Profibus-DP 프로토콜을 이용한 필드버스 시스템 구현

Implementation of a Fieldbus System Based on Profibus-DP Protocol

배 규 성, 김 종 배, 최 병 욱, 임 계 영
(Gyu-Sung Bae, Jong-Bae Kim, Byoung-Wook Choi, and Kye-Young Lim)

Abstract : In this paper, we describe a slave chip based on the Profibus-DP protocol and a system board to verify the developed slave chip. The Profibus-DP protocol is designed using VHDL and implemented on FPGA. The system board adopting the developed FPGA is designed in which the firmware is implemented on Intel 8051 by using C language. Among the Profibus-DP protocols, low level layers from the physical layer to the data link layer is implemented in the form of hardware that we are able to greatly reduce the CPU load in processing protocols, and then higher layers could be processed by software. These technologies result in an IP to make terminal devices in the distributed control systems. Therefore, many digital logics as well as communication logics can be implemented onto SOC(System On a Chip) and it could be applied to various fieldbus-related areas.

Keywords : Profibus-DP, FPGA, ASIC, VHDL, synthesis, logic simulator, emulation

Ⅰ. 서론

생산자동화 환경에서 이기종의 장비들간의 통신을 위한 연구는 80년대 초반부터 활발하게 진행되어 왔으며, 최근에 와서는 산업 현장에서 사용되는 여러 자동화 기기들 간에 통신의 중요성은 점점 증가하고 있다. 통신의 연구는 자동화 분야 뿐 아니라 다양한 분야에서 연구되었으며, OSI(Open Systems Interconnection) 상의 7계층이 제안되었고 이 기종간의 통합 기술로 표준화되었다. 그러나 공장 자동화 환경에서 매우 다양한 통신 기능을 제공하는 네트워크 시스템을 7계층으로 수용하여 각종 기기들간의 실시간 통신을 지원하도록 하기에는 여러 가지 면에서 적합하지 않았다. 이러한 배경으로 80년 후반부터 현장에 설치된 제어 기기 및 자동화 관련 장비들에서 생성된 데이타들의 실시간 통신을 지원하며 동시에 저렴한 시스템의 개발이 필요하게 되었으며, 이를위해 필드버스라는 네트워크 시스템이 제안되었다[1] [2] [3].

필드버스가 출현함으로써 배선의 복잡성 및 전달 과정에서 발생할 수 있는 잡음에 대하여 신호를 디지털화 함으로써 전송의 신뢰도를 향상 할 수 있었다. 이러한 직접적인 효과 이외에 필드버스에서는 동일한 배선으로 여러 개의 센서들에서 발생하는 중복신호를 동시에 처리할 수 있고, 네트워크 상의 각종 기기들을 모니터링할 수 있을 뿐만 아니라 센서의 주기적 교정과 같은 조치를 네트워크 망을 통하여 자동으로 수행할 수 있어 시스템의 운용 및 유지 보수에 소요되는 비용을 절감할 수 있게 되었다. 필드버스 기술에 대한 연구와 개발은 ISA(Instrument Society of America)와 IEC(International Electronics Committee)에서 필드버스에 대한 국제 표준안 제정을 공동으로 추진하면서 매우 활발히 추진되어 왔지만, 각국의 이해 관계로 국제 표준안 제정이 지연되고 있다. 현재 개발 중이거나 사용되는 필드버스에는 국제표준인 IEC/ISA 필드버스와 WorldFIP가 있고 미국의 주도로 개발된 Fieldbus Foundation와 유럽이 주도하는 profibus등이 있다. 그 이외에도 필드버스에서 제공하는 기능에 비하여 축소된 기능만을 제공하는 센서버스들이 상용화되어 사용되고 있으며, 이러한 센서버스에는 CAN, INTERBUS-S등이 있다[4]. 앞서 언급한 여러 종류의 필드버스 중 Profibus는 생산 자동화 환경에 가장 적합하며, 이미 세계적으로 200만 이상의 노드가 설치되어 그 우수성과 안정성이 증명되었다. 또한 VDC(Venture Development Corporation)의 99년 Market Survey자료를 살펴보면 2003년 PC Based Control System 및 PLC System 분야등에서 Profibus가 가장 많은 시장 점유율(24.5%)을 차지할 것 이라는 보고도 나와 있다. 현재 유럽, 미국, 일본 등의 기술 선진국에서는 필드버스를 이용한 자동화 시스템의 개발 및 구축이 활발히 진행되고 있으며, 년간 필드버스의 출하동향이 연평균 35.6%의 비율로 증가할 것으로 예상하며 2000년경에는 아시아 시장이 필드버스의 전체 시장의 30%까지를 차지할 것으로 예상하고 있다[5].

한편 국내에서는 아직 필드버스를 이용한 자동화 시스템의 구축 기술이 축적되지 못한 상태이나 자동화 시스템의 구축 및 디지털화가 점차적으로 증대되고 있으나, 모든 장비 및 기술이 선진 기술국으로 부터 수입에만 의존하고 있는 실정이다. 본 논문에서는 Profibus-DP 프로토콜를 구현할 수 있는 칩을 개발하고 통신용 보드를 설계함으로써 산업용 자동화 시스템 분야의 디지털 통신 기술을 구현함을 목적으로

접수일자 : 1999. 12. 20., 수정완료 : 2000. 5. 25.
배규성, 김종배, 임계영 : LG 산전연구소
최병욱 : 선문대학교 제어 및 기계공학부

Profibus-DP 프로토콜의 물리 계층, 데이터 링크 계층을 하드웨어로 구현하였으며, 구현 방법으로는 VHDL을 사용하여 하드웨어를 설계하고 기능 검증을 위하여 FPGA를 이용하였다. 또한 FPGA로 구현된 프로토콜의 처리를 위하여 8032를 사용하여 전체 시스템을 구성하였으며, 구현된 시스템을 사용하여 Profibus-DP 통신을 검증하였다.

