EIA-709.1 Control Network Protocol을 이용한 필드버스 시스템 구현

Implementation of a Fieldbus System Based on EIA-709.1 Control Network Protocol

최 병 욱, 김 정 섭, 이 창 희, 김 종 배, 임 계 영
(Byoung-Wook Choi, Jung-Sub Kim, Chang-Hee Lee, Jong-Bae Kim, and Kye-Young Lim)

Abstract : EIA-709.1 Control Network Protocol is the basic protocol of LonWorks systems that is emerging as a fieldbus device. In this paper, the protocol is implemented by using VHDL with FPGA and C program on an Intel 8051 processor. The protocol from the physical layer to the network layer of EIA-709.1 is implemented in a hardware leval. So it decreases the load of the CPU for implementing the protocol. We verify the commercial feasibility of the hardware through the communication test with Neuron Chip, based on EIA-709.1 protocol, which is used in industrial fields. The developed protocol based on FPGA becomes one of IP and we are able to implement SOC for the terminal device in the distributed control systems. Also, the result can be applicable to various industrial field because it is implemented by VHDL.

Keywords : EIA-709.1 control network protocol, VHDL, Verilog HDL, FPGA

I. 서론

필드버스는 분산 제어 및 자동화 시스템에서 필드에 설치된 장비들 간에 실시간으로 데이터를 교환하도록 하는 디지털 직렬 통신망이다. 필드버스에는 센서, 루프 제어기, PLC, 모터, 밸브, 로봇, NC 머신, operator station 등의 각종 산업용 제어, 자동화 장비들이 접속된다. 공정 제어를 위한 신호 전송 체계로는 1950년대까지는 3-50psi의 공압 계측 신호가 표준으로 사용되었고, 1960년대부터는 4-20mA의 전류 또는 전압의 아날로그 신호가 표준으로 채택되었다. 1980년대 중반부터 디지털 기술을 이용하는 통신망 신호 전송 체계의 필요성이 대두되기 시작하면서 필드버스 기술이 개발되기 시작하였다. 공정 제어 및 각종 자동화 시스템에서 필드버스를 도입함으로써 얻을 수 있는 장점으로는 다음과 같은 사항들이 있다[1][2].

- 필드버스는 단일 배선을 이용한 일-대-다 통신 기술을 사용함으로써 자동화 시스템의 초기 설치비용을 크게 줄일 수 있다.
- 필드버스에서는 디지털 신호를 사용함으로써 기존의 아날로그 신호에 비하여 외부의 잡음에 의한 영향이 감소되고, 여러 개의 중복 신호를 동시에 전송할 수 있다.
- 필드버스는 양방향 통신을 가능하게 하여 자동화 시스템의 모니터링 및 유지 보수에 소요되는 비용을 크게 절감시킬 수 있다.
- 필드버스는 시스템의 변형을 용이하게 하여 자동화 시스템의 유연성과 확장성을 증대시키며, 따라서 자동화 관련 신기술의 도입이 용이해진다.
- 필드버스는 스마트센서의 도입을 가능하게 하여 센서 신호의 전처리 과정이 스마트 센서 내에서 모두 이루어질 수 있도록 한다. 또한, 기존의 시스템에서는 제어 센터의 컴퓨터에서 처리되던 제어기능이 필드 장비에서 바로 구현될 수 있어 자동화 시스템의 분산화가 가능해진다.

이런 필드버스 프로토콜 중의 하나로써 EIA-709.1 Control Network Protocol이 있다[3]. 이것은 분산 제어와 빌딩 자동화 시스템에서 지능화된 기기들 간에 peer-to-peer 방식의 통신을 제공하기 위한 통신망으로, 미국의 Echelon사에서 개발되었으며, 다른 필드버스 프로토콜들과는 달리 ISO(International Standards Organization)의 OSI-7(Open System Interconnection) 계층 모델을 채택하고 있다.

물리 계층(Physical Layer)의 전송 속도는 전송 거리에 따라 4.833Kbps에서 1.25Mbps까지 지원하고, 데이터 링크 계층(Link layer)은 predictive p-persistent CSMA (Carrier Sense Multiple Access) 방식으로 동작된다. 네트워크 계층(Network Layer)에서는 router기능을 제공하며, 트랜스포트 계층(Transport Layer)에서는 end-to-end 연결 관계간에 데이터 전송의 신뢰도를 보장한다. 세션 계층(Session Layer)에서는 전송측이 데이터를 전송할 권한이 있는가를 인증하는 기능 등을 제공하며, 프리젠테이션 계층(Presentation Layer)에서는 응용계층에서 이미 정의된 약 100 가지의 SNVT(Standard Network Variable Types)의 변수 데이터를 처리한다. 응용 계층(Application Layer)은 메시지 전송을 위한 다양한 서비스를 제공한다.

EIA-709.1 프로토콜은 현재까지는 3개의 8비트 파이프라인방식의 마이크로 프로세서로 설계된 Neuron Chip으로 구현되며, 사용자를 위한 개발 툴인 Lon-Builder Developer's Workbench와 프로그래밍을 위한 Neuron C를 이용한다. EIA-709.1은 넓은 범위의 응용분야에 적용될 수 있는 프로토콜로써, 1996년 5월에

접수일자 1999. 12. 20., 수정완료 : 2000 3 27.
최병욱 · 선문대학교 기계 및 제어 공학부
김정섭, 이창희, 김종배, 임계영 · LG산전 중앙연구소

Echelon은 어떤 프로세서에서도 사용할 수 있도록 개방화를 선언하였고, 1998년 EIA에서, 1999년 ANSI에서 표준화된 프로토콜로 인정되었다.

