Implementing the User Defined Function to Configure Serial Parameters of the Modbus Based System

[1]Sneha Kunte, [2]J.A.Shaikh

[1,2]Electronics Engineering Department PVPIT Budhgaon

Abstract- Modbus Remote Monitoring and Control usually uses RS-485 and RS-232 transport for collecting data from Modbus slaves using established protocols. Of the two, Modbus RS-485 is more common due to its support for multi-drop communication. In this research the user defined function is analyzed, which can efficiently configure the serial parameters like buadrate, parity and slave address, of the any Modbus slave connected to modbus master in multi-drop network. With this function data switching work can be done in more flexible ways with configurable serial parameters. For better commercial usage of the modbus, the master should efficiently communicate and configure the any modbus slave attached to the multi-drop network. This user defined function is more fast and economical way to configure serial parameters than traditional way which consumes more time and money. It is encouraged the effective use of function in the multi-drop system platforms.

Index Terms- Modbus, Multi-drop networks.

I. Introduction

A short time ago, analog and purpose built communications systems use to be predominant technologies on industrial plants. It was not common to find either interoperability (Interoperability describes the extent to which systems and devices can exchange data, and interpret that shared data. For two systems to be interoperable, they must be able to exchange data and subsequently present that data such that it can be understood by a user) or compatibility among them. Later communication Networking began to be used in Direct Digital Control. Today MODBUS protocols are used in different industrial system widely, according to a study MODBUS was the world widespread protocol and involved on the control of lots of different kind of industries, including critical infrastructures. The protocol was primarily designed to lay over serial communication protocols so extensive usage of the modbus protocol was on the serial communication networks. Modbus is a protocol that traditionally uses serial communication lines. These serial lines connect the modbus master to modbus slave devices for collecting register and coil information. Modbus over traditional serial networks is still found in many industrial applications. Modbus Remote Monitoring and Control usually uses RS-485 and RS-232 transport for collecting data from Modbus slaves using established protocols. Of the two, Modbus RS-485 is more common than RS-232 due to its support for multi-drop (The term multi-drop describes an interface in which there are several receivers and one transmitter) communication. As being in multi drop network, the modbus master should efficiently communicate and configure the modbus slave attached to it. The good serial communication is determined by the various serial parameter that need to be set for appropriate communication between two nodes e.g. buadrate, slave address and parity. Here the discussed user defined function can efficiently help master to configure the serial parameters like buadrate, parity and slave address, of the any Modbus slave connected to that master in multi-drop network & data switching work can be done in more flexible ways with configurable serial parameters. A Modbus master can communicate with up to 247 Modbus Remote Telemetry Units (RTUs) or Intelligent Electronic Devices (IEDs). It uses a unique Modbus address assigned to each RTU. E.g. Modbus master want to configure the slave who is currently having slave address (0x09) to new slave address (0x05) this will be possible with the user defined function. This user defined function is more fast and economical way to configure serial parameters than traditional way which consumes more time and money, and also the proposed function will not disturb the communication flow means after the correct response is sent by the slave, the slave will restart its system and will get initialized with new serial parameters mentioned in query and further communication will happen according to the new serial parameters requested in the previous query.

