 사용자의 지시에 따라 하위 디바이스 제어 네트워크의 개별 디바이스들이 일대일로 정보를 교환하도록 하며 교환되는 정보를 사용자의 웹 브라우저 화면에 제공한다. 이때 사용자는 웹 브라우저 화면상에서 서버에서 제공한 정보에 대한 제어 및 감시를 하게 된다. 이 방법은 지역 디바이스 제어 네트워크를 미리 LNS 응용 프로그램으로 구성해야 하고 그 구성 정보를 이용해야 하는 단점이 있지만, 웹 브라우저를 통하여 어느 환경에서도 라우터의 웹 서버를 통하여 디바이스 제어 네트워크에 대한 통합 관리, 제어 및 감시가 가능하다는 장점이 있다.

가상 디바이스 네트워크에 관한 운영 및 관리를 위해 마이크로소프트 윈도우 기반 응용 프로그램을 이용할 수도 있다. 이 방법은 사용자가 클라이언트로서 소켓을 이용하여 라우터의 웹 서버에 접근하여 라우터들 간의 연결을 설정한다. 그리고 서버로부터 하위의 로컬 디바이스 제어 네트워크에 대한 정보를 바탕으로 서로 분리되어 있는 로컬 디바이스 제어 네트워크의 개별 디바이스끼리 정보를 교환 할 수 있도록 가상 디바이스 네트워크를 구성하고 가상 디바이스 네트워크를 관리하고 감시하는 방법이다.

이러한 세 가지 경우 모두 LNS 응용 프로그램을 필요로 한다. 디바이스 제어 네트워크와 데이터 네트워크의 공통노드가 두 네트워크에 대한 게이트웨이 기능과 LonWorks 라우팅 기능 및 웹 서버 기능을 갖춘다면 사용자는 라우터를 이용하여 가상 디바이스 네트워크를 좀 더 다양한 방법으로 손쉽게 운영하고 관리 할 수 있다.

Ⅳ. LonWorks/IP 라우터

1. LonWorks/IP 라우터의 개요

전체적인 LonWorks/IP 라우터 시스템의 기능은 서로 분리된 디바이스 제어 네트워크를 연결하여 가상 디바이스 네트워크 환경을 만들어 가상으로 연결된 디바이스 제어 네트워크 사이의 통신이다. LonWorks/IP 라우터의 하드웨어적인 구성은 데이터 네트워크 부분을 담당할 마스터 프로세서와 디바이스 제어 네트워크 부분을 담당할 슬레이브 프로세서로 나뉜다. 마스터 프로세서는 인텔사의 PXA250을 사용하였고, 슬레이브 프로세서는 에쉘론사의 뉴런 프로세서 FT3150을 사용하였다. 그림 2에는 위에서 설명한 마스터 부분과 슬레이브 부분

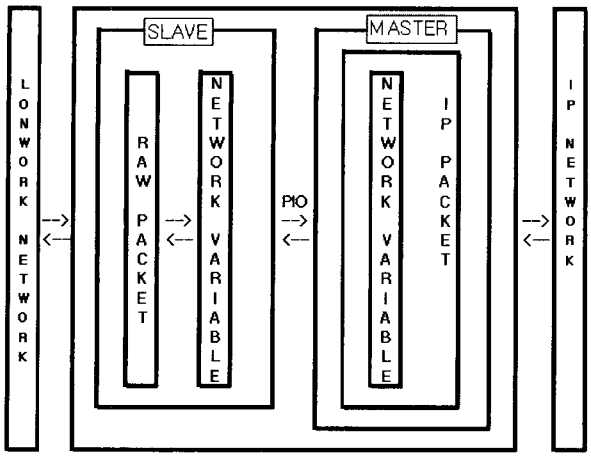


그림 2. LonWorks/IP 라우터 시스템 블록도
Fig. 2. Blockdiagram of the LonWorks/IP router system.

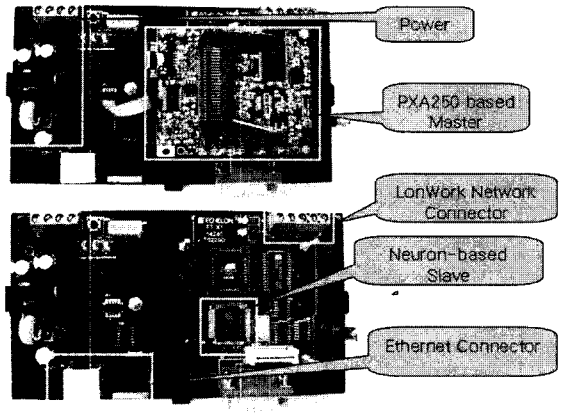


그림 3. 개발된 라우터
Fig. 3. The developed router.