본 논문에서는 EIA-709.1 프로토콜의 물리 계층, MAC(Medium Access Control) 계층, 데이터 링크 계층을 하드웨어로 구현하였으며 구현 방법으로는 VHDL을 사용하였고 기능 검증을 위하여 FPGA를 이용하였다. FPGA로 구현된 프로토콜의 처리를 위하여 8032를 사용하여 전체 시스템을 구성, 기존의 Neuron Chip과 상호 테스트를 수행하여 동작을 검증하였다. 본 논문에서 II장은 EIA-709.1 프로토콜에 대한 전반적인 개요, III장은 기존의 구현 방법을 사용할 때 발생할 수 있는 문제점인 기존 개발 환경의 종속성을 범용 프로세서를 이용한 상위 시스템으로 개발함으로써 해결 가능함을 보이며, IV장은 개발한 시스템의 구성 요소에 대한 내용, V장은 실제로 테스트를 수행한 실험 결과. VI장은 결론으로 구성된다.

II. EIA-709.1 프로토콜의 개요

EIA-709.1은 그림 1과 같은 계층 구조를 가진다.

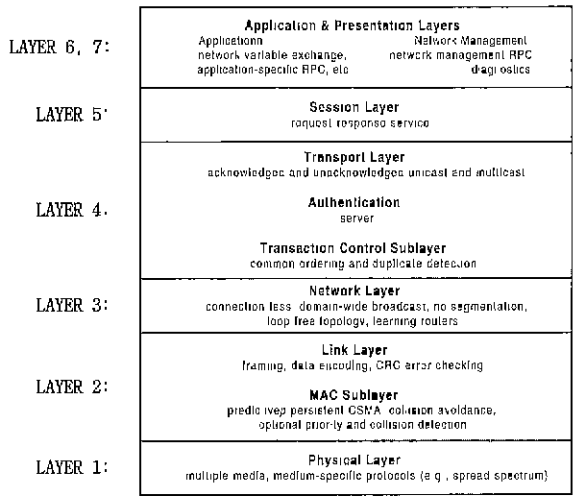

그림 1. EIA-709.1프로토콜 계층 구조.
Fig. 1. EIA-709.1. protocol layering.

각 계층에서 제공되는 서비스는 다음과 같다. 물리 계층에서는 다양한 데이터 엔코딩 기법이 사용된다. 각 엔코딩 방식은 미디어에 따라 달라진다. 예를 들어 twisted-pair 선에는 Differential Manchester Encoding이 사용되며 Power line에는 FSK modulation이나 modified direct sequence spread spectrum system이 사용되며 RF에는 FSK modulation이 사용된다.

데이터 링크 계층에서는 충돌에 대해 충돌 감지를 처리하기 위하여 predictive p-persistent CSMA 방식의 충돌 회피 알고리즘을 사용하는 MAC sublayer가 사용된다. 예측 p-persistent CSMA 알고리즘은 각 노드는 데이터를 전송하기 이전에 버스의 상태를 감지하며, 버스의 상태가 비활성일 때 준비된 메시지를 전송한다.

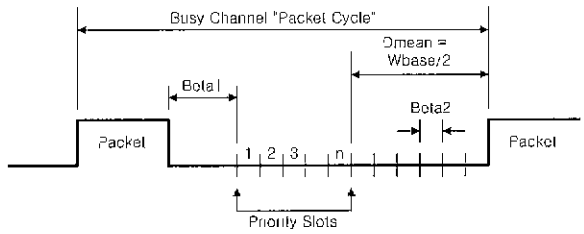

그림 2. p-퍼스스턴트 CSMA 알고리즘.
Fig. 2. Predictive p-persistent CSMA algorithm.

이때 만일 두개 이상의 노드가 동시에 메시지를 전송하면 충돌이 발생하며, 충돌된 노드들은 무작위(random) 시간 동안 기다렸다가 재전송을 시도한다. 따라서 다음 전송시에 재충돌의 확률은 감소된다. 여기서 p-persistent란 전송의 기회를 가진 노드가 p의 확률로 메시지를 전송하거나 1-p의 확률로 다시 무작위 시간 동안 대기하는 것을 말하며, predictive는 네트워크 내의 부하를 예측하여 무작위 대기 시간의 범위를 조정하는 기능이다.

그림 2는 예측 p-persistent CSMA 알고리즘에 관한 그림이다. 여기서 Beta1 Time은 패킷 전송이 끝나면 동작하기 시작하여, beta1 시간동안 채널이 비활성 상태를 유지하면, 채널에 패킷이 없다고 판단하고 다음 상태로 넘어가게 된다. Priority slot은 우선 패킷을 보낼 때 사용하는 것으로, 우선 패킷을 보낼 때는 이 슬롯 중의 하나를 이용해 패킷을 보내게 된다. 우선 패킷이 없는 노드는 그 채널의 최대 priority slot 수만큼은 대기해야 한다. 마지막으로 Random slot은 각 노드에 대하여 임의의 값이 선택되어 이 시간동안에도 채널이 비활성 상태이면 패킷을 보내게 된다.

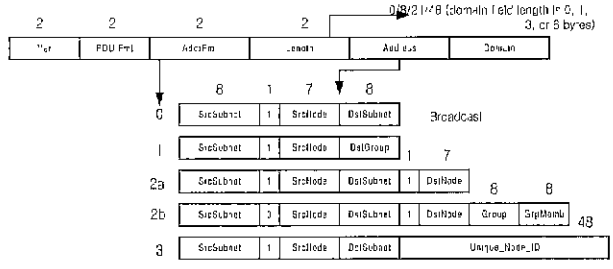

그림 3. EIA-709.1의 어드레스 형식.
Fig. 3. EIA-709.1 address format.