본 논문의 II장은 Profibus-DP 프로토콜에 대한 전반적인 개요, III장은 Profibus-DP 프로토콜 구현을 위한 FPGA 내부 설계 로직과 슬레이브용 테스트 보드의 하드웨어 구성도, IV장은 개발한 시스템의 구성요소에 대한 검증 과정 및 로직 시뮬레이션의 결과 및 실제로 테스트를 수행한 실험 결과, VI장은 결론으로 구성된다.

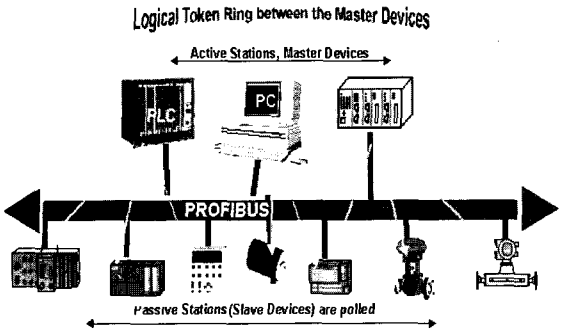

그림 1. Profibus 네트워크 구성도.
Fig. 1. Structure of profibus network.

II. Profibus-DP Protocol 개요

일반적으로 통신망 시스템은 통신 기능을 7 계층 즉, 물리(Physical Layer), 데이터 링크(Data Link Layer), 네트워크(Network layer), 트랜스포트(Transport layer), 세션(Session layer), 프리젠테이션(Presentation layer), 어플리케이션(Application layer)으로 구분하며, 이는 ISO 7498: Reference Model for Open System Interconnection(OSI-RM)에 근간을 두고 있다. 하위 4개의 계층은 전송 노드와 수신 노드간에 안정적으로 정보를 교환하도록 하는 통신 서비스 기능을 수행하고, 상위 3 계층은 응용 서비스의 수행을 위한 조정 기능을 담당한다. 필드버스는 디지털 필드 레벨 기기들과 제어 시스템을 연결하는 실시간, 양방향 디지털 통신 시스템으로 짧은 메시지를 고속으로 전송하는 생산 자동화 분야와 저속으로 Intrinsic Safety(IS) 요구 사항을 만족시키는 공정 제어 분야에서 주로 사용되고 있다.

1. 필드버스의 계층 구조

필드버스의 계층 구조는 앞에서 언급한 바와 같이 OSI-RM 의 7 계층 가운데 3-6 계층(네트워크, 트랜스포트, 세션, 프리젠테이션)이 실시간 통신을 위해 제거되고 물리, 데이터 링크 및 어플리케이션의 3 계층에 사용자 계층이 추가된 4 계층 구조로 구성되며, 여기에 Network Management 와 System Management 기능이 추가된다. 물리 계층은 데이터 링크 계층으로부터 전송할 데이터를 수신하고, 이를 전기적인 신호인 2 진 정보로 부호화하여 전송하거나, 버스상의 기기로부터 수신한 전기적인 신호를 복호화 하여 2 진 데이터를 데이터 링크 계층으로 전달하는 기능을 수행한다. 데이터 링크 계층은 MAC(Medium Access Control)과 LLC(Logical Link Control)의 두 개의 논리적인 개체로 구성되며 통신망의 노드들 간에 데이터 프레임 전송을 위한 서비스를 제공한다. LLC 에는 Connectionless, Connection-oriented, Acknowledged Connectionless 의 세 가지 형태가 있으나, 필드버스에서는 일반적으로 첫 번째와 세 번째가 사용된다. 어플리케이션 계층은 사용자를 위한 어플리케이션 인터페이스와 각종 어플리케이션 서비스를 제공하고, 사용자 서비스 요구를 데이터 링크 계층에 맞는 메시지 객체로 전환하는 기능을 수행한다. 사용자 계층은 필드 기기 내에서 수행되는 각종 데이터의 수집 및 제어에 필요한 기능을 정의한다. 즉, 각종 계측기와 제어기가 주어진 동작을 수행하는데 필요한 기능 블록 및 이에 대한 속성과 모드 등에 대한 데이터베이스의 구조를 정의한다. 이러한 기능 블럭은 주로 공정 제어를 위한 필드 기기에 사용된다.

2. Profibus 특성

Profibus 는 지멘스(SIEMENS)사에 의하여 주도되는 콘소시움에 의해서 개발되어 독일 표준(DIN 19 245)과 유럽 표준(EN 50170)으로 제정된 필드버스 프로토콜로써 적용 분야에 따라 Profibus-FMS, Profibus-DP 및 Profibus-PA 로 나뉘어진다. PA 가 주로 아날로그 신호를 처리하는 공정 제어 분야에 적합한 반면, FMS 와 DP 는 주로 이산 신호를 처리하는 제조 응용 분야에 적합하다. 또한, 같은 시스템과 같은 기기 내에서 FMS 와 DP 를 모두 구현하는 것이 가능하다.

- Profibus-FMS(Fieldbus Message Specification): FMS는 주로 셀(cell) 제어와 같은 단말 제어기와 지능형 필드 기기 사이에 많은 양의 정보를 전송하는데 사용되며 주요 특징은 VFD(Virtual Field Device)를 기반으로 하는 Object-oriented, Client-server구조에 근거하여 사용자 응용 레벨에서 통신 서비스를 제공한다. MAC(Medium Access Control)는 토큰 패싱과 마스터(Master)/슬레이브(Slave)가 결합된 방법을 사용하며, 하위 계층과의 인터페이스는 LLI(Lower Layer Interface)를 통하여 수행되고, FMS는 Context Management, Variable Access, Domain Management, Program Invocation Management, Event Management, VFD Support, OD(Object Dictionary) Management 등 39가지 서비스를 제공하나 이러한 서비스들은 시간적 제약이 필요한 환경에서 동작되지는 않는다.
- Profibus-DP(Decentralized Peripherals): DP는 자

동화 시스템과 분산된 주변 장치 사이에 실시간이 요구되는 등의 시간적 제약이 요구되는 I/O 통신을 제공하며, MAC 계층의 프로토콜로는 마스터/슬레이브 방식이 사용된다. DP에는 응용 계층이 없으며, 사용자 인터페이스는 DLLM(Direct Data Link Mapper)를 통하여 데이터 링크 계층에 직접 접속된다.