의 구성을 볼 수 있다.

슬래이브 프로세서는 그림 3.2에서처럼 24개의 아날로그 입출력, 32개의 디지털 입출력, 총 56개의 LonWorks 네트워크 인터페이스를 가지며 LonWorks 네트워크의 로 패킷(raw packet)을 네트워크 변수(network variable)의 형태의 데이터로 변환하여 마스터 프로세서로 전송하거나 마스터 프로세서에서 네트워크 변수의 데이터를 받아 로 패킷으로 변환하는 역할을 한다. 마스터 프로세서와 슬래이브 프로세서가 통신하는 방식은 슬래이브 뉴런 프로세서의 펌웨어 라이브러리에서 제공하는 핸드셰이크 프로토콜(handshake protocol)의 병렬 I/O 통신 방식을 사용한다.

마스터 프로세서는 슬래이브 프로세서로부터 네트워크 변수의 데이터를 읽어 IP 패킷으로 변환하여 IP 네트워크를 통하여 목적지의 라우터로 보내거나 IP 네트워크에서 받은 IP 패킷으로부터 네트워크 변수의 데이터를 추출하여 슬래이브 프로세서로 전송한다. 슬래이브 프로세서로부터 얻은 패킷을 어떤 라우터로 보낼지는 라우팅 테이블에 기반을 두어 결정하게 되며 좀 더 빠른 라우팅을 위해 라우팅 알고리즘이 수행되게 된다. 또한, 라우터는 웹 기반 설정을 위해 웹 서버로서의 기능도 수행하게 된다. 마스터 프로세서의 소프트웨어는 이러한 여러 일들을 한꺼번에 효율적으로 수행하기 위해 멀티 쓰레드 환경이 요구된다. 이를 위해서는 운영체제가 필요하며 본 논문에서는 마이크로소프트사의 운영체제인 Windows CE.NET을 선택하였다. Windows CE는 멀티 쓰레드 환경을 제공하며, 라우터에 필요한 웹 서버도 간단히 구현할 수 있도록 라이브러리가 제공된다. 그림 3은 LonWorks/IP 라우터를 실제로 구현한 사진이다.

2. LonWorks/IP 라우터 계층 간의 프로토콜

LonWorks/IP 라우터는 그림 4 에서와 같이 CNL(component network layer), INL(intermediate network layer), IPL(IP network layer)의 3개의 계층으로 나누어 구현하였으며, CNL 계층은 마스터 프로세서와 슬래이브 프로세서를 모두 포함한다.

마스터 프로세서의 소프트웨어에서 각 계층 간 데이터의 이동은 계층 간 공유된 데이터 저장 공간을 통하여 이루어진다. 다른 계층으로 데이터를 전달할 때는 데이터 저장 공간에 데이터를 넣고, 이 데이터를 받는 계층에서는 데이터 저장 공간에서 데이터를 꺼내서 처리를 하면 된다. 하지만 공유된 데이터이기 때문에 여러 계층에서 동시에 데이터 저장 공간에 접근 하는 경우가 생기게 되며 데이터가 깨질 수 있다. 그래서 안전한 데이터 처리를 위해 동기화 기법이 요구된다. 동기화란 복수의 쓰레드가 보조를 맞추어 실행하도록 함으로써 경쟁 상

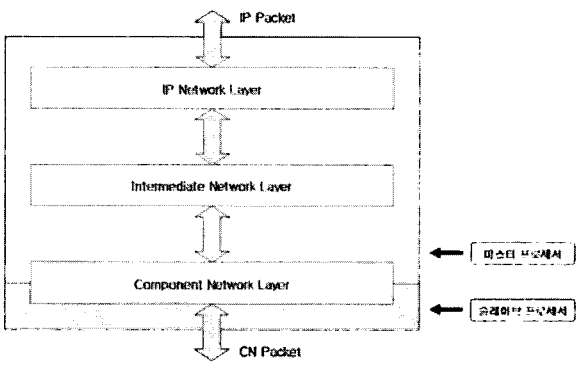


그림 4. LonWorks/IP 라우터 프로토콜 스택
Fig. 4. Protocol stack of the LonWorks/IP router.

태(race condition)나 교착상태(deadlock)를 해소하는 것이다. 라우터의 마스터 프로세서는 하나의 프로세스로 구현되었으므로 동기화를 위해 크리티컬 섹션을 선택하였다. 크리티컬 섹션은 공유 자원에의 접근과 같은 다른 쓰레드에 의해 방해받지 말아야 할 작업을 할 때 이 영역을 크리티컬 섹션으로 둘러싸 주어 공유 자원의 독점권을 보장해주는 것이다.

구체적으로 3개 계층의 동작은 다음과 같다.