링크 계층은 simple connection-less 서비스를 지원한다. 링크 계층은 프레임 전송, 프레임 엔코딩, 에러 검출 기능으로 제한되며 재전송에 의한 에러 복구 기능은 지원하지 않는다. 이 프로토콜의 어드레스 형식은 모두 5개의 모드를 지원하는 데, 자세한 사항은 그림 3과 같다. 모두 2가지의 브로드캐스트 모드와 2개의 서브넷, 노드 통신, 그리고 Unique Node ID 통신을 수행한다.

네트워크 계층은 단일 도메인 내에서 패킷의 전송에 대한 것을 처리하며 도메인 사이의 통신은 지원하지 않

는다. 네트워크 계층의 서비스는 connection-less, unacknowledged 서비스이며 메시지의 분할과 재조립 기능은 지원하지 않는다.

트랜스포트 계층과 세션 계층의 공통적인 transaction control sublayer는 트랜잭션을 순서화하고 중복 메시지를 감지하는 기능을 한다. 트랜스포트 계층는 하나 또는 여러 노드로 connection-less reliable 전송을 제공한다. 메시지 전송자의 식별을 위한 인증 기능(Authentication)은 선택적으로 제공된다. 인증 서버는 그 기능을 하기 위하여 Transaction Control sublayer만을 필요로 한다. 따라서 트랜스포트 계층과 세션 계층의 메시지는 브로드캐스트 이외의 모든 어드레싱 모드를 사용하여 인증될 수 있다. 세션 계층은 원격지의 서버에 접근할 수 있는 simple Request-Response 메커니즘을 제공한다.

여기서 사용되는 통신 서비스 형식은 모두 5가지의 형식이 있다. 먼저 Unacknowledged 서비스는 acknowledge frame을 요구하지 않는 서비스이고, 이 서비스 중의 하나인 Simple unacknowledged 메시지는 중복 메시지 감지가 필요 없다. Acknowledged 서비스는 acknowledge frame을 요구하고, Repeated 서비스는 acknowledge frame을 요구하는 대신, 지정된 횟수만큼 반복하여 메시지를 보낸 후 서비스를 종결한다. Request/Response 서비스는 acknowledged frame 대신 response frame을 요구하는 서비스이고, 마지막으로 Authentication 서비스는 보안상의 불안을 회피하기 위하여 사용하는 기능으로 사용되는 서비스 형식이다.

프리젠테이션 계층과 응용 계층에서는 네트워크 변수(Network Variable) 개념을 포함한 메시지를 전송하고 받는 모든 일반적인 서비스를 제공한다. 또한 네트워크 망의 관리 및 보안등 모든 유지 보수에 관련된 서비스도 이 계층에서 이루어진다.

III. 기존의 구현 방법을 이용한 개발시 문제점

EIA-709.1 프로토콜은 Echelon사에서 개발한 것으로써 현재까지 나와있는 구현 방법으로는 앞에서 언급한 바와 같이 Toshiba나 Cypress사의 Neuron Chip을 하드웨어로 이용하여 보드를 디자인하고, Echelon사의 LonBuilder Developer's Workbench와 Neuron C를 이용해서 기본 구성 및 펌웨어와 어플리케이션 프로그램을 컴파일하고 Neuron Chip에 탑재하는 방식이 사용되고 있다. EIA-709.1은 개방형 프로토콜임에도 불구하고, 이렇게 하드웨어와 소프트웨어가 일부 특정회사에 종속되기 때문에 여러 문제가 발생할 수 있다.

먼저 Neuron Chip이라는 하드웨어로만 구성이 가능하기 때문에, Neuron Chip에서 제공하지 않는 네트워크 구성은 사용할 수가 없다. Neuron Chip에서 정해진 통신 속도 이외에 다른 통신 속도를 이용할 수 없기 때문에 네트워크망의 효율성을 높일 수 없고, 네트워크에 문제가 발생했을 때, 칩 레벨에서 디버깅이 불가능하기 때문에 네트워크 문제를 처리할 수 없다. 또한, Neuron Chip과 LonBuilder를 이용한 통신은 펌웨어 처리상의 문제로 데이터 패킷을 보내고, 다음 데이터 패킷을 보낼 때까지의 시간이 EIA-709.1로 구현할 수 있는 이론상의 시간보다 상대적으로 상당히 길다고 할 수 있다. 이런 현상은 일반적인 경우에는 크게 문제가 되지는 않겠지만, 네트워크 상에 패킷량이 많아진다던가, 네트워크에 어떤 에러가 발생했을 때에는 큰 문제가 될 수 있다.

또 다른 문제로는 소프트웨어 구성시 Neuron C라는 ANSI-C의 변형된 C를 사용해야 하기 때문에, C의 일반적인 적용이 아니라, LonBuilder에서 지원하는 방식으로 응용 프로그램을 작성해야 한다. 그러나, 이렇게 되면 LonBuilder의 개발 환경에 종속되어, 원하는 다양한 기능을 구현할 때, 개발 환경이 지원하지 않는 기능은 사용할 수 없게 된다. 따라서, 응용 프로그램의 유연성이 전혀 없어지게 되고, 프로그램의 구성시 제한을 받게 된다.

IV. 개발한 시스템의 구성

본 시스템은 EIA-709.1 프로토콜에서 물리 계층, MAC 계층과 링크 계층, 그리고 네트워크 계층을 하드웨어로 구현하였으며, VHDL을 이용하였다. 이를 위해 FPGA를 사용하였고, 기능 검증을 위해 펌웨어와 테스트 보드를 설계하였다. 이 테스트 보드의 목적이 일반 프로토콜의 구현이 아니라, 이 프로토콜을 이용하는 엘리베이터의 홀보드 설계에 있었기 때문에, 일부 기타 기능이 추가되어 설계되었다.

