IMPROVING WORK EFFICIENCY BY REDUCING INTERFACE NOISE GENERATION EXPERIENCED IN CONNECTING MODBUS PROTOCOL TO ARDUINO.

H Dasanayake, JAH Harshana, R Wijenayake, KVP Dhammika#

kvpdhammika@gmail.com

***Abstract**: Modbus is a standard open communication protocol which is extensively used in industry and is the most commonly available means of connecting industrial electronic devices in many industries today. The Modbus was developed by "Modicon" in 1979. It is typically used to transmit signals from instrumentation and control devices back to a main controller or a data gathering system. There are many versions of the Modbus protocol for serial lines (Modbus RTU and Modbus ASCII) and for Ethernet (Modbus TCP).*

In industrial automation, engineers use expensive devices to implement SCADA systems. Only mega industries can afford these systems due to the high cost. Mid-level industries use low cost Arduino Mega microcontrollers instead of very expensive devices as Modbus slave devices. One major technical barrier during usage is the Electro-Magnetic Interference (EMI) and noise generation during the connection of the Modbus master device with the Arduino micro-controller (slave device), and sensors with the slave device.

This research attempts to reduce the noise level to an acceptable level through specially designed hardware circuitry and software tools to improve performance. This allows to customize the Modbus slave device according to the industrial requirements where it can work efficiently with SCADA systems.

***Key Words:** Modbus, SCADA, Interface Noise, EMI and Protocol Conversion.*

1. INTRODUCTION

Industrial automation is the use of various control systems for operating equipment such as machinery, processes in factories, boilers and heat treating ovens, switching on telephone networks, steering and stabilization of ships, aircraft and other applications with minimal or reduced human intervention. Some processes have been completely automated [1].

The biggest benefit of automation is that it reduces labour involvement; however, it is also used to save energy and materials and to improve quality, accuracy and precision. With the advancement of technology, survival is the main concern. Challenges faced by many is the lack of sufficient capital in the medium scale industries. Concept of using Modbus protocol with the low cost Arduino module would be an ideal solution. One of the technical barriers experienced during the process of interfacing is noise .Hence reduction of interface noise may result in improvement of efficiency of the Arduino based industrial automated systems.

Modbus is an open source protocol [2], meaning that it's free for manufacturers to build into their equipment without having to pay loyalties. It has become a standard communications protocol in the industry, and is now the most commonly available means of connecting industrial electronic devices. It is widely used by many manufacturers. Modbus is typically used to transmit signals from instrumentation and control devices back to a main controller or data gathering system, for example: a system that measures temperature and humidity and communicates the results to a computer. Modbus is often used to connect a supervisory computer with a remote terminal unit (RTU) [3] in supervisory control and data acquisition (SCADA) [4] systems. Versions of the Modbus protocol exists for serial lines (Modbus RTU and Modbus ASCII) and for Ethernet (Modbus TCP).

An application layer messaging protocol at Level 7 of the OSI Model that provides client/server communication between devices connected on different types of buses or networks. The Modbus messaging structure belongs to that and was developed by "Modicon" in 1979.

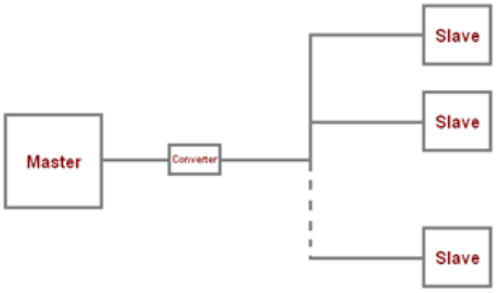

Fig 1. Master and Slave on Sample Modbus Network.

2. CONCEPT OF MODBUS

Different versions of Modbus are available today. From that Modbus RTU plays a major role in industrial automation field. RTU is based on serial communication like RS485 and RS232, Modbus ASCII and Modbus TCP, which is the Modbus RTU protocol embedded into TCIP packets. It's an open protocol, which means the specification is available free of charge for download, and there are no licensing fees required for using Modbus or Modbus TCP/IP protocols. MODBUS devices communicate using a master-slave technique in which only one device can initiate transactions (queries).

