Implementation of PROFIBUS-DP Master Protocol (ICCAS 2003)

Eui-Seob Kim*, and Seung-Ho Hong**

* Department of Electronic, Electrical And Control Instrument Engineering, Hanyang University, Seoul, Korea
(Tel : +82-31-400-4084; E-mail: creme2@dreamwiz.com)
**School of Electrical Engineering and Computer Science, Hanyang University, Ansan, Korea
(Tel : +81-31-400-5213; E-mail: shhong@hanyang.ac.kr)

Abstract : PROFIBUS is an open industrial communication network for a wide range of application in manufacturing automation and process control systems. PROFIBUS-DP(Decentralized Peripherals) are mainly used to connect smart automation devices via a fast serial link. PROFIBUS-DP adopts master/slave mechanism for communication service. In this paper, we present an implementation method of the protocol stacks for the master station of PROFIBUS-DP.

1. INTRODUCTION

PROFIBUS protocol is widely used for process control and automation systems. The PROFIBUS was developed from Germany and it was established with standard of Germany and Europe. It was also adopted as one of the eight fieldbus international standards from IEC[1]. The PROFIBUS is composed of the FMS(Fieldbus Message Specification), the PA(Process Automation) and the DP(Decentralized Peripherals)[2-3]. In this paper, we present an implementation method of the protocol stacks for the master station of PROFIBUS-DP. PROFIBUS-DP is mainly used to connect automation devices such as programmable controllers, input/output-devices, sensors, actuators and smart devices. The main purpose of PROFIBUS-DP is the fast cyclic exchange of data between a powerful master (automation system) and several simple Slaves (peripheral devices). Thus, this system uses mainly the Master-Slave type of communication services.

This paper is composed of 5 chapters. Chapter 2 describes the PROFIBUS-DP master protocol, and its communication model and structures. Chapter 3 explains the whole structures of PROFIBUS-DP master software. Chapter 4 explains the principle of operation and software implementation method of internal layer such as DDLM(Direct Data Link Mapper), USIF(User Interface) and USER(using graphic user interface). Finally, chapter 5 presents conclusions and further works of this study.

2. PROFIBUS-DP MASTER PROTOCOL

The PROFIBUS-DP defines only 3 layers among ISO OSI 7 layers in order to be suitable for real-time process control. (see Figure 1). In a typical remote I/O configuration, single master configurations are mainly used to fulfill reaction time requirements. PROFIBUS-DP uses the polling principle for communication using master-slave method. This means a DP -Slave station needs a master-request to exchange information.

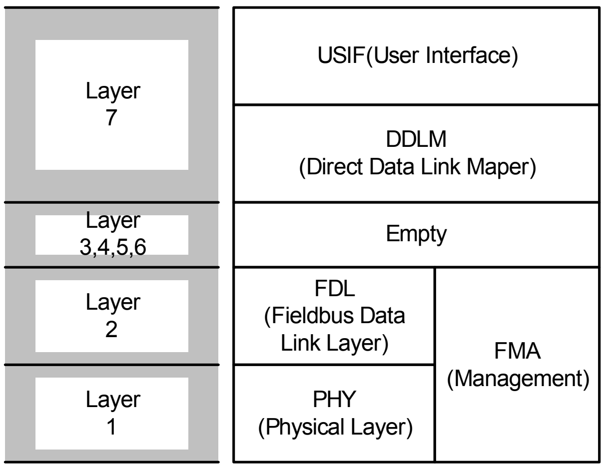

Figure 1. PROFIBUS-DP layer model

A message cycle mainly consists of a request frame from the active station followed by the corresponding acknowledge /response frame of the addressed station.

PROFIBUS-DP system consists of several communicating units such as DP-Master(Class 1,2) and DP-Slave. DP-Master(Class 1) controls several DP-Slaves according to a defined algorithm. This polls the associated slaves to submit data to its local user. The DP-Master(Class 1) can communicate to a DP-Master(Class 2) by a set of functions. DP-Master(Class 1) acts both as requester and responder. DP-Master(Class 2) is a management device. A set of functions supports management and diagnostics of a complex DP-System. DP-Slave implements a defined set of responder functions. (see Figure 2) PROFIBUS-DP offers the following basic functions: Master-Diagnostic, Parameter Up/Download, Activate/Deactivate Slaves, Slave Diagnostics, Set parameters of DP-Slave, Check configuration, Data exchange and Changing station address of a slave. These functions consist of a set of primitives such as request, indicate and confirm. The primitives are adapted to the PROFIBUS-DP protocol architecture.

