工學碩士 學位論文

마이크로컨트롤러 制御를 위한 LonWorks 通信 모듈의 設計 및 具現

Design and Implementation of LonWorks Communication Module for Microcontroller Control

指導敎授 金 基 文

2003年 2月

韓國海洋大學校 大學院

電子通信工學科

崔 栽 赫

工學碩士 學位論文

마이크로컨트롤러 制御를 위한 LonWorks 通信 모듈의 設計 및 具現

Design and Implementation of LonWorks Communication Module for Present Micro-Controller Control

指導敎授 金 基 文

2003年 2月

韓國海洋大學校 大學院

電子通信工學科

崔 栽 赫

목 차

표 목차

그림 목차

Abstract

In this paper, we implemented universal controller for distributed intelligent control system using LonWorks technology of Echelon co. and developed LON system.

LON is a important element in low-level control network that issues in the automatic as well as in economic aspect.

It can change large control system to small by using distributed control technique with intelligent device. Developed system is based on network communication technique using LonWorks technology. LonWorks communication module has the advantage that field bus is connected without protocol modification.

LonWorks communication module can be divided hardware module and software module. First, hardware module is divided into microcontroller attaching sensors and hardware communication module for working together control network and data network. Second, software module is realized with neuron C.

Hardware module is making use of neuron chip(TMPN3120FE3M), microcontroller(PIC16F873), transceiver(FTT-10A), Lon card(PCLTA-20 PCI LonTalk adapter). And operating software is realized with neuron C using NodeBulider 3.0 development tool.

Produced and implemented LonWorks communication module is pretested using LTM-10A, Gizmo 4 I/O board, parallel I/O Interface. For field test, microcontroller module part is tested by hyper-terminal, communication procedure in data network is certified by transmitting and receiving short message using LonMaker for Windows tool.

This thesis is composed of five chapters. Chapter 1 describes the historical background, goals of LonWorks and outline of remaining chapters. Chapter 2 describes the concepts of LonWorks. Chapter 3 describes design and implementation of LonWorks communication module for microcontroller control. Chapter 4 describes the experimental result of LonWorks communication module and consideration to the result. Chapter 5 summarizes all results obtained in this thesis and includes the further reseach topics and the work to be supplemented.

제 1 장 서 론

1.1 연구의 배경 및 목적

현대 산업사회에서 제어 시스템은 다량의 정보처리와 고속의 정보통신 및 통합적인 정보 관리가 요구되고 있다. 선진국에서 개발되고 있는 최근의 분산제어 및 자동화 시스템에서는 컴퓨터를 이용하여 분산된 공정을 자동화하고 있다. 따라서 이들을 다시 수직, 수평적으로 통합하여 전체 공정을 일관되게 관리함으로써 생산성을 향상시키고, 비용을 절감시키는 동시에 자동화 공정의 설계, 구축 및 유지관리에 유연성과 신뢰도를 극대화시키는 효과를 거두고 있다[1]~[3].

이러한 첨단 자동화 시스템은 궁극적으로 모든 공정을 통합하는 것으로 미래의 분산제어 및 자동화 시스템에 있어서 네트워킹 기술이 핵심기술로 부각되고 있다. 네트워킹 기술을 바탕으로 각 제어기들은 공정에 맞는 독자적인 운용 프로그램을 가지며 고도로 분산화, 독립화 되어 운영된다. 또한 공정제어에 있어서는 분산제어 기법이 발달함에 따라 공정제어 시스템이 거대화되어 가고 있고 개방형 통신망 구조가 제안된 이후, 여러 회사에서 개발된 고성능의 전력선 통신(PLC : Power Line Communication), 프로세서 컨트롤러(processor controller) 등을 기존의 독립적인 구성에서 벗어나 전체적으로 결합시켜 하나의 종합적인 시스템으로 구축하려는 노력이 이루어지고 있다. 과거의 각종 필드장치는 50년대에는 유공압을 이용하여 제어하였으며, 80년대까지는 주로 4 - 20 mA의 변조신호를 사용하는 쌍꼬임선(TP : Twisted Pair Line)에 의해 일대일(1:1)로 연결되어 왔다[1], [5].

이러한 연결 방식은 전자파 노이즈에 덜 민감하고 오류의 감지가 쉬워서 제어 시스템에 많이 사용되긴 하지만, 길게 연결되는 수많은 전선 다발, 노이즈, 열, 설치비용, 관리 문제 그리고 시스템 업그레이드(upgrade)가 어

려운 문제점이 있다. 이러한 이유로 필요한 각 제어 기기간의 통신, 데이터의 교환 등의 문제를 해결하기 위해 표준 통신망을 각 나라에서 제안했는데 그 대표적인 예가 필드버스(Field(생산현장) + Bus(통신))이다. 필드버스는 분산제어 또는 자동화용 디지털 직렬통신망으로서, 자동제어 분야에서 통신을 이용한 제어방식을 가능케 하는 중요한 기반 기술이다.

이것은 각 제어기의 고속통신을 이용한 정보교환을 통하여 유기적인 결합을 이루고 멀티입력/멀티출력의 복잡한 시스템의 제어를 간단한 구성으로서 가능하게 한다. 특히 이는 국제 표준화 기구(ISO : International Standardization Organization)의 개방형 시스템 상호접속(OSI : Open System Interconnection) 참조 모델을 기초로 하고 있다[1-3, 5].

따라서 본 논문에서는 여러 가지 필드버스 중에서도 개방형 통신 프로토콜(protocol)이 우수하고 경제성이 뛰어난 산업용 통신망인 LonWorks를 이용하여 마이크로컨트롤러 제어를 위한 LonWorks 통신 모듈을 설계 및 구현하였다.

1.2 연구의 내용

LonWorks의 LON(Local Operating Network)이 LAN(Local Area Network)의 개념을 따라 구현되었듯이, 과거의 컴퓨터 산업은 마스터/슬레이브(master/slave) 방식에서 중형 컴퓨터, 워크스테이션을 구축하고, 이제는 개인 컴퓨터가 막강한 능력을 갖추어 서로 네트워크를 구성하는 클라이언트/서버(client/server) 구조로 변천되어 왔다[2].

또한 기존의 제어 시스템은 독자적이고 폐쇄적인 벤더 중심의 제어 시스템을 구성한다. 기존의 컨트롤러와 단위 센서류까지 일대일(1:1)로 구성된 전기 연결 방식에서 발전되어 디지털 명령 시스템(DCS : Digital Command System) 같은 컨트롤러로 독자적이고 폐쇄적인 서브시스템을 구축하고, 각각의 서브시스템은 단순한 프로토콜 변형을 위한 게이트웨

이(gateway)를 사용하던 방식으로 변화되었으나, 이제는 각 단위 디바이스(device)에 지식(intelligence)을 부여하고 플랫(flat, peer-to-peer) 구조로 제어 네트워크를 구성해 가고 있다[2], [3].

따라서 디바이스와 시스템 통합 작업에 공통의 프로토콜을 사용함으로써 시스템이 개방적이고 많은 벤더(vendor)의 참여와 서로 다른 프로토콜의 변형 없이 쉽게 연결될 수 있는 필드버스 장점을 이용하여 제어 네트워크와 데이터 네트워크 연동을 위한 LonWorks 통신 모듈 구현에 대해 연구하였는데 LonWorks는 아래와 같은 장점을 가진다.

디지털 데이터 전송으로 강한 내잡음성 제공, 통신을 이용한 멀티타겟(multi-target)/멀티제어(multi-control) 제공, 시스템의 신뢰도, 유연성, 확장성 증대, 스마트 센서(smart sensor)를 사용함으로써 센서 레벨까지 분산화 제공, 양방향 통신을 통하여 필드 기기 모니터링 및 제어 기능 제공, 시스템 유지 보수비용 절감 등을 가능하게 한다.

논문의 구성은 다음과 같다. 제 2 장에서는 필드버스의 한 종류로서 실시간 제어 시스템 구현에 이용된 LonWorks의 개념을 설명하고, 제 3 장에서는 실제적인 마이크로컨트롤러 제어를 위한 LonWorks 통신 모듈의 설계 및 구현에 대하여 기술한다. 즉 LonWorks 통신 모듈의 하드웨어 제작, 그리고 하드웨어로 제작된 LonWorks 통신 모듈의 운용을 위한 펌웨어(firmware) 구현에 대해서 논한다. 제 4 장에서는 실제 제작된 LonWorks 통신 모듈의 실험 및 고찰, 그리고 제 5 장에서는 본 연구의 결론과 향후 연구방향을 제시하였다.

제 2 장 LonWorks의 개념

2.1 LonWorks 시스템의 개요

LonWorks는 애쉘론사(Echelon)가 창안한 제어용 네트워크 시스템이다. LonWorks는 제어나 감시등 센서를 사용하는 모든 부문에 적용할 것을 전제로 설계되었다[2], [3], [6], [8]~[14].

먼저 LonWorks 네트워크를 데이터 네트워크의 대표적인 예인 LAN과 비교하면 다음과 같다. LAN과 LON은 모두 네트워크이지만 사용 목적이 다르다. LAN은 컴퓨터, 프린터, 파일서버 등의 사이에서 문서나 이미지 등 대량의 데이터 처리를 목적으로 한다. 이에 반하여 LON은 제어를 위한 피드백(feedback) 정보나 각 노드(node)간의 명령이나 상태 등 소량의 데이터를 교환하여 네트워크 전체 또는 그룹단위로 고도로 조직화된 제어를 목적으로 한다. 표 2-1은 LonWorks와 LAN의 비교 내용을 나타내고 있다.

표 2-1. 데이터 네트워크와 제어 네트워크의 비교
Table 2-1. Comparison of Data Network and Control Network

구 분	데이터 네트워크(LAN)	제어 네트워크(LON)
정보의 양	많다	적다
전송 빈도	불규칙	규칙
지연 시간	비실시간	실시간
사용 목적	컴퓨터, 프린터 등 문서나 대용량의 데이터 처리	제어를 위한 피드백 정보, 노드간의 명령이나 상태 등의 최적화된 데이터

LonWorks는 빌딩 자동화(BA : Building Automation) 분야에서는 사실상의 표준(de factor standard)으로 인정받고 있으며, 공장 자동화(FA : Factory Automation), 홈 오토메이션(HA : Home Automation) 등 넓은 응용 범위를 가진다. 작은 시스템으로는 복사기나 자판기와 같은 복잡한 기능을 갖는 기계류에서부터 엘리베이터 제어, 경비행기의 운항 제어, 공항의 수화물 처리 시스템 제어 등 광범위한 영역에서 사용되고 있다. 특히 최근에는 환경, 에너지 관련분야, 교통 및 철도, 차량 등 많은 응용분야를 확대해 가고 있다[2].

LonWorks/LonTalk는 BACnet(빌딩제어)의 하부 LAN으로도 정의되어 있고, 미국전기전자학회(IEEE : Institute of Electrical and Electronics Engineers) 1473(기차/전동차 제어), 미국전자공업협회(EIA : Electronic Industry Association) 709(제어 네트워킹), SEMI(Semiconductor Equipment Manufacturing)의 표준으로 포함되어 있다. 또한 산업 디바이스 버스 리서치로 잘 알려진 ARC(Automation Research Corporation)와 VDC (Venture Development Corporation)는 LonWorks가 이미 이 분야의 마켓 리더임을 인정하고 있다. 유럽의 정유회사 컨소시엄인 IFSF(International Fore-court Standards Forum)는 유럽의 각 주유소와 정유소, 본사와의 네트워킹을 LonWorks로 구현하고 있다. 홈 오토메이션 시장을 놓고 경쟁 중인 시스코 사의 인터넷 홈 네트워크, 마이크로소프트 사의 HAPI (Home Application Programming Interface), 썬(Sun) 사의 JINI(Java Intelligent Network Infrastructure) 기술 모두 하부 디바이스용 제어 네트워크로 LonWorks를 선택하였다. LonWorks에 관한 세부사항은 다음 제 3 장에서 논한다.

2.2 LonWorks 시스템의 구성 요소

LonWorks는 필드 제어 네트워크를 구성하는데 필요한 모든 요소를 갖추고 있다는 것과 여기에는 개방형 프로토콜인 LonTalk가 있다. 이는 디바이스 개발자, 시스템 통합 관리자, 최종 사용자 모두에게 동일한 통신 규약을 갖도록 하는 장점이 있다.

이 개방형 기술에는 뉴런 칩(neuron chip), LonTalk, 트랜시버(transceiver), LNS(LonWorks Network Services) 등이 포함되어 있다. LonWorks 시스템의 구성 요소를 LAN 시스템과 비교하면 다음 표 2-2와 같다[2], [3].

표 2-2. LAN과 LonWorks의 구성 요소 비교
Table 2-2. Comparison of LAN and LonWorks Component

LonWorks	LAN
뉴런 칩(Neuron Chip)	Pentium Ⅱ
트랜시버(Transceiver)	LAN Card
LNS	Win NT
LonTalk	TCP/IP
윈도우즈용 LonMaker	MS-Excel

2.2.1 뉴런 칩(Neuron Chip)

뉴런 칩은 LonTalk 프로토콜 스택(stack)을 펌웨어(firmware)로 포함하는 마이크로 프로세서이다. 즉 뉴런 칩에는 통신 및 연산을 위한 3개의 8비트 프로세서를 탑재하고 있으며, LonTalk 프로토콜을 포함하고 있는 일반적인 용도로 응용(application) 구현이 가능한 마이크로컨트롤러이다. 모든 LonWorks 노드는 뉴런 칩 또는 미국표준협회(ANSI :

American National Standard Institute)/미국전자공업협회(EIA 709.1) 프로토콜을 수행하는 마이크로컨트롤러를 포함하여야 한다. 칩 내부의 메모리 내장 유무에 따라서 3120 계열과 3150 계열로 나누어진다.

프로세서 자체가 통신을 위한 프로토콜을 내장하고 있으므로 디바이스 개발자는 디바이스의 응용 프로그램만 작성하고 네트워크 통신을 위한 프로토콜은 일일이 구현할 필요가 없다. 또한 뉴런 칩에는 2개의 16비트 타이머/카운터, 20 mA 싱크 포트(port) 등 11개의 입·출력(I/O) 포트를 이용한 34개의 입·출력 모드를 제공한다. 또한 네트워크 통신 전용 포트를 갖추고 있어 여러 종류의 트랜시버와 다양한 모드로 인터페이스를 제공한다[2], [3], [7]. 뉴런 칩의 내부 구성도는 그림 2-1과 같다.