1. CN(component network) 계층의 동작

CN 계층에서 슬레이브 프로세서는 디바이스 제어 네트워크 패킷을 분석해서 마스터 프로세서가 필요로 하는 네트워크 변수 데이터를 얻고, 마스터 프로세서에 병렬 I/O 인터페이스 방식을 통해서 전달한다. 이 데이터를 받은 마스터 프로세서는 CNL PDU(protocol data unit)를 만들고, 이 데이터를 저장한다.

1) CNL PDU

그림 5는 CN 계층에서 쓰는 PDU인 CNL PDU를 보여준다. 헤더 부분은 1 바이트로 데이터 부분의 길이를 알려주며, 데이터 필드의 길이는 최대 256 바이트가 된다. 데이터 필드에는 네트워크 변수 데이터가 최대 256 바이트까지 여러 개가 실리게 된다. 네트워크 변수 데이터는 아날로그 타입은 8 바이트, 디지털 타입은 4 바이트이다.

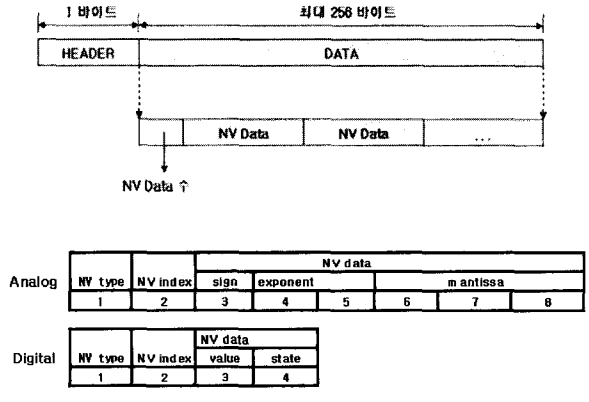


그림 5. CNL PDU
Fig. 5. CNL PDU.

2) 병렬 I/O 인터페이스

마스터와 슬레이브 프로세서 사이의 통신 방법으로 병렬 I/O 인터페이스 방식을 선택하였다. 물리적인 핀 배치는 그림 3.5에 나타내었으며 병렬 I/O 인터페이스는 11 개의 I/O 핀을 이용하며, 8bit의 데이터 라인, 칩 셀렉트 핀, 읽기/쓰기 핀, 그리고 나머지 하나는 통신할 때 동기를 맞추기 위한 핸드셰이킹 핀으로 사용된다. 마스터 프로세서와 슬레이브 프로세서는 데이터 라인을 점유하는 토큰을 주고받으며 핸드셰이킹 프로토콜에 기반을 두어 통신을 한다.[9]

2. IN(intermediate network) 계층의 동작

IN 계층은 CN 계층과 IP 계층 사이에서 CNL PDU를 IPL PDU로 만들거나, IPL PDU를 CNL PDU로 만드는 일을 한다. 즉, 데이터 네트워크 패킷과 디바이스 제어 네트워크 패킷을 서로 변환하여 주는 일을 수행한다. 네트워크 패킷 변환 과정에는 터널링(tunneling) 기법과 MD5를 통한 인증 방법이 사용된다.

1) 터널링

라우터의 기본적인 동작은 서로 연결되어 있지 않은 디바이스 제어 네트워크 사이의 패킷의 교환이다. 분리된 네트워크 사이의 통신을 위해서는 두 네트워크를 연결해 주는 채널이 필요하게 되며 이 역할을 데이터 네트워크가 맡게 되는 것이다. 이러한 개념으로 터널링은 데이터 네트워크를 터널로 삼아 디바이스 제어 네트워크 패킷을 다른 네트워크와 주고받는 기법으로 EIA/CEA-852의 표준을 따른다. 이를 위해 IN 계층에선 디바이스 제어 네트워크 패킷인 CN 패킷을 데이터 네트워크 패킷인 IP 패킷으로 변환하여 주거나, IP 패킷을 CN 패킷으로 변환하여 주는 일을 하게 된다.

2) 인증 방법

라우터가 IPL PDU를 주고받을 때는 IETF RFC 1321에 명시된 MD5를 이용한 인증(authentication)기법을 사용한다. 간단히 설명하면 암호화된 인증키를 주고받으며 유효한 인증키가 사용되었을 경우에만 인증을 하는 것이다. IPL PDU를 보면 헤더와 데이터를 제외하고 128 비트의 공간과 공유키 공간이 있는데 인증키로 128 비트의 공간을 채우게 된다. 먼저, 보내기 위한 IP 패킷을 만들기 위해서는 인증키를 위한 128 비트를 '0'으로 초기화하고 헤더, 데이터, 128 비트, 암호 키의 연속된 데이터로 MD5 알고리즘을 수행한다. 알고리즘의 결과는 128 비트의 MD5 Digest값이고 이 값을 '0'으로 초기화 되었던 공간에 넣어서 IP 패킷을 완성하게 된다. 이 패킷을 받은 쪽에서도 마찬가지로 받은 데이터의 128 비트를 다시 '0'으로 초기화하고 보낼 때와 같은 방법으로 MD5 알고리즘을 수행한다. 여기서 나온 MD5 Digest값과 받