1 FPGA의 구현

FPGA의 설계는 VHDL(VHSIC Hardware Description Language: VHSIC는 Very High Speed Integrated Circuit의 약어)을 이용하여 Logic 설계를 하였으며 Synopsys사의 Design Compiler를 이용하여 Synthesis하고, Cadence사의 Verilog-XL simulator를 이용하여 Logic simulation를 수행하였다. 마지막으로 Altera의 MaxplusII Tool를 이용하여 FPGA P&R(Place & Route) 및 FPGA Fitting을 하였다.

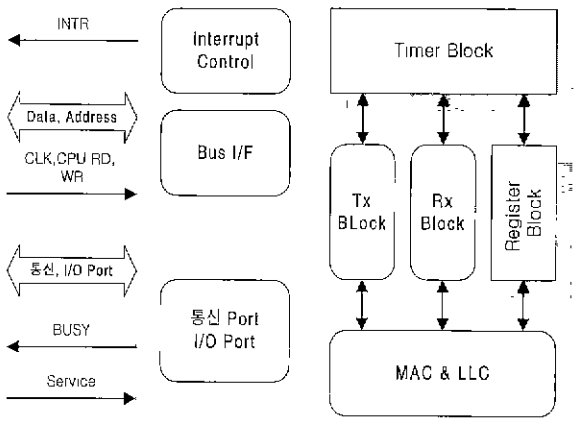

그림 4. FPGA의 블록 다이어그램.
Fig. 4. FPGA block diagram.

본 FPGA는 MAC 블록, 송신부 블록, 수신부 블록, 타이머 블록, 인터럽트 블록 등 크게 6개 블록으로 나누어 진다. 이에 대한 전체 블록 다이어그램은 그림 4와 같다.

2 MAC 블록의 구현

이 블록에서는 가변 통신 보드 레이트 설정과 차동 맨체스터 엔코더/디코더(Differential Manchester Encoder/Decoder), 통신 포트의 디지털 필터와 CRC 생성 및 검출 블록 등으로 구성된다. 또한, 랜덤 시간 생성에 의한 predictive p-persistent CSMA 알고리즘이 이 블록에서 구현되었다.

Twisted Pair선에서는 모든 통신 데이터는 NRZ 형태가 아니라, 항상 차동 맨체스터 코딩 방식을 이용해서 통신을 수행한다. 차동 맨체스터 코딩이란 1 비트 시간의 시작점에서 꼭 한번의 전이(Transition)를 발생하고, 1인 경우는 1 비트 시간동안 현재의 값을 유지하고, 0인 경우는 1/2 비트 시간 시점에서 한번의 전이가 더 발생하는 방식이다. 차동 맨체스터 엔코더와 디코더의 설계 알고리즘는 그림 5과 같다.

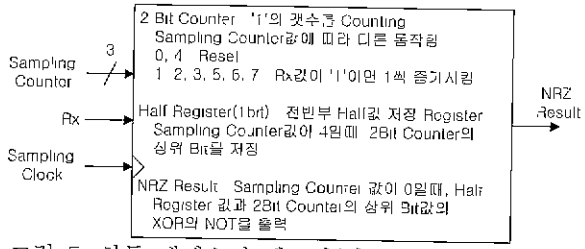

그림 5. 차동 맨체스터 엔코더/디코더 알고리즘.

Fig. 5. Differential Manchester Encoder/Decoder Algorithm.

CRC 생성과 검출 블록은 그림 6와 같다. 전송하는 매체에 있어서의 에러를 수신측에서 판단하기 위해 프레임에 포함되어 전송한다. EIA-709.1 프로토콜에서는 CCITT-CRC16 알고리즘을 이용해서 구현한다. 다음은 알고리즘에서 사용하는 프레임 전송을 위한 polynomial을 나타낸다.

$$G(x) = X^{16}+X^{12}+X^{5}+1$$

그림 6는 CRC 생성부의 블록 다이어그램을 나타낸다. CRC 생성부는 XOR와 쉬프트 레지스터(Shift Register)를 이용해서 구현될 수 있다. 이때 레지스터의 초기값은 1이고, 계산 후 통신 출력 포트로 나가는 값은 반전되어 나가게 된다.

그림 6. CRC 생성부의 블록 다이어그램.

Fig. 6. CRC Generator block diagram.

라인 활성(Line Busy)과 비활성(Line Idle)의 구분은 라인에서 전이 신호가 발생하면 일정 시간 동안 라인 활성으로 인식하고, 어떤 시간 이상 전이 신호가 라인에 발생하지 않으면 비활성으로 인식하게 된다. CSMA의 random timer는 Linear Feedback Shift Register 알고리즘을 이용해서 생성하였다. 재전송에 관한 부분은 S/W에서 에러를 보고 처리하도록 하였다.

3. 송신부(Transmit) 블록의 구현

송신부 블록은 송신을 하기 위한 프레임의 생성 및 콘트롤 신호 생성 및 각 프레임 생성 신호에 따른 스테이트 변화 블록으로 생성되어진다. 송신부 블록의 블록 다이어그램은 그림 7와 같다.

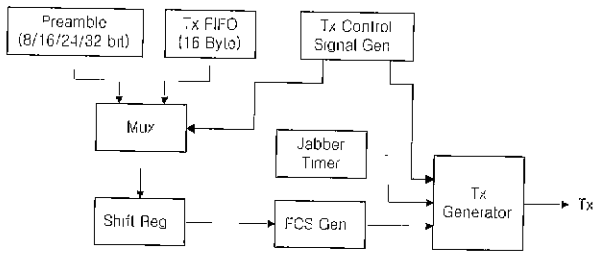

그림 7. 송신부 블록 다이어그램.