The other devices respond by supplying the requested data to the slave, or by taking the action requested in the query. A slave is any peripheral device (I/O transducer, valve, network drive, or other measuring device), which processes information and sends its' output to the master using MODBUS. Masters can address individual slaves, or can initiate a broadcast message to all slaves. Slaves return a response to all queries addressed to them individually, but do not respond to broadcast queries.

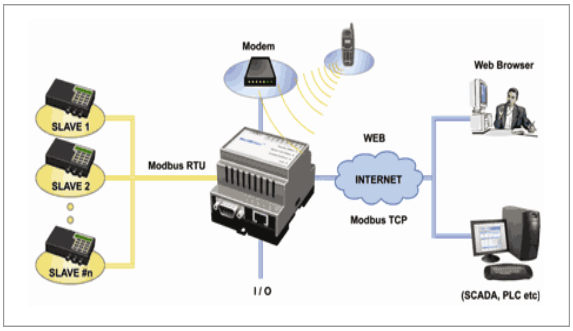

Fig 2. The complex diagram used in industry for Modbus data transmission

MODBUS functions operate on register map to monitor, configure, and control module I/O.

Register Name	Size	Details
Coil Status(0x)	1 bit	R and W
Input Status(1x)	1 bit	R
Input Register(2x)	1 word	R
Holding Register(3x)	1 word	R and W

R- Read W-Write

The transmission mode defines the bit contents of the message bytes transmitted along the network, and how the message information is to be packed into the message stream and decoded.

Standard MODBUS networks employ one of the two types of transmission modes:

1.1 ASCII Mode

1.2 RTU Mode

The mode of transmission is usually selected along with other serial port communication parameters (baud rate, parity, etc.) as part of the device configuration.

Modbus is open, common, relatively easy to use, and has few restrictions in its applications. However, it has its shortcomings. In many cases, a user has no choice but to use it because of hardware constraints such as existing sensors or other devices that must talk over Modbus. Whether user must use Modbus or has any other option, user should understand the benefits and restrictions inherent to the protocol.

Modbus is transmitted over serial lines between devices. The simplest setup would be a single

serial cable connecting the serial ports on two devices, the Master and the Slave.

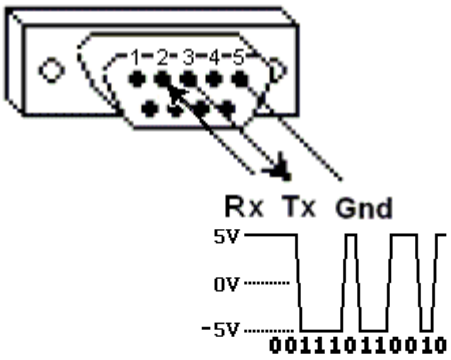

Fig 3. Serial data representation

The data is sent as series of ones and zeroes called bits. Each bit is sent as a voltage. Zeros are sent as positive voltages and ones as negative. The bits are sent very quickly. A typical transmission speed is 9600 bits per second (baud rate).

The function code field of the message frame will contain two characters (in ASCII mode), or 8 binary bits (in RTU Mode) that tell the slave what kind of action to take. Valid function codes are from 1-255, but not all codes will apply to a module and some codes are reserved for future use. MODBUS networks employ two methods of error checking: parity checking and frame checking [5].

When troubleshooting problems, it can be helpful to see the actual raw data being transmitted. Long strings of ones and zeroes are difficult to read, so the bits are combined and shown in hexadecimal. Each block of 4 bits is represented by one of the sixteen characters from 0 to F.

0000 = 0 0100 = 4 1000 = 8 1100 = C

0001 = 1 0101 = 5 1001 = 9 1101 = D

0010 = 2 0110 = 6 1010 = A 1110 = E

0011 = 3 0111 = 7 1011 = B 1111 = F

Each block of 8 bits (called a byte) is represented by one of the 256 character pairs from 00 to FF (Hexadecimal).