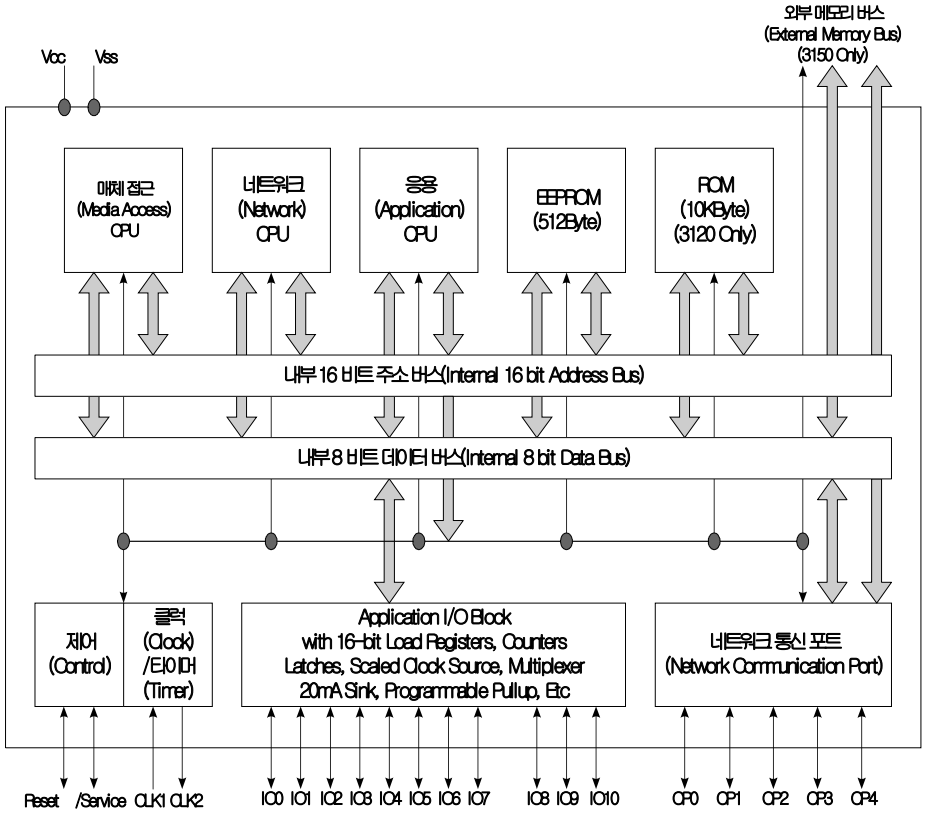

그림 2-1. 뉴런 칩의 내부 구성도

Fig. 2-1. Internal Composition of Neuron Chip

현재 뉴런 칩은 모토롤라(Motorola)와 도시바(Toshiba)가 생산하여 판매하고 있다. 또한 최근에는 사이프레스(Cypress)사에서도 뉴런 칩을 생산 공급할 것을 발표하였다. 아래의 표 2-3은 도시바 사에서 생산되는 뉴런 칩의 사양을 나타내고 있다.

표 2-3. 뉴런 칩의 사양
Table 2-3. Specification of Neuron Chip

Device	TMPN3120					TMPN 3150
	BIAM	EIM	A20/M	FE3M	FE5M	
Max, Clock	10MHz	10MHz	20MHz	20MHz	20MHz	10MHz
EEPROM	0.5KB	1KB	1KB	2KB	3KB	0.5KB
ROM	10KB	10KB	16KB	16KB	16KB	None
RAM	1KB	1KB	1KB	2KB	4KB	2KB
Package	32 pin SOP	32 pin SOP	44PIN QFP32/ pinSOP	32 pin SOP	32 pin SOP	64 pin QFP
Firmware Ver.	4	6	10	9	10	N/A
On-board A/D	No	No	No	No	No	Yes
Ext. Memory	No	No	No	No	No	Yes

또한 최근에 개발되고 있는 뉴런 칩은 기능이 더욱 보강되어 칩 내부에 자바(Java) 코어(core)를 내장시킨 자바론(Javalon)과 32 비트 축소명령집합컴퓨터(RISC : Reduced Instruction Set Computer) 타입의 프로세서에 이더넷(ethernet) 포트를 내장시킨 타입(일명 펜타곤(pentagon) 칩)이 발표될 예정이라고 한다. 이러한 기능들은 향후 디바이스의 강력한 응용과 인터넷을 통한 시스템의 통합 제어 및 모니터링을 가능하게 할 수 있다. 그림 2-2는 펜타곤 칩의 내부 구성도를 보이고 있다[2].

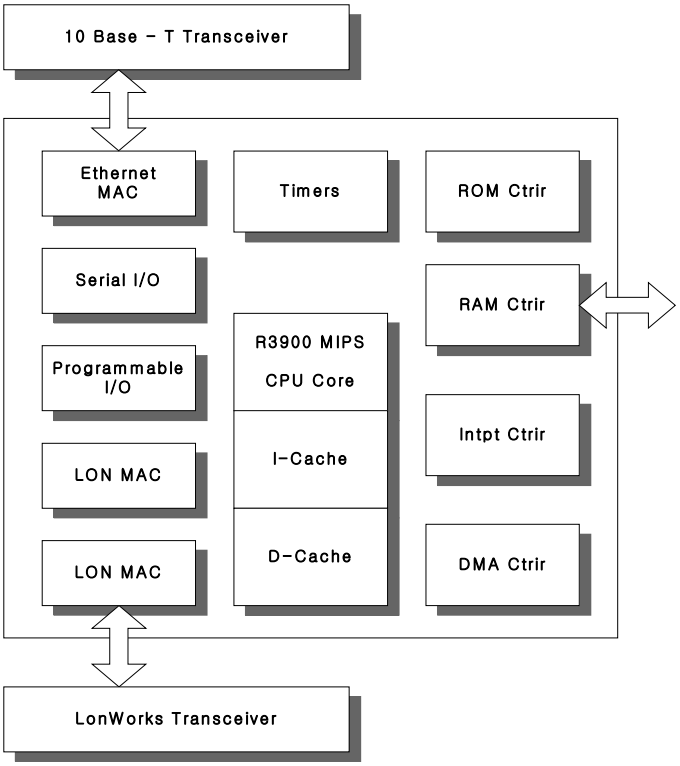

그림 2-2. 펜타곤 칩의 내부 구성도

Fig. 2-2. Internal Composition of Pentagon Chip

2.2.2 LonTalk

통신 시스템의 가장 중요한 부분은 바로 통신 프로토콜이다. LonTalk 프로토콜은 반도체 칩에 내장된 펌웨어와 하드웨어를 사용하여 OSI 모델의 7계층 모두를 가지고 있다. LonTalk의 특징은 다음과 같다[2], [3], [12], [15].

(1) LonTalk 메시지 서비스

LonTalk 프로토콜은 기본적으로 ACK TYPE과 UNACK TYPE의 메시지 서비스를 제공한다.

(가) ACK TYPE

1) ACKD(종단간 확인 : End-to-end Acknowledged) 서비스

노드 혹은 노드의 그룹에 메시지를 보내고 나서 ACK 신호를 받지 못하면 트랜잭션(transactions) 타이머가 파기되면서 메시지를 다시 보내게 된다. 데이터 전송에 있어서 얼마나 기다릴 것인가와 몇 번을 시도할 것인가를 설정하는 것이 가능하다. ACK 서비스는 응용 프로세서와는 무관하며 네트워크 프로세서에서 발생한다.

2) 요청/응답(Request/Response) 서비스

LonTalk는 중복된 메시지 검출 서비스를 제공하는데 일반적으로 메시지가 중복되면 단 한 번만 메시지를 보낸다. 그러나 명시적(explicit) 메시지를 이용하여 데이터를 보내고자할 때는 이 서비스를 이용하여 메시지를 여러 번 보낼 수 있다. 메시지는 응답을 발생시키기 전 수신측 응용 프로세서에서 처리된다. 이 서비스는 클라이언트/서버 응용 원격 프로시저 호출(RPC : Remote Procedure Call)에 적합하다.

(나) UNACK TYPE

1) 반복(비확인 반복) 서비스(Repeated(Unacknowledged Repeated) Service)

메시지를 노드 또는 노드 그룹에 여러 번 보내고 응답을 기대하지 않는다. 이 서비스는 큰 네트워크 그룹에 브로드캐스팅(broadcasting)하고 모든 노드의 응답에 의한 네트워크 부하를 방지하기 위해 사용한다.

2) 비확인(Unacknowledged) 서비스

메시지를 보내고 나서 응답을 기대하지 않는다. 높은 성능(performance)이 요구되거나 많은 양의 데이터를 전송할 때 사용한다. 자유 버퍼 대기(free-buffer wait) 타이머를 사용하며, 이 타이머는 송신 측에서 자유 버퍼를 이용하여 노드가 기다릴 수 있는 최대 시간을 결정한다. 타이머가 파기될 때까지 버퍼에 데이터를 확보하지 못하면 치명적인 에러로 간주하고 리셋된다.

(2) 네트워크 변수(Network Variable)와 명시적 메시지(Explicit Message)

LonTalk은 네트워크 변수(NV : Network Variable)와 명시적 메시지(explicit message) 같은 특징을 가지고 있다. 응용 프로그램은 네트워크 변수라고 불리는 특별한 형태의 입·출력 객체(object)를 선언할 수 있으며, 출력으로 선언된 하나의 네트워크 변수의 값을 갱신하게 되면 그 출력 네트워크 변수와 연결된 입력 네트워크 변수를 가지는 모든 노드로 해당 값이 전파되어 값이 갱신된다.

이와 같이 한 노드에서 출력 네트워크 변수를 다른 노드의 입력 네트워크 변수에 연결하여 동시 갱신이 되도록 하는 과정을 바인딩(binding) 작업이라고 한다. 네트워크 변수에 의한 통신의 장점은 응용 프로그램에서 통신에 대한 고려 부분이 전혀 없이 통신이 되는 것이며, 따라서 응용 프로그램의 변경 없이 바인딩 정보만 바꾸어 줌으로서 쉽게 네트워크 통신이 가능하다. 네트워크 변수의 크기는 객체마다 다르지만 최대 32바이트이며, 뉴런 칩을 호스트로 사용하는 노드에서는 최대 64개의 네트워크 변수를 선언할 수 있다. 아래 그림 2-3은 네트워크 변수의 선언 예를 나타낸다[2], [3], [9], [10].

```
network input/output [class] type identifer [=initial value]
network input byte nviInputData;      // 입력 네트워크 변수 선언
network output bit nvoOutData         // 출력 네트워크 변수 선언
```

그림 2-3. 네트워크 변수 선언의 예

Fig. 2-3. Example of Network Variable Declaration

네트워크 변수 중에는 LonWorks를 사용하는 기기간의 상호 호환을 위한 표준화 기구인 LonMark 상호운용 단체(LonMark Interoperability Association)에서 정한 것을 특별히 표준 네트워크 변수 타입(SNVT : Standard Network Variable Types)라 부른다.

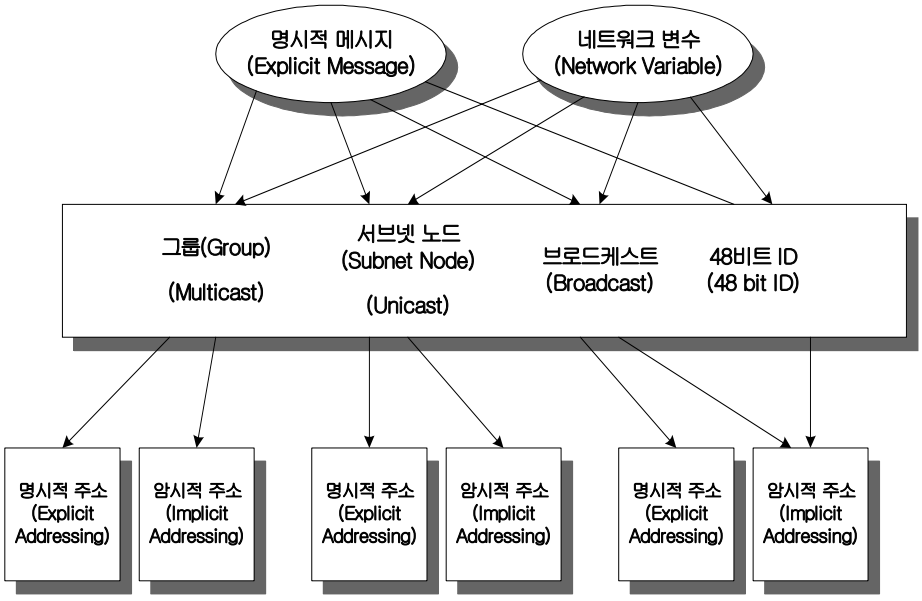

그림 2-4. LonTalk의 통신 방법

Fig. 2-4. Communication Method of LonTalk

이를 이용하여 구성된 시스템은 개방형 시스템으로서 타 LonWorks 기기와 추가적인 변경 없이 통신이 가능하다. 명시적 메시지에 의한 통신 방법은 네트워크 변수를 이용한 방법과는 달리 응용 프로그램에서 직

접 어드레싱(addressing) 정보와 데이터를 가공하여 전송하는 방법이다.

명시적 메시지에 의한 통신은 실제적으로 네트워크 관리와 응용 프로그램에서 사용된다. 결론적으로 LonTalk에서 지원하는 통신 방법은 그림 2-4와 같다.

2.2.3 LonTalk 프로토콜 계층

LonTalk는 OSI 모델의 7 계층을 모두 지원하며, 이와 같이 OSI 모델에 의거 정의되는 것은 다른 종류의 LonWorks와의 상호 운용성을 위해 중요하다. LonTalk 프로토콜은 이미 뉴런 칩에 하드웨어, 펌웨어 형태로 포함되어 있다. 따라서 디바이스 개발자는 LonTalk 프로토콜의 자세한 패킷 규격에 대해 모르고서도 응용 프로그램을 구현할 수 있다. LonTalk는 OSI 모델을 따르며, 계층 구조는 그림 2-5에 나타나 있다[2], [3], [12], [15].