- Profibus-PA(Process Automation): PA는 주로 공정 제어 분야에 사용되며, 물리 계층은 IS(Intrinsic Safety)전송을 위해 IEC/ISA 물리 계층 표준안(31.25 Kbps 전압 모드)을 채택하였고 데이터 링크 계층은 DP를 사용하며, 응용계층은 FMS를 사용한다. 사용자 계층에서는 장비들 간에 상호호환성을 보증하기 위해 Instrument Pro-file A와 B가 정의되어 있다.

3. 프레임

Profibus 에서는 스테이션들 간의 데이터 교환을 위해 전달하는 데이터들을 프레임이라고 부르는데, 이들 프레임의 종류에는 매체 접근 권리를 나타내는 토큰 프레임, 다른 스테이션으로 데이터를 보내는데 사용하는 액션 프레임, 그리고 액션 프레임에 대한 확인 또는 응답에 사용되는 확인/응답 프레임이 있으며. 마스터 스테이션이 보낼 수 있는 프레임의 종류에는 토큰 프레임, 액션 프레임, 확인/응답 프레임이 있으며 슬레이브 스테이션은 확인/응답 프레임만을 보낼 수 있다.

FDL(Fieldbus Datalink Layer)에서 사용되는 프레임의 구조 및 종류는 그림 2와 같으며 종류를 살펴보면 다음과 같다. SD1 프레임은 FDL의 상태를 요구하는 프레임 이며, SD2 프레임은 1~246 Bytes의 가변적인 데이터를 SD3 프레임은 8bytes의 고정적인 테이타를 전송하는 프레임이다. SD4 프레임은 토큰 프레임이며, 3바이트의 크기를 가지고 있다. SD5 프레임은 요구에 대한 올바르게 수신했음을 알리는 응답 프레임이다. 이들 프레임안의 구조는 Start Delimiter(SD), 상대 주소(DA), 자기주소(SA), FC, FCS(Frame Check Sequence), End Delimiter등으로 이루어지며 FCS는 Checksum방식을 사용하고 있다.

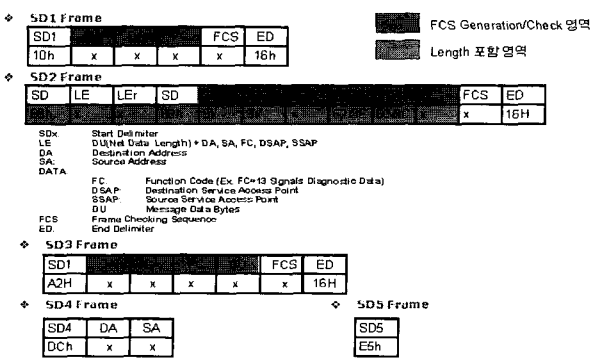

그림 2. Profibus 프레임의 구조.
Fig 2. Structure of Profibus frame.

1옥테트의 정보 (8bit)는 그림 3과 같이 3비트의 제어 비트(Start bit, Parity bit, Stop bit)가 추가되어 11 비트로 변환되어 전송되는데 문자 포맷의 구조는 그림 3과 같다.

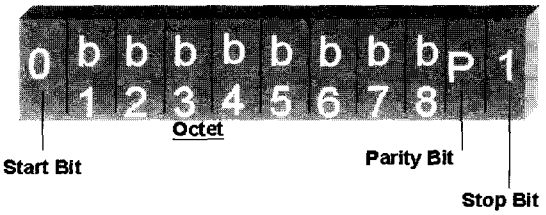

그림 3. 문자 포맷.
Fig. 3. Character format.

4. Function code

Function Code 는 프레임의 종류를 구별하고 에러 확인등의 기능을 수행하며 구체적인 기능은 다음과 같다. 그림 4 의 7 번 비트는 프레임 형태 비트로 0 이면 응답 프레임이고 1 이면 요구 프레임이다. FC 안에 있는 FCV(Frame Count Bit Valid), FCB(Frame Count Bit)를 이용하여 간단한 flow control 기능을 수행하여, 에러를 발견하고 재전송 요구를 하도록 하고 있다. 요구 프레임을 전송하는 스테이션에서 수신하는 스테이션에 처음 메시지를 보내는 것이면, FCV=0, FCB=1 로 설정하여 메시지를 전송한다. 전송하는 스테이션은 같은 수신지 스테이션으로 다음 요구 프레임을 전송할 때에는 FCV 를 1 로 설정하고 FCB 는 토글(0 이면 1 로, 1 이면 0 으로)을 해서 전송한다. 수신 스테이션은 FCV=0, FCB=1 인 요구 프레임을 받으면, 그 프레임의 전송지 주소를 SAM 이라는 변수에 저장해 두고, FCB 를 FCBM 이라는 변수에 저장해 둔다. 새 요구 프레임을 수신하여, 같은 전송지 주소이고 FCV=1 이고 FCB 와 FCBM 이 같지 않으면, 새로운 요구 프레임에 알맞은 응답 프레임을 전송한다. 이 때에는 앞에 보낸 응답 프레임을 에러가 없이 수신하고 새 요구 프레임을 전송한 것으로 판단한다. 만일 FCB=FCBM 이면 앞에서 보낸 응답 프레임을 재 전송하라는 의미로 인식하여, 보관해 놓은 응답 프레임을 전송 한다. 그림 4 는 프레임 Function Code 의 구조를 보여주고 있다.

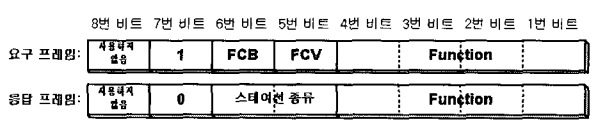

그림 4. Function code 의 구조.
Fig. 4. Structure of function code.