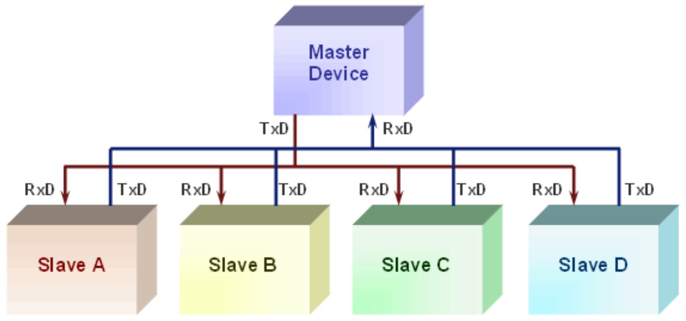

FIGURE 1: Multi-drop network

II. MODBUS PROTOCOL ANALYSIS

MODBUS is the most popular industrial protocol being used today, for good reasons. It is simple, inexpensive, universal and easy to use. The main reasons for the use of Modbus in the industrial environment are, it is developed with industrial applications in mind, openly published and royalty-free, easy to deploy and maintain, moves raw bits or words without placing many restrictions on vendors. Even though MODBUS has been around since the past century early 30 years almost all major industrial instrumentation and automation equipment vendors continue to support it in new products. Although new analyzers, flow meters and PLCs may have a wireless, Ethernet or field bus interface, MODBUS is still the protocol that most vendors choose to implement in new and old devices. Another advantage of MODBUS is that it can run over virtually all communication media, including twisted pair wires, wireless, fiber optics, Ethernet, telephone modems, cell phones and microwave. This means that a MODBUS connection can be established in a new or existing plant fairly easily. In fact, one growing application for MODBUS is providing digital communications in older plants, using existing twisted pair wiring. OSI reference model is a seven-layer's structure, which can support a strong network communication. When achieving communication between the workshop level in the industrial field, the designer should consider the following factors: First, to constitute a real open interconnection system, how to create and choose the proper network Communications reference model, whether Open Systems Interconnection model meets the conditions of the special status of industry or not, and simplifying it still meets the control network or not. According to different application areas, communication protocols were carried out to integrate and simplify, therefore, data can communicate only requires simple interface. According to the international OSI 7 layer network model, the standard Modbus protocol defines the communication physical layer, link layer and application layer. Physical Layer is the asynchronous serial communication standard of RS232 and RS485; Link Layer provides the number identification based station, master / slave mode of medium access control, Application Layer provides the information specification (or message format) and communication services; There are two communicate modes about Modbus protocol ASCII mode and RTU (Remote Terminal Unit) mode. ASCII mode is a byte of two ASCII characters to send, while RTU mode is in hexadecimal form of data, a byte is one frame, so efficiency of data transmission is more than ASCII mode, the majority of industrial controllers are using RTU mode. In the same network, regardless of the master or slave, must use the same communication patterns and the same transmission rate. At present, commonly transmission rate of Modbus protocol is 1200 bit/s ~19200 bit/s. During communications on a Modbus network, the protocol determines how each controller will know its device address, recognize a message addressed to it, determine the kind of action to be taken, and extract any data or other information contained in the message. Controllers communicate using a master/slave technique where only one device, the master, can initiate transactions or queries. The other devices, slaves, respond by supplying the requested data to the master or by taking the action requested in the query. Typical master devices include host processors and programming panels. Typical slaves include programmable controllers.