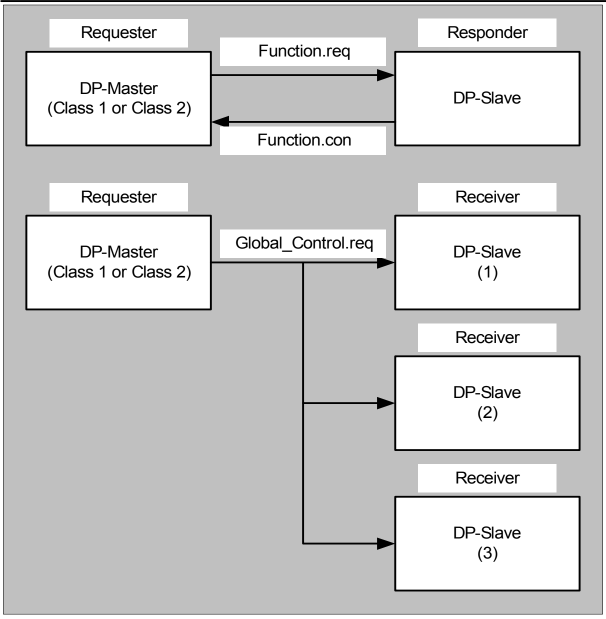

Figure 2. Master-Slave communication

3. COMPOSITION OF PROFIBUS-DP MASTER PROTOCOL SOFTWARE

The PROFIBUS-DP master protocol implemented in this study is operated on PC which has a PCI interface board for FDL(Fieldbus Data Link) layer[4]. Device driver which interfaces PC and FDL board was developed by using Windriver tool[5]. Device driver is implemented by integrating generated source codes for execution of Windriver to the DP master program codes. The software of PROFIBUS-DP master protocol stack developed in this study consists of DDLM(Direct-Data-Link-Mapper), USIF (User Interface) and USER(Graphic User Interface) (see Figure 3). The protocol stacks were implemented on the Window 2000 operating system by using Visual C++(API/MFC)[6].

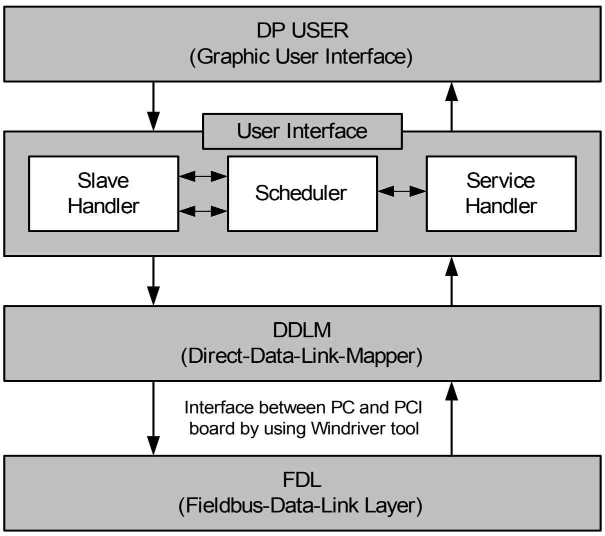

Figure 3. Structure of the PROFIBUS-DP master protocol

3.1 DDLM(Direct Data Link Mapper) layer

DDLM module provides a connection between the USIF and FDL layers. It follows the communication services which are defined in the PROFIBUS-DP specification. DDLM software module accomplishes the functions such as Slave-Diagnostic(DDLM_Slave_Diag), Set parameters of DP-Slave(DDLM_Set_Prm), Check configuration (DDLM_Chk_Cfg), and Data exchange(DDLM _Data_Exchange). Upper layer(USIF) and lower layer(FDL) are communicated using these functions of DDLM module.