그림 4 중 응답프레임의 5 번, 6 번 비트를 스테이션 종류라고 하는데 이들 비트가 00 이면 응답스테이션이 슬레이브 스테이션이라는 의미이고, 01 이면 아직 논리적인 토큰링안에 들어가지 않은 마스터 스테이션이라

는 의미이다. 10 이면 논리적인 토큰링안에 들어갈 준비가 되어 있는 마스터 스테이션이라는 뜻이며 마지막으로 11 이면 논리적인 토큰링안에 들어가 있는 마스터 스테이션임을 의미한다. 그림 4 중 Function 에서는 요구프레임이 데이터 전송 서비스 중 SDA(Send Data with Acknowledge)인지 SDN(Send Data with No acknowledge)인지 아니면 SDR(Send and Request Data)인지를 나타내고, 우선순위가 "low"인지 "high"인지를 나타낸다. 표 1 에 Function 에 따른 요구 프레임과 응답프레임의 의미를 나타내고 있다.

	요구 프레임	프레임 형태	응답 프레임	프레임 형태
0	IEC 870-5-2, FC Code 0,1,2		+ACK (OK)	SD1 Frame
1			-ACK (UE) FDL/FMA 1/2에서 에러 발생	SD1 Frame
2			-ACK (RR) 전송할 데이터가 없슴	SD1 Frame
3	SDA, low	SD2,3 Frame	-ACK (RS) 서비스가 비활성화 상태임.	SD1 Frame
4	SDN, low	SD2,3 Frame	Reserved	
5	SDA, high	SD2,3 Frame	Reserved	
6	SDN, high	SD2,3 Frame	Reserved	
7	Reserved		Reserved	
8	IEC 870-5-2, FC code 8		DL SRD 서비스에서 올바르게 수신하였고 Low 우선순위 데이터를 전송함.	SD2,3 Frame
9	Request FDL Status with Reply	SD1 Frame	-ACK (NR) SRD 서비스에서 올바르게 수신하였으나, 전송할 데이터가 없슴.	SD1 Frame
10	Reserved		DH SRD 서비스에서 올바르게 수신하였고 High 우선순위 데이터를 전송함.	SD2,3 Frame
11	Reserved		Reserved	
12	SRD, low	SD1,2,3 Frame	RDL SRD 서비스에서 수신 데이터에 대한 저장영역이 없고, Low 우선순위 데이터를 전송함.	SD2,3 Frame
13	SRD, high	SD1,2,3 Frame	RDH SRD 서비스에서 수신 데이터에 대한 저장영역이 없고, High 우선순위 데이터를 전송함.	SD2,3 Frame
14	Request Ident with Reply	SD1 Frame	Reserved	
15	Request LSAP Status with Reply	SD1,2,3 Frame	Reserved	

표 1. FC 안의 functions.
Table 1. Functions in FC.

III. Profibus-DP 프로토콜 구현

1. FPGA 설계

본 FPGA(Field Programmable Gate Array)의 내부 블록은 전송부(Transmitter Logic), 수신부(Receiver Logic), MAC(Medium Access Control) & LLC(Logical Link Control) Logic, Baudrate Generation Logic, 버스 인터페이스 로직, Watchdog Timer Logic, Input/Output Port Logic, Reset Logic등으로 나누어 설계하였으며, VHDL이라는 하드웨어 기술 언어를 사용하여 Top Down방식으로 설계 하였다. 그림 5는 설계된 FPGA의 내부 블록 다이아그램이다.

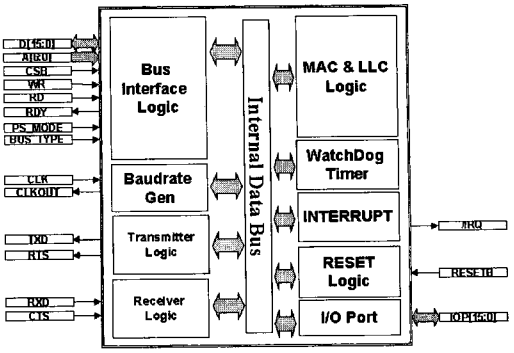

그림 5. 전체 블록 다이아그램.
Fig. 5. Overall block diagram.

1.1 전송부

전송부(Transmitter Logic)은 보내고자하는 프레임를 형식에 따라 자동적으로 생성하는 로직이며 그림 6과 같이 Transmit Control logic, Jabber Timer, Transm-itter FIFO, Transmit Shift Register 로 구성된다.

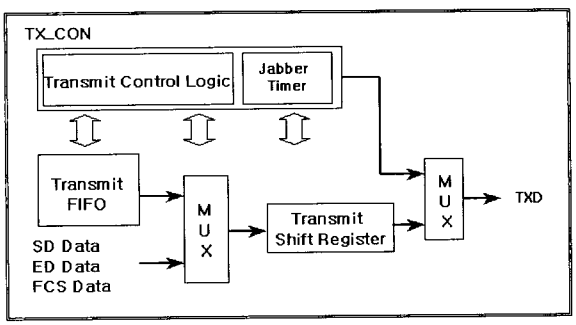

그림 6. 송신부 로직의 블록 다이아그램.
Fig. 6. Block diagram of transmitter logic.

Transmitter Logic은 프레임 전송을 위해 Line을 점유할 수 있는 최대시간을 설정하는 Jabber Time의 동작을 제어하는 Jabber Timer Control과 전송할 프레임 형식에 따라 프레임을 자동 생성하며, 사용자가 설정한 재전송 횟수를 제어하는 Transmit State Control로 구성된다. Jabber timer는 전송이 시작되면 가동하며 설정된 시간을 초가하면 즉시 프레임 전송을 중단된다. 만약 전송이 완료되면 Clear된다. Transmit state control의 전송 제어 상태도는 그림 7과 같다.

