International Journal of Advance Research and Innovation
Vol. 3(2), Apr-Jun 2015, pp. 116-121
Doi: 10.51976/ijari.321526
www.gla.ac.in/journals/ijari

Article Info
Received: 05 Apr 2015 | Revised Submission: 30 Apr 2015 | Accepted: 20 May 2015 | Available Online: 15 Jun 2015

Implementation of Industrial Communication Protocal for Power Electronic Device

Sathya K and R. S. Geetha***

ABSTRACT

In this modern world, the need for optimizing the automation equipment is among the critical factors to consider. Normally the advancement of the drive control lies in the efficiency and operation range. But, nowadays the factor called "communication protocol" also comes into account for advanced control. There are many wired and wireless buses available in the market for various specific applications. But the majority of them are based onhigh speed Ethernet bus. This set up can be made economical by using the open source tools.This project deals with the deployment of the industrial communication protocol using open-source tools with the standards being maintained. The necessary libraries are designed and developed for the same as a proof of concept.In the present work, the Modbus protocol is considered for the open source evaluation.

Keywords: *IEDs (Intelligent Electronic Device), Modbus, SCADA (Supervisory Control & Data Acquisition).*

1.0 Introduction

The existing substation network has the SCADA (Supervisory control and Data acquisition), IEDs (Intelligent Electronic Device) and the communication protocols for protection and control of load. But, the cost of each SCADA and IEDs are very expensive, since they are all proprietary [10].The difference between Open source and Proprietary are as follows : Open source have Open license, Open Standard, more secure at less cost and Open Library. But, proprietary have Custom License, approved Standard by Certain Industries/Associations, less secure if Pirated and also Libraries are customized [10].

In order to avoid the purchase of the custom license at very high cost [10] and to maintain the high security, in this paper the same concept has been implemented using the open source tools.

As seen from the literature survey, the necessary structure for protection and control of load comprises of a numerical relay, SCADA (Supervisory control and Data acquisition) and the connecting bus (process bus). A typical Numerical Relay (NR) control system is based on the conventional closed loop control of drives, wherein, the critical output parameter is being sent back as feedback. The feedback in the present context is the current fed to the numerical relay. The relay decides upon the functionalities, by comparing with the threshold. Once any abnormal value is being detected, then the relay triggers the circuit breaker, which may take the necessary action on the load.

The relay then sends the value to the SCADAvia a bus [1] [2] [3].There prevails only one bus called the process bus, where the central controller (SCADA/HMI) and the IEDs (Intelligent Electronic Device) are in the same bus [1]. Due to the need for faster response and multi device connecting capability, the need for a high speed "station bus "is necessary. Hence the SCADA is connected with the IEDs through the station bus and the IEDs are in turn connected with process bus [1] [2].

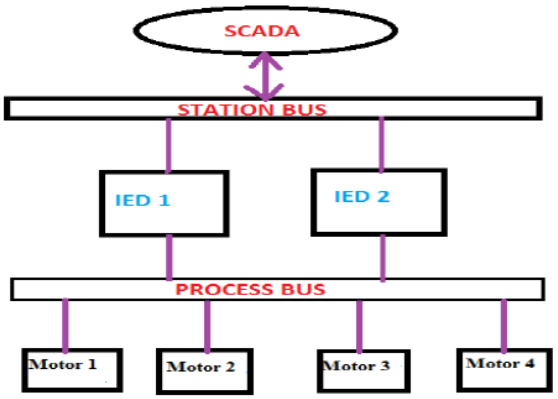
Thus the high speed data bus is used for protection and the process bus is used for the control of IEDs and Load.

The controlling device with a memory and intelligence is called as Intelligent Electronic Device (IEDs) [1]. Some of the examples are PMU (Phasor Measurement Unit), PLC (Programmable Logic Controller), NR (Numerical Relay),&AT (Automated Timer). The Process bus plays vital role in collecting sampled values of Pressure, Temperature, Voltage, Current from the Current Transformer, Potential Transformer etc., [2] [3] [4]. Sampled values

**Correspoinding Author: Department of Electrical & Electronics Engineering, BMS College of Engineering, Bangalore, India (E-mail: Sathya.k1991@gmail.com)*
***Department of Electrical & Electronics Engineering, BMS College of Engineering, Bangalore, India*

collected are given to IEDs. Sampled values from IEDs are transferred to the SCADA through Station bus, which is also the Protection bus [1][3][4].The setup is shown below in the Figure1.

