

SOFTWARE IMPLEMENTATION OF A REAL-TIME MONITORING AND CONTROLLING SYSTEM FOR THREE-PHASE INDUCTION MOTOR USING ZIGBEE AND IOT

Ali Husein Benhusein & Muhammed Fatih Kiliçaslan

Research Scholar, Department of Material Science and Engineering, Kastamonu University, Kastamonu, Turkey

Received: 24 Sep 2019

Accepted: 04 Oct 2019

Published: 31 Oct 2019

ABSTRACT

A Real time remote monitoring and controlling wireless system for three-phase induction motor using the Zigbee communication protocol and IOT is realized. The designed Graphics User Interface (GUI) installed in the main PC allows local or remote user to monitor the three-phase induction motor parameters and can be started/stopped locally using ZigBee protocol or remotely from anywhere around the world using IOT. This study complements the Hardware Implementation of a Real Time Monitoring and Controlling System for Three-Phase Induction Motor Using ZigBee and IOT study. A Low-level code for handling sensors data from the two Nodes to the main Node using ZigBee protocol is designed. Also, a GUI for the presents of motor parameters, Starts/Stops the motor, saves the motor parameters in Microsoft Access database file and Requests/Gets data and commands to/from the cloud is developed.

KEYWORDS: *Zigbee, Xbee, Three Phase Induction Motor, IOT*

INTRODUCTION

Choosing The term “Internet of Things” which is first time used by The British technology pioneer Kevin Ashton in 1999 for describing the system in which all the devices with sensors in the real world connected to each other. In the same year, Kevin Ashton, David Brock and Sanjay Sarma founded Auto-ID Labs which is the research-oriented successor to the MIT Auto-ID Center. They helped to develop the Electronic Product Code (EPC), a global RFID-based item identification system intended to replace the UPC bar code [1].

According to Cisco Internet Business Solutions Group (IBSG), The Internet of Things was born in between 2008 and 2009 at simply the point in time when more “things or objects” were connected to the Internet than people.

19xx–Present: A complete set of IOT platforms (Pachube, Thingspeak, etc.), protocols (6LoWPAN, Dash7, etc) hardware and software (Contiki, Tiny OS, Arduino. etc) have developed.

2011: The public launch of Internet protocol IPV6 – By using this protocol its will be possible to address a number of 2¹²⁸ device or as Steven Leibson said, “we could assign an IPV6 address to every atom on the surface of the earth, and still have enough addresses left to do another 100+ earths” [2] which can be used for addressing every device around us.

Nowadays, IOT devices are active research area and daily grows up.

Related Works

During the last years, significant efforts have been dedicated to the induction motors efficiency monitoring and many techniques have been proposed. Therefore, in this section a brief characterization of the main techniques used presented in the literature, also their advantages and disadvantages are presented.

Monitoring and Control System for Three Phase Induction Motor [3] was designed by M. P. Bodkhe, K. N. Pawar. They used IC ADE7758 for Poly Phase Multifunction Energy Metering and ZigBee Protocol. The ZigBee network topology was point to point topology, which means, the ZigBee network is limited to two nodes only. One ZigBee node for reading all motor parameters (voltage, current, number of rotates RBM and temperature of stator winding) and the other for base station node which is used as the system monitoring and controlling. The system is designed to work in limited areas "for local monitoring only".

Note: there are No buttons used for control choices showed in the GUI.

While, Mehmet FatihIşık, Mustafa Reşit Haboğlu, and Büşra Yartaşı [4] proposed the Energy Monitoring System for 3 Phase Induction Motors using smart phone. The proposed system used a PLC for controlling the motor.

The Pc named as CMT-SVR is connected to the plc using Ethernet connection, all the motor parameters are collected and processed in the CMT-SVR pc and available for accessing from any IOS/Android based mobile device. The connection between the CMT-SVR and the mobile devices is done by using wireless communication (Wi-Fi) by connecting them with an access point.

The authors used a frequency converter type A4022EE from speed controller series of Omron 3G3RX for controlling the speed of the motor.

The parameters monitored in the developed system were voltage, current, the motor speed and the frequency used to operate the motor. Also, the designed system allows the user to start/stop the motor and control the speed of the motor by changing the frequency of the power applied to the motor.