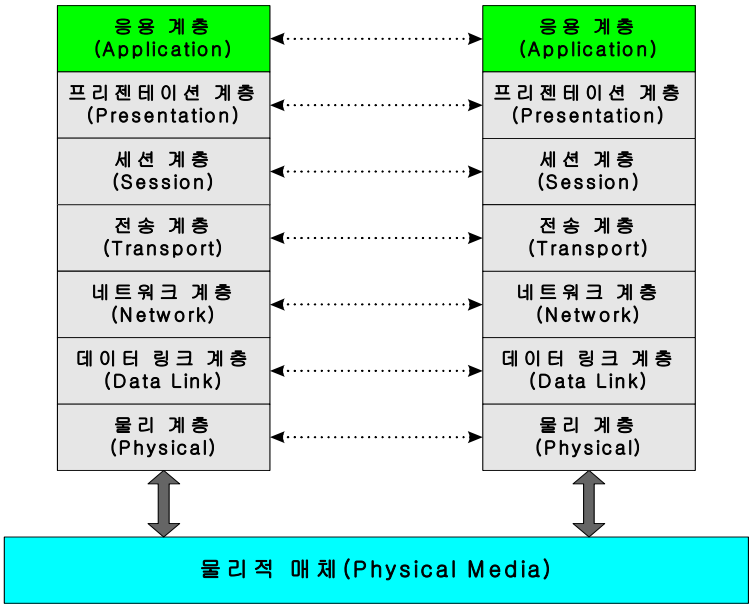

그림 2-5. OSI 7 계층 모델

Fig. 2-5. OSI 7 layers model.

(1) 물리 계층(Physical Layer)

물리 계층에서는 데이터를 인코딩 하는 방법이 매체에 따라 다르다. 예를 들면 쌍꼬임선에는 차분 맨체스터 인코eld(differential manchester encoding) 방식이 사용되고 전력선(power line)에는 FSK(Frequency Shift Keying) 변조와 modified direct sequence spread-spectrum이 사용된다. 그리고 RF(Radio Frequency)에는 FSK 변조가 사용된다.

(2) MAC 서브계층(Sublayer)

매체 액세스 제어(MAC : Media Access Control) 서브 계층에서는 충돌 회피 알고리즘인 predictive persistence 반송파 감지 다중 액세스(CSMA : Carrier Sense Multiple Access)를 사용한다.

(3) 링크 계층(Link Layer)

링크 계층에서는 멀티캐스트 프로토콜(multicast protocol)의 간소화와 호환성을 위해 비연결형(connectionless) 서비스를 지원한다. 비연결형 서비스는 프레임(frame) 인코딩이나 에러 검출에 한계가 있고 재전송에 의한 에러 복구를 지원하지 않는다.

(4) 네트워크 계층(Network Layer)

네트워크 계층은 하나의 도메인 내에서 패킷의 전송을 다룬다. 네트워크 서비스 역시 비연결형, 비확인응답(non acknowledge) 서비스를 지원하며 메시지의 분할(segmentation)이나 조합(reassemble)은 지원하지 않는다. 이 계층에서 라우팅 알고리즘을 지원한다.

(5) 트랜스포트 계층(Transport Layer)과 세션 계층(Session Layer)

트랜스포트 계층과 세션 계층은 처리순서와 중복 메시지 검출을 다룬다. 트랜스포트 계층은 비연결형 서비스이고, 하나 또는 여러 목적지 노드에 신뢰할 수 있는 메시지 전송을 제공한다. LonTalk 프로토콜의 트랜스포트 계층이나 세션 계층 메시지는 주로 인증(authenticated) 메시지를 사용한다. 세션 계층은 원격 서버를 위한 간단한 요구/응답 메카니즘을 수행한다.

(6) 표현 계층(Presentation Layer)과 응용 계층(Application Layer)

표현 계층과 응용 계층은 LonTalk 노드를 위한 상호운영을 기본적으로 제공한다. 응용 계층은 보내지거나 받을 모든 메시지를 위한 통상적인 서비스를 제공한다. 표현 계층은 응용 프로토콜 데이터 단위(APDU : Application Protocol Data Unit)가 어떻게 네트워크 변수의 갱신에 이용되는가를 알기 위한 APDU 헤더에 정보를 제공한다.

이런 응용과 무관한 데이터 해석은 여러 노드 사이에 우선 배열에 관계없이 데이터를 공유할 수 있게 한다. 표 2-4에 각 계층별 기능 및 수행 프로세서에 대해서 정리하였다.

표 2-4. 계층별 기능 및 수행 프로세서

Table 2-4. Layer Function and Achievement Processor

OSI 계층		목 적	제공 서비스	뉴런칩 프로세서
7	응 용	application compatibility	standard network variable type	응 용
6	프리젠테이션	data interpretation	network variables, foreign frame transmission	네트워크
5	세 션	remote action	request/response, authentication, network management	네트워크
4	전 송	end to end reliability	acknowledge/unacknowledged, unicast/multicast, authentication, common ordering	네트워크
3	네트워크	destination addressing	addressing, routers	네트워크
2	데이터 링크	media access and framing	framing, data encoding, CRC, checking, priority, predictive CSMA, collision avoidance	MAC
1	물 리	electrical interconnect	media-specific interface and modulation schemes	MAC Xcvr (트랜시버)

2.2.3 트랜시버(Transceiver)

LonWorks 트랜시버는 매체와의 통신을 담당하는 칩으로 볼 수 있으며, OSI 7 계층 중 1 계층인 물리계층에 해당한다. 즉 뉴런 칩이 컴퓨터의 프로세서라면 LAN 카드(NIC : Network Interface Card)로 비유될 수 있다. LonWorks의 장점 중 하나는 데이터 전송매체 선택이 자유롭다는 점에 있다. 즉 쌍꼬임선, 전력선(AC100/200), 링크 파워(link power, DC 24V 선에 데이터를 실어 보냄), 무선, 동축케이블(coaxial), 광섬유(fiber optic), 적외선(infrared) 등 다양한 매체를 지원하는 트랜시버가 여러 업체에 의해 제공되고 있다. 표 2-5는 애쉴론 사의 제품사양을 표기한 것이다.

표 2-5. LonWorks의 트랜시버 종류
Table 2-5. Transceiver Kind of LonWorks

구 분	트랜시버	Bit Rate	Topology	Distance	비 고
쌍꼬임선	FTT-10A	78 Kbps	Free-Bus, Star, Ring, Other combination	500m free topology, 2700m doubly terminated bus	transformer-isolated
	TPT/XF-1250	1.25 Mbps	Bus	130m (0.3m stub)	transformer-isolated
링크 파워	LPT-10	78 Kbps	Free-Bus, Star, Other combination	500m free topology, 2200m doubly terminated bus	transformer-isolated
전력선	PLT-22	5 Kbps	Free or Bus	Depends on transmitter-to receiver attenuation and noise seen at receiver	Dual Carrier Frequency (132 KHz / 115 KHz)

LonWorks 트랜시버의 특징은 네트워크 형상(topology) 구성에 제한이 없다하여 자유 형상(free-topology) 트랜시버라고 한다. 따라서 버스, 링, 스타 등의 형상을 자유롭게 또는 혼합하여 네트워크를 구축할 수 있다. 이는 시스템 설치자는 현장에 맞는 네트워크 구성을 구현할 수 있다.

LonWorks가 특정 분야가 아닌 여러 산업분야에 사용될 수 있었던 이유 중 하나는 이렇듯 매체 선택이 자유롭고, 매체간의 연결을 라우터(router)를 써서 간단히 구현할 수 있기 때문이다. LonWorks의 라우터는 네트워크의 확장 외에도 이기종 매체의 연결(예를 들면, 전력선을 쌍꼬임선으로, 쌍꼬임선을 광섬유로)을 위해서도 사용된다. 또한 어떤 업체들은 LonTalk을 이더넷, 모뎀(MODEM : Modulator-Demodulator), 종합정보통신망(ISDN : Integrated Service Digital Network)으로 연결시켜주는 라우터를 개발하여 판매하기도 한다.

2.2.4 LNS(LonWorks Network Service)

LonWorks가 다른 필드버스와 차별화 될 수 있는 점은 일관된 네트워크 관리 해결책(solution)까지 갖춘 제어용 네트워크라는 것이다. 1997년 말에 발표한 LNS는 LonWorks 네트워크의 운영체제(OS : Operating System) 또는 네트워크 관리 시스템(NMS : Network Management System)이라 할 수 있다.

LNS는 시스템 설치자 및 통합자가 서로 다른 벤더의 제품을 묶어서 전체 네트워크를 구성하는데 편리한 장점을 제공한다. 즉 LNS는 개인 컴퓨터를 비롯하여 다른 마이크로프로세서 컨트롤러에 구현되어 LonWorks 네트워크를 구성, 설치, 모니터링, 컨트롤할 수 있는 응용을 작성할 수 있게 한다. LNS는 클라이언트/서버 아키텍처 구조를 지원하고 특히 32비트 윈도우즈(Windows) 응용을 위하여 LonWorks 네트워크 요소에 대해 객체 계층을 구조화하여 보다 쉽게 MMI(Multi Modal Interface)/HMI(Human Man Interface)를 작성할 수 있게 한다.

LNS는 네트워크 관리 프로그램을 쉽게 구현해 준다는 장점만 가진 것은 아니다. 현재 많은 디바이스 제조업체가 LNS 응용에 탑재되는 디바이스 플러그인(plug-in)을 개발하여 디바이스의 구성 값(parameter)을 손쉽게 조회하고 수정할 수 있게 한다.

그림 2-6은 LonWorks 네트워크 구성 및 설치, 그리고 네트워크 변수(메시지), 모니터링/컨트롤을 위한 응용 프로그래밍의 기반이 되는 클라이언트/서버 아키텍처의 LNS 컴포넌트(component)를 보이고 있다. 그림 2-6에서 하위의 LonWorks 디바이스와 LonPoint에 의해 지능이 부여된 더미(dummy) 센서 및 액츄에이터(actuator)가 네트워크로 연결되고, 이들간의 바인딩 및 네트워크의 논리적 구성, 속성 설정 등의 작업을 LNS 기반 응용인 윈도우즈용 LonMaker(LonMaker for Windows)를 통해서

한다. 윈도우즈용 LonMaker는 네트워크의 논리적인 구성 상태를 각각의 LonWorks 디바이스에 커미션(commission)하며, 그런 상태 값을 데이터베이스(DB : DataBase) 형태로 보관하게 된다.

이에, MMI(HMI)는 그 네트워크 데이터베이스를 기반으로 LonWorks 네트워크 상의 디바이스 주소 및 네트워크 변수/메시지 이름을 확인(identify)하여 시스템을 모니터링/컨트롤 할 수 있게 한다.

여기에는 동적 데이터 교환(DDE : Dynamic Data Exchange) 방식으로 상위 응용을 구현할 수도 있고, ActiveX 컨트롤을 사용하여 VB(Visual Basic), VC++, 델파이(Delphi)로 쉽게 LonWorks 네트워크 모니터링/컨트롤 응용을 구현할 수 있다.

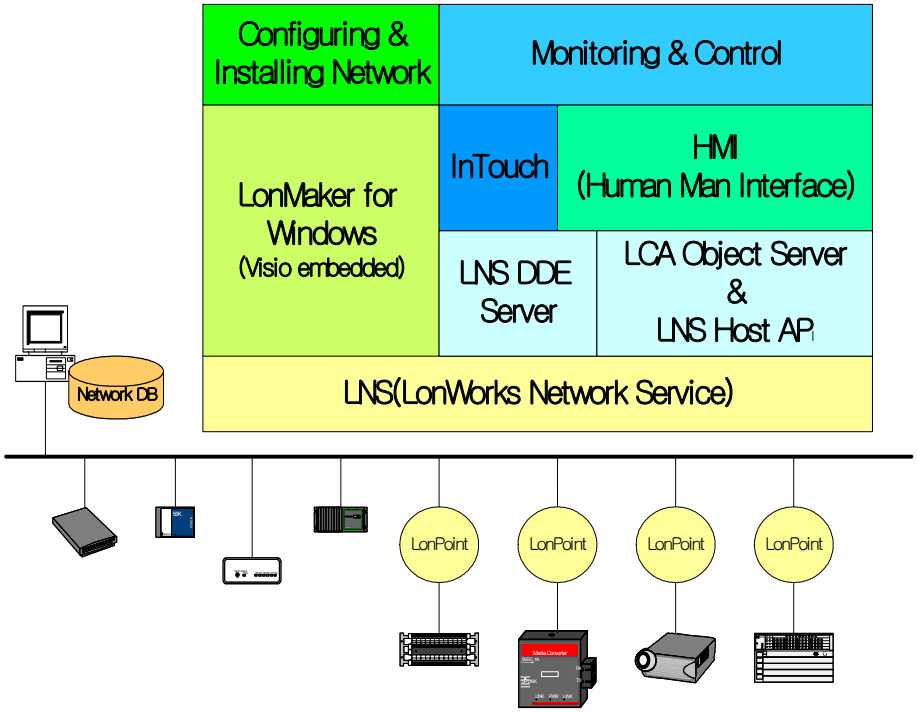

그림 2-6. LNS 컴포넌트

Fig. 2-6. LNS Component

제 3 장 LonWorks 통신 모듈의 설계 및 구현

본 장에서는 실제로 마이크로컨트롤러 제어를 위한 LonWorks 통신 모듈(디바이스)을 제작, 구현한다. 이를 위해서 표준형 론 노드(SLN : Standard LON Node)에 대해서, 이를 응용하여 실제 하드웨어 제작에 필요한 구성 요소와 제작에 관해서, 그리고 실제 하드웨어로 제작한 LonWorks 통신 모듈의 운용을 위한 펌웨어(소프트웨어) 구현에 관하여 논한다.

3.1 표준형 론 노드(SLN)

디바이스는 LON 노드의 가장 기본이 되는 구성 요소들로 이루어져 있으며, 어떤 응용을 개발하든지 공통으로 포함되는 가장 기본이 되는 회로이다. 이런 의미에서 이름을 표준형 론 노드라고 부르기도 한다. SLN은 LON 시스템에 있어서 단순한 기능을 가지는 노드 구성 시 사용되어 진다. 그림 3-1에서 보는 바와 같이 기본적으로 뉴런 칩과 펌웨어를 탑재하기 위한 메모리, 네트워크 매체 접속을 하기 위한 트랜시버로 구성되어 있다. 그림 3-1에서는 도시바사의 TMPN3150B1AF (10 MHz)를 사용하였으며, 외부 메모리의 ROM(Read Only Memory)은 EEPROM (Electrically Erasable and Programmable Read Only Memory)인 AT28C256과 62256 RAM을 사용하였다. 네트워크 매체는 쌍꼬임선을 사용하였으며, 네트워크 접속을 위해 78 Kbps를 지원하는 애쉴론 사의 TTT-10A 트랜시버를 사용하였다. 외부로 나와 있는 11개의 I/O 포트를 이용하여 34가지 모드의 I/O 기능을 구현할 수 있으며 LonTalk 프로토콜의 계층화된 주소(address) 할당을 통하여 3,200 이상의 노드에 동시에 접속할 수 있다[2], [3], [16].