Fig: 1. The SCADA Connected to IEDs Through Buses



In this paper, the Station bus protocol (Modbus Protocol) is considered for the open source evaluation, where, the sampled values from IEDs are transferred to the SCADA using the Modbus Industrial Communication Protocol, maintaining the Standards. In addition, libraries are designed and developed for validating the concept. The Modbus considered for open source evaluation is explained in section I.

2.0 Modbus and System Architecture

"Modbus" is the most important industrial standard communication protocol. The Modbus communication protocol can be used as serial and Ethernet standard for industrial control devices [5] [6] [7]. Serial buses can be derived from RS 232, which functions in full duplex mode or RS 485 that functions in half-duplex mode [8]. The Modbus TCP/IP works with the existing Ethernet stack. The communication mode is master-slave / server-client driven [8]. The desired mode can be selected, along with the serial port communication parameters such as baud rate, parity mode, etc. during the configuration of each controller [8]. The mode and serial parameters must be the same for all devices on a Modbus network. Each message must be transmitted in a continuous stream [8]. It has pre-defined function codes and registers for data binding and manipulation.

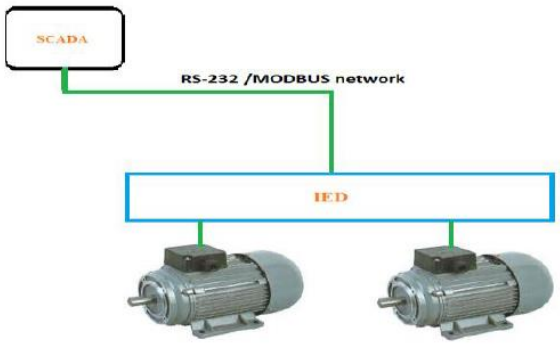
The star connection of Rs232 is not possible as it uses a single serial twisted pair cable, which facilitates the detection of a single slave/master. To have a connection with multi master/slave, then the MODBUS TCP/IP has to be used with Ethernet Bridge [9].

Some of the advantages of Modbus communication Protocols are [5]

- Many IEDs can be connected on one cable
- It has Flexible network topology
- Scalable real time
- High availability
- It is highly secured

The architecture of the system considered in the present work and the proposed methadology are briefly explained in the following paragraph.

Fig: 2. Architecture of the System



The system setup considered for study is shown in Fig 2. Here SCADA, which is the Master Device [3] [9] is connected to the Slave (IED).Generally more than one slave can be connected to the SCADA. The IED in turn controls the load connected to it. The instructions are obtained from the Master/SCADA to the IED, in which, the Master acts like monitoring and controlling device [3].

The simulations of the above system using Open Muc Framework and hardware implementation have been carried out. Detailed description is presented in section II and III respectively.

3.0 Simulation Using Open MUC Framework

In this work, simulation has been carried out using Open Muc framework in which, a dummy driver is considered. The real time values of dummy driver such as, voltage, current and their plot can be seen on the Monitor similar to that in SCADA system.

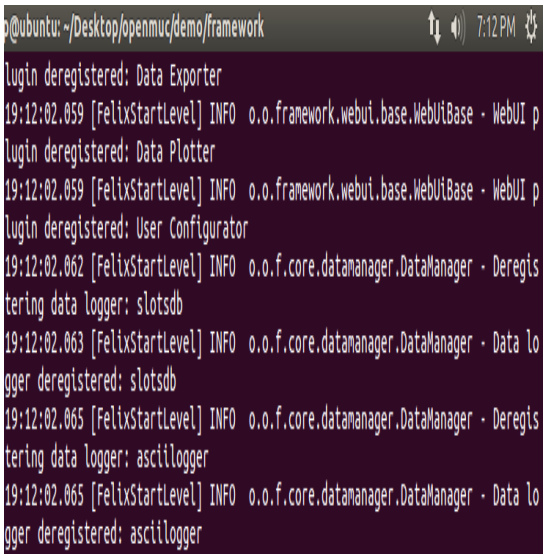
In the MUC framework, a Dummy driver is available that provides "dummy" data in form of a sine wave to the Open MUC core. Normally an open MUC driver implements a specific communication protocol (e.g. Modbus, IEC61850) [3][9] that can be used by the Open MUC Data Manager to get data from connected devices. The dummy driver does not

carry out any communication, but only returns dummy data.

The Dummy driver (also called as project) will be running inside the framework that also has different supporting files such as, Felix, Bundle and Conf that are used in configuring the dummy driver. Framework starts running with run-openmuc.sh and "Java –jar felix.jar" commands in the Linux Ubuntu platform.