3.2 USIF(User Interface) layer.

The USIF layer which exists between the USER and the DDLM analyzes the data that is sent by USER. It accomplishes the actual functions which control the operation of the PROFIBUS-DP system. The USIF module is composed of three functional blocks: Salve-Handler, Scheduler and Service-Handler.

3.2.1 Slave-Handler

The Slave-Handler block keeps a control to only one DP-Slave. If the PROFIBUS-DP system starts, the diagnostic request will descend from the USER(master). The DP-Slave set the default parameters, and recognizes the slave-station-address inside the PROFIBUS-DP system. After this act finishes, the data exchange between the master and slave will be accomplished.

3.2.2 Scheduler

The Scheduler block controls Slave-Handler block. When the PROFIBUS-DP system starts, the Scheduler block initializes and resets the DDLM and Slave-Handlers. It performs the error control inside PROFIBUS-DP system. It also analyzes the condition of the global control of PROFIBUS -DP system and delivers these contents to the USER. The Scheduler is operated on four modes: Offline, Stop, Clear, Operate.

3.2.3 Service-Handler

The Service-Handler block reads the bus parameters and the control parameters which are found in the operational condition of the Scheduler and transmits these parameters to the USER. It is mainly operated in Master-Master communication.

3.3 USER layer.

The USER layer is implemented as a GUI(Graphic User Interface) by demand of the system user's intention. It is able to approach both a master and slave modules, and the application which is interfaced by system user. The USER module provides I/O-window of the connected slave.

4. IMPLEMENTATION OF PROFIBUS-DP PROTOCOL

4.1 Implementation of DDLM layer.

DPRAM(Dual Port RAM) in FDL PCI board is used to interface between DDLM and FDL board, and PC memory is used to interface between DDLM and USIF modules. The DDLM module consists of transmitter part and receiver part. Transmitter part uses *Send_FDL()* function to deliver a frame to the DPRAM of FDL board. Receiver part uses *DDLM_InterruptHandlerRoutine()* function to indicates the condition of DPRAM interrupt, and *Read_Frame()* function to read a frame from DPRAM. After a FDL data frame is received, it is analyzed using *Analyze_Frame()* function and delivered to USIF module(see Figure 4)

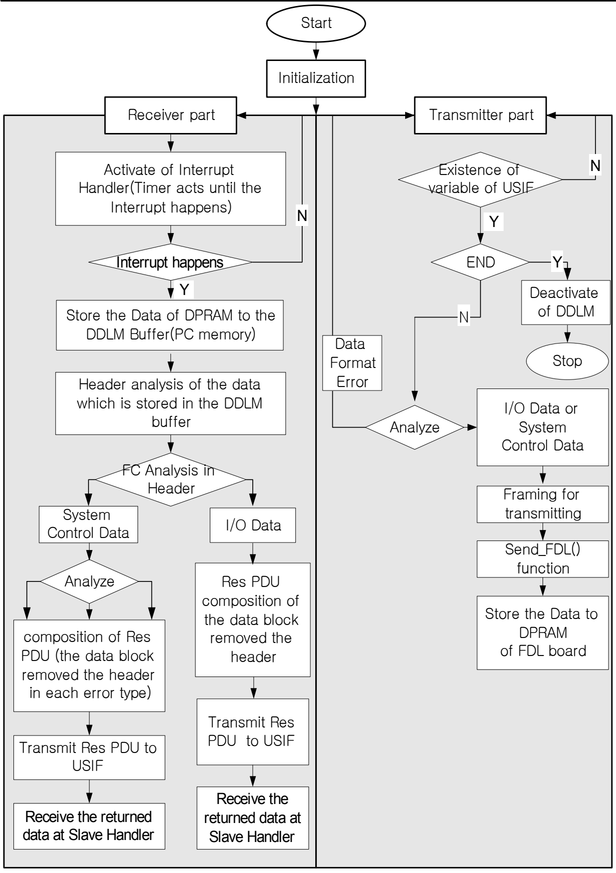

Figure 4. Execution of DDLM

4.2 Implementation of USIF

The USIF module is composed of three functional blocks: Slave-Handler, Scheduler and Service-Handler. These blocks are implemented in separate functions.