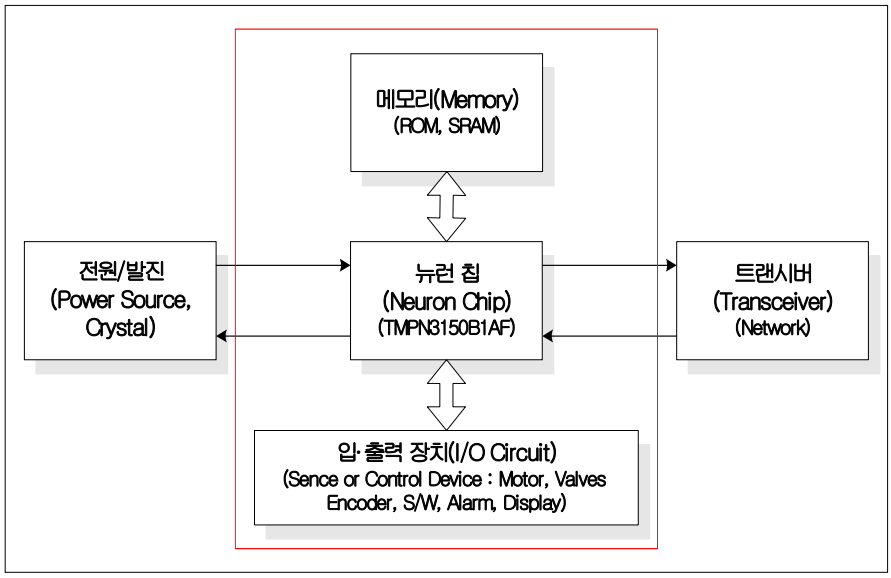

그림 3-1. 표준형 론 노드의 구조

Fig. 3-1. Structure of SLN(Standard LON Node)

3.2 LonWorks 통신 모듈 제작의 구성 요소

본 장에서는 LonWorks 통신 모듈을 제작, 구현하는데 필요한 중요 구성 요소에 대해서 논하며 표 3-1과 같다.

뉴런 칩은 내부의 메모리 내장 유무에 따라서 3120 계열과 3150 계열로 나누어진다. 뉴런 칩 자체가 통신을 위한 프로토콜을 내장하고 있으므로, 개발자는 디바이스의 응용 프로그램만 작성하고 네트워크 통신을 위한 프로토콜은 일일이 구현할 필요가 없다. 또한 네트워크 통신 전용 포트를 갖추고 있어 여러 종류의 트랜시버와 다양한 모드로 인터페이스 할 수 있다. 본 논문에서는 도시바사의 TMPN3120FE3M을 사용한다. 즉 그림 3-1의 SLN에서 3120 계열의 뉴런 칩을 사용하게 되면 외부 메모리가 없게 되므로 보다 간단한 시스템을 구현할 수 있다.

트랜시버는 다양한 전송매체를 지원하는 네트워크와 노드간의 전기적인 매체 인터페이스를 제공하기 위한 것이다.

표 3-1. LonWorks 통신 모듈 제작을 위한 구성요소
Table 3-1. Component for LonWorks Communication Module Implementation

구 분	모델(제품명)	비 고
뉴런 칩	TMPN3120FE3M	하드웨어 제작
마이크로컨트롤러	PIC16F873(PIC chip)	
트랜시버	FTT-10A	
Lon Card	PCLTA-20 PCI LonTalk 어뎁터	
개발 툴	노드빌더 3.0, LTM-10A, Gizmo 4 I/O 보드, 병렬 I/O 인터페이스	펌웨어 (소프트웨어) 구현

본 논문에서는 가장 일반적으로 널리 사용되는 애쉴론사의 쌍꼬임선용인 FTT-10A를 이용한다. 마이크로컨트롤러는 제어 포인트들의 상태 정보를 받아들여서 처리하고, 이 정보를 제어 네트워크를 통해서 정보를 전달하는 기능을 한다. 이를 위해서 마이크로칩 테크놀로지(Microchip Technology) 사의 PIC16-F873을 이용한다.

PC에서 LonWorks를 인터페이스 할 수 있는 방법은 다양하다. PCI (Peripheral Component Interconnect), ISA(Industry Standard Architecture), PCMCIA(Personal Computer Memory Card International Association) 타입의 LonTalk 어댑터(adapter)를 사용할 수도 있고, SLTA(Serial LonTalk Adapter)와 같은 제품을 사용하여 PC와 시리얼 인터페이스를 사용할 수도 있다. 또한 아이론(iLon)과 같은 제품은 PC와 LonWorks 네트워크를 IP(Internet Protocol) 상에서 접속할 수 있게 한다. 본 논문에서는 PCLTA-20 PCI LonTalk Adapter를 이용한다.

뉴런 칩 또는 스마트 트랜시버는 LonWorks 프로토콜을 내장하고 있을 뿐 아니라, 다목적 I/O, 응용 메모리 등등, 디바이스의 다양한 응용을 수용할 수 있는 다목적 프로세서이다. 이들 프로세서에 응용을 구현하기 위해서 사용하는 언어는 뉴런 C이며, 애쉴론사에서는 이 스크립트

(script)를 이용하여 LonWorks 디바이스 응용을 개발하고 테스트해 볼 수 있는 장비를 공급하고 있다. 이 제품이 노드빌더(NodeBuilder)이다.

노드빌더는 LonWorks 기반의 디바이스 개발자에게 꼭 필요한 개발 환경이다. 이전과 달리 노드빌더 3.0은 마법사(wizard) 기능이 대폭 강화되어 보다 편리하고 빠르게 제품 개발을 할 수 있다. 디바이스 개발에 필요한 컴파일러(compiler), 빌더(builder), 디버거(debug), 플러그인(plug-in) 생성기뿐만 아니라, 개발된 디바이스를 구성, 설치해볼 수 있는 LonMaker 통합툴과 간단한 모니터링/제어 테스트를 할 수 있는 LNS DDE 서버를 함께 제공하므로, 디바이스 개발자가 개발 및 테스트를 해볼 수 있는 환경을 제공한다.

또한 LTM-10A와 Gizmo 4 보드는 프로그래밍 연습 및 테스트해볼 수 있는 하드웨어 환경을 제공한다. 즉 개발자는 LTM-10A와 Gizmo 4로 타겟 하드웨어 환경을 증설해볼 수 있다.

3.3 LonWorks 통신 모듈의 구성

기존의 컨트롤러와 단위 센서류까지 전기적으로 연결하던 방식에서, DCS 같은 컨트롤러로 독자적이고 폐쇄적인 서브시스템을 구축하고, 각각의 서브시스템은 단순한 프로토콜 변형을 위한 게이트웨이를 사용하던 방식으로 변화되었으나, 현재는 각 단위 디바이스에 지식을 부여하고 플랫(flat, peer-to-peer) 구조로 제어 네트워크를 구성한다.

그림 3-2는 기존의 자동제어 아키텍처를 보이고 있다. 게이트웨이를 이용한 시스템 통합은 서로 다른 벤더의 폐쇄적이고 독자적인 시스템이 빌딩의 서브시스템에 각각 적용되었을 때 이들 서브시스템간의 상위에서 게이트웨이를 통하여 이루어질 수밖에 없다. 이러한 게이트웨이가 하는 일은 폐쇄적인 서브시스템간의 통신을 위한 것으로서 양쪽의 메시지를 변경하여 처리하도록 기능을 담고 있다. 시스템 통합 시, 이러한 게이트

웨이의 구현은 그 자체가 고가일뿐더러, 응용마다 매번 구성해 주어야 하고, 또 유지보수의 작업도 어렵다는 단점이 있다. 또한 서브시스템 별로 다른 프로토콜을 사용하기 때문에 같은 방안 또는 같은 계층에서 일어나는 작업도 매번 게이트웨이를 거쳐 제어되어야 하므로 네트워크의 효율도 떨어진다.

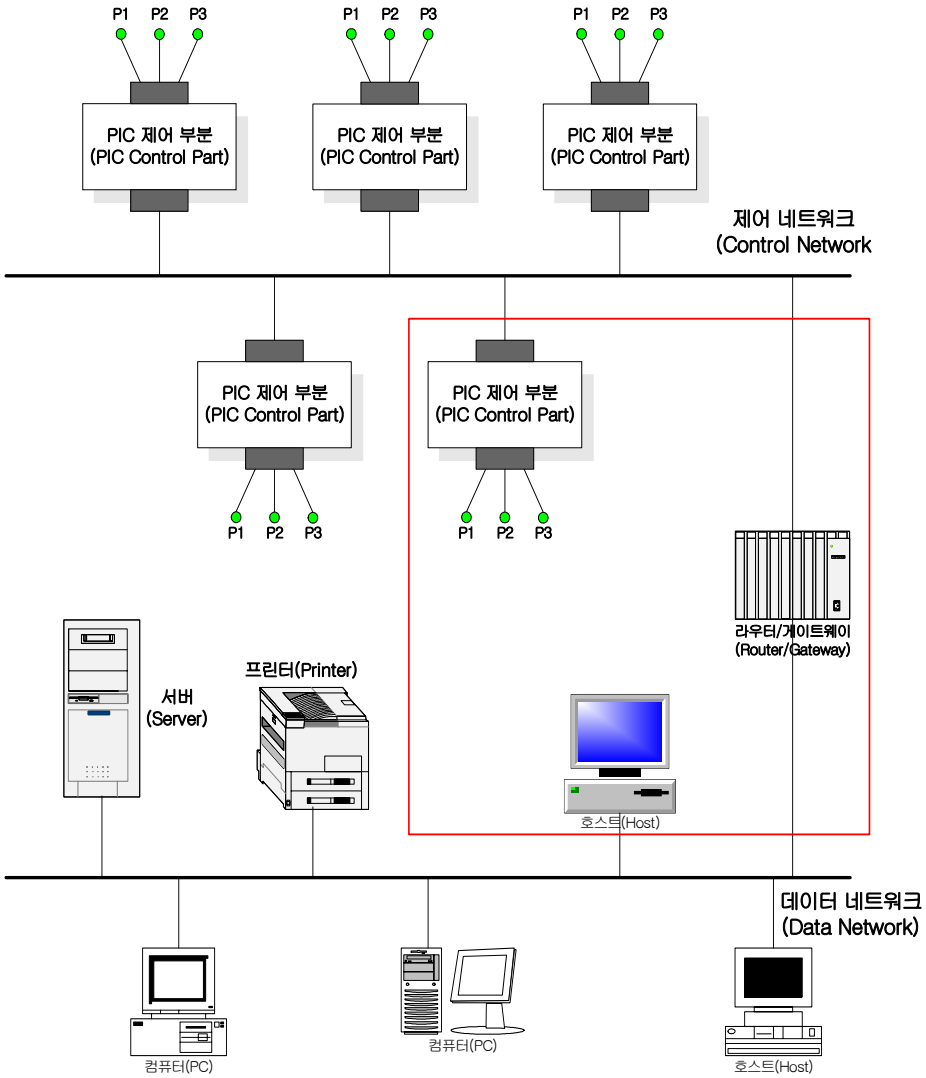

그림 3-2. 기존 자동제어 아키텍처

Fig. 3-2. Present Automatic Control Architecture

LonWorks 디바이스는 ANSI/EIA 709.1의 제어 프로토콜을 사용하는 네트워크 디바이스를 의미한다. 이 프로토콜은 공개된 스펙이고 EIA 문서와 애쉴론 사에서는 구현 소스코드를 제공하기도 하지만, 대부분의 개발자는 ANSI/EIA 709.1 프로토콜이 이미 내장된 칩을 사용하여 디바이스 응용을 작성하는 것이 일반적이다.

LonWorks를 구성하기 위해 필요한 기능은 대부분 뉴런 칩에 포함되어 있고, 전송 매체로서 쌍꼬임선, 전력선, 적외선과 무선방식 등을 모두 적용할 수 있다. 이렇게 제어 네트워크를 구성하여 뉴런 칩은 많은 수의 센서와 엑츄에이터의 정보처리 능력과 제어 기능이 있고 데이터 처리속도가 1 Mbps 대인 폭 넓은 시스템을 구축할 수 있다.

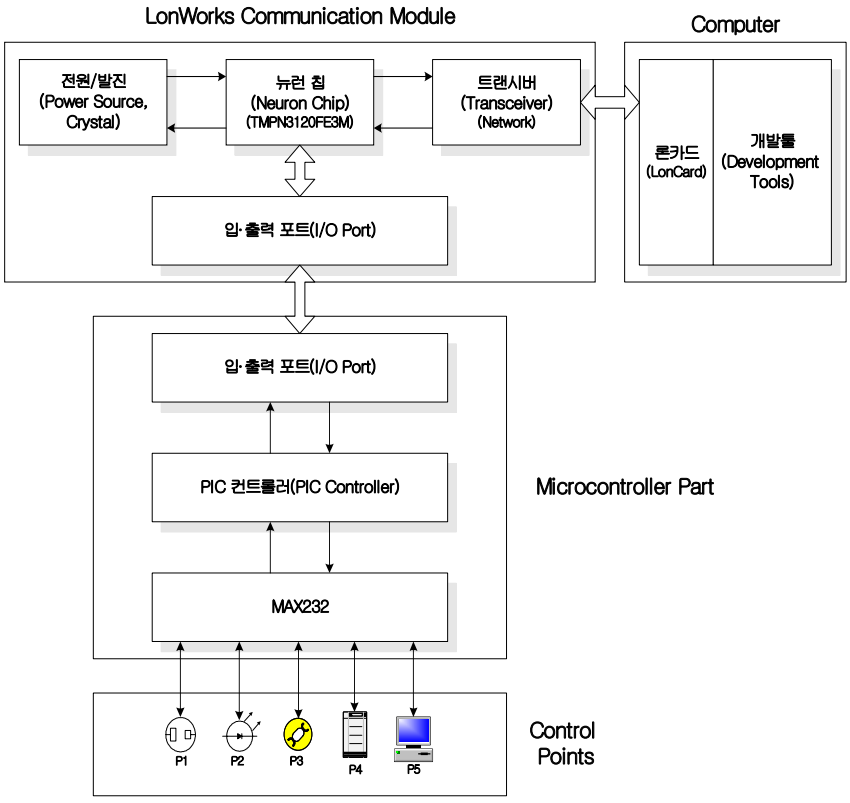

그림 3-3. LonWorks 통신 모듈 구성

Fig. 3-3. LonWorks Communication Module Configuration

이와 같이 뉴런 칩 또는 트랜시버는 통신 및 디바이스의 고유 응용을 함께 수행할 수 있는 다목적 칩이지만, 본 논문에서 구현하고자 하는 것과 같이 다른 프로세서(컨트롤러)와 뉴런 칩을 함께 사용하기도 한다. 그리고 다른 프로세서 계열은 PIC(Programmable Interrupt Controller)이고 뉴런 칩과의 통신은 직렬 또는 병렬 포트를 사용할 수 있으며, DPRAM(Dual Port Random Access Memory)과 같은 보조 메모리를 사용하여 구현 할 수도 있다.