Fig. 7. Transmit block diagram.

송신부의 송신 순서는 먼저 라인 비활성 상태에서 시작된다. 라인 비활성 상태에서는 자동적으로 Beta1 타이머가 동작하여, Beta1 시간을 조사하게 된다. Beta1 타이머가 동작 중이거나, 파기되었거나, 아직 동작 전일 때, 외부에서 송신 시작(Transmit Start Enable) 신호가 활성화되면, 일단 Beta1 타이머가 파기되기를 기다린다. 그후, Priority Slot과 Random Slot의 Beta2 시간을 기다리는 동작, 계속 라인 비활성 상태가 유지되면, 패킷을 보내기 시작한다.

먼저 정해된 Preamble Byte수만큼 쉬프트 레지스터에 Preamble이 실린다. Preamble은 BitSync와 ByteSync로 구성되며, BitSync는 '1'의 값을 ByteSync는 '0'의 값을 갖는다. 먼저 쉬프트 레지스터에 전부 '1's를 전송하며, 마지막 Preamble은 마지막 비트에 '0'을 가지고 실리게 된다.

Preamble 전송 완료 후, 패킷가 전송되는데, 피포(FIFO)에서 쉬프트 레지스터로 차례대로 하나씩 실려서 전송된다. 피포에서 하나씩 전송됨에 따라, 송신 시작 전에 쓰여진 송신 데이터 길이 수를 하나씩 감소시켜, 모든 패킷이 전송되었을 때 '0'이 되면 정상 동작을 수행하고, 일치하지 않으면 데이터 길이 불일치 에러(Data Length Mismatch Error)를 발생시킨다. 또한, 피포의 모든 값을 전송하면 CRC 생성부의 동작을 수행해, 16 비트 CRC를 내보내게 된다. 또한, Jabber Timer에 의해 비정상적인 송신 출력 라인 점유를 조사한다. CRC의 전송이 끝나면, 2.5 비트 시간동안 송신 라인을 현상태로 유지(Code Violation)하여 패킷의 끝을 알려 준다. 그림 8은 송신부의 스테이트 다이어그램을 나타낸다.

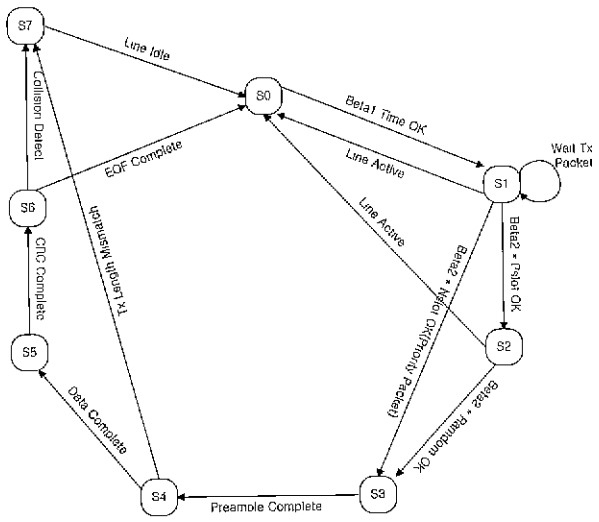

그림 8. 송신부 스테이트 다이어그램.
Fig. 8. Transmit block state diagram.

4. 수신부(Receive) 블록의 구현

수신부 블록은 수신부 스테이트 조정부, 32바이트의 FIFO, 쉬프트 레지스터 및 어드레스 검출 블록으로 구성되고, 블록 다이어그램은 그림 9와 같다.

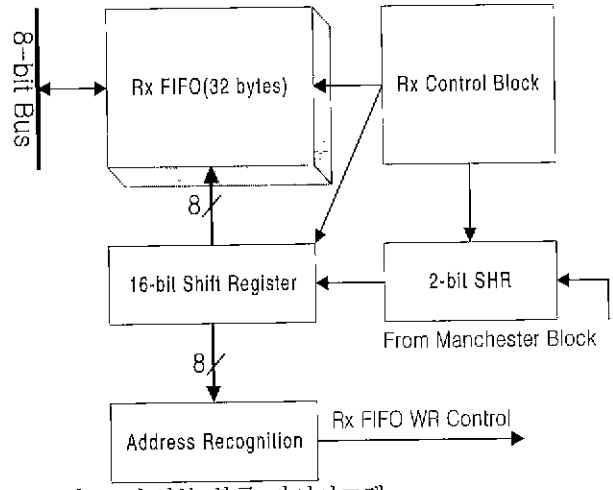

그림 9. 수신부 블록 다이어그램.
Fig. 9. Receive block diagram.

차동 맨체스터 디코더에서 새로운 패킷를 인식하면 쉬프트 레지스터로 시리얼 데이터를 넘겨준다. 쉬프트 레지스터에 16비트의 데이터가 저장되면 상위 8 비트를 피포에 쓴다. 동시에 쉬프트 레지스터에 있는 어드레스 영역을 인식하여 더 이상 수신해야 할 데이터인지 판단한다. 어드레스가 맞지 않는 경우에는 피포에 데이터를 쓰지는 않지만, 최종적으로 쉬프트 레지스터에 있는 CRC를 조사하여 CRC 에러 여부를 판단하며 랜덤 수 생성 제어에 사용한다.

수신 스테이트 제어 블록(Rx control block)은 수신부의 피포, 쉬프트 레지스터, 어드레스 검출 블록의 제어를 위해 사용한다. 그림 10은 수신 블록의 스테이트 다이어그램을 나타낸다. 6 개의 독립적인 스테이트로 구성되며 각 스테이트의 기능은 다음과 같다.