The screenshot of the result obtained with these commands is presented in Fig 3.

Fig: 3.Output of Open MUC



The Dummy Driver provides some data which is read or written by the Data Manager. The App accesses this data via the Data Manager and triggers the Data Manager to write data to the Driver. To see the data that are actually sampled by the Data Manager, the following steps have to be followed.

- Run the Demo via run-openmuc.sh. and "Java –jar felix.jar". When the demo is running open browser and go to http://localhost:8888/openmuc to show Open MUC's web interface.
- Login with Username: admin and password: admin as shown in Fig 4.
- Click on the Data Plotter node that is available under Application menu as shown in Fig 5.
- Click on the live plotter tab to obtain the real time values of parameters such as, Voltage, Power, Current etc. as shown in Fig 6.
- Set Refresh to 1 sec as shown in Fig 7 in order to obtain the real time values every one second
- Select channels to plot the real time values of voltage and power as shown in Fig 6
- Click on Plot Data
- The real time plotting of parameters begin.

Fig: 4. Logging

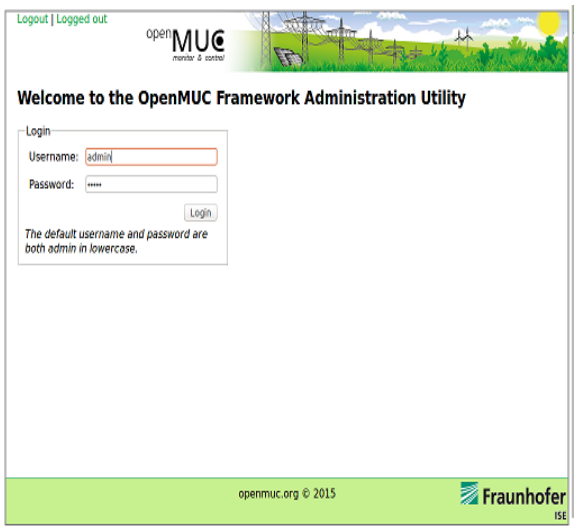


Fig: 5.Channel Configurator

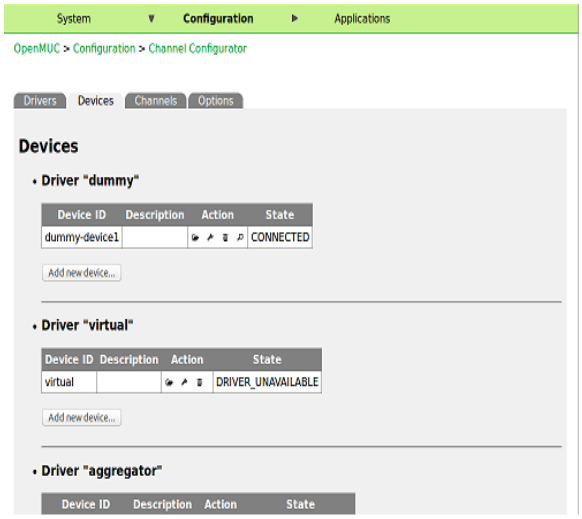


Fig: 6. Data Plotter

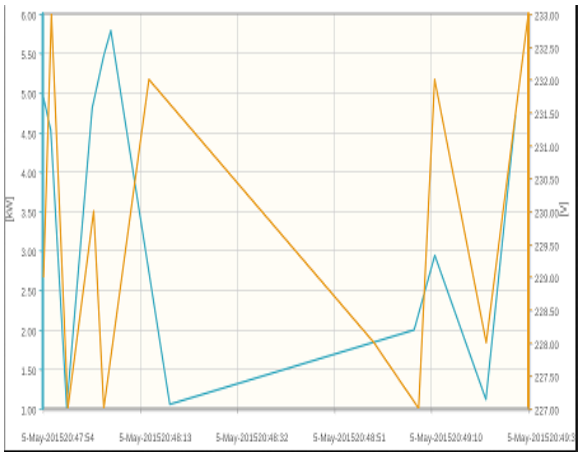


Fig: 7. Channel and Refresh

4.0 Hardware Implementation

The Hardware implementation has been carried out using the Raspberry Pi, RS 232, Pic controller and two Motors as Load. Raspberry Pi acts like SCADA, wherein, the Monitoring and Rectifying action have been carried out. Serial communication RS-232, is the type of MODBUS used as the Communication Protocol [5] [7], which is placed between Raspberry Pi and PIC for the effective communication. PIC controller is used as an IED, which has intelligence and memory in it.