그림 3-3은 그림 3-2의 일부분을 본 논문에서 구현하고자 하는 뉴런 칩과 다른 프로세서를 함께 사용한 시스템의 구성도이다. 본 논문에서는 그림 3-1의 SLN을 응용하여 기능상의 한계를 극복하기 위하여 다른 마이크로프로세서와 병렬 프로세서 형태로 구성하였다. 즉 제어 포인트(control point)들의 상태를 마이크로컨트롤러 제어 모듈에서 정보를 받아들이고, 이 정보를 제어 네트워크(LonWorks)를 통해서 모니터링 또는 컨트롤 시스템에 전달된다. 그림 3-3의 LonWorks 통신 모듈 구성에서와 같이 본 논문은 제어 네트워크에 연결할 수 있는 LonWorks 통신 모듈을 제작, 구현하고자 한다.

3.4 LonWorks 통신 모듈의 제작

3.4.1 주 구성 회로

그림 3-3의 구성에 대한 회로는 마이크로컨트롤러 모듈인 PIC16F-873-047SP 부분과 LonWorks 통신 모듈인 뉴런 칩 부분으로 그림 3-4, 3-5와 같다.

그림 3-4에서 보는 바와 같이 회로는 크게 뉴런 칩 회로와 발진회로, 리셋 회로, 메모리 회로 그리고 트랜시버 회로로 나눌 수 있고, 또한 독특하게 서비스 핀(service pin) 회로가 있다. 다른 회로는 일반적인 다른

프로세서와 유사하나 서비스 핀 회로와 트랜시버 회로는 주의있게 보아야 할 부분이다.

그림 3-4의 구성 요소는 뉴런 칩(TMPN3120FE3M), 서비스 핀, 리셋 회로, 트랜시버(FTT-10A), 외부 인터페이스를 위한 11개의 I/O 라인, RS-232C 시리얼 통신 포트, 발진 회로 등이 있다.

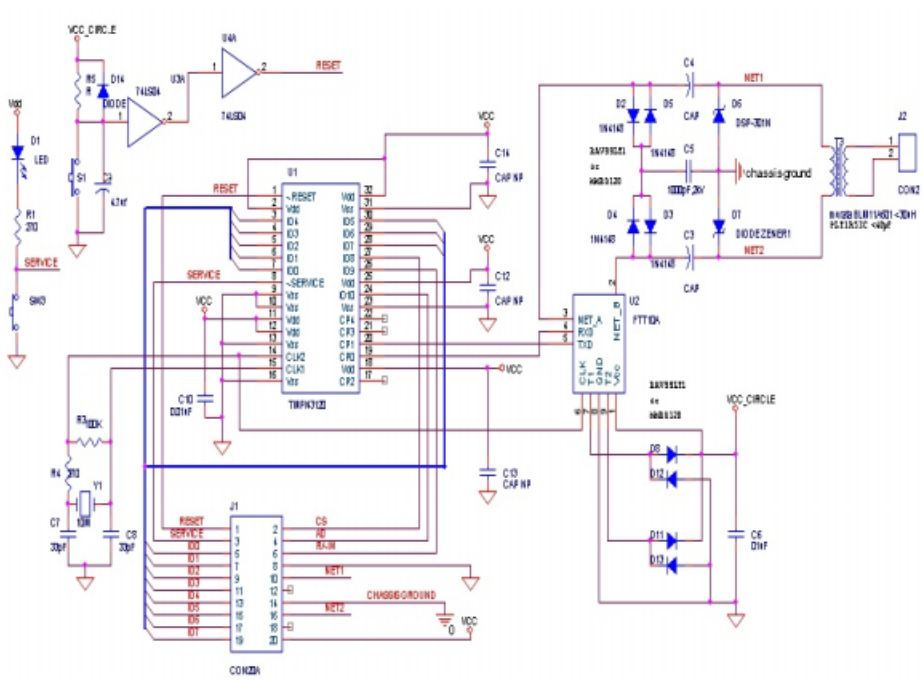

그림 3-4. LonWorks 통신 모듈의 뉴런 칩 부분

Fig. 3-4. Neuron Chip Part of LonWorks Communication Module

그림 3-5의 구성요소는 PIC 칩(PIC16F837-047SP), MAX232, 제어 포인트와 연결하는 시리얼 통신 포트 부분으로 구성되어 있다. 즉 LonWorks의 주변 회로를 나타내고 있다. 그림에서 보는 바와 같이 외부 확장 인터페이스를 통해 주변기기들과 연결이 가능하며, 다양한 응용을 구현할 수 있다. 또한 필요 시 RS-232C 포트를 사용할 수 있도록 구현하였다. 기본적으로 네트워크 통신기능 및 RS-232 시리얼 통신 기능을

갖추고 있으며, 사용자 응용 인터페이스를 위한 확장 포트를 가지고 있으므로, 이것을 이용하여 DIO(Direct Input Output), AIO(Analog Input Output), 접점 출력 등 다양한 사용자 응용을 구성하여 외부 인터페이스를 할 경우 미니 PLC와 같은 기능도 구현할 수 있다.

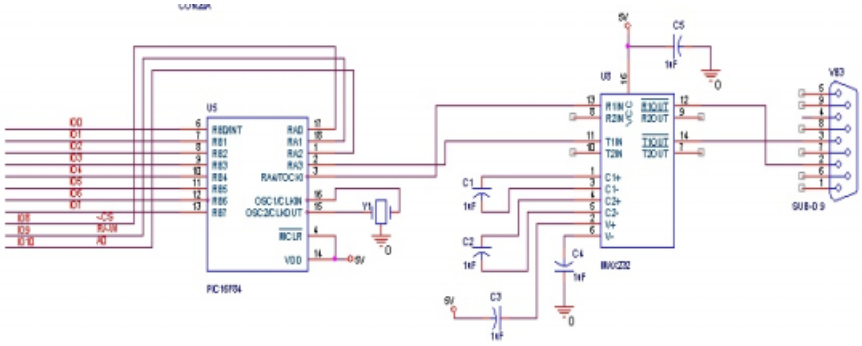

그림 3-5. LonWorks 통신 모듈의 PIC 칩 부분
Fig. 3-5. PIC Chip Part of LonWorks Communication Module

3.4.2 서비스 핀과 메모리 구성

(1) 서비스 핀

서비스 핀은 LON 노드의 구성, 설치, 유지를 하는 동안에 사용되는 특수한 포트이다. 이 포트는 입·출력 기능을 모두 가지며, 출력일 때 서비스 핀은 외부 발광다이오드(LED : Light Emitting Diode)를 켜기 위해 액티브-로우(active-low)에서 동작된다. LED는 노드에 유효한 응용 프로그램이 없거나 on-chip failure가 있을 때 불이 들어온다. 또한 노드는 네트워크 주소 정보가 구성되어 있지 않으면 LED가 1/2 Hz의 비율로 깜빡인다.

서비스 핀에서 logic-low 입력은 네트워크 상에서 뉴런 칩이 가지고

있는 네트워크 관리 메시지를 포함하고 있는 뉴런 ID를 네트워크 상으로 송출한다. 이런 두 가지 기능을 수행하기 위해서 이 포트는 50 % 듀티 사이클(duty cycle)을 가지는 76 Hz에서 입력과 출력사이에서 다중 통신을 하게 된다. 그림 3-6은 서비스 핀 구성의 회로도를 나타내고 있다.

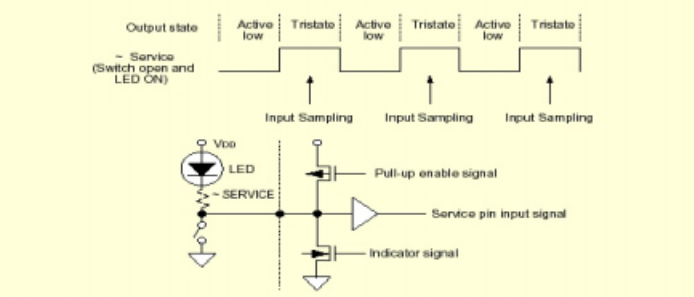

그림 3-6. 서비스 핀 구성의 회로
Fig. 3-6. Circuit of Service Pin Component

(2) 메모리 구성

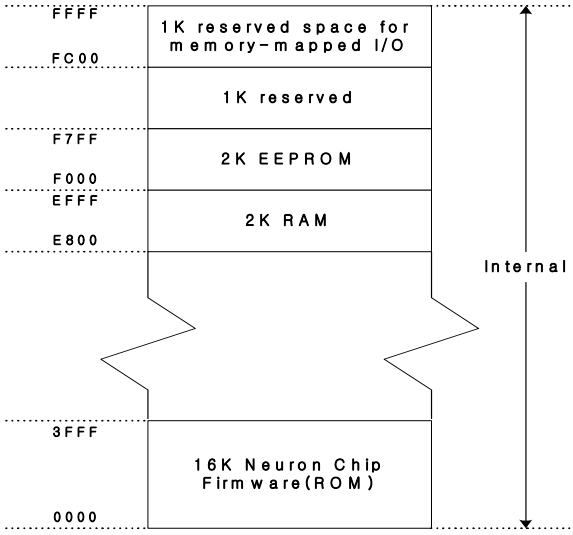

그림 3-7 뉴런 칩 메모리 맵
Fig. 3-7. Neuron Chip Memory Map

그림 3-7에서 뉴런 칩 메모리 맵을 나타내었다. ROM 영역은 0xF000-0xF7FF 번지까지 할당하였으며, RAM은 0x8000-0xEFFF 번지까지를 할당하였다. 따라서 사용자의 필요에 의해서 주변장치를 확장할 경우에는 RAM 영역의 일부를 확장 디바이스 전용번지로 할당함으로서 쉽게 인터페이스 할 수 있다. 즉 DIO, AI, AO 등을 마치 메모리의 일부처럼 쉽게 액세스 할 수 있다.

3.4.3 PCB 제작

그림 3-8, 3-9는 LonWorks 구성 회로도에 의해서 PCB(Printed Circuit Board)로 제작된 실제 사진이다.

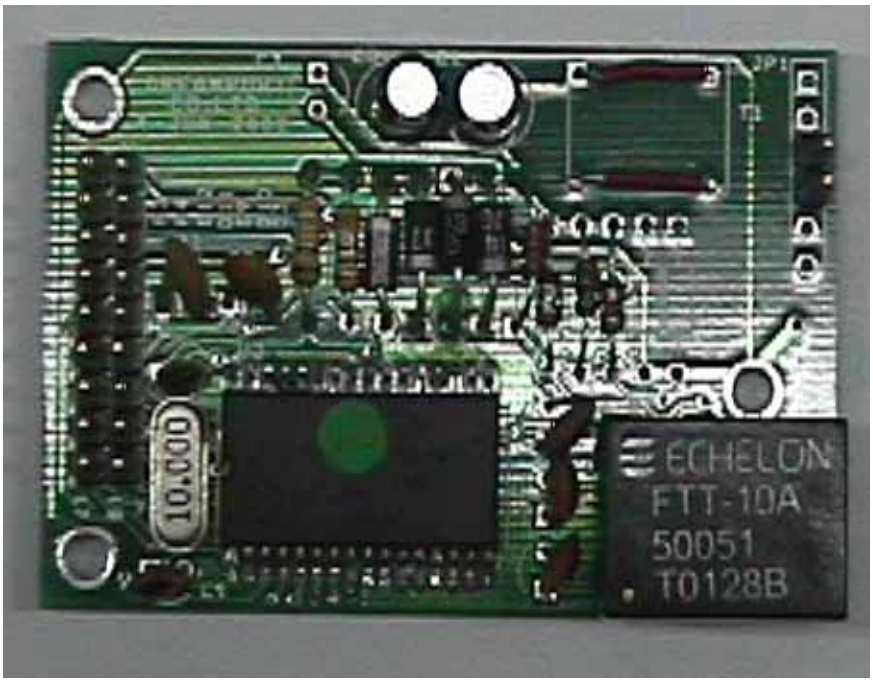

그림 3-8. PCB로 제작한 뉴런 칩 부분
Fig. 3-8. Neuron Chip Part Manufactured by PCB

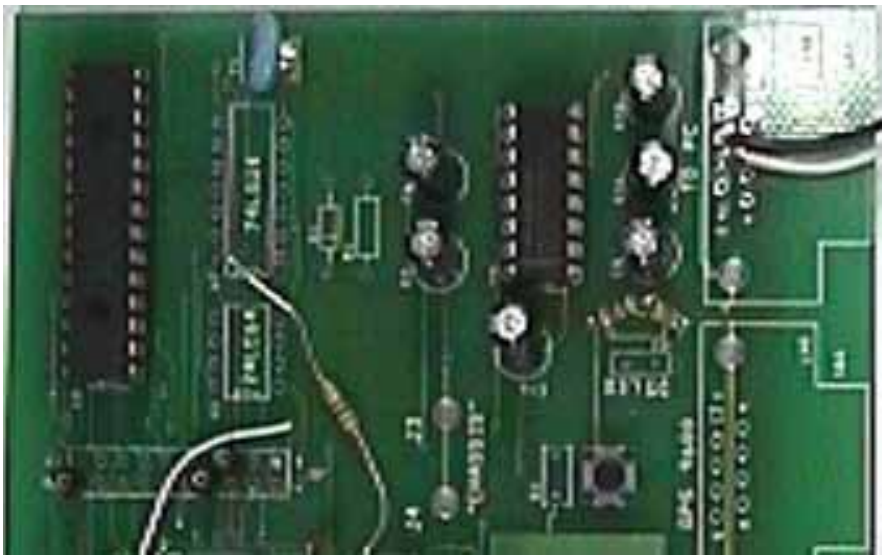

그림 3-9. PCB로 제작한 PIC 칩 부분
Fig. 3-9. PIC Chip Part Manufactured by PCB

3.5 LonWorks 통신 모듈의 펌웨어 구현

3.5.1 펌웨어 구현 시 고려사항

펌웨어 구현 시, 사용자는 개발된 펌웨어의 상태를 결정해 주어야 한다. 보통 다음과 같은 3 가지의 경우가 있는데 이것은 시스템이 많은 수의 네트워크 기반 디바이스를 가질 때 설치의 용이함, 유지보수 등 여러 가지 측면을 고려하여 선택해야만 한다.