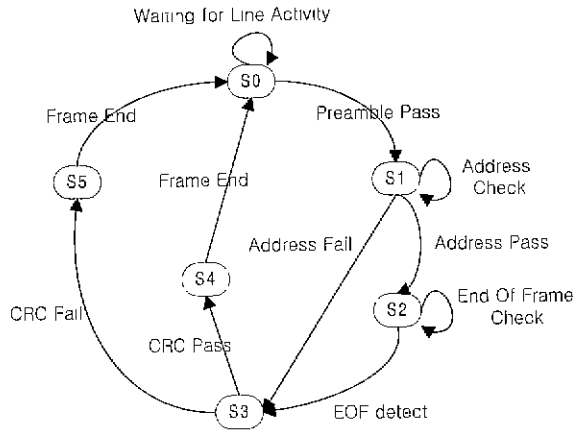

그림 10. 수신부 스테이트 다이어그램.
Fig. 10. Rx block state diagram.

1) S0(state 0) : 비활성 및 동기화 상태

라인의 활성화를 감시한다. 차동 맨체스터 디코더는 Bit/Byte Sync.를 수행한다. 비활성 상태에서 라인에 전이가 발생하면 데이터가 쉬프트 레지스터로 입력된다. Preamble 데이터 수신이 완료되면 S1으로 천이한다. BitSync Threshold 값은 4-7비트사이의 하나로 정해진다.

2) S1 : 어드레스 인식 상태

NPDU(Network Protocol Data Unit)에 포함된 Version. PDUFmt, AddrFmt, Domain Length, Address, Domain은 따로 레지스터에 저장하여 어드레스 매칭을 수행한다. 어드레스 매칭을 수행하는 동안 데이터는 피포에 쓰여진다. 어드레스 인식이 완료되었을 때 동일 어드레스 패킷이면 S2로 천이하고 아니면 S3로 천이한다.

3) S2 : 수신 완료 상태

EOF(End of Frame)를 검출할 때까지 데이터를 수신하여 피포에 쓴다. EOF를 검출하면 S3 상태로 천이한다.

4) S3 : CRC 조사 상태

EOF가 검출되면 쉬프트 레지스터의 상위 16 비트가 CRC 데이터가 된다. CRC 조사 결과 이상이 없으면 S4 상태로 천이하고 에러가 발생하면 S5 상태로 천이한다. S1 상태에서 어드레스 불일치된 경우에도 CRC를 조사해서 BL(BackLog) 값을 처리해 주어야 한다.

5) S4 : BL 처리 상태

CRC 값이 조사되면 S1 상태에서 저장한 BL 값을 처리한 후 S0 상태로 천이한다.

6) S5 : 에러 처리 상태

CRC 에러가 발생하면 이를 보고하고 BL 값을 처리한 후 S0 상태로 천이한다.

5. 인터럽트 콘트롤 블록의 구현

인터럽트 콘트롤 블록은 Interrupt Pin에 의한 처리를 지원하며, 총 8개의 인터럽트 소스에 대하여 우선 순위에 따라 인터럽트를 선택적으로 처리한다. 인터럽트 소

스는 Edge Trigger Mode만 지원하며, 블록 다이어그램은 그림 11과 같다.

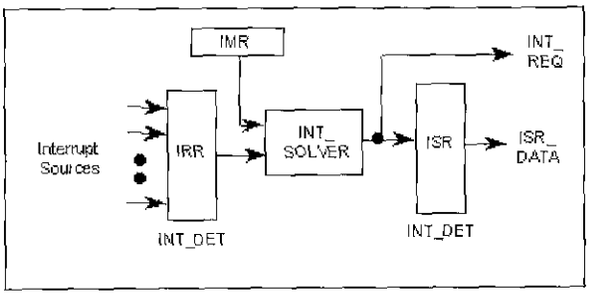

그림 11. 인터럽트 콘트롤 블록 다이어그램.
Fig. 11. Interrupt control block diagram.

6. 기타 블록의 구현

기타 구현으로는 디버깅의 편리를 위해서 루프백(LoopBack) 모드를 지원하고, 홀 보드의 일반적인 입출력 포트 사용을 위한 16개의 입력 포트와 16개의 출력 포트를 지원한다. 또한, 소프트웨어 리셋 기능을 제공하고, 현재 통신 상태를 나타내 주는 출력 포트를 두었다.

7. 테스트 보드의 구현

본 테스트 보드는 CPU는 8032를 사용하고, ROM과 RAM은 각각 32K 바이트로 구성되었다. 그리고, 디버깅을 위한 RS-232 포트를 구성하였으며, FPGA는 Altera의 EPF10K100GC503-4를 이용하여 구현하였다. 자체 제작한 통신 트랜스버를 통해서 통신을 수행하도록 하였고, 홀보드의 기타 블록으로 구성되었다. 전체적인 블록 다이어그램은 그림 12와 같다.

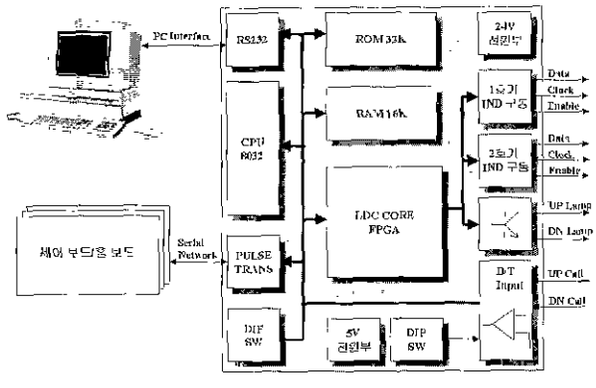

그림 12. 테스트 보드의 블록 다이어그램.
Fig. 12. Test board block diagram.