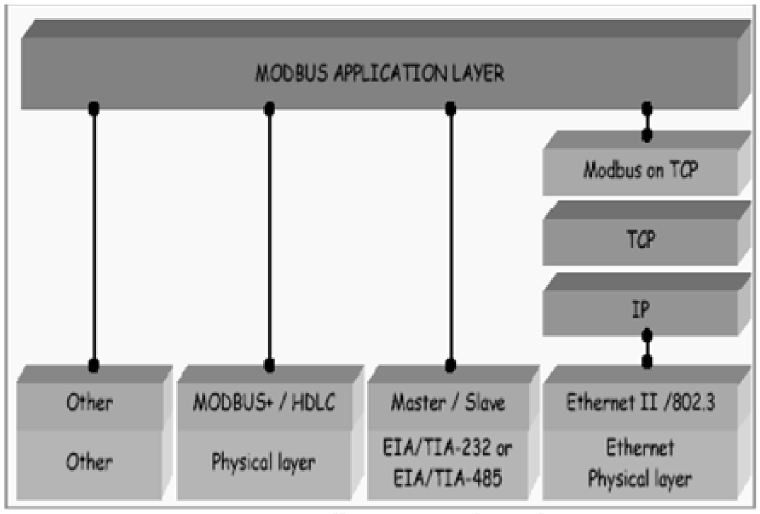

FIGURE2: Modbus Protocol Stack

Reading ASCII and RTU Modbus Devices via RS232, RS422, RS485 or TCP/IP

In Modbus systems, a master or client initiates queries to a slave or server (the measurement device). The slave/server responds either by supplying data or taking an action. Slaves only respond to queries from the master. The Modbus protocol defines two modes of transmission: ASCII and RTU (Remote Terminal Unit). Windmill supports both modes. In ASCII mode each 8-bit byte in a message is sent as two ASCII characters. It allows intervals of up to 1 second between characters, without causing an error. Messages start with a colon and end with a Carriage Return followed by a Linefeed. The advantages of ASCII mode is that it allows intervals of up to a second to occur between characters without causing an error. ASCII mode is only used over serial (RS232, RS422 and RS485) lines.

The RTU mode uses binary coding. Each 8-bit byte in a message contains two 4-bit hexadecimal characters. Greater character density allows better data throughput than ASCII for the same baud rate. Each message is transmitted in a continuous stream. The final part of a serial RTU message is a cyclic redundancy check, CRC. This calculates its value based on all earlier bytes in the message, it then adds its 2 bytes into the message. The computer therefore knows when it has received a corrupted message and can ask the instrument to resend its data. RTU mode is used over serial and network (TCP/IP) lines

III. Using MODBUS PROTOCOL in serial line

Modbus Serial protocol messages are transmitted between a master and slave devices over serial lines using the ASCII or RTU transmission modes.

The messages have three components:
(i) Slave address.
(ii) Modbus application protocol data unit (PDU).
(iii) An error checking field. (CRC or LRC)

The slave address in a request message identifies the recipient; the corresponding address in a response message

identifies the responding slave. A unicast message has an address in the [1, 247] range that identifies an individual slave. A broadcast message uses a slave address of zero. Values in the [248, 255] range are reserved addresses. The Modbus PDU has two fields, a one-byte function code and function parameters (maximum 252 bytes). The function code field in a request message specifies the operation requested by the master; the corresponding field in the response message is used to convey status information to the master (e.g., error information when an exception occurs in the slave device). The function parameters field contains data pertaining to functions invocation (request messages) or function results (response messages). Modbus function codes specify read and write operations on slaves, diagnostic functions and error conditions.
Modbus has three types of function codes:
A] Public codes,
B] User defined codes
C] Reserved codes.

Public codes correspond to functions whose semantics are completely defined in the Modbus standard. Valid public codes fall in the noncontiguous ranges: [1, 64], [73, 99] and [111, 127]. User-defined codes in the [65, 72] and [100, 110] ranges are not considered in the Modbus standard; their implementations are left to vendors. Reserved function codes are public codes that may be used to ensure compatibility with legacy systems. Function code values in the unused range [128, 255] indicate error conditions in response messages. Response messages have the same structure as request messages. The Modbus specification defines positive and negative responses to request messages. A positive response informs the master that the slave has successfully performed the requested action; in this case, the function code of the request message is included in the response message. A negative or exception response notifies the master that the transaction could not be performed by the addressed slave. The function code for a negative response is computed by adding 128 to the function code of the request message; thus, function codes in the [128, 255] range denote error conditions. A negative response also includes an exception code as a function parameter, which provides information about the cause of the error. The Modbus specification defines nine exception responses whose format and content depend on the issuing entity and the type of event producing the exception.

Reading ASCII and RTU Modbus Devices via RS232, RS422, RS485 or TCP/IP

In Modbus systems, a master or client initiates queries to a slave or server (the measurement device). The slave/server responds either by supplying data or taking an action. Slaves only respond to queries from the master. The Modbus protocol defines two modes of transmission: ASCII and RTU (Remote Terminal Unit). Windmill supports both modes. In ASCII mode each 8-bit byte in a message is sent as two ASCII characters. It allows intervals of up to 1 second between characters, without causing an error. Messages start with a colon and end with a Carriage Return followed by a Linefeed. The advantages of ASCII mode is that it allows intervals of up to a second to occur between characters without causing an error. ASCII mode is only used over serial (RS232, RS422 and RS485) lines.