The controlling action is taken care by the PIC controller. The instructions given to the PIC Controller is obtained from the Raspberry Pi and Motor load is connected to the PIC Controller. The hardware setup is as shown below in the Fig 8.

Fig: 8. Hardware Module

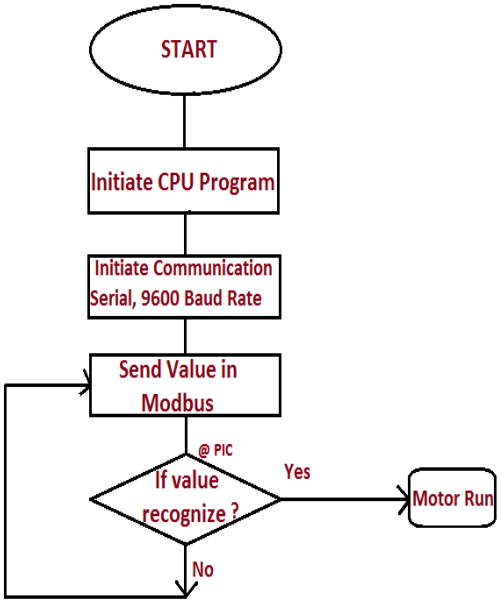
To validate the process of the communication between Raspberry PI acting like SCADA and PIC controller acting like IED ,during fault condition a manual fault has been applied by using a switch at the input terminals of the motor. The communication occurs such that, motor halts once the fault is recognized. PIC receives the message from Raspberry PI regarding the occurrence of the fault and hence turns off the motor. The fault condition has been implemented using a switch across the terminals of the motor.

In the system, the values presented in Table1are considered for validation of effective communication between Raspberry Pi and Pic controller to control the load.

Table: 1. List of values used in Hardware implementation

Values	Motor control
1	Front rotation
2	Left Rotation
3	Right Rotation
0	Halt

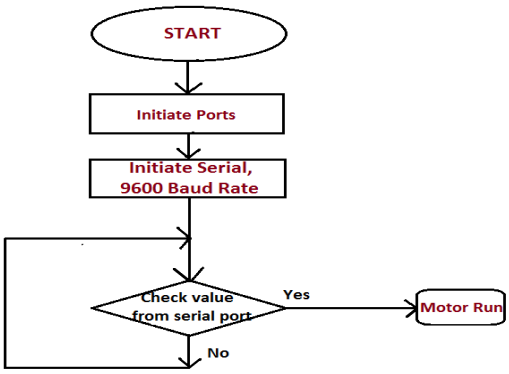
Fig: 9. Activation of Raspberry pi



The Raspberry Pi operating system gets activated with initiation of the Pic controller, which is nothing but the CPU of our Module. The initiation of the serial communication at 9600 Baud Rate sends the value from Raspberry pi to Pic controller through Modbus. Once the value is recognized by the PIC controller, the motor runs, if it is not recognized, then

the process of recognizing the value is repeated as shown in Fig 9.The Pic controller activation is briefly explained below.

Fig: 10. Activation of Pic controller



Pic controller starts with the initiation of Ports. The serial communication is initiated at 9600 Baud Rate. PIC checks the value from serial ports, if value obtained then Motor Runs else the process of recognizing the value is repeated.

The results obtained during hardware implementation are presented in section IV.

5.0 Results

As presented in section III, two motors (Right and Left) are considered for validation. The developed system is tested for rotation of motor in different directions as specified in Table 1. Fig: 11. Illustrates the Front rotation where the two motor rotates in the clockwise direction as shown in LCD display. The plot of the voltage, current and power can be obtained by providing GUI interface to the Raspberry PI, however this interface has not been implemented in the present work.