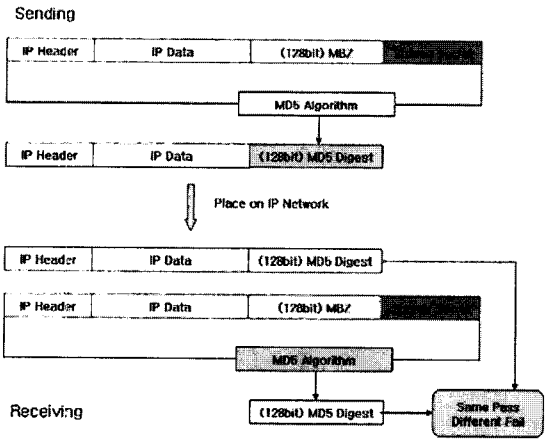


그림 6. MD5 암호화를 통한 데이터 인증
Fig. 6. Data encryption through MD5 authentication.

은 MD5 Digest를 비교하여 같다면 유효한 데이터로 인증을 하는 것이다. 그러므로 암호키가 공유되지 않은 라우터 간에는 패킷을 주고받을 수 없다. 이러한 인증 방법은 인증 이외에도 데이터 에러를 인식할 수 있는 방법이 되기도 한다. 이러한 과정을 그림 6에 나타내었다.[10]

3. IP(IP network) 계층의 동작

IP 계층의 기본 동작은 다른 라우터 또는 클라이언트와의 데이터 네트워크를 통한IPL PDU의 교환이다. 그리고 라우터와 통신할 때는 RNI가 구성된다.

1) IPL PDU

그림 7은 IP 계층에서 쓰는 PDU인 IPL PDU를 보여준다.[10]

20 바이트의 헤더에는 데이터 영역의 크기를 나타내는 필드를 포함하여 패킷을 표현하는데 필요한 정보들이 위치하게 된다. 데이터 영역에는 CN PDU가 들어가게 되는데, CN PDU의 크기에 따라 달라진다. MBZ(must be zero)로 예약된 128 비트의 공간에는 인증키가 위치하게 되고, 공유키 필드에는 라우터 간 통신

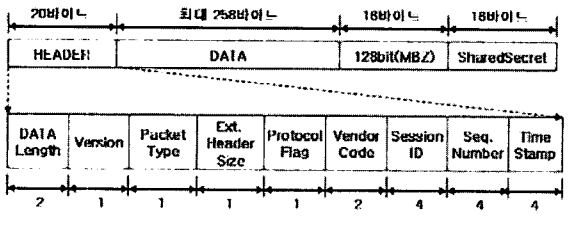


그림 7. IPL PDU
Fig. 7. IPL PDU.

을 인증을 위한 공유키가 위치하게 된다. 인증키를 만들 때는 이 IPL PDU가 쓰이게 된다.

2) IPL 블록도

IP 계층에선 데이터 네트워크를 이용하여 다른 라우터나 클라이언트와 TCP 소켓 통신을 하며 폴링 방식으로 동작한다. 다시 말해 데이터를 받는 블록과 데이터를 주는 블록이 번갈아 가면서 호출되다 조건을 만족하면 해당 동작을 수행하게 된다. 데이터를 받는 블록에서는 데이터를 받으면 INL에 메시지를 보내 데이터를 처리할 수 있도록 한다. 데이터를 주는 블록의 블록 도는 그림 8 에 나타내었다.

소켓 통신을 통해서 받은 데이터는 캐릭터형의 데이터의 연속이므로 이 데이터를 IPL PDU 용으로 타입 변환을 해주어야 한다. 패킷의 타입에 따라 수행할 블록이 다르므로 PDU의 패킷 타입 필드를 검사해서 적절한 처리를 해준다. 대부분의 패킷이 VNI 데이터이므로 VNI 타입을 제일 먼저 검사한다. VNI 데이터는 CNL로 보내야할 데이터이므로 IN 계층에서 처리하도록 큐에 패킷을 저장하고 IN 계층에 이벤트를 발생시켜 패킷을 처리하도록 한다. 패킷 타입이 RNI 요청인 경우는 웹 클라이언트나 윈도우 기반 클라이언트 프로그램이 라우터 하위 디바이스 제어 네트워크의 내용을 감시하기 위해 요청한 패킷이다. 이 패킷을 보낸 클라이언트는 CNL PDU를 원하기 때문에 라우터는 CNL PDU를 클라이언트들에게 보내기 위해 바인딩 리스트에 해당 클라이언트를 추가시켜 이후 발생하는 CNL PDU를 클라이언트가 받을 수 있도록 한다. 패킷 타입이 바인딩 리스트 요청인 패킷은 이 패킷은 네트워크를 구성하는 윈도우 기반 응