The RTU mode uses binary coding. Each 8-bit byte in a message contains two 4-bit hexadecimal characters. Greater character density allows better data throughput than ASCII for the same baud rate. Each message is transmitted in a continuous stream. The final part of a serial RTU message is a cyclic redundancy check, CRC. This calculates its value based on all earlier bytes in the message, it then adds its 2 bytes into the message. The computer therefore knows when it has received a corrupted message and can ask the instrument to resend its data. RTU mode is used over serial and network (TCP/IP) lines

Modbus messages sent over Serial Lines: RS232, RS422 and RS485

Each message comprises four parts: device address, function code, data, and error check. The Device or Slave Address identifies your instrument. It contains one byte of information. In ASCII it is coded with two hexadecimal characters, in RTU with one byte. Valid addresses are between 0 and 247. The Function Code specifies the type of message. It contains one byte of information. In ASCII it is coded with two hexadecimal characters, in RTU with one byte.

Modbus RS-232 Allows Concurrent, Two-Way Flow of Data.

Modbus via RS-232 sends data in the form of time-series of bits. It is a standard for communication between data terminal and data circuit termination equipment. Transmission (Tx) and receipt (Rx) for data occurs on different circuits when using Modbus RS-232 lines. That means that data is able to flow both ways at the same time.

Modbus RS-485 Indicates Values Using Differences in Voltage.

RS-485 is similar, but distinct, from RS-232. The two wires, multipoint connection communicates data by indicating values by sending different voltages across the two wires. These differences between these voltages are related to one and zero values, which make up the Modbus RS-485 communications.

Modbus Com Port Settings

ASCII
Start Bit=1
Data Bits=7
If Parity is even or off then Stop Bits = 1
If Parity is none then Stop Bits = 2

RTU
Start Bit = 1
Data Bits=8,
If Parity is even or off then Stop Bits = 1
If Parity is none then Stop Bits = 2

4. Parsing User Defined function

Modbus function codes are in the range 1-127 (decimal), as 129(1+128)- 255(127+128) represents the range of error codes.

Public

Are guaranteed to be unique and specify well defined functions that are publicly documented. These are validated by the community and a conformance test exists.

User-Defined

Are available for user-defined functions, thus their codes might not be unique. The specification defines the code ranges 65-72 and 100-110 for user-defined functions.

Reserved

These are currently used by some companies for legacy products and are not available for public use (these are not discussed any further in the specification).

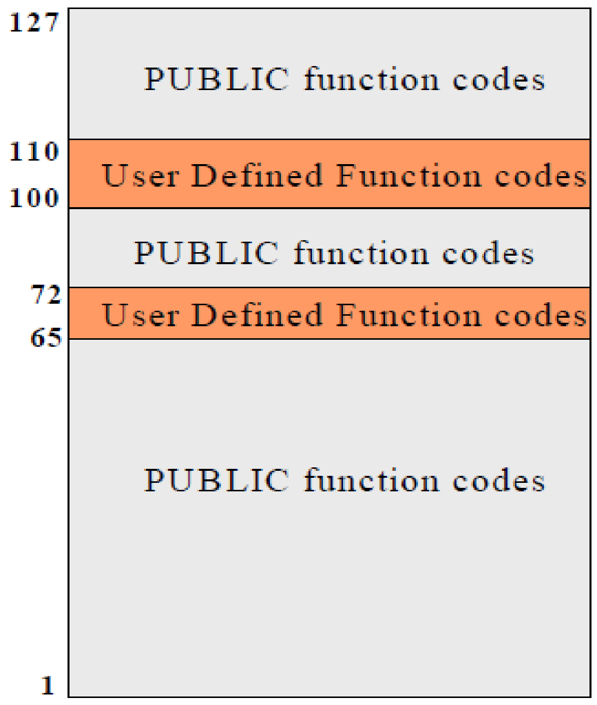

FIGURE 3: Modbus Function Code Categories

There are two ranges of user-defined function codes, i.e. 65 to 72 and from 100 to110 decimal. User can select and implement a function code that is not supported by the specification. There is no guarantee that the use of the selected function code will be unique if the user wants to re-position the functionality as a public function code, he must initiate an RFC (RESERVED FUNCTION CODE) to introduce the change into the public category and to have a new public function code assigned. MODBUS Organization, Inc expressly reserves the right to develop the proposed RFC.