Request

This command is requesting the content of analog input register # 30009 from the slave device with address 17.

```
11 04 0008 0001 B298
```

11: The Slave Address (11 hex = address17)
04: The Function Code 4 (read Analog Input Registers)
0008: The Data Address of the first register requested.
(0008 hex = 8 , + 30001 offset = input register #30009)
0001: The total number of registers requested. (read 1 register)
B298: The CRC (cyclic redundancy check) for error checking.

Response

```
11 04 02 000A F8F4
```

11: The Slave Address (11 hex = address17)
04: The Function Code 4 (read Analog Input Registers)
02: The number of data bytes to follow (1 registers x 2 bytes each = 2 bytes)
000A: The contents of register 30009
F8F4: The CRC (cyclic redundancy check).

Fig 4. Example of "Request" and "Response" of Modbus master and slave

3. PROTOTYPE DEVELOPMENT

This research explains about implementation of Modbus protocol for medium scale industries by replacing Modbus slave as Arduino UNO. Also methods to protect the circuitry from EMI and reduce interface noise level to an acceptable level through specially designed hardware circuitry. In this prototype, industrially used "ADAM 4000" slave device was replaced by using Arduino UNO board which was comparatively very low in cost and it performed similarly.

Fig 5. ADAM 4000 Modules

Most of the industrial Modbus slave devices are specified for one register type at once. It's not economically feasible for medium scale industries those who deals with analogue, digital data. In this research four main types of registers were used in a single Modbus slave device. Therefore all the operations can be done using this device.

In this prototype, calibration of sensors and signal processing can be done inside the slave device. Therefore separate controllers or devices are not needed.

Arduino UNO can be programmed easily. Therefore customizing, maintaining and servicing can be easily done according to the customer's requirement.

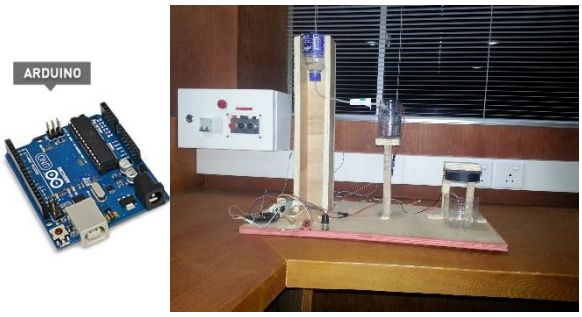

Fig 6. Prototype development for the beverage factory

The electromagnetic interference (EMI) is the level of conducted or radiated electrical noise sourced by the equipment. Conducted emission propagates along a cable or any interconnection line. Radiated emission propagates through free space.

It is the energy which causes undesirable response to any instrumentation and may be generated by power circuit switches, activation of inductive and resistive loads, sparking on motor brushes, relay activation, switches, circuit breaks, fluorescent lamps, heaters, automotive ignitions, atmosphere discharges and even electrostatic discharges between persons and equipment, microwave sets, mobile communication equipment, etc. All of this could cause over-voltage, under-voltage, peaks, transients, etc. that will cause an impact on a communication network as well as the embedded systems.

As a solution for the EMI analogue and digital signals were filtered and isolated using specially designed electronic circuitry.

Digital signals were isolated and filtered by using following opto coupler and Schmitt trigger based circuitry.

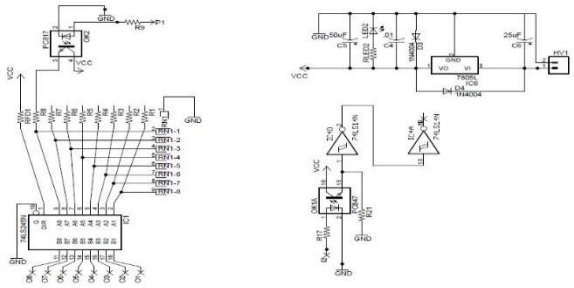

Fig 7. Isolation of digital Input and Output

Filtering of analogue signals were done by using software tools and combination of hardware and software tools.

Cascading together a single low pass filter circuit with a high pass filter circuit, band pass filter can be created. This filters pass a selected range or band of frequencies that can be either narrow or wide while attenuating all those outside of this range.