8. 펌웨어의 구현

펌웨어는 C언어를 이용하여 i8051용 프로그램을 작성하였다. IAR Systems의 IAR 8051 C Compiler를 이용하여 컴파일하였으며, 테스트용 프로그램은 송수신 데이터를 RS-232 포트로 터미널 상에 디스플레이함으로써 수행하였다.

V. 실험 결과

1 테스트 시스템 구성

테스트 환경은 그림 13에서 보여지는 바와 같이, 1개의 EIA-709.1 프로토콜을 구현한 FPGA를 탑재한 홀 보드와 1개의 기존의 홀 보드, 그리고, 전체적인 데이터를 관장하는 제어반 보드로 구성되어있다. 이 외에도 버튼이나 랜턴과 같은 주변 부품들로 구성되어 패킷의 송신과 수신을 테스트하였다. 그림 14는 EIA-709.1 프로토콜을 구현한 FPGA를 탑재한 홀 보드에 대한 자세한 사진이다. CPU는 14.7456MHz, FPGA는 20MHz의 클럭을 사용하고, 통신 속도는 78.125Kbps로 정해서 사용하였다.

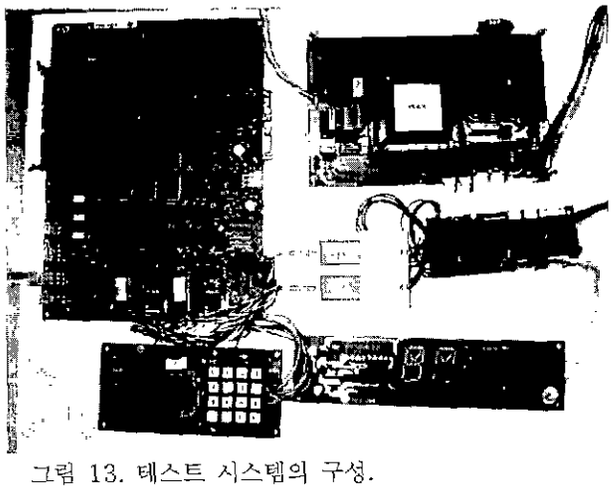

그림 13. 테스트 시스템의 구성.
Fig. 13. Test system configuration.

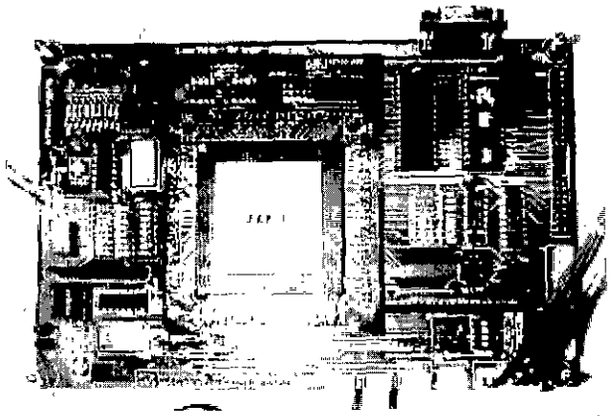

그림 14. EIA-709.1 프로토콜을 구현한 FPGA를 탑재한 보드.
Fig. 14. The board with FPGA implementing EIA-709.1 protocol.

2. 검증 결과

먼저 시뮬레이션 결과는 Cadence사의 Verilog-XL과 SimWaves를 이용해서 검증하였다. 그림 15는 그룹 어드레스 방식의 Unrepeated 서비스 통신일 때, 송신부에서 만들어지는 통신 패킷을 보여 준다. 총 10개의 데이터를 쓰고, 송신 시작을 했을 때, 각각의 상태에 따른 패킷이 생성되어서 나가는 것을 볼 수 있다.

그림 16은 수신부의 시뮬레이션 파형을 보여준다. 이것은 두번째 패킷을 수신할 때 시뮬레이션 파형으로 11번째 수신 피포부터 19번째 수신 피포까지 총 9개의 데이터를 수신하여 저장하고, 각 상태에 맞는 콘트롤 신호를 생성하는 것을 볼 수 있다.

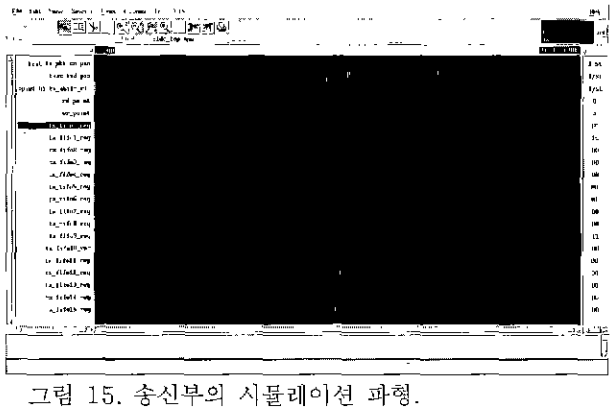

그림 15. 송신부의 시뮬레이션 파형.
Fig. 15. Transmit unit simulation waveform.

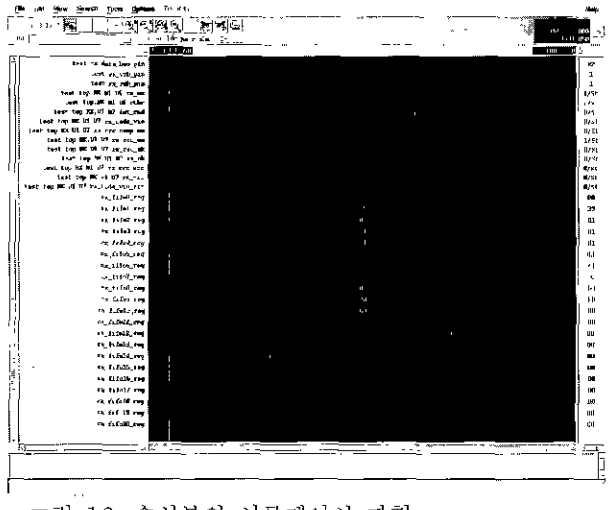

그림 16. 수신부의 시뮬레이션 파형.
Fig 16. Receive unit simulation waveform.