4.2.1 Implementation of Slave-Handler

The Slave-Handler block handles diagnostic request, parameterization and configuration of slave nodes(see Figure 5). To develop these three main processes, the parameters which are defined in the specification were declared with the structure body using API code. In this module, Bit-masking method is used to change declared structure.

4.2.2 Implementation of Scheduler

The Scheduler has four modes : Offline, Stop, Clear, Operation. They are used to control the *Start_SlvHdl(), Continue_SlvHdl()* and *Stop_SlvHdl()* functions of Slave -Handler. When three processes (Diagnostic, Parameterization and Configuration) are finished from the Slave-Handler, these information are returned to the USER module. After the initialization of the Slave-Buffer, Operation mode activates. The state information of the master and the slave are stored in each variable. USER(GUI) indicates the value by calling with window-message. (see Figure 6)

4.2.3 Implementation of Service-Handler

The Service-Handler is the module which mainly used for master-master communication. In the operation of master-slave communication, when the Scheduler acts the Operate mode, *Read_Scheduler_Status()* function defined in Service-Handler is executed. It reads the condition of the current scheduler and stores declared variable. The user calls and indicates the value of variable.

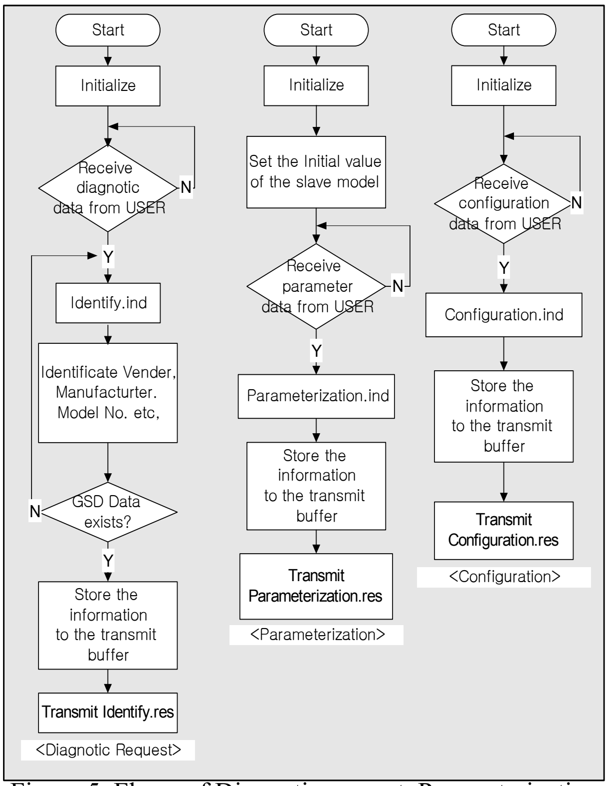

Figure 5. Flows of Diagnotic request, Parameterization and Configuration in Slave-Handler module