Fig: 11. Front rotation of Motors

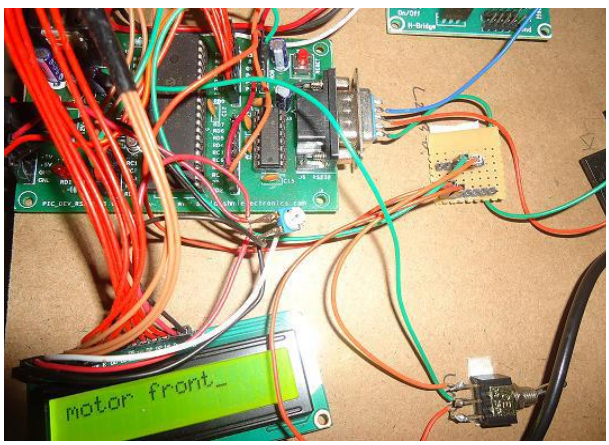
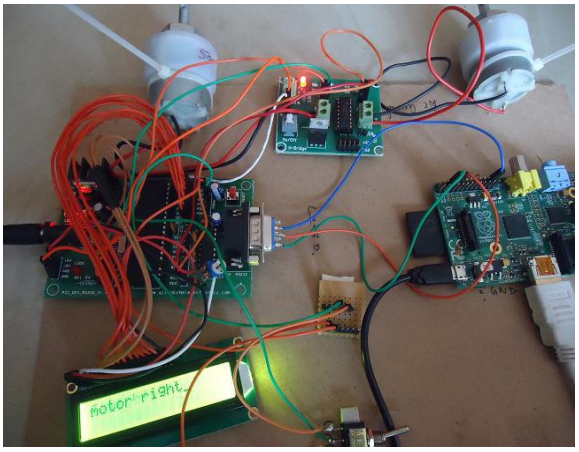
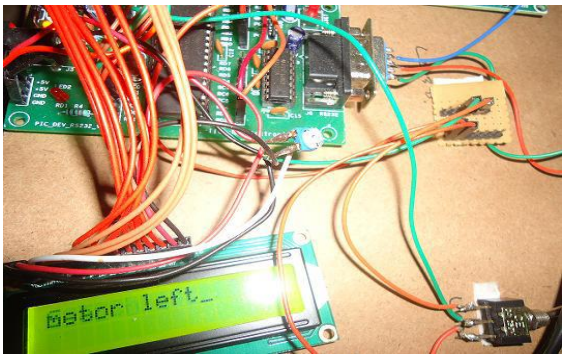


Fig: 12. Rotation of Right Motor



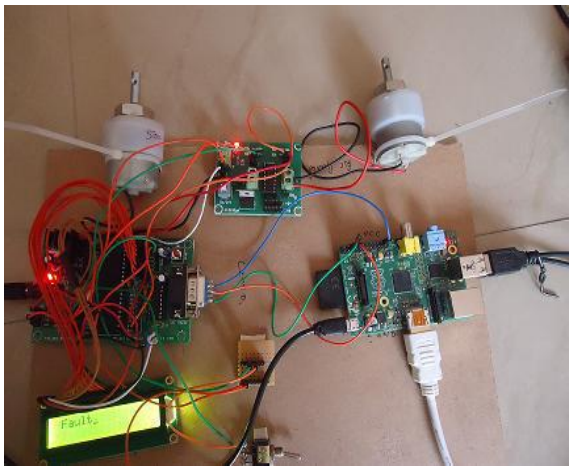
In the second case the commands are provided for Raspberry pi to rotate only the Right motor and the result is shown in Figure 12.

Fig: 13. Rotation of Left Motor



In the third case the command are provided for Raspberry pi to rotate only the Left motor and the result is shown in Figure 13.

Fig: 14. Fault to the Motor



In the fourth case the fault is considered, both the motors stop once it is recognized and the result is shown in Figure 14.

6.0 Conclusion

The SCADA and IED's role with effective communication is validated by the deployment of the Modbus which is the industrial communication protocol using open source tools wherein standard being maintained, libraries are designed and developed as a proof of concept. The cost can be effectively reduced with the high speed communication. The simulated and experimental results are obtained to verify the theoretical analysis.

Acknowledgment

This work is supported by the BMS College of Engineering, Bangalore. The authors would like to thank BMS College of Engineering for supporting this implementation by encouraging and supplying the necessary tools.

References

[1] An lEC61850-compliant distributed PMU for electrical substations, 2012

[2] A. Apostolov, B. Vandiver, IEC 61850 process bus - principles, Applications and benefits, Protective Relay Engineers, 63rd Annual Conference for, 2010

[3] Standards and committee drafts IEC 61850: Communication networks and systems in substations; http://www.scc-online.de/std/61850

[4] IEC Communication networks and systems for power utility automation, IEC 61850 Ed. 2, 2011.

[5] MODBUS Application Protocol Specification, http://www.MODBUS.org, 2002

[6] Wilson. Common Industrial Communications Protocols. The International Journal of Thermal Technology, digital edition, 2011

[7] IEEE Technical Report 1550 (1999): Utility Communications Architecture, UCA

[8] Integration Techniques of the Embedded Distributed Systems Using Programming Environments and Industrial Standard Communication Protocols, 2006

[9] Performance of an Industrial Data Communication Protocol on Ethernet Network, 2008

[10] Standard IEC 61850 for Substation Automation and Other Power System Applications, 2002