(1) Applicationless

단지 통신관련 파라미터(parameter)만 뉴런 칩의 내부 EEPROM에 로드된다. 이때 서비스 핀(service pin)의 상태는 온(ON) 상태를 유지한다.

(2) Configured State

EEPROM에 바인딩 정보를 갖고 있어 이런 노드로 네트워크를 구성하여 전원만 제공하면 노드끼리 이미 제공된 바인딩 정보를 이용하여 통신을 한다. 만약 새로운 한 노드를 첨가하려면 기존의 노드와 새로운 노드의 EEPROM에 각각의 새로운 바인딩 정보를 제공해야 한다. 이때 서비스 핀은 오프(OFF) 상태를 유지한다. 이 모드는 어떤 장비(장치)내의 통신처럼 네트워크 구성의 변화가 적은 시스템에 주로 적용된다.

(3) Unconfigured state

기존 네트워크를 구성하고 있는 노드 사이에 새로운 노드를 첨가하면 새로운 노드에서 서비스 핀을 통해 자신의 뉴런 칩 고유 ID를 네트워크에 알리고 네트워크 관리 툴(LonMaker)을 이용해 새로운 바인딩 정보를 구성하여 통신을 한다. 이때 서비스 핀의 상태는 0.5 Hz로 점멸한다. 주로 빌딩 자동화에서 네트워크 구성요소의 변화가 많은 시스템에서 사용되는 모드이다.

3.5.2 뉴런 C 구현 조건

뉴런 C는 ANSI 표준 C에 준해 만들어진 고급 프로그램 언어이다. 뉴런 C에서는 메시지 수신이나 I/O 제어기의 상태 변화 등을 이벤트(event)로서 관리한다. 또한 메시지의 전송이나 I/O 제어기기의 취급에 있어서 사용자는 하드웨어나 소프트웨어의 저 수준 입·출력 제어를 하지 않아도 된다. 이들의 이벤트 스케줄링(scheduling)이나 통신, 하드웨어 인터페이스의 대부분은 뉴런 칩이나 뉴런 C에서 해석되고 수행한다. 따라서 사용자는 통신에 관련된 부분은 신경 쓰지 않고 자신의 응용 프

로그램만 구현함으로서 쉽게 네트워크를 구축할 수 있다[17].

(1) 이벤트 기반 방식

```
When(Event_1) {
        // Event_1 이벤트 발생시 수행할 내용
 }
   . . . . . .

 priority When(Event_2) {
        // Event_2 이벤트 발생시 수행할 내용
```

그림 3-10. 뉴런 C 프로그램의 when 문장
Fig. 3-10. When Sentence of Neuron C Program

뉴런 C에서는 응용 프로그램내의 이벤트를 관리하기 위해 “when” 이란 문장을 도입하였다. 즉 뉴런 칩 내부의 스케줄러는 해당하는 이벤트가 발생 시 ”when” 절의 내용을 평가하여 해당 내용을 수행한다. 또는 특정 “when” 절에는 우선순위를 부여할 수 있어 다른 이벤트에 우선해 이벤트 처리를 할 수도 있다. 즉 통상적으로 뉴런 C 프로그램은 그림 3-10처럼 “when” 절이 모여서 구성된다.

(2) 소프트웨어 타이머의 지원

뉴런 C는 응용 프로그램 내에서 15개의 소프트웨어 타이머를 사용할 수 있다. 각 타이머는 독립적으로 동작하며 초(s) 단위, 밀리초(ms) 단위로 설정해서 사용할 수 있다. 타이머가 종료되면 해당 이벤트가 발생한다.

(3) 뉴런 칩 펌웨어 스케줄러

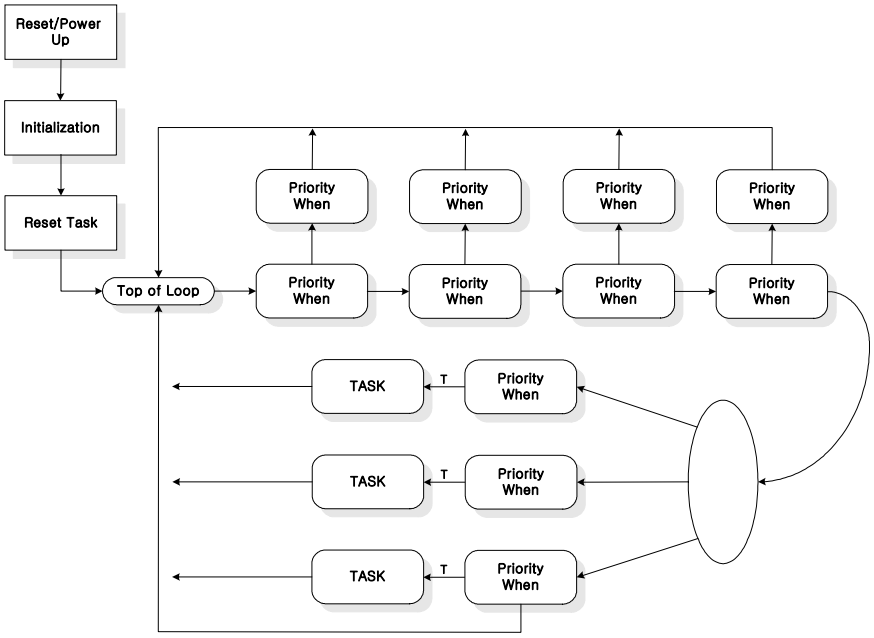

그림 3-11. 뉴런 칩 펌웨어 스케줄러
Fig. 3-11. Neuron Chip Firmware Scheduler

그림 3-11은 뉴런 C 프로그램의 운용 알고리즘을 나타내었다. 기본적으로 “when” 절의 모임으로 구성되어 있고 라운드 로빈(round robin) 방식으로 루틴(routine)이 수행되는 것을 알 수 있다.

3.5.3 데이터 전송

그림 3-12에서 보는 바와 같이 마이크로컨트롤러(PIC16F83-047SP)는 마스터 프로세서로 동작하며, 뉴런 칩은 슬레이브 프로세서로 동작한다. 두 프로세서는 HS(handshake) 신호를 주고받음으로써 핸드쉐이크 방식으로 사용권을 획득한다. 또한 두 프로세서의 병렬 I/O 모드에 대한 뉴런 칩 핀의 할당을 나타내고 있다.

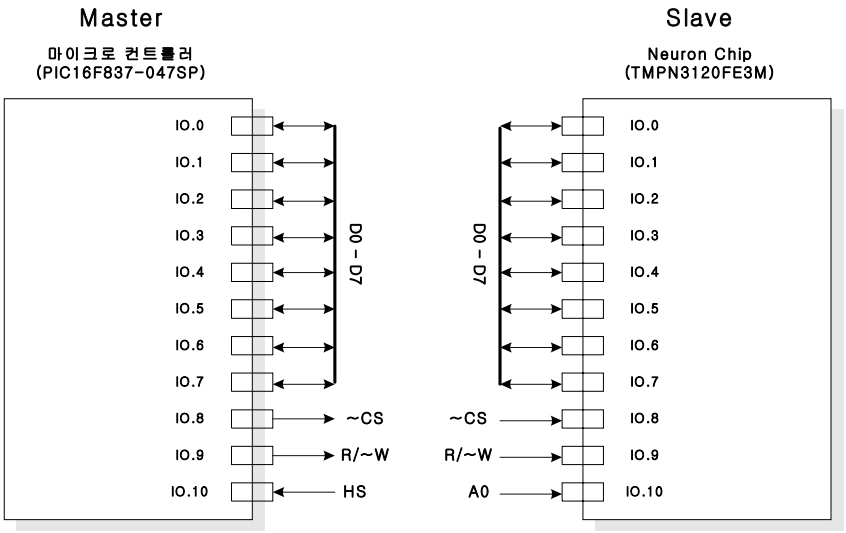

그림 3-12. 마스터와 슬레이브 사이의 데이터 전송

Fig. 3-12. Data Transmission between Master and Slave

CS(Chip Selection) 라인은 항상 마스터에 의해서 구동되고, 활성화될 때 1 byte씩 전달된다. 이 라인의 하위 펄스(low pulse)는 마스터나 슬레이브로부터 데이터를 스트로브(strobe) 한다. 마스터 관점에서 CS 라인이 하위 펄스 일 때 읽기, 쓰기 레벨(level)이 된다. R/W(Read/Write) 라인은 마스터에 의해서 구동되고, 뉴런 칩 슬레이브 상의 양방향 버스 드라이브의 방향을 결정하다.

HS 라인은 항상 슬레이브의 비지(busy) 신호에 의해서 구동된다. 이것은 마스터에게 슬레이브가 비지라는 것을 알려준다. 만약 상위(high) 신호일 때 슬레이브가 액션(읽기, 쓰기 명령과 데이터)을 취할 차례이며, 하위 신호일 때는 마스터가 버스를 액세스할 차례이다.

A0 핀은 마스터에 의해서 구동되고, 데이터 레지스터와 HS 비트를 포함하는 컨트롤 레지스터 사이에서 선택하는 주소 핀이다. HS 비트는 컨트롤 레지스터의 최하위 비트(D0 라인)이다. 컨트롤 레지스터의 남아있는 비트는 사용되지 않는다. 슬레이브인 뉴런 칩을 위해서는 D0 라인이 상위 레벨로 유지되어야 한다.

3.5.4 핸드쉐이크 프로토콜(Handshake Protocol)

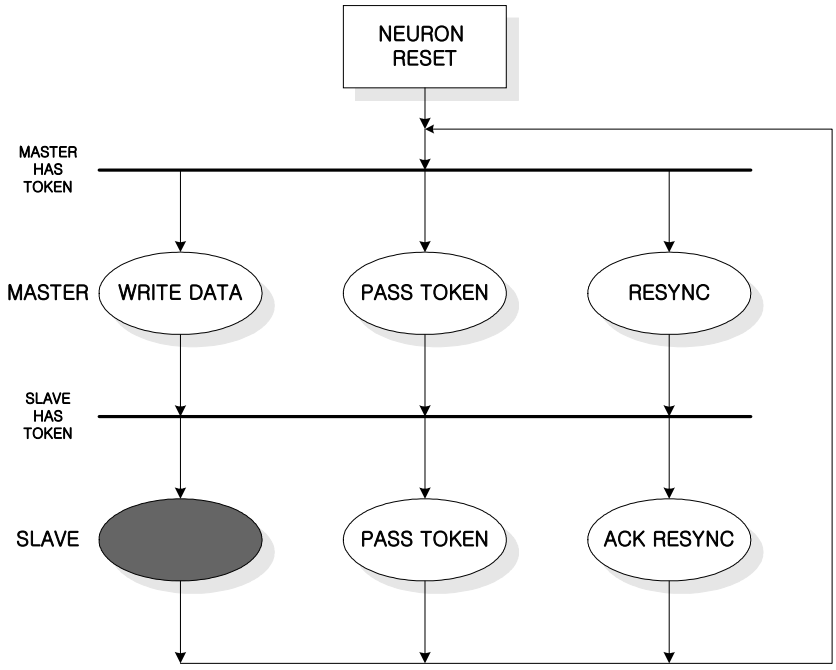

그림 3-13. 마스터와 슬레이브 사이의 핸드쉐이크 프로토콜
Fig. 3-13. Handshake Protocol between Master and Slave

뉴런 칩 펌웨어에서 구현된 핸드쉐이크 프로토콜은 공통 버스 상에 여러 디바이스가 공존하는 것을 허용한다. 어느 임의의 시간에, 오직 하나의 디바이스만이 버스를 사용할 수 있는 권한을 부여받는다. 가상 쓰기 토큰(virtual write token)은 계속해서 버스 상에서 마스터와 슬레이브 사이에서 전달된다. 일단 토큰을 소유한 디바이스는 그림 3-13과 같이 여러 동작들에서 하나를 수행한다. 즉 데이터 쓰기, 토큰 전달, 재동기화(마스터만 가능), 재동기화 확인(슬레이브만 가능) 등의 기능을 수행한다.

토큰 홀더(holder)가 사용하는 명령 옵션의 데이터 포맷은 그림 3-14와 같다. 이들 명령들은 뉴런 칩 병렬 I/O와 그 주변장치(장비) 사이에서 모든 통신의 기반이 되는 데이터 포맷(format)이다. 이들 명령 중에서 오

직 하나만 임의의 시간에 토큰 홀더에 의해서 수행될 수 있다. 명령을 끝내면 토큰은 다른 디바이스로 넘겨진다. 이로서 다음 디바이스가 명령을 실행할 기회를 가진다. 따라서 쓰기 토큰은 마스터와 슬레이브간에 비결정적으로 주고받는다.

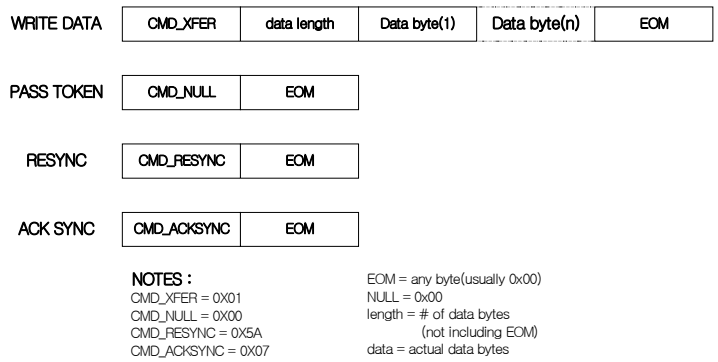

그림 3-14. 데이터 전송을 위한 패킷 구조
Fig. 3-14. Packet Structure for Data Transmission

그림 3-14에서 각각의 블록은 1 byte를 의미한다. 데이터의 포맷에서 헤더 부분은 명령의 종류를 나타내는 구분자가 있다. 즉 CMD_XFER, CMD_NULL, CMD_REYNC, CMD_ACKSYNC는 명령의 종류를 의미한다. 따라서 마스터 프로세서와 슬레이브 프로세서는 토큰을 1 byte씩 주고받음으로써 통신이 가능하다. 그림 3-15는 그림 3-14의 각각의 명령에 대한 처리 과정을 상세히 나타내고 있다.