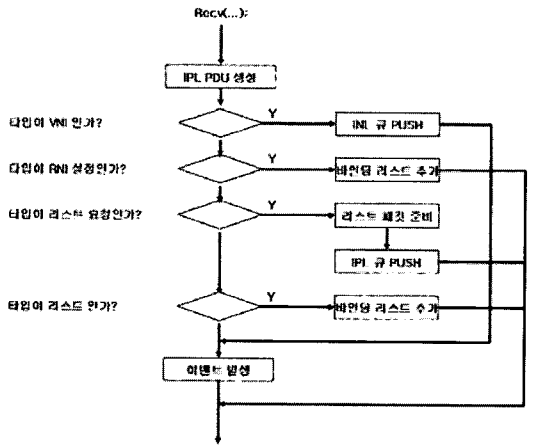


그림 8. IPL 수신측 블록도
Fig. 8. Blockdiagram of the IPL receiver.

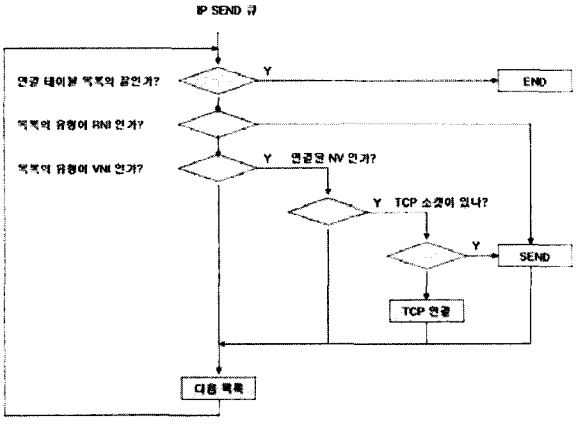


그림 9. IPL 송신측 블록도
Fig. 9. Blockdiagram of the IPL transmitter.

용 프로그램이나 웹 클라이언트에서 보내온 패킷으로 현재 구성된 가상 디바이스 네트워크의 구성 정보를 해당 클라이언트에 보내준다. 패킷 타입이 바인딩 리스트인 패킷은 네트워크를 구성하는 윈도우 기반 응용 프로그램이나 웹 클라이언트에서 보내온 패킷으로 가상 디바이스 네트워크에서 디바이스 제어 네트워크의 네트워크 변수 사이의 가상 연결 설정을 위한 패킷이다. 라우터는 이 패킷을 기반으로 가상 네트워크 변수의 연결을 설정하거나 해제 한다. IN 계층으로부터 받은 데이터를 다른 라우터 또는 클라이언트에 보내는 일을 하는 IP 계층의 송신부 블록도를 그림 9에 나타내었다.

패킷을 보내기 위해서 큐에서 IPL PDU를 읽으며 보낼 때는 연결 리스트의 설정을 따른다. 이 연결 리스트가 라우팅 테이블이 되는 것이다. 송신의 기본 동작은 IPL PDU를 원하는 곳으로 보내는 것인데, 하나의 IPL PDU는 한곳으로만 전달되는 것이 아니므로 연결 리스트 전체를 검색하면서 리스트의 설정과 조건이 맞을 때마다 패킷을 전달하게 된다. 위의 그림에서 RNI로 설정된 클라이언트와는 별다른 조건문이 없는데, RNI 설정은 클라이언트가 연결 요청을 하면 연결 리스트에 추가가 되고 클라이언트와의 연결이 끊어지면 연결 리스트에서도 제거되기 때문이다. 그러나 VNI로 설정된 클라이언트는 클라이언트가 존재 할 수도 있고 없을 수도 있기 때문에 존재 유무 검사가 필요하다.

3. 라우팅 알고리즘

가상 디바이스 네트워크에서의 라우터의 역할은 서로 분리되어 있지만 가상으로 연결된 디바이스 제어 네트워크 사이의 CNL PDU의 교환이다. 라우터의 CN 계층에서 만들어지는 CNL PDU의 주 데이터는 네트워크

SOURCE NV	SOURCE IP	TARGET NV	TARGET IP	TYPE
SOURCE NV	SOURCE IP	TARGET NV	TARGET IP	TYPE
SOURCE NV	SOURCE IP	TARGET NV	TARGET IP	TYPE
...	...	...	...	...
SOURCE NV	SOURCE IP	TARGET NV	TARGET IP	TYPE

그림 10. 연결 테이블 구조
Fig. 10. Structure of the mapping table.