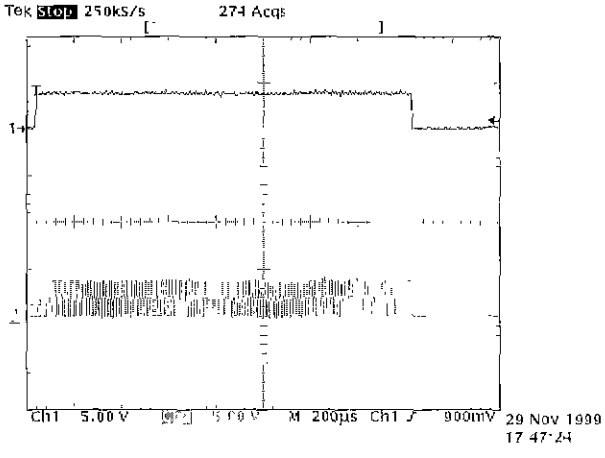

그림 17. 송신부의 에뮬레이션 파형
Fig. 17. Receive unit emulation waveform.

보드 테스트 결과는 송신부는 키가 입력될 때, 패킷을 송신하도록 하여 기존의 Neuron 보드가 패킷을 받아서 렌턴의 값을 바꾸는 것으로 테스트하였고, 수신부는 입력되는 패킷을 RS-232를 이용하여 터미날 상에 디스플레이함으로써 테스트하였다. 그림 17은 송신할 때, FPGA 보드에서 나오는 데이터 신호를 디지털 오실로스코프로 저장한 것이다. 채널 1번은 패킷이 전송 중이라는 신호이고, 채널 2번은 전송 패킷의 데이터로써 정상 동작을 수행함을 볼 수 있다.

VI. 결론

본 논문에서는 EIA-709.1 Control Network Protocol을 구현하였다. 기존의 Neuron Chip을 이용하는 방법에서 여러 단점을 회피하였다. 그 중에서 유연성을 높일 수 있도록 물리 계층부터 네트워크 계층까지는 VHDL로 구현하여 하드웨어 계층에서 구현되어서, FPGA에 검증되었으며, 기타 상위 계층은 소프트웨어로 처리하였다. 이렇게 함으로써 하위 계층에 대한 직접적인 제어가 가능해져서, 네트워크 단선이라던가 패킷 오류 등에 대한 효과적인 대처를 할 수 있게 되었다. 또한, CPU의 선택에 따라 자유자재로 통신 속도 및 통신 프로세스를 제어함으로써 높은 효율성을 얻을 수 있게 되었다. 따라서, 본 연구의 결과를 이용하면 EIA-709.1 프로토콜을 사용할 때 좀더 유연성을 높일 수 있고, 기존의 CPU를 사용하면서도 구현이 가능해져서 보다 유용하고, 쉬운 접근 방법으로 활용될 수 있을 것이다. 하드웨어 부분이 VHDL로 설계되어 IP로 구현되었기 때문에, 어떤 응용 제품을 구현한다고 하더라도 바로 적용이 가능하며, 이후 분산제어 시스템의 SOC 구현시 탁월한 선택이 될 수 있을 것이다. 향후 과제로는 보다 최적화된 하드웨어를 설계하여야 하며, 펌웨어 부분에서 지금보다 훨씬 진보된 프로그램을 설계하여야 할 것이다. 또한 범용 프로세서를 사용할 경우에 통합 개발 환경의 구축이 필수적이며, 특히 어플리케이션 계층과 네트워크의 유지 보수 기능에 관련된 부분에 대한 연구가 수행되어야 한다.

참고문헌

[1] 홍승호, "필드버스 전개과정 및 발전전망," 월간 *CONTROL*, pp. 48-56, Sept., 1996.
[2] *Intech: Field Buses Special Issue*, ISA Publication, Nov., 1996.
[3] Control Network Protocol Specification, EIA-709.1 : 1998.
[4] Motorola LonWorks Techology Device Data, 1997.
[5] "분산제어 네트워크 LonWorks의 기술," 월간 신전자, pp. 64 -72, Nov., 1998.
[6] Altera Device Data Book, 1999.
[7] Douglas L. Perry, *VHDL Second Edition*, R. R. Donnelley & Sons Company, 1993.
[8] Samir Palnitkar, *Verilog HDL*, Prentice Hall, the United States of America, 1996.
[9] 박성균, 장인용, *i8051 C실습*, Ohm사, 서울, 1994.

최 병 욱

제어 · 자동화 · 시스템공학 논문지 제5권, 제5호, 참조.

김 정 섭

1995년 연세대 전기공학과 졸업. 동대학원 석사(1997). 1997년-현재 LG산전 연구소 주임연구원. 관심분야는 필드버스, 실시간 제어 시스템 및 ASIC.

이 창 희

1996년 아주대 전자공학과 졸업. 동대학원 석사(1998). 1998년-현재 LG산전 연구소 연구원. 관심분야는 멀티미디어 및 필드버스.

김 종 배

1984년 중앙대 전자공학과 졸업. 동대학원 석사(1986). 1988년-현재 LG산전 연구소 책임연구원. 관심분야는 필드버스, SOC 및 ASIC

임 계 영

제어 · 자동화 · 시스템공학 논문지 제5권, 제5호. 참조.