International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 4, April - 2014

Fault Tolerance System in Automation Industry using Profibus Protocol

Saad Ahmed Shaikh
Electronics Department

Prof. S .P. Karmore
Computer department (GHRCE)

***Abstract*- The Profibus protocol is gaining great interest from the industrial community relate automation. In an industry it is very important to automate the fault tolerance methods for the task to be carried out without any interrupt. The Profibus protocol provides the gateway to implement such systems which will be self correcting. Thus without any human monitoring the fault in the system will be self rectified from the system. This paper provides idea of intelligent fault tolerance system design using Profibus protocol. The simulations of various transmitter blocks are shown with the idea of implementation.**

***Keywords*—FPGA and Profibus Fieldbus.**

I. INTRODUCTION

FOUNDATION fieldbus is an all-digital, serial, two-way communications system that serves as a Local Area Network for factory/plant instrumentation and control devices. The fieldbus environment is the base level group of the digital networks in the hierarchy of plant networks. FOUNDATION fieldbus is used in both process and manufacturing automation applications and has a built-in capability to distribute the control application across the network. Foundation fieldbus is neither owned by any individual company, nor regulated by a single nation or standards body. The technology is controlled by the fieldbus Foundation.

PROFIBUS PROTOCOL

The profibus equipment mode mainly consists of three blocks namely, Communication Interface, Object dictionary and application.
The profibus Medium Access Control (MAC) protocol uses a token passing procedure to bus access to masters, and a master-slave procedure used by masters to communicate with slaves.

II.COMMUNICATION MODE OF PROFIBUS

The communication interface defines four communication objects as configuration data CFC, process data object PDO, service data object SDO, and application data object ADO. Shown in figure1.

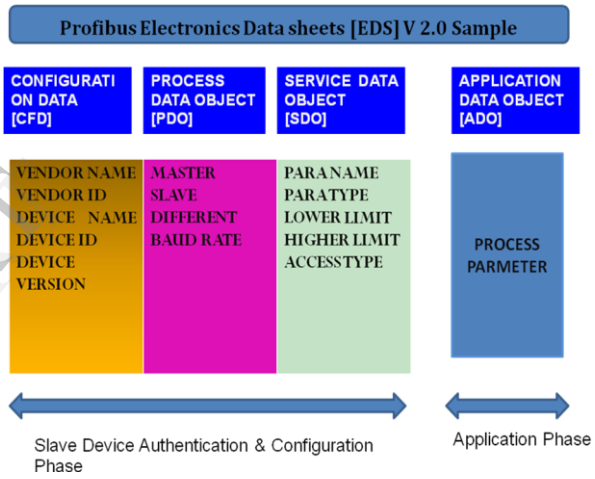

Figure1.

III. RELATED WORK ON FIELDBUS

Fieldbus become influencing term. There has been fieldbus technique developed for implementing any industrial plant. If company wants to implement a new slave device they will have to reconfigure the server setting, this will take a lot more time to implement a new device into the plant.

IV SYSTEM ARCHITECHTURE

In an application depending on motor in the industry when there is some fault in motor, the motor may stop and the working of units gets interrupted. Using the concept an un interrupted system can be designed. Two motors are to be connected. The speed and voltage of the main motor ie the first motor is to be monitored, while controlling a motor, we only need to set basic parameters such as current, speed, position, acceleration and so on. Whenever it is found that in spite of the proper applied voltage the speed of the motor is

not as per the specifications, then the second motor started. When the speed of both the motor match, The control from first motor gradually shifted to the second motor. This is done for the uninterrupted operation of the unit in the industry. This speed and voltage of motor is observed and compared from the object dictionary of the Profibus protocol. One of such system can be seen in paper. Shown in Figure2, and slave device application frame generation simulation result is shown in figure4.

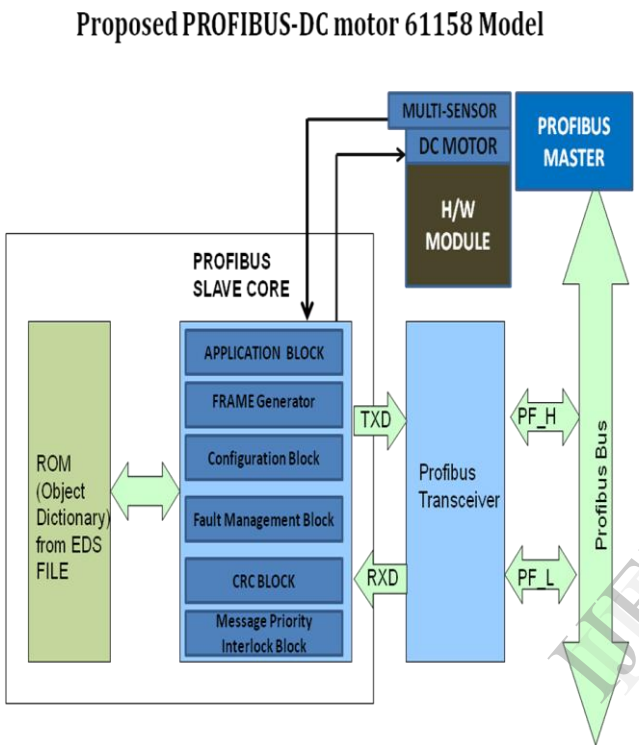

Fig 2 - System Architecture

V.PROFIBUS CONTROLLER

The block diagram of the Profibus controller consists of Transmitter and Receiver blocks. The transmitter parts mainly consist of the Transmitter buffer, Data/Remote Frame Generator, Parallel to Serial converter, CRC generator, bit stuff and serialized frame transmitter.