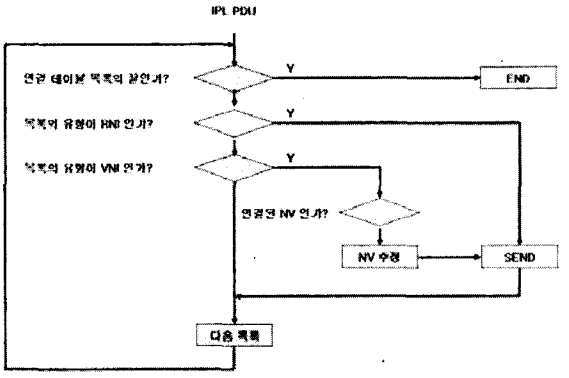


그림 11. 라우팅 과정 블록도
Fig. 11. Blockdiagram of the routing process.

변수(network variable)라 불리는 LonWorks 네트워크에서 돌아다니는 일종의 변수이며, 라우터의 기본적인 역할은 이러한 네트워크 변수를 하나의 라우터에 속한 LonWorks 네트워크와 다른 라우터에 속한 LonWorks 네트워크와 주고받을 수 있도록 하는 것이다. 그러므로 라우터는 네트워크 변수 데이터에 기반을 두어 라우팅을 수행하게 된다. 라우팅을 하기 위해서는 가상으로 연결된 네트워크 간 네트워크 변수의 가상 연결 상태 정보가 필요하며, 이 정보는 라우터의 웹서버나 윈도우 기반 응용 프로그램을 통해 설정이 가능하다. 가상 연결을 위해서는 연결되는 라우터의 IP 주소와 LonWorks 네트워크 디바이스들의 네트워크 변수 정보가 필요하다.

라우터로 연결된 가상 디바이스 네트워크는 네트워크 변수 기반 가상 연결이기 때문에 라우팅을 수행하기 위해서는 네트워크 변수만 검사를 하면 된다. 네트워크 변수 기반 라우팅 과정을 그림 11에 나타내었다. 하나의 네트워크 변수가 두개 이상의 가상 네트워크 변수 연결을 가질 수 있기 때문에 하나의 CNL PDU에 대하여 연결 테이블을 처음부터 끝까지 검색할 필요가 있다.

VNI로 연결된 네트워크 변수의 경우에는 네트워크 변수의 번호를 타겟의 네트워크 변수 번호로 바꿔서 보냄으로서 패킷을 받은 쪽에서 네트워크 변수 값 수정을 위해 다시 한 번 연결 리스트를 검색할 필요가 없게 한다.

V. 라우터의 데이터 구조

라우터의 주요한 동작은 라우터 내부에서 각 계층 간에 공유된 데이터 저장 공간을 읽고 쓰는 것이기 때문에 그로 인해 발생하는 복사 시간에 따라 라우터의 성능이 좌우된다. 디바이스 제어 네트워크와 데이터 네트워크로의 변환 과정에서 발생한 여러 계층 간의 빈번한 데이터 복사는 자칫 라우터의 성능을 저하 시킬 수 있기 때문에 실시간으로 제어되는 네트워크 기반 제어 시스템에서 데이터 구조의 중요성은 더욱 강조된다. 데이터 저장 방식은 큐(queue)형태의 동적 데이터 구조와 배열 형태의 정적 데이터 구조 방식이 있다.

1. 큐(queue)

큐는 Fist-Input/First-Output방식의 데이터 구조로 기존의 라우터에서 취하는 데이터 구조이며 일종의 데이터의 버퍼역할을 할 수 다. 데이터가 추가 될 때마다 새로 메모리를 할당하여 큐의 리스트에 추가시키므로 메모리 용량이 정적인 배열 구조의 데이터 형식보다 유연성이 있지만 데이터를 큐에 저장하거나 꺼낼 때마다 데이터를 복사해야 한다. 즉, 네트워크로부터 실시간으로 읽혀지는 데이터를 라우터가 허용하는 메모리 공간까지 계속 저장할 수 있지만 그로 인해 지연시간이 발생할 수 있다. 그림 3.14의 블록도를 보면 라우터 내부의 계층 간에 공유되는 큐 형태의 데이터 구조를 알 수 있다.