This user defined function is intended to be similar as the Write Multiple holding register which writes particular values in particular values of the holding register. When Slave processes the query of this user defined function code, it sends the proper response to the requested query first then it restarts the system with new mentioned serial parameter.

E.g. If master wants to communicate with slave currently having the slave Id 0x09 and current buadrate 19200 (0x4B00) and parity none (0x00), but master wants to change this slave address to 0x05 with new buadrate 9600 (0x2580) and to new parity Odd (0x02). (None – 0x00, Even – 0x01, Odd-x02). Then Function Code is 100 (0x64) then PDU of Proposed User defined query will looks like:

REQUEST QUERY

SLAVE ID	0x09
FUNCTION CODE	0x64
NEW SLAVE ID	0x05
BUAD RATE HIGH	0x25
BUAD RATE LOW	0x80
NEW PARITY	0x02
CRC HIGH	0xE7
CRC LOW	0x98

RESPONSE Form Slave

SLAVE ID	0x09
FUNCTION CODE	0x64
NEW SLAVE ID	0x05
BUAD RATE HIGH	0x25
BUAD RATE LOW	0x80
NEW PARITY	0x02
CRC HIGH	0xE7
CRC LOW	0x98

After the correct response is sent by the slave, this slave will restart its system and will get initialized with new serial parameters mentioned in query and further communication will happen in that manner.

IV. CONCLUSION

MODBUS is popular serial communication protocol widely accepted over all Industrial sectors and enormously used in world. In Multi-Drop networks it is very necessary to manage network attributes within less time and with cost-effective manners. This user defined function is reliable approach, which can efficiently configure the serial parameters like buadrate, parity and slave address, of the any Modbus slave connected to Modbus master in multi-drop network & data switching work can be done in more flexible ways with configurable serial parameters and it is more fast and economical way to configure serial parameters than traditional way which consumes more time and money. For better commercial usage of the modbus it is encouraged the effective use of function in the multi-drop system platforms.

REFERENCES

[1] Thomas H. Morris, Bryan A. Jones, Rayford B. Vaughn, Yoginder S. Dandass ”Deterministic Intrusion Detection Rules for MODBUS Protocols” IEEE 2012.

[2] Javier Jiménez Díaz, Robert Vandenbrink “Using SNORT for intrusion detection in MODBUS TCP/IP communications” December 7th, 2011.

[3] Caswell, B. Bealeand, J., Foster, J. and Faircloth, J. “Snort2.0 Intrusion Detection,” Syngress, Feb. 2003.

[4] Peterson, D. "Quickdraw: Generating Security LogEvents for Legacy SCADA and Control System Devices," Conference for Homeland Security, 2009.CATCH '09. Cybersecurity Applications & Technology, pp.227-229, 3-4

[5] "SCADA Network Insecurity: Securing Critical Infrastructures through SCADA Security Exploitation" Giovanni A. Cagalaban, Yohwan So, Seoksoo Kim. 2009

[6] "The Use of Attack Trees in Assessing Vulnerabilities in SCADA Systems" Eric J. Byres, Matthew Franz and Darrin Miller 2005

AUTHORS

First Author – SNEHA KUNTE, BE Electronics, PVPIT Budhgaon. Skunte43@gmail.com.
Second Author – J.A.SHAIKH, ME Electronics, PVPIT Budhgaon, jubershaikh@rediffmaill.com.

Correspondence Author – SNEHA KUNTE, BE Electronics, PVPIT Budhgaon. Skunte43@gmail.com, 9960057384