Start_State 는 데이타 전송을 위한 준비 단계이다. 전송부의 FIFO(First In First Out)의 상태와 프레임 Start 라는 명령에 의해 다음 상태로 천이한다. Sdx_State 는 전송하고자 하는 프레임 형식을 해독하여 다음 천이할 스테이트를 결정하는 단계이다. SD1, SD3, SD4 인경우 da_State 로 천이하고 SD2 인경우 Sd2le_State 로 천이한다. Sd2le_State 은 프레임 형식에 따라 Start Delimiter (68h)와 SD2 의 전송할 Data length 를 생성한 후 da_state 로 천이한다. SD5 인 경

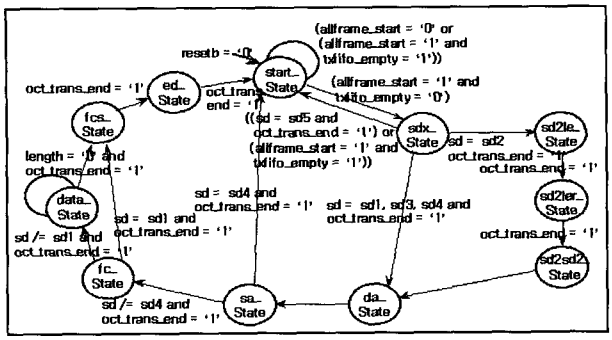

그림 7. 전송부 제어 상태도.
Fig. 7. Transmitter control state machine.

우에는 Start_state 상태로 천이 한다. DA_State 는 Destination Addres 를 생성하고, Sa_State 는 Source Address 을 생성하는 스테이트이다. 만약 보내려는 프레임이 SD4 인 경우에는 Start_State 로 천이한다. Fc_State 는 Frame Contro 을 생성하고 data_State 는 사용자가 보내려고 하는 데이터를 생성한다. 이때 사용자가 설정한 데이터 양과 실제 데이터 양이 일치하지 않으면 에러를 발생하고 정상인 경우 fcs_state 로 넘어간다. fcs_State 는 보내려고 하는 프레임의 값을 DA 에서 DU 까지 Checksum 한 데이터를 생성하고 ed_State 로 천이한다. ed_State 는 End Delimiter 로 모든 프레임에 16h 을 입력한다. ed_State 가 끝나면 초기 상태로 천이한다.

1.2 수신부

수신부(Receiver Logic)은 라인상에서 들어오는 프레임을 형식에 따라 자동으로 받아들이며 그림 8.과 같이 Receive Control Logic, SD & ED Detect, Line idle check, Receive Shift Register, Receive FIFO 로 구성된다.

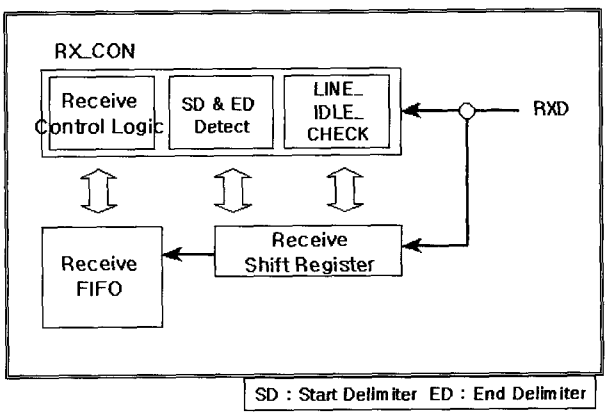

그림 8. 수신부 로직 블록 다이아 그램.
Fig. 8. Block diagram of receiver logic.

Receiver Control Logic은 수신 데이터를 체크하고 있다가 검출되는 Line Inactivity Checker와 프레임 형식에 따라 자동으로 프레임을 수신하는 Receive State Control로 구성된다. 라인이 액티브된 순간부터 모든 Logic은 동작하며 Address Recognition Logic이 존재하여 Address가 일치하는 프레임만 수신한다.
Receiver state control의 수신 제어 상태도는 그림 9 와 같다

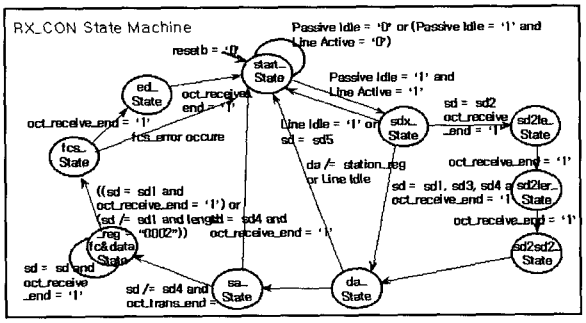

그림 9. 수신부의 제어 상태도.
Fig. 9. Receiver control state diagram.

수신부의 제어 상태도는 전송부의 상태도를 역으로 수행하는 과정이라고 할 수 있다. Start_State 는 프레임을 받을 수 있는 준비 단계이다. 데이터 라인(RXD Pin)으로 테이타가 들어오면 sdx_state 상태로 천이하면 프레임의 종류를 판별한다. SD1, SD3, SD4 인 경우에는 da_state 로 천이하고 SD2 인 경우 sd2le_State 로 천이하여 Start Delimiter (68h)와 Data Length 를 받아 들인다. SD5 프레임인 경우 Start_State 상태로 천이한다. da_State 는 Destination Address 를 Sa_State 는 Source Address 를 확인하고 받은 프레임이 SD4 인 경우 Start_State 상태로 천이한다. Fc&data_State 는 Function Control Data 와 User Data 를 받아들이면 사용자가 보낸준 데이터 양과 실제 수신된 데이터 양이 일치하지 않으면 에러를 발생하고 Start_State 로 천이한다. Fc&data_State 가 이상 없으면 fcs_State 로 천이하고 fcs_State 는 FCS 데이터를 확인하여 그림 2. 의 DA 부터 DU 까지 Checksum 한 결과와 일치하지 않는 경우 에러를 발생하고 Start_State 로 천이하고 이상이 없는 경우 ed_State 로 천이하여 End Delimiter(16h)를 확인한 후 초기 상태로 천이한다