The system designed for work in limited area "for local monitoring only".

Mahendra P. Bodkhe, K. N. Pawar, [5] introduced a parameter monitoring system for three phase induction motor using ZigBee protocol. The developed system can monitor motor parameters such as voltage, current, winding temperature, the number of revaluations for the motor and start/stop it in emergency case.

The developed system consists of two sections, the first section contains a Pc where the monitoring software is, and the second section called Induction motor control circuit, it's consists of AT Mega microcontroller and sensors, Speed and other parameters of motor controller. The two sections connected to each other via wireless communication link using ZigBee protocol.

The system is designed to work in limited areas "for local monitoring only".

Geethi .P& V. Saravanan[6] did the same work using another technology. They used microchip solutions for building their system, the data acquisition system built using PIC microcontroller.

V.S.D Rekha, K. Srinivasa Ravi [7] developed a system for monitoring and controlling a single phase induction motor using IOT. The developed system uses a Raspberry Pi as data collection and gateway to the cloud. The Raspberry Pi connected to the cloud via wired Ethernet link.

All the sensors used in the system are connected directly to the inputs of raspberry Pi.

The sensor used in the developed system was current, voltage, temperature, vibration, moisture and speed sensor.

Software Implementation

The software part is used to implement the proposed system. The software used to implement the system is divided in two parts: Low level program and High level program.

Low Level Program

Means the group of instructions installed in the microcontroller for configuring the different pins in the microcontroller for handling and processing the signals from/to different units connected to it. This program also contains with appropriate code for enabling the Arduino board to send and receive the data and commands between the XBee nodes, extracts and forms them into useable data form. We programmed the Arduino Microcontroller using the open-source Arduino Software (IDE) compiler "Arduino.cc"[8].

Every Arduino board in our developed system is programmed to handle and process its sensor's data "sensors connected to it". We can explain every node's code in the following:

Node One Code

As we know, node one is designed for monitoring and controlling the power fed to the induction motor.

The monitoring process is belonging to the group of power sensors which are:

- Three single phase AC voltage sensors connected to the Arduino Leonardo analog ports A0 for Phase 1 voltage, A2 for phase 2 voltage and A4 for phase 3 voltage.
- Three AC current sensors connected to Arduino Leonardo analog ports A1 for phase 1 current, A3 for phase 2 current and A5 for phase 3 current.

All the Readings of these sensors are handled to the XBee module which is connected to Arduino Leonardo digital pin D0 "RX pin" and pin D1 "TX pin". At the same time, all these readings are showed using 20x4 LCD which is connected on SDA LCD pin to the Arduino Leonardo digital port D2 and SCL LCD pin to Arduino Leonardo digital port D3.

The controlling process belongs to the two Relays which used to control the three-phase contactor.

The two relays can be activated by applying logic 0 "0 volts" to their data input ports "Relay in pins" from Arduino Leonardo digital ports D5 and D6 by configuring these ports to work as digital outputs. Also, we can deactivate them by applying logic 1 "5 volts" to their data input ports from the same digital ports of Arduino Leonardo board.

The code developed and uploaded to the Arduino Leonardo board contains the appropriate instructions for converting electrical signals comes from different analog inputs (A0 to A5) to represent the actual physical readings values

which is detected by the voltage and current sensors, and finally transfers them to the main node in string form via RF signals using XBee module. The string form contains the following information: (phase 1 voltage "3 numbers of integer values / phase 1 current "3 numbers of Real values / phase 2 voltage "3 numbers of integer values / phase 2 current "3 numbers of Real values / phase 3 voltage "3 numbers of integer values / phase 3 current "3 numbers of Real values). Figure 1 shows a sample code for forming and sending sensors data from Node 1 to the Main Node.

Also, the developed code contains the group of instructions which can activate and deactivate the relays according to the special commands sent from the main node which received using RF via XBee module.

Node Two Code

Node two is designed for measuring the motor's number of rotates per minute and the stator temperature which is one of the methods used for discovering any damage occurs to the three-phase induction motor on early time.