그림 12에서처럼 슬레이브 프로세서와 마스터 프로세스간의 큐, CN 계층과 IL 계층 간의 큐, IL 계층 과 IP 계층 간의 큐, IP 계층과 IP 네트워크간의 큐 등 3 종류의 큐가 필요하며 데이터를 받는 계층 간의 이동과 데이터를 보내는 계층 간의 이동에서의 큐는 따로 관리 되어야 하므로 두 배의 큐, 즉 총 6 개의 큐가 필요하게 된다. 쓰레드가 큐에 접근하면 크리티컬 섹션에 의해 다른 쓰레드의 프로세스는 중단되기 때문에 큐에 의한 빈번한 데이터 복사는 서로 다른 쓰레드의 프로세스에 영향을 주어 전체적인 프로세스의 성능을 저하시키게 되고 그것은 다른 계층 간의 데이터 이동 속도를 떨어뜨리게 되는 등 성능 저하의 악순환이 반복된다. 이런 식으로 데이터가 큐에 계속 누적되어 라우터로 인해 발생되는 지연시간이 점점 길어진다면 실시간 네트워크 기반 제어 시스템의 의미가 퇴색될 수 있다. 그래서 큐를 다루는 프로그램 최적화에 세심한 주의가 필요하다.

2. 배열 구조

큐 형태의 데이터 구조의 개선안으로 배열 형태의 데이터 구조를 제안한다. 여기서 말하는 배열 구조란 큐와는 동일한 기능을 하면서 메모리 특성이 정적인 배열 구조와는 좀 다르다. 그것은 슬래이브의 뉴런 프로세서 네트워크 인터페이스의 특성에 따른 배열 구조이다. 슬래이브의 네트워크 인터페이스는 아날로그 입력 12 개와 출력 12 개, 디지털 입력과 16 개와 출력 16 개의 네트워크 변수로 이루어져 있기 때문에 그것을 기반으로 큐와는 다른 데이터 구조를 만들어 낼 수 있다. 그 데이터 구조의 기본은 마스터 프로세서에서 각각의 네트워크 인터페이스만큼 CNL PDU를 저장할 수 있는 배열를 할당하는 것이다. 그리고 각 배열의 요소마다 갱신되었음을 표시하는 플래그 형태의 변수도 할당하여 병렬 I/O 를 통해 들어온 네트워크 변수의 정보를 갱신하고 갱신되었다는 것을 플래그 변수에 표시하고 데이터를 사용했

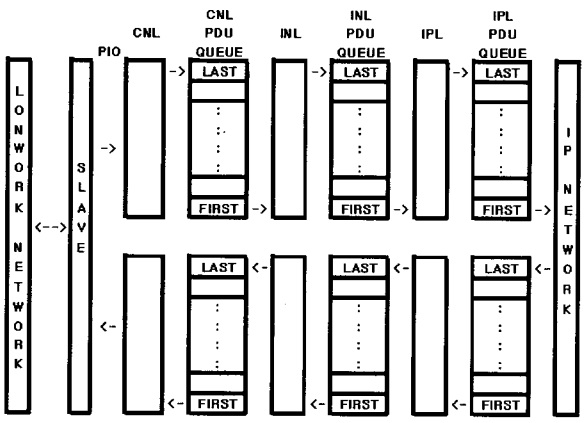


그림 12. 큐를 이용한 데이터의 이동
Fig. 12. Data transfer using queue.

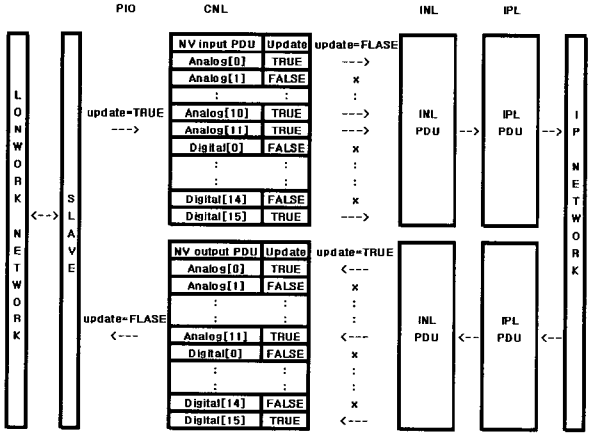


그림 13. 배열 구조의 데이터 이동
Fig. 13. Data transfer of array.

을 경우 갱신되었다는 표시를 삭제한다. 이것의 단점은 갱신된 데이터를 라우터의 상위 계층에서 읽어 들이지 못한 상태에서 새로 갱신된 데이터는 이전 데이터 위에 덮어씌우기 때문에 데이터의 손실이 발생한다는 것이다. 그 구조는 그림 13과 같다.