그림 3-15에서 토큰 소유에 대한 몇 가지 주의사항이 있다. ~CS 신호와 R/~W 신호는 마스터 프로세서에 의해서 제어되고, HS 신호는 슬레이브에 의해서 제어된다. HS 신호가 하위(0) 값을 가지면 토큰 소유가 슬레이브에 있고, 상위(1) 값을 가지면 마스터 프로세서에 소유권이 있다. 마스터 프로세서의 읽기, 쓰기는 ~CS 신호가 하위 값을 가질 때 운용된다. 그리고 모든 데이터의 마지막에는 EOM(End Of Message) byte로서 마지막을 나타낸다.

그림 3-15. 핸드쉐이크 프로토콜의 마이크로 운용

Fig. 3-15. Micro-operations of the Handshake Protocol

3.5.5 뉴런 C 자원 알고리즘

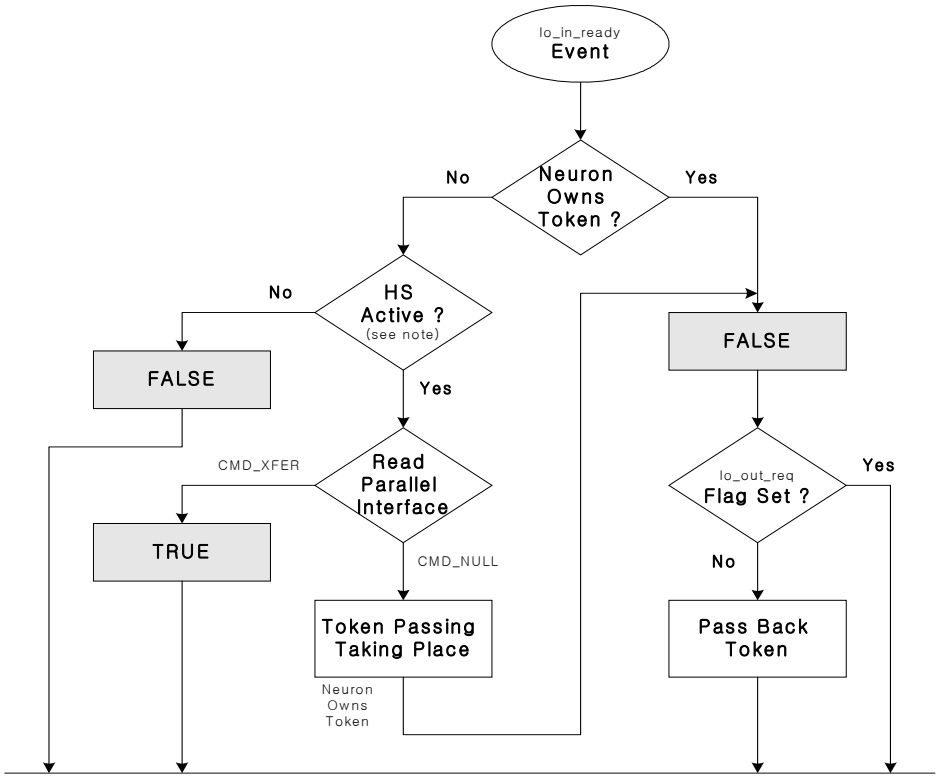

그림 3-16. io_in_ready() 이벤트 값을 위한 뉴런 칩 알고리즘
Fig. 3-16. Neuron Chip Algorithm for Evaluating the io_in_ready() Event

뉴런 칩의 병렬 I/O 객체는 내장된 뉴런 C 기능과 이벤트로 쉽게 액세스된다. 다음 기능과 이벤트는 병렬 I/O 객체와의 사용을 위해서 제공되는 함수들이다. 특히 io_in() 호출에 대한 이벤트의 평가를 위한 알고리즘을 나타내고 있다.

- io_out_request()

이 기능은 I/O 객체를 위한 io_out_ready 지시를 요청하기 위해 사용된다. 이 기능을 호출하는 것은 io_in_ready 또는 io_out_ready 평가의 끝에서 쓰기 토큰을 포기하는 것을 막기 위하여 뉴런 칩의 플래그를 설

정한다. 이것은 io_out_ready 이벤트가 참(true) 일 때까지 데이터를 버퍼링하는 응용에 의존한다.

- io_in_ready

이 이벤트는 메시지가 읽혀져야만 하는 병렬 버스에 도착할 때마다 참이 된다. 이 응용은 데이터를 가져오기 위하여 io_in() 함수를 호출한다. 그리고 이 이벤트의 평가를 위해 사용된 정확한 알고리즘이 그림 3-16에 나타나 있다.

- io_out_ready

이 이벤트는 병렬 버스가 쓰기로 될 수 있고, io_out_request() 함수가 이전에 호출되어졌을 때 참이 된다.

- io_in()

CMD_XFER 블록(바이트)이 이미 읽혀졌다. 이 함수는 길이 바이트와 나머지 전송 데이터를 읽는다. 일단 길이 바이트가 읽혀지면 뉴런 칩은 블록 읽기 명령을 수행한다.

- io_out()

CMD_XFER 바이트는 인터페이스에 쓰여질 수 있다. 다음에 이어서 뉴런 칩은 나머지 전송 데이터를 연속적으로 수반하는 길이 바이트만큼 쓰기 위해서 블록 쓰기 명령을 수행한다. 뉴런 칩은 EOM을 수행하고, 그것은 마스터나 슬레이브로서 구성되어져 있는 지에 따라 달라진다. 이 함수는 역시 io_out_request 플래그를 클리어(clear) 한다. 그림 3-17은 위와 같은 함수들에 마스터 프로세서와 슬레이브 뉴런 칩의 인터페이스 루틴에 대한 병렬 I/O 인터페이스 처리이다.

```
IO_0 parallel slave s_bus;
#define DATA_SIZE 255
struct parallel_io_interface
{
    unsigned int length;              // length of data field
    unsigned int data [DATA_SIZE];
 } piofc;
when(io_in_ready(s_bus))              // ready to input data
{
    piofc.length = DATA_SIZE;  // number of bytes to read
    io_in(s_bus, &piofc);             // get 10 bytes of incoming data
 }
when(io_out_ready(s_bus))                    // ready to output data
{
    piofc.length = 10;                // number of bytes to write
    io_out(s_bus, &piofc);            // output 10 bytes from buffer
 }
when(. . . .)                         // user defined event
{
    io_out_request(s_bus);            // post the write transfer request
 }
```

그림 3-17. 병렬 I/O 처리 인터페이스 루틴

Fig. 3-17. A Parallel I/O Processing Interface Routine

제 4 장 구현 결과 및 고찰

4.1 구현 결과

3.5절의 알고리즘을 기반으로 뉴런 C 소스를 작성하였다. 개발툴은 에쉘론사의 노드빌더 3.0을 이용하여 뉴런 C 소스를 작성하고, 컴파일 하였다. 그림 4-1에는 노드빌더 3.0을 이용하여 뉴런 C 소스 작성 과정을 나타내고 있다.

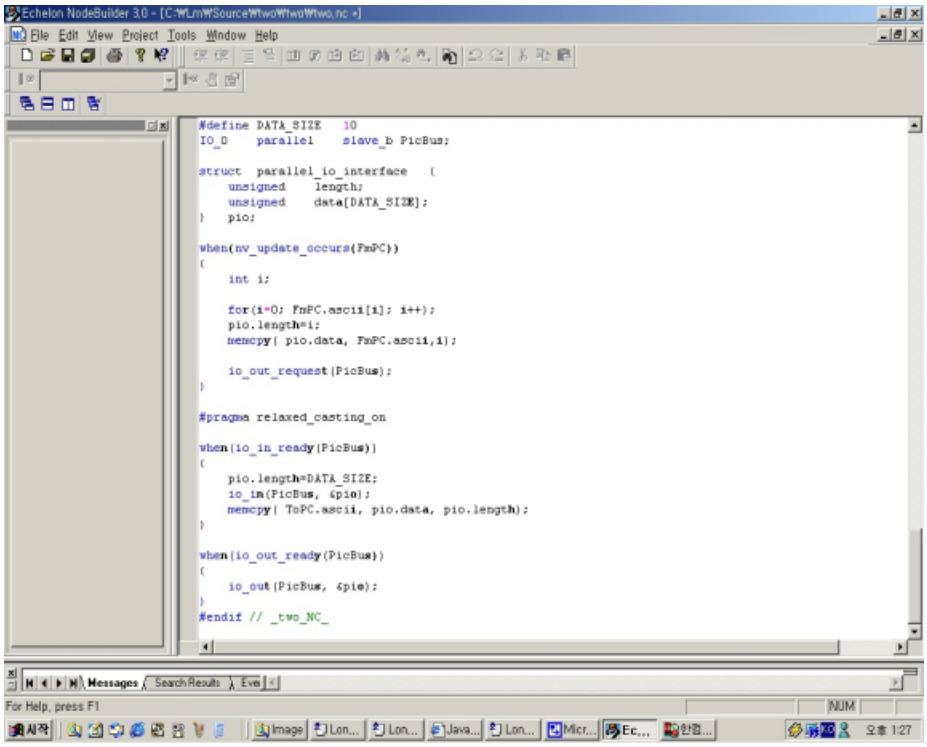

그림 4-1. 뉴런 C 소스 작성 및 컴파일(노드빌더 3.0)

Fig. 4-1. Neuron C Source Creation and Compile(NodeBuilder 3.0)

완성된 프로그램은 개발용(development)과 상업용(release) 두 가지 형태로 생성된다. 이 결과 프로그램을 LTM-10A, Gizmo 4 I/O 보드, 병렬 I/O 인터페이스 툴을 이용하여 디버깅(debugging)함으로써 사전 테스트를 완료한다. 그림 4-2는 사전 테스트를 위한 LTM-10A, Gizmo 4 I/O 보드를 나타내고 있다. 노드빌더 3.0을 이용해서 완성된 소스 프로그램은 컴퓨터와 LTM-10A를 연결하고, 결과를 디스플레이 할 수 있는 Gizmo 4 I/O 보드를 LTM-10A를 연결하여 사전 테스트를 완료한다.

그림 4-2. LTM-10A와 Gizmo 4 I/O 보드
Fig. 4-2. LTM-10A and Gizmo 4 I/O Board

4.2 실험 환경

본 장에서는 앞서 개발한 LonWorks 통신 모듈을 이용하고 실제 LON 네트워크를 구성하여 짧은 메시지 송・수신을 통하여 실험을 하였다.

그림 4-3은 이 실험을 위한 LON 네트워크 구성도를 나타내고 있다. 그림 4-3에서 P1은 PIC 제어 부분에 연결되고, 전송 매체는 RS-232C 케이블을 이용한다. 호스트(host) 컴퓨터는 LonWorks 통신 모듈과 연결된다. 이 때, 호스트 컴퓨터에는 론 카드(LON card)가 설치되고 전송 매체로는 쌍꼬임선을 이용한다. 따라서 호스트 컴퓨터도 짧은 메시지를 송・수신할 수 있다.

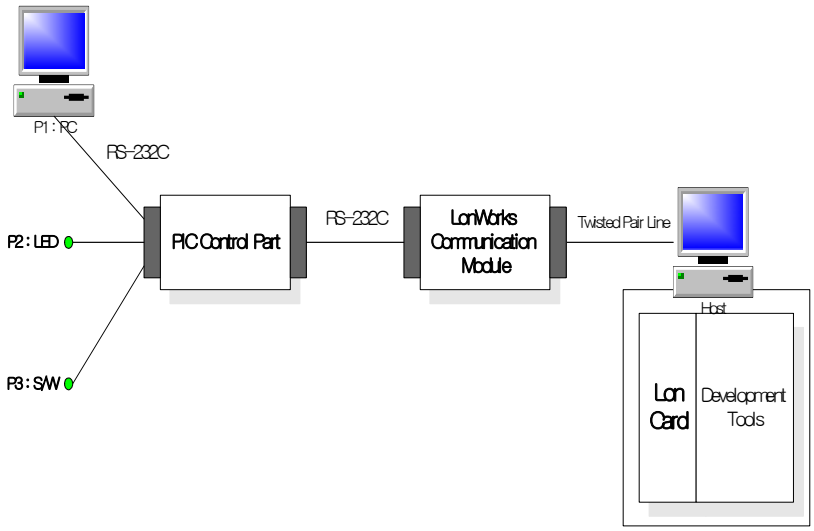

그림 4-3. 실험을 위한 LON 네트워크 구성

Fig. 4-3. LON Network Configuration for Experiment

4.3 실험결과

결과를 확인할 수 있도록 호스트 컴퓨터에서는 윈도우즈용 LonMaker를 이용하였으며, P1 컴퓨터는 하이퍼터미널(hyperterminal) 프로그램을 이용하였다. 윈도우즈용 LonMaker는 LNS 아키텍처 하에 LonWorks 네트워크를 구성하고 설치하는 프로그램으로서 비지오(Visio)를 내장하고 있다. 따라서 사용자는 비지오가 제공하는 다양한 형태와 함께 네트워크 구성 도면을 작성할 수 있다.

윈도우즈용 LonMaker에서의 작업은 네트워크 상의 디바이스 생성, 디바이스 내의 함수 블록을 불러와서 함수 블록 내의 여러 가지 속성(configuration property)을 설정하고, 함수 블록 내의 네트워크 변수와 다른 함수 블록의 네트워크 변수를 바인딩하는 순서로 이루어진다.

그림 4-4는 윈도우즈용 LonMaker로 그린 네트워크 구성도이다. 윈도우즈용 LonMaker는 비지오 V5.0을 내장하고 있으므로 네트워크가 구성된 상태를 CAD(Computer Aided Design) 도면같은 형태로 보여주게 된

다. 윈도우즈용 LonMaker의 스텐실(stencil)에는 개별 디바이스와 해당 디바이스가 갖고 있는 LonMaker 객체를 표시하는 기능함수(FB : Functional Block)가 있다. 사용자는 도면에 채널을 설정하고 해당 채널에 놓일 디바이스를 드래깅하여 도면에 얹고, 사용할 기능함수를 드래깅한 후 해당 디바이스의 플러그인(LNS Device plug-in)을 불러내어 속성값을 설정하거나 네트워크 변수를 바인딩 하는 작업을 하게 된다. 바인딩 작업은 디바이스의 외부 인터페이스 정보를 읽어서 LonMaker의 기능을 쉽게 구현 가능하다.