1. Transmitter Buffer

During the transmission of data, the data is first stored in the transmitter buffer. Transmitter buffer consist of ten 8bit registers. First two bytes contains the arbitration field and the control field. Then depending on DLC, the data bytes are stored in the buffer. The contents from this transmitter buffer are given to Data/Remote frame generator to construct the frame to be transmitted.

2. Data/Remote Frame Generator

The Data/Remote frames for transmission are generated in this block. The arbitration field, control field, and data are taken from the transmitter buffer and the Data/Remote frame is generated depending on the RTR bit. If RTR bit is recessive ie logic‘1’, the frame generated is the remote frame. In Remote frame the data bytes are not considered, whatever may be the value of DLC. Hence all the data bytes are represented as ‘1’ as shown in the simulation results. If RTR bit is dominant ie logic‘0’ then the frame generated is data frame. The number of data bytes depends on DLC. It can be from zero to eight bytes. Shown in figure3.

3. CRC Generator

The function of CRC generator block is to calculate CRC. The generator polynomial is used to calculate the CRC frame. This generator polynomial is called Profibus polynomial. The CRC input sequence from the serial to parallel converter, is divided by the generator polynomial. The CRC sequence transmitted in the message frame is nothing but the remainder of this polynomial division. The simulation result of the CRC generator.

4. Bit Stuff

Bit Stuff block performs stuffing of the string as per the specification in the profibus protocol. This makes the string suitable for transmission. As per the specification, the bit stuffing can be performed only on data and remote frames. In profibus protocol, if five consecutive bits are ‘0’ then the sixth bit is ‘1’ and if five consecutive bits are ‘0’ then the sixth bit is ‘1’.

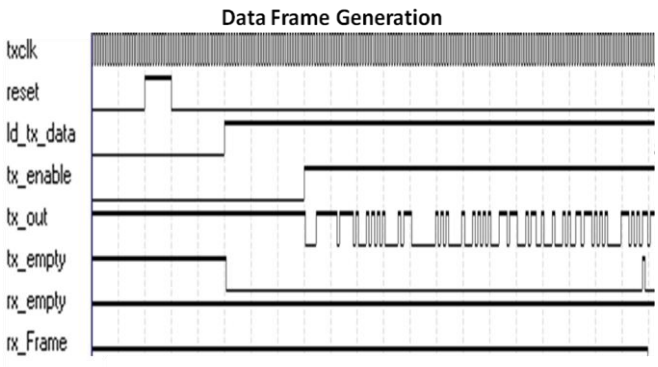

Figure3- Data Frame Generation

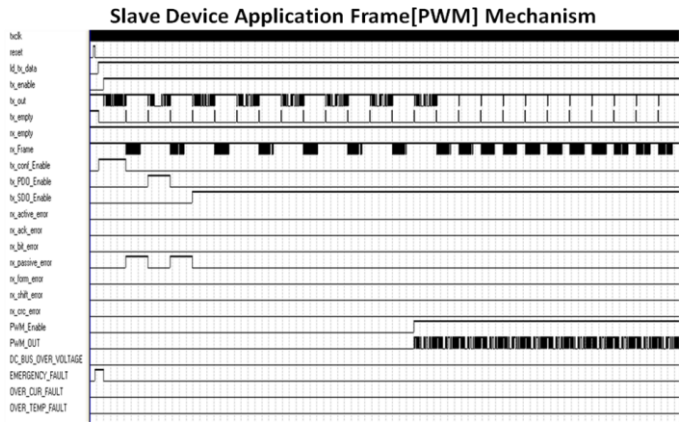

Figure4-Slave Device Application Frame

VI.CONCLUSION

The transmitter block of profibus for the stated application is designed and simulated. The results of the simulation are shown for the respective blocks. Similarly receiver block can be designed. The next approach will be for designing receiver block and then interfacing this profibus controller between the motor and the bus. The proposed work can be found beneficiary in any industrial plants to reduce the access time, cost and ease to implement.

VII.FUTURE WORK

The existing work will be further extended to design the receiver part. Thus the complete profibus controller can be seen as a trans-receiver. The controller will have a unique object dictionary which will have the definitions voltages, current, speed and torques of the various motors connected to the controller bus. Fault tolerance will be incorporated in the system with auxiliary motor (idle in the system). Depending upon fault, fault motor will communicate with the other controllers and will initiate the idle performance of motor into the circuit. Depending on the speed and voltage of the first motor second motor will be synchronized.

REFERENCES

1. Hui Hu, Ruixue Cui, Xuejie Wei, "VHDL Implementation of CAN Fieldbus Modeling Based on Petri nets," The Ninth International Conference on Electronic Measurement & Instruments, ICEMI'2009
2. Lu Yong, Haibin Yu, "Fieldbus interoperation technologies," In Proceeding of the 5th World Congress on Intelligent Control and Automation, 2004.
3. JACEK W S, "Embedded internet technology in process control devices," IEEE Factory Communication Systems , Porto, Portugal, 2000
4. Ferreira L,Tovar E, "Profibus protocol extensions for enabling intercell mobility in bridge-based hybrid wired/wireless networks," In Proceeding of the 5th IFAC International Conference on Fieldbus Systems and their Applications, 2003.
5. Huber, P., Jensen, K., Shapiro, R. M. Hierarchies in coloured
6. petrinets [A]. Rozenberg, G., (Ed.), Advances in Petri Nets[C].
7. Lu Yong, Haibin Yu, "Fieldbus interoperation technologies," In Proceeding of the 5th World Congress on Intelligent Control and Automation, 2004.
8. Guohun Zhu, Congmin Wang, Optimization VHDL Design With Petri Nets Model, Journal of System Simulation, Vol 15,pp. 83-84. Aug. 2003.