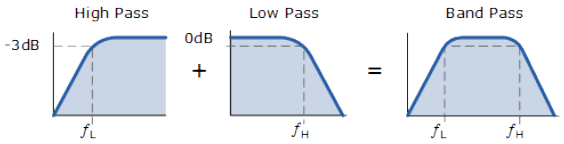

Fig 8. Bode plot of Band pass filter

In some environments, with large amounts of EMI, this type of band-pass filter can be very helpful to filter analogue signals. In this research pass band of the filter had been chosen as 132 MHz to 148 MHz

By using digitally tunable capacitors [10], band pass of the filter circuit can be customized or tuned. Therefore, digitally tunable band pass filter circuits [11] were designed.

These digitally tunable capacitors are made up of several high-Q-factor metal-insulator-metal (MIM) capacitors, which are digitally controlled FET switches.

The DTC used in the design of this research is the Peregrine PE64102, which has a capacitance range of 1.88 pF to 14.0 pF in steps of 391 fF, totaling 32 states that are controlled with 5 bits through a serial peripheral interface (SPI) bus.1 The Peregrine PE64102 works at 100 MHz to 3 GHz, and can handle up to 26 dB of RF power.

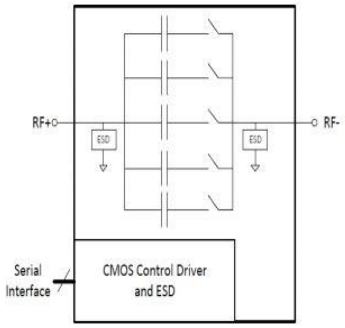

Figure 9. Digitally tunable capacitor

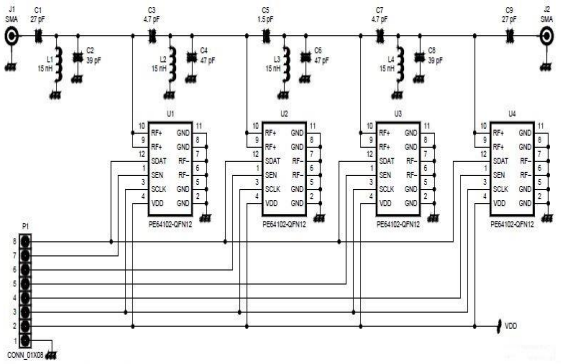

Figure 10. Schematic of Digitally tunable band pass filter circuit

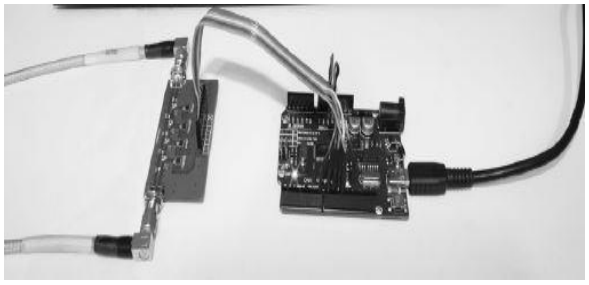

Figure 11. Digitally tunable band pass filter circuit connected to Arduino Uno.

Because of the Arduino programming language, software based filter technique was designed to help above.

To use this MODBUS protocol, beverage factory has been implemented as an example. Soft drink manufacturing starts with water, which is treated and cleansed to meet exacting quality-control standards, usually exceeding the quality of the local water supply. This process is critical in achieving high product quality and consistent taste profiles. As ingredients are being compounded, the treated water is filled into large, stainless-steel tanks. This is the stage at which various ingredients are added and mixed. Diet beverages are mixed with artificial, non-nutritive sweeteners such as aspartame or saccharin, whereas regularly sweetened drinks typically use liquid sugars like fructose or sucrose. It is during this stage of the production process that food colouring may be added. Flavoured, sparkling water get the desired flavour at this stage, while plain water are stored in the mixing tanks until the filling line calls for them.