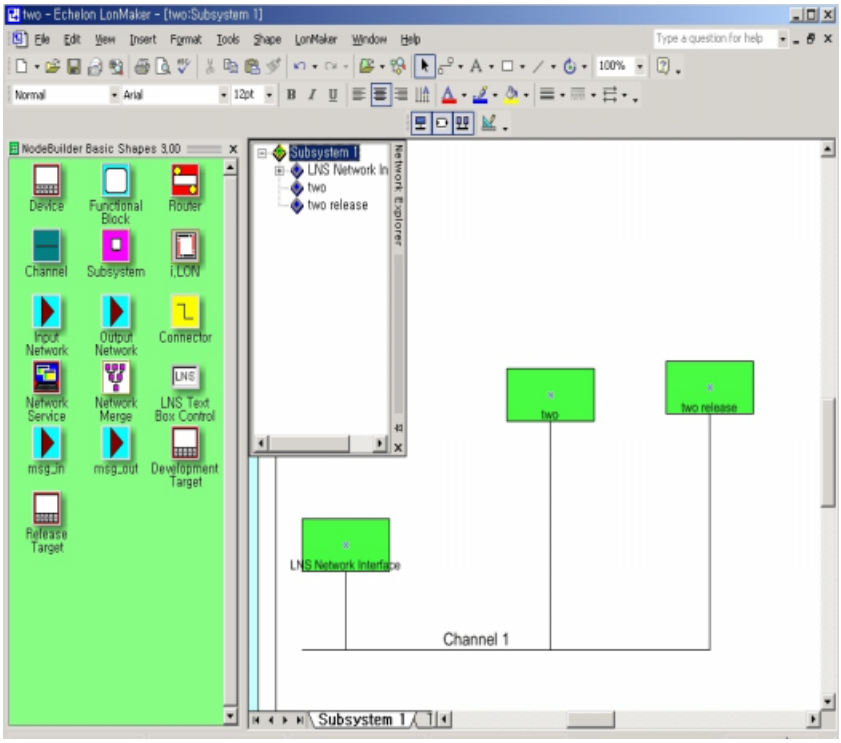

그림 4-4. 윈도우즈용 LonMaker를 이용한 네트워크 구성도

Fig. 4-4. Network Architecture using LonMaker for Windows

그림 4-5는 그림 4-4에서처럼 모든 네트워크 구성이 완료된 후, 구성된 네트워크 구성요소에 대한 디바이스, 속성 값 등을 확인할 수 있는

LonMaker Browser의 초기 화면을 나타내고 있다. 그림 4-5에서 P1의 기능함수(FB)는 가상 함수 블록이고, 네트워크 변수(NV)는 FmPC이다. 호스트 컴퓨터의 기능함수는 가상 함수 블록이고, 네트워크 변수는 ToPC이다.

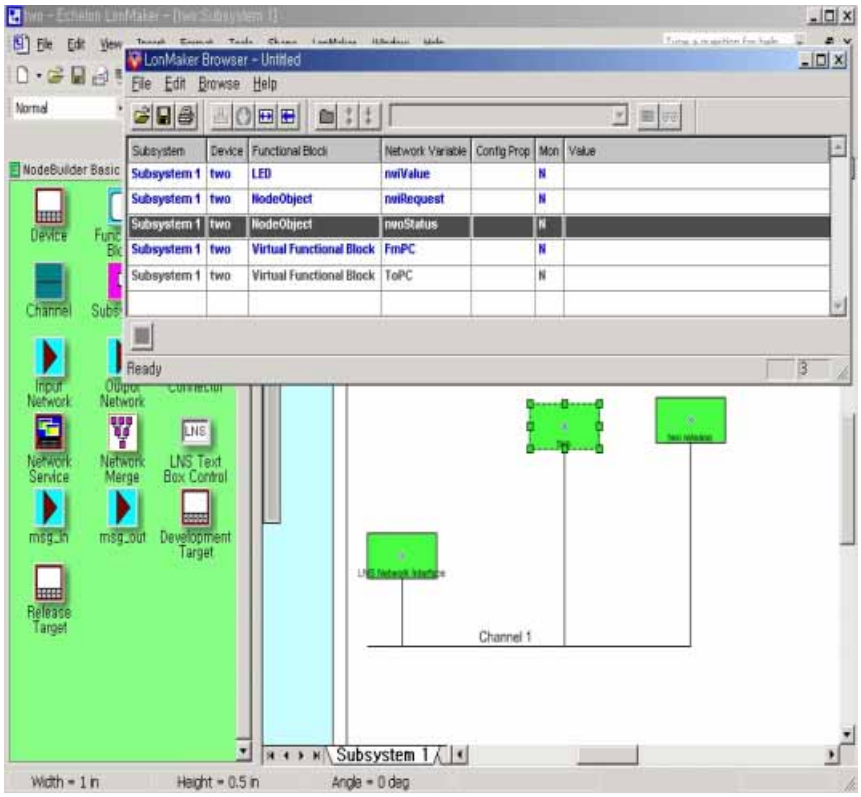

그림 4-5. LonMaker Browser의 초기 화면
Fig. 4-5. Initial Screen of LonMaker Browser

그림 4-6에서 보는 것과 같이 P1과 호스트 컴퓨터 사이에 짧은 메시지를 교환함으로서 통신을 확인할 수 있다. 즉, 호스트 컴퓨터에서 윈도우즈용 LonMaker 툴인 LonMaker Browser를 이용하여 확인하고 있다. 그림 4-7은 P1에서 하이퍼터미널을 이용하여 메시지 송·수신을 확인하고 있다.

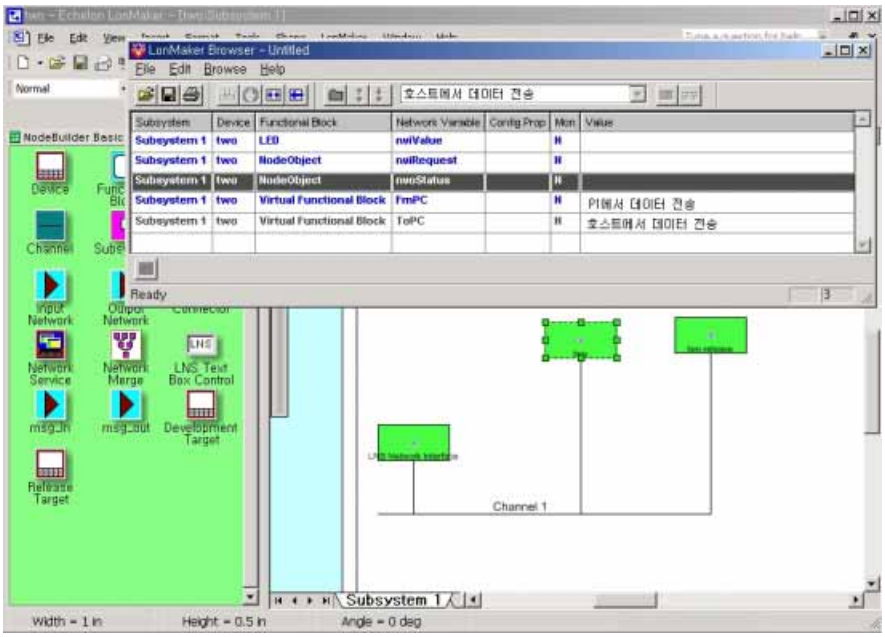

그림 4-6. 호스트 컴퓨터의 데이터 송·수신 결과

Fig. 4-6. Data Send/Receive Result of Host Computer

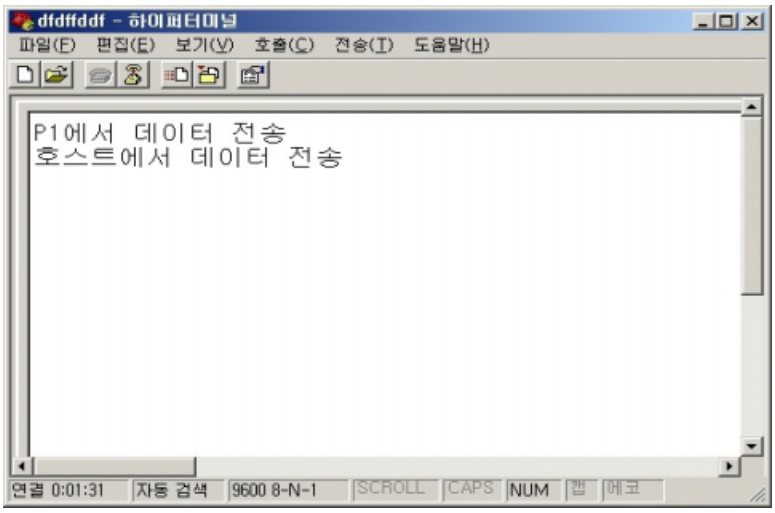

그림 4-7. P1의 데이터 송·수신 결과

Fig. 4-7. Data Send/Receive Result of P1

제 5 장 결 론

현대의 제어 시스템은 고속의 정보처리와 통합적인 정보 관리를 기본으로 하고 있다. 그리고 분산제어 및 자동화 시스템에서는 컴퓨터를 이용하여 분산된 공정을 자동화하고 전체 공정을 일관되게 관리함으로써 생산성을 향상시키고 비용을 절감하는 자동화 시스템을 구축하고 있다. 이러한 분산제어 및 자동화 시스템에 있어서 네트워킹 기술이 핵심 기술로 부가되고 있다. 즉 공정 제어에 있어서는 분산제어 기법이 발달함에 따라 공정제어 시스템이 거대화되어 가고 있고, 개방형 통신망 구조가 제안된 이후, 여러 회사에서 개발된 독립적인 구성에서 벗어나 전체적으로 결합시켜 하나의 종합적인 시스템으로 구축하려는 노력이 이루어지고 있다.

따라서 본 논문에서는 각 제어기간의 통신, 데이터의 교환 등의 문제를 해결하기 위해 표준 통신망인 OSI 참조 모델에 기초하고, 여러 가지 필드버스 중에서도 개방형 통신 프로토콜이 우수하고 경제성이 뛰어난 에쉴론사가 창안한 제어용 네트워크 시스템인 LonWorks를 이용하여 마이크로컨트롤러 제어를 위한 LonWorks 통신 모듈을 설계 및 구현하였다. 또한 LonWorks 통신 모듈을 이용하여 제어 포인트(P1)에 해당하는 컴퓨터와 호스트 컴퓨터간에 메시지 전송 네트워크를 구성, 실제 메시지 송·수신 여부를 실험하였다.

이를 위해서 LonWorks 통신 모듈은 크게 하드웨어 제작, 그리고 하드웨어로 제작된 LonWorks 통신 모듈의 운용을 위한 펌웨어 부분으로 나누어져 있다. 하드웨어 제작에는 뉴런 칩(TMPN3120FE3M), 마이크로컨트롤러(PIC16F873), 트랜시버(FTT-10A), LonCard(PCLTA - 20 PCI LonTalk 어뎁터) 등을 이용하여 PCB로 제작하였다. 펌웨어(소프트웨어) 구현에는 노드빌더 3.0, LTM-10A, Gizmo 4 I/O 보드, 병렬 I/O 인터페이스 등을 이용하여 구현 및 사전 테스트를 하였다. 그리고 이를 이용하여 제어 포인트 부분에서는 하이퍼터미널, 호스트 컴퓨터 부분에서는 윈

도우즈용 LonMaker를 이용하여 양단간에 짧은 메시지 송·수신 과정을 실험하였다.

본 논문에서 구현한 LonWorks 통신 모듈은 뉴런 칩을 내장하는 노드뿐만 아니라, 타 프로세서에 병렬로 연결된 경우에도 사용가능 하도록 범용성과 확장성을 확보하였다. 본 연구를 통해서 네트워크를 이용한 자동화 해결책을 제공함으로서, 이를 활용할 경우 생산성 및 품질 등 경쟁력 향상이 기대되고, 공장 자동화용 통신장비의 국산화를 통해 외화절감 및 추후 고성능의 지능형 분산 제어형 제어기를 개발할 수 있는 기초 기술력 확보가 가능하다고 사료된다. 또한 빌딩자동화(BA), 공장자동화(FA) 및 홈 오토메이션(HA), 멀티 노드 DC/AC 모터 제어, 선박엔진 제어 등 디지털 분산제어를 요구하는 거의 모든 분야에서의 적용이 가능하리라 본다.

본 논문의 연구는 메시지 교환, 실시간 제어 등의 하위 레벨 네트워크 응용에 치중하고 있으므로, 향후 연구과제로 각 계층 간의 시뮬레이션 모델의 개발을 통한 성능의 해석이 필요하고, 따라서 상위계층인 LNS를 이용한 시스템의 구축 및 펌웨어의 연구개발이 필요하다고 사료된다.

참고 문헌

[1] 박홍성, 권욱현, “산업용 네트워크와 그 응용”, ICASE Magazine, 제2권, 제4호, pp. 4-18, Jul, 1996.

[2] 이희승 외1명, 「LonWorks 기술의 이해」, 애쉴론 코리아 기술 지원팀, 1999.

[3] 백우진, 「지능분산제어를 위한 LON 시스템 구축」, 울산대학교 공학석사 학위논문, 1998. 12.

[4] “CIM/FA에 혁명이 기대되는 분산형 지적 제어 네트워크”, 자동화 기술 8월호, pp. 39-47, 1997.

[5] 홍승호외 3인, “분산제어 및 자동화 시스템과 필드버스”, ICAE Magazine, 제2권, 제4호, pp. 19-29, Jul, 1996.

[6] C.Xie, J.-S. Pu, P. R. Mooere, “A case study on the development of intelligent actuator components for distributed control system using LONWORK neuron chips”, Machatronics 8, pp. 103-119, PERGAMON, 1998.

[7] Toshiba, 「Neuron Chip Local Operating Network LSIs」, 2001

[8] Echelon, 「LonBuilder Users's Guide」, 2001

[9] Echelon, 「Neuron C Programmer's Guide」, 2001

[10] Echelon, 「Neuron C Reference Guide」, 2001

[11] Echelon, 「LonWorks Engineering Bulletins」, 2001

[12] Echelon, 「LonTalk Protocol Secification Version 3.0」, 2001

[13] Motorola, 「MOTOROLA LonWorks Technology Device Data」, 1996.

[14] Echelon, 「The '95-'96 Echelon LonWorks Products Databook」, 1996.

[15] 이희승 외1명, 「LonTalk 프로토콜의 이해」, 애쉴론 코리아 기술 지원팀, 1999.

[16] 이희승 외1명, 「LonWorks Device 개발」, 애쉴론 코리아 기술 지원팀, 1999.

[17] 이희승 외1명, 「LonWorks Firmware 개발 및 LonMark Guide line 의 이해」, 애쉴론 코리아 기술 지원팀, 1999.