1.3 MAC & LLC logic

MAC(Medium Access Control) & LLC(Logical Link Control) Logic 은 FPGA 의 핵심 Part 로써 메시지 또는 데이터 송수신을 위한 Function Code Decoder, Address Recognition, Frame Check Sequence Generation/Checker, Transmit/Receive Length Count, Transmit/Receive Data Length Compare Logic 으로 구성된다. Address Recognition 는 라인상에 전달되는 프레임중 상대 주소(DA)와 자기 주소(SA)를 비교하여 일치하면 나머지 프레임도 모두 수신하고 만약 일치하지 않는 경우에는 프레임을 수신하지 않는 기능을 수행한다. Transmit/Receive Length Count & Tran-

smit/Receive Data Length Compare Logic 는 전송하거나 수신하는 프레임의 length 를 체크하는 기능을 수행한다. Frame Check Sequence Generation/Checker 는 프레임중 FCS 부분을 Check Sequence Generation 로 생성하거나 수신된 FCS 와 Frame Check Sequence Checker 에 의해서 생성된 FCS 와 비교하는 기능을 수행한다.

1.4 Baudrate generation logic

Baudrate Generation Logic은 입력 Clock을 Division Factor와 Baud Rate Register(BR)값으로 Baudrate Clock를 생성하며 생성된 Bardrate Clock은 Transmitter Logic, Receiver Logic의 Transmit/Receive Shift Register 구동을 위한 기준 Clock으로 사용되여 1Clock에 1비트를 송수신한다. 외부 입력 Clock은 48MHz를 사용하여 9.6Kbps, 19.2Kbps, 45.45Kbps, 93.75Kbps, 187.5Kbps, 500Kbps, 1.5Mbps, 3Mbps, 6Mbps, 12Mbps로 Profibus-DP 통신을 가능하게 한다. 보드레이트는 사용자의 목적에 맞게 보드레이트 레지스터를 설정함에 따라 자유롭게 보드레이트를 결정할 수 있다.

1.5 Interrupt logic

Interrupt Control Block 은 인터럽트 소스에 의해 액티브된 에러를 하드웨어적으로 CPU 에게 알려주는 기능을 수행하며 인터럽트 소스에는 Watchdog Timer Over run, FIFO overflow, FCS error, Parity bit error 등이 있다. 인터럽트 로직의 내부에는 IMR(Interrupt Mask Register), ISR(Interrupt State Register), IRR(Interrupt Request Register)등의 레지스터로 구성되어 있다. 인터럽트 소스가 액티브 되면 Rising Edge 에서 IRR 에 Set 된다. 만약 현재 ISR 이 모두 Clear 상태이면 현재 설정된 IRR 의 값 중 최상위 (MSB 로부터 최상위) 인터럽트 소스에 대해 CPU 에게 인터럽트를 요청하면서 동시에 내부 System Clock 에 동기시켜 ISR 에 설정한다. CPU 가 ISR 을 읽으면 Interrupt Request Pin 과 해당 IRR 비트를 Clear 한다. 이후 CPU 가 ISR 에 임의 값을 쓰면 최상위 ISR 이 Clear 된다. 그림 10.은 인터럽트 로직의 블록 다이아그램이다.

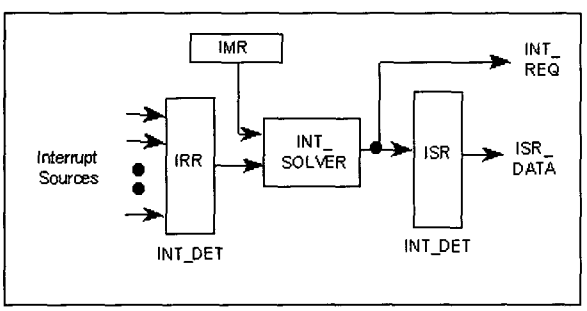

그림 10. 인터럽트 로직 블록 다이아그램.
Fig. 10. Block diagram of interrupt logic.

1.6 Bus interface logic

Bus Interface logic 은 8 비트 CPU 나 16 비트 CPU 를 모두 사용할 수 있도록 Logic 을 구현 하였으며 Motorola 계열이나 Intel 계열 CPU 를 모두 사용할 수 있는 버스 인터페이스 로직이 포함되어 있다.

2. Board 설계

Profibus-DP 슬레이브용 통신 보드의 CPU는 인텔 8032를 사용하였으며 시스템 프로그램을 위한 16Kbyte의 롬과 입 출력 프레임을 저장하는 16Kbyte의 램, 외부기기와 인터페이스를 위한 2Kbyte의 듀얼 포트 램으로 구성 되였으며 FPGA는 Altera의 EPF10K100GC503-4를 이용하여 구현하였다. Profibus-DP 통신을 위한 RS485 Driver와 9.6Kbps 에서 12Mbps까지 통신하기 위한 48Mhz의 오실레이터 사용하였으며 테스트용 프로그램의 송수신 데이터를 RS-232 포트로 터미널 상에 디스플레이 하기위하여 RS-232 Driver와 CPU를 구동하는 11.0591Mhz의 크리스탈로 구성 하였으며 아날로그 파트와 디지털 파트의 분리 및 노이즈 차폐 효과를 위해 시스템과 통신 드라이브의 파워단을 분리하였다. 그림 11은 Profibus-DP 슬레이브용 통신 보드의 개략적인 블록 다이아그램이다.