In order to carbonate (absorption of carbon dioxide (CO_2)), soft drinks are cooled using large, ammonia-based refrigeration systems. This is what gives carbonated products their effervescence and texture. CO_2 is stored in a liquid state and piped into carbonation units as needed. This process can be manipulated to control the required rate of beverage absorption. Depending upon the product, soft drinks may contain from 15 to 75 psi of CO_2. Once carbonated, the product is ready to be dispensed into bottles and cans.

The filling room usually is separated from the rest of the facility, protecting open product from any possible contaminants. Again, the highly automated filling operation requires a minimal number of personnel. Filling room operators monitor the equipment for efficiency, adding bulk lids or caps to the capping operation as necessary. Empty bottles and cans are transported automatically to the filling machine via bulk material-handling equipment.

Water level sensor, Thermistor, 12V DC Motor & Peltier cooler are the devices which have been used for develop this factory structure. Using “Indusoft Web Studio v8.0” software developed the SCADA (Supervisory Control and Data Acquisition) architecture. Also to support medium scale industries open source SCADA like “Rapid SCADA” is used.

SCADA is a remote monitoring and controlling system which operates with coded signals over communication channels (using typically one communication channel per remote station). The

control system may be combined with a data acquisition system by adding the use of coded signals over communication channels to acquire information about the status of the remote equipment for display or for recording functions. It is a type of industrial control system (ICS). Industrial control systems are computer-based systems that monitor and control industrial processes that exist in the physical world.

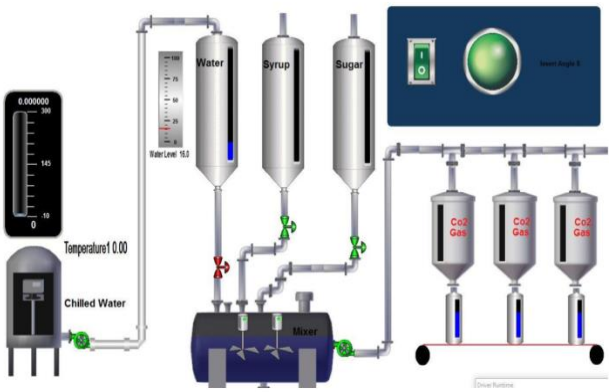

Fig 12. SCADA interface for the Prototype System.

6. CONCLUSION

The use of this MODBUS slave device is beneficial for industries those who find large scale investments not possible. Noise reduction techniques which was used in this research will help medium scale industries to work parallel and compete with mostly used Modbus slave devices in the industry.

REFERENCES

[1] Bennett, S. (1993). A History of Control Engineering 1930-1955. London: Peter Peregrinus Ltd. On behalf of the Institution of Electrical Engineers. ISBN 0-86341-280-7.

[2] Drury, Bill (2009). Control Techniques Drives and Controls Handbook (PDF) (2nd ed.). Institution of Engineering and Technology. pp. 508.

[3] Gordon R. Clarke, Deon Reynders, Edwin Wright, Practical modern SCADA protocols: DNP3, 60870.5 and related systems Newnes, 2004 ISBN 0-7506-5799-5 pages 19-21.

[4] Boys, Walt (18 August 2009). "Back to Basics: SCADA". Automation TV: Control Global - Control Design.

[5] Cf: Wendell ODOM, Ccie #1624, Cisco Official Cert Guide, Book 1, Chapter 3: Fundamentals of LANs, Page 74

[6] "A real-time digital signal processing (DSP) library for Arduino", http://playground.arduino.cc/Code/Filtersand 10 May 2016.

[7] "Arduino Tutorial: Simple High-pass, Band-pass and Band-stop Filtering"https://www.norwegiancreations.com/2016/03/arduino-tutorial-simple-high-pass-band-pass-and-band-stop-filtering/ 12 May 2016.

[8] "INTRODUCTION TO ELECTROMAGNETIC COMPATIBILITY (EMC) Analog Device "http://www.analog.com/media/en/training-seminars/tutorials/MT-095.pdf MT-095 TUTORIAL.

[9] http://www.ti.com/lit/an/sbaa195/sbaa195.pdf

[10] www.psemi.com/pdf/datasheets/pe64102ds.pdf

[11]www.chumalab.com.br/digitally-tunable-bandpass-filter