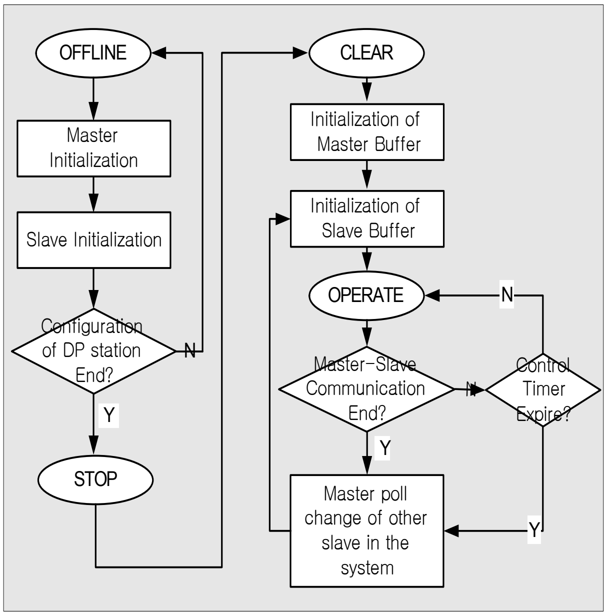

Figure 6. Algorithm of Scheduler module

4.3 Implementation of USER

The USER module exists above the protocol stacks defined in the PROFIBUS-DP specification. The structure of the USER program is composed of four parts. The USER developed by MDI(Multi-Dialogued base-Interface) in MFC. The menu bars in operation of USER program indicate the functions that are implemented: Initialize(*DP_Init()*), Reset (*DP_Reset()*), Data exchange(*DP_In_Slave(), DP_Out_Slave()*), System control data(*DP_Global_Ctrl()*) and Diagnostic data(*DP_Read_Diag()*). In USER module, it will be able to control the returned data from the USIF and the input data received from the GUI user. When it receives the data from the USIF, it stores the received data in the buffer. Using the data in the buffer, window-messages are composed and transmitted to each user window. Oppositely, if the data is entered from the GUI, the data is divided functionally. The PDU is delivered to the USIF when the initial connecting processes(Diagnostic, Parameterization and Configuration) are accomplished (see Figure 7).

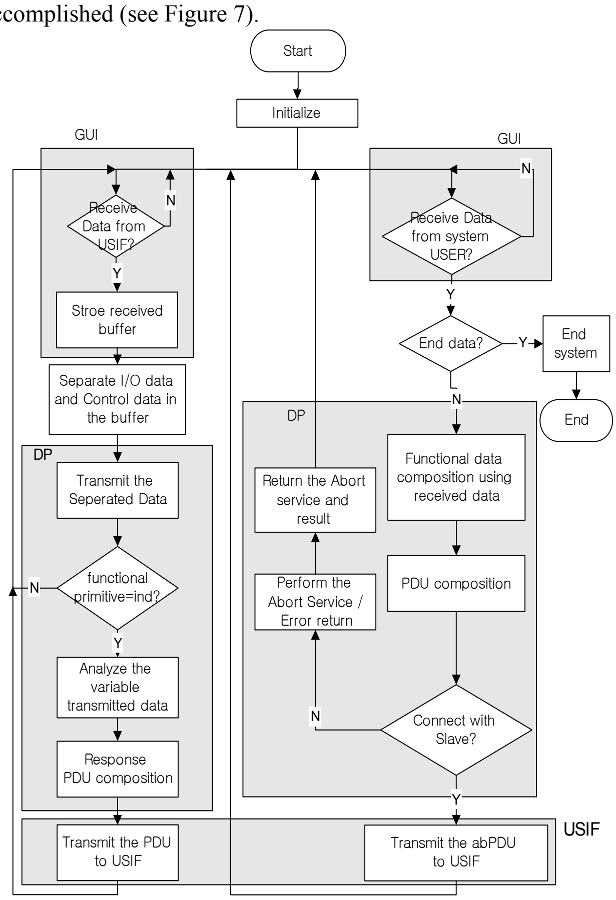

Figure 7. Flow algorithm of USER(GUI)

5. CONCLUSION

In this study, we implemented the PROFIBUS-DP master protocol. Using the developed protocol stack, we will develop the experimental model of the PROFIBUS-DP system that consists of DP master and slave nodes. Using the experimental model, we will investigate the effect of the change of DP network parameters on the performance of the PROFIBUS-DP system.

REFERENCES

[1] Standards of IEC 61158-1 to 61158-7 and 61784-1, 2003.
[2] DIN 19245 PROFIBUS Standards Part 1 and Part 3, 1993.
[3] Klaus Bender, "PROFIBUS-The Fieldbus for Industrial Automation, Munchen and Englewood Cliffs, NJ : Carl Haner Verlag and Prentice Hall, 1993.
[4] Jong Min Park, "Experimental Performance Evaluation of PROFIBUS-FDL and Implementation of PROFIBUS Master Board", Master Thesis, Hanyang Univ., 2002
[5] Windriver(Ver.4.32) User's manual, WindRiver Press, 1999
[6] Charles Petzold, "Programming Windows 5'th Edit, Microsoft Press, 1999.