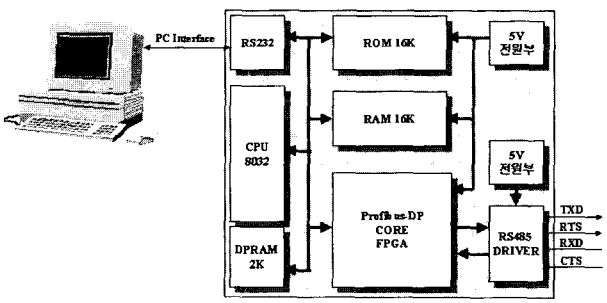

그림 11. Profibus-DP 슬레이브용 통신 보드 블록 다이아그램.
Fig. 11. Block diagram of communication board for Profibus-DP slave.

IV. 실험 결과

1. 로직 설계

VHDL(VHSIC Hardware Description Language: VHSIC 는 Very High Speed Integrated Circuit 의 약어)이라는 하드웨어 기술 언어를 이용하여 Topdown 방식으로 로직 설계를 하였다. 내부 VHDL 블록 구성은 전송부, 수신부, MAC(Medium Access Control) & LLC(Logical Link Control) Logic, Baudrate Generation Logic, 버스 인터페이스 VHDL(VHSIC Hardware Description : VHSIC 는 Very High Speed Integrated 의 약어)이라는 하드웨어 기술언어를 이용하여 Topdown 방식으로 로직설계를 하였다. 내부 VHDL 블록구성은 전송부, 수신부, MAC(Medium Access Control) & & LLC(Logical Link Control)Logic, Baudrate Generation Logic, 버스 인터페이스 로직, Watchdog Timer Logic, In-

ut/Output Port Logic, Reset Logic 구조로 Coding 하였으며 설계된 VHDL 소스(Source)를 Synopsys 사의 Design Compiler 를 이용하여 로직 합성(Synthesis) 하였다. 합성된 회로를 로직 시뮬레이션(Logic Simlation)하기 위하여 Cadence 사의 Verilog-XL simulator 를 사용 하였으며 시뮬레이션 결과를 검증하기 위하여 Altera 사의 Maxplus II Tool 를 이용하여 P&R(Place & Route) 및 FPGA Fitting 하였다. 마지막 단계로써 CPU 8032 를 사용하여 실제 보드를 제작하고 Keil Compiler 를 이용하여 펌웨어를 작성하여 Profibus-DP 프로토콜 통신을 테스트하였다.

2. Logic simulation 결과

시뮬레이션은 설계자가 설계한 것이 제대로 동작할 것인지 테스트하기 위하여 컴퓨터에 그 설계와 입력 신호(test vector)를 주고 출력이 원하는 대로 나오는지 컴퓨터로 계산하는 과정이다. 설계된 Design 을 시뮬레이션 하기위해 Cadence 사의 Verilog-XL 이라는 Simulator Tool 를 사용하였으며 Sim-wave tool 을 이용하여 시뮬레이션 결과를 확인하였다. 그림 12 는 전송부의 SD1 Frame 생성 과정 및 프레임 전송을 로직 시뮬레이션한 결과 파형이며, Baudrate Clock 를 사용하여 전송부 내부에 있는 FIFO(First In First Out) 데이터를 각 상태에 맞도록 콘드롤 하여 프레임을 생성하는 과정을 보여주고 있다. 내부적인 상태도 구조는 그림 7 의 전송부 제어 상태도와 같으며 내부적인 상태는 3.1.1 전송부의 제어 과정에 따라 생성되며 프레임 종류(SD1), 상대 주소(Destination Address), 자기 주소(Source Address), FC(Function Code), FCS(Frame

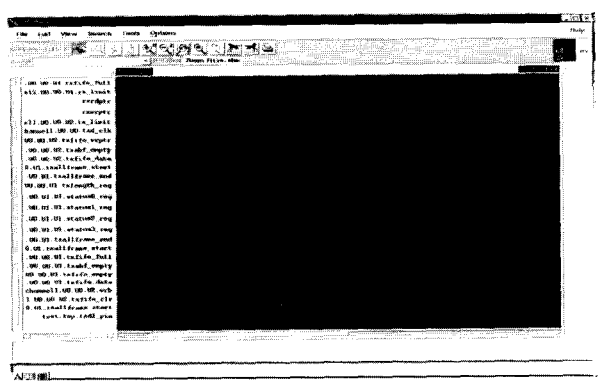

그림 12. Profibus-DP 시뮬레이션 결과.
Fig. 12. Simulation result of Profibus-DP.

Check Sequence), ED(End Delimiter)순으로 전송부의 출력단에 생성됨을 볼 수 있다. 그림 12 의 가장 하단에 존재하는 Signal(test.top.txdt.pin)이 칩에서 발생되는 SD1 프레임의 형식이다.

3. Emulation 결과

에뮬레이션(Emulation)이란 로직 시뮬레이션에서 검증된 사항이 실제 하드웨어상에서도 원하는 대로 출력되는지 확인하는 것이다. VHDL 로 설계된 Design 을 검증하기 위하여 Altera 사의 Flex10k100 FPGA Chip 을 사용하였다. 구현된 FPGA 를 확인하기 위하여 CPU(인텔 8032), RAM(16Kbyte), ROM(16kbyte), DPRAM(2kbyte), RS485 통신용 Driver, RS232 통신용 Driver 등을 이용하여 시스템을 구성 하였으며, 펌웨어는 Keil 소프트웨어의 8051 C Compiler 를 이용 프로그램 하였다. 테스트용 프로그램은 Profibus-DP 프로토콜 모니터링을 위하여 RS-232 포트를 이용하여 터미널 상에 디스플레이함으로써 시스템간의 통신을 확인하였다. 그림 13 은 Profibus-DP Protocol 에 대한 통신 테스트 환경이다.

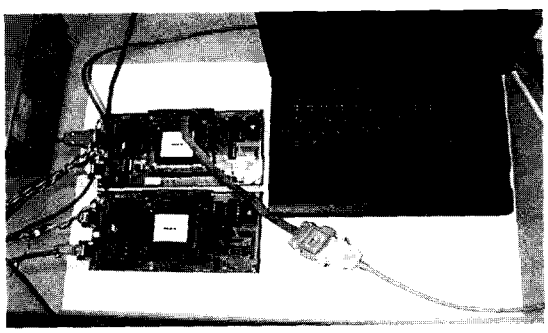

그림 13. Profibus-DP 프로토콜 실험 환경.
Fig. 13. Environment for Profibus-DP protocol test.