이 구조는 큐 형태의 데이터 구조에서 라우터의 지연 시간으로 인해 큐의 리스트에 데이터가 과다하게 누적되어 동일한 시작점, 동일한 목적지이지만 시간은 다른 데이터들이 큐의 리스트 안에 공존할 때 지연 시간이 생기더라도 모든 데이터를 보존해야 하는지 아니면 최신의 데이터만 유효하도록 하는 지에 대한 선택에 따라 그 정당성이 확보된다. 하지만 전자의 경우는 실시간 제어가 이루어지지 않을 확률이 높기 때문에 실시간 네트워크 제어라는 전제에서는 후자를 선택하기 쉽다. 후자의 경우에 지연 시간의 감소로 인하여 디바이스 제어 네트워크에서 네트워크 변수가 갱신되는 평균 속도이상의 성능이 나온다면 무리 없이 쓸 수 있다. 네트워크 변수 데이터들이 아날로그 데이터이거나 단순한 스위치 입출력의 디지털 정보이기 때문에 마스터 프로세서에서 갱신된 데이터를 읽지 못하고 새로운 데이터로 갱신되어 데이터 손실이 발생하는 경우가 약간 발생하더라도 문제가 없다. 데이터가 특정 메시지를 갖는다면 데이터의 손실이 문제가 될 수 있다. 그런 경우 큐와 배열의 구조를 적절하게 조합한 방법이 가능하다. 배열구조에서 각각의 배열의 요소가 큐의 형태를 취하는 방법이다. 물론 여기서 큐가 데이터를 누적할 있는 공간은 작은 수로 제한되어 있어 어느 정도의 데이터가 누적되면 가장 오래된 데이터를 버리는 방식을 취하여 데이터 누적으로 인한 지연시간을 줄인다.

VI. 결 론

본 연구에서는 가상 디바이스 네트워크의 필요성을 설명하고 가상 디바이스 네트워크 기반 제어 시스템을 구현하기 위하여 임베디드 시스템을 이용한 라우터의 개념과 알고리즘을 제안하였다. 또한 실제로 LonWorks/IP 라우터를 구현하였다. LonWorks/IP 라우터와 IP 네트워크를 통해 원격의 제어 네트워크들을 하나의 네트워크로 통합하는 목적을 달성하기 위해서는 차후에 실험실이 아닌 실제 현장에서 큰 규모의 가상 디바이스 네트워크를 구성하여 네트워크 기반 제어에 관한 실험이 필요하다.

참 고 문 헌

[1] G. H. Choi, G. S. Choi, and J. S. Kim, "Lonworks Based Virtual Device Network(VDN) for Predictive Maintenance", ICMIT'01, Yamaguchi Seminar Park, Japan, pp. 372-376, 2001.

[2] Feng-Li Lian, James Moyne, and Dawn Tilbury, "Network Design Consideration for Distributed Control Systems", IEEE, pp 297-307, March 2002..

[3] Wei Zhang, M. S. Branicky, and S. M. Phillips, "Stability of Networked Control Systems", IEEE Control Systems Magazine, pp. 84-99, February 2001.

[4] Byoung-Hee Kim, Kwang-Hyun Cho, and Kyoung-Sup Park, "Toward LonWorks technology and its applications to automation", pp197-202, Science and Technology 2000.

[5] Echelon, Introduction to the LonWorks system, 078-0183-01A.

[6] H. Shahnasser and Q. Wang, "Controlling Industrial Devices over TCP/IP by Using LonWorks," Proc. IEEE, pp. 1309-1312, 1998.

[7] O. Vahamaki, A. Allen and J. Gaff, "High Speed Peer-to-Peer Communication System for Integrated Protection and Control in Distribution Network", Development in Power System Protection, IEE, pp.243-246, 1997.

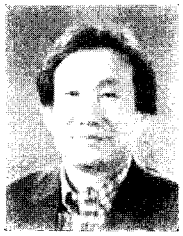
[8] H.Reiter, C. Kral, "Interaction Between JAVA and LonWorks," Proc. IEEE, pp. 335-339, 1997.

[9] Echelon, Engineering Bulletin, 1995.

[10] ANSI/EIA/CEA-852-2002.

저 자 소 개

현 진 욱(정회원)
2005년 2월 서울시립대학교 전자전기컴퓨터공학부(공학사)
2007년 2월 서울시립대학교 전자전기컴퓨터공학부 (공학석사)
2007년 2월~현재 LG이노텍 연구원
<주관심분야: Mechatronics, Distributed Control>



최 기 흥(정회원)
1982년 2월 한양대학교 기계공학과 (공학사)
1990년 5월 University of California, Berkeley (Ph.D, Mechanical Engineering)
1995년 3월~현재 한성대학교 기계시스템공학과 교수
<주관심분야: Safety Engineering, Mechatronics, Distributed Control>

최 기 상(정회원)
1982년 2월 서울대학교 기계공학과 (공학사)
1990년 5월 University of California, Berkeley (Ph.D, Mechanical Engineering)
1991년 4월~현재 서울시립대학교 전자전기컴퓨터공학부 교수
<주관심분야: Mechatronics, Distributed Control, Musical Acoustics>