스테이션 1에서 스테이션 2로 요구 프레임을 보내고 보낸 요구 프레임을 확인한다. 스테이션 2는 스테이션 1으로부터 전송된 요구 프레임을 확인 하고 이에 해당하는 응답 프레임을 전송한다. 이런 과정을 RS232 포트를 사용하여 화면상에 디스플레이하여 반복적인 데이터 통신을 확인하였다. 스테이션 1과 2 사이의 전송 케이블은 twisted shielded pair cable를 사용하였으며 그림 14는 스테이션간의 통신중 SD2 프레임에 대한 측정 데이터를 Tektronix사의 오실로스코프로 측정한 결과 파형이다.

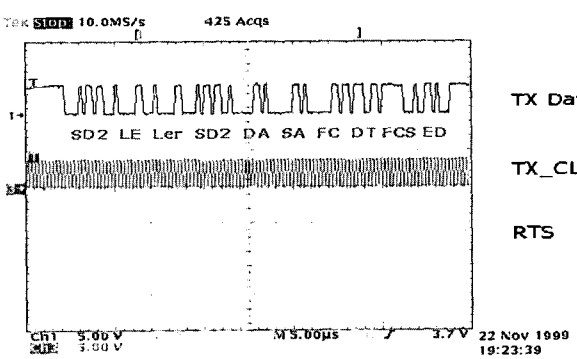

그림 14. Profibus-DP 에뮬레이션 결과.
Fig. 14. Emulation result for Profibus-DP.

VI. 결론

본 논문에서는 자동화 분야 및 분산 제어 분야에 가장 적합한 Profibus-DP 프로토콜을 하드웨어로 구현한 FPGA 를 개발 하였고 CPU 를 사용한 인텔리전트 슬레이브(Intelligent Slave) 통신용 보드를 설계하였다.

설계한 FPGA 와 통신보드가 Profibus-DP 에서 요구하는 9.6Kbps ~ 12Mbps 의 속도로 시스템간의 데이터 통신이 오류없이 수행됨을 확인하였다. 따라서 선진국의 기술에만 의존하였던 필드버스 기술을 확보 할 수 있을뿐만 아니라 물리계층 및 데이터 링크 계층을 VHDL 로 설계하여 향후 Profibus-DP 프로토콜을 사용하는 모든 단말장치에 사용가능한 IP(Intellectual Property)를 확보하게 되었으며 본 논문의 결과는 Profibus 용 통신 보드개발을 위한 자료로써 활용 가치를 가질 수 있다고 하겠다. 현재 산업계에서 많이 사용되며 상용화에 유리한 인텔리전트 슬레이브 칩만을 구현하였으나 향후 수행할 과제로는 CPU 에 의존하지 않고 Profibus-DP 프로토콜을 수행할 수 있는 Simple Slave 용 칩 개발 및 상위 어플리케이션 프로그램과 인터페이스 할 수 있는 최적화된 소프트웨어 개발을 2000 년 말까지 수행할 예정이며 현재 구현된 FPGA 에 Master 에서 필요한 MAC 와 LLC 를 추가하므로써 Profibus-DP Master 용 칩 및 보드 개발이 가능할 것으로 예상된다.

참고문헌

[1] 홍승호, "필드버스 전개과정 및 발전전망," 월간 CONTROL, pp. 48-56, Sept., 1996.

[2] 김기암, 홍승호 "자동화 시스템에서 Profibus 네트워크 인터페이스 구현 및 성능 평가", 제어.자동화.시스템공학 pp. 113-122 1998.

[3] 이철민, 이재환, 장태정, 남부희 "PROFIBUS Protocol 및 인터페이스 H/W 의 구현", 제어계측.자동화.로보틱스연구회합동학술발표회, pp. 88-91, 1997.

[4] 홍승호, 김기암, 김지용, 고성준, "분산제어 및 자동화시스템과 필드버스", 제어.자동화.시스템 학회지, 제 2 권, 제 4 호, pp. 19-29, 1996. 7.

[5] 김용래 "국내 필드버스 산업의 현황과 발전 대책" Profibus News 첫호, pp. 2-4, Dec., 1999.

[6] J.R. Jordan, Serial Networked Field Instrumentation, John Willey & Sons, 1995.

[7] DIN 19 245 Profibus Standard Part 1 : 1991.

[8] DIN 19 245 Profibus Standard Part 2 : 1991.

[9] Fieldbus Standard for Use in Industrial Control Systems Part 2 : Physical Layer Specification and Service Definition.

[10] J. R. Jordan, Serial Networked Field Instrumentation, John Willey & Sons, 1995.

[11] PROFIBUS Technical Guideline : Profibus-PA Protocol Specification Version : 1.0, 1.2. 1995.

[12] Profibus Communication Interface Layer7 for CP5412-A1 Controller, SOFTING GmbH, 1994.

[13] J. R. Piementel, Communication Networks for Manufacturing Prentice Hall, 1990.

[14] Klaus Bender, "PROFIBUS The Fieldbus for Industrial Automation" Prentice Hall, 1993.

[15] C 로 쓰는 8051, Ohm 사.

[16] 어셈블리 & C 언어를 사용한 8051 길라잡이, 두남.

배 규 성

1992 년 조선대 전자공학과 졸업. 동대학원 석사(1994). 1994 년 1996 년 10 월 ㈜하이트론 시스템 선임팀원 1996 년 - 현재, LG 산전 연구소 선임연구원 관심분야는 필드버스, & ASIC.

최 병 욱

제어 · 자동화 · 시스템공학 논문지 제5권, 제5호, 참조.

김 종 배

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

임 계 영

제어 · 자동화 · 시스템공학 논문지 제5권, 제5호, 참조.