We used Infrared obstacle avoidance sensor module for calculating the motor's number of rotates (RPM). We connected a digital pin D7 of the Arduino Leonardo board with "in" pin of the Infrared obstacle avoidance sensor module after configuring D7 pin in the Arduino Leonardo board as a digital Input.

We also connected lm 35 temperature sensor "in" pin to the analog pin A0 of the Arduino Leonardo board.

The number of rotates and the motor stator temperature is showed in the 16x4 LCD which connected in pin D2 and D3 in the Arduino Leonardo board. All the data collected from different ports will be sent via RF using an XBee module which is connected serially to the digital pin D0 and D1.

D0 "RX" pin of Arduino Leonardo board connected with Dout pin of the XBee module and D1 "TX" pin of Arduino Leonardo board connected with Din pin of XBee module.

The code developed and uploaded to the Arduino Leonardo board contains the appropriate instructions for calculating motor RPM, we used an Interrupt service Routine for calculating the Number of Rotates.

Every revolution generates Interrupt, we used interrupt number 4 configured to start in every falling edge of pin D7 input of Arduino Leonardo board "means in every change of input state from logic 1 to logic 0 of pin D7 the interrupt is starts". We used timer for calculating the amount of time between two interrupts "in milliseconds", the resulting number will be transferred to minute. The final presented value will be the average value of 10 readings to get the stable value.

```

1 // (LX)
2 // (LX)
3 // (LX)
4 // (LX)
5 // (LX)
6 // (LX)
7 // (LX)
8 // (LX)
9 // (LX)
10 // (LX)
11 // (LX)
12 // (LX)
13 // (LX)
14 // (LX)
15 // (LX)
16 // (LX)
17 // (LX)
18 // (LX)
19 // (LX)
20 // (LX)
21 // (LX)
22 // (LX)
23 // (LX)
24 // (LX)
25 // (LX)
26 // (LX)
27 // (LX)
28 // (LX)
29 // (LX)
30 // (LX)
31 // (LX)
32 // (LX)
33 // (LX)
34 // (LX)
35 // (LX)
36 // (LX)
37 // (LX)
38 // (LX)
39 // (LX)
40 // (LX)
41 // (LX)
42 // (LX)
43 // (LX)
44 // (LX)
45 // (LX)
46 // (LX)
47 // (LX)
48 // (LX)
49 // (LX)
50 // (LX)
51 // (LX)
52 // (LX)
53 // (LX)
54 // (LX)
55 // (LX)
56 // (LX)
57 // (LX)
58 // (LX)
59 // (LX)
60 // (LX)
61 // (LX)
62 // (LX)
63 // (LX)
64 // (LX)
65 // (LX)
66 // (LX)
67 // (LX)
68 // (LX)
69 // (LX)
70 // (LX)
71 // (LX)
72 // (LX)
73 // (LX)
74 // (LX)
75 // (LX)
76 // (LX)
77 // (LX)
78 // (LX)
79 // (LX)
80 // (LX)
81 // (LX)
82 // (LX)
83 // (LX)
84 // (LX)
85 // (LX)
86 // (LX)
87 // (LX)
88 // (LX)
89 // (LX)
90 // (LX)
91 // (LX)
92 // (LX)
93 // (LX)
94 // (LX)
95 // (LX)
96 // (LX)
97 // (LX)
98 // (LX)
99 // (LX)
100 // (LX)

```

Figure 1: Sample Code for Sending Sensors Data from Node 1 to the Main Node.

Also the code converting electrical signals comes from analog input (A0) to represent the actual physical readings values which is detected by the LM35 temperature sensor, and finally transfers them to the main node in string form via RF signals using the XBee module. The string form contains the following information: (temperature "4 numbers of Real value" / RPM "4 numbers of integer values). Figure 2 shows the sample code for forming and sending sensors data from NodeIto the Main Node.

The Main Node's Code

We programmed the Arduino Leonardo board for this node using Arduino IDE compiler for fetching data from the buffers of XBee nodes in string form which contains the data collected from different sensor data readings connected to each node and transfers it to the main PC as a string via serial port in the form (phase 1 voltage "3 numbers of integer values / phase 1 current "3 numbers of Real values / phase 2 voltage "3 numbers of integer values / phase 2 current "3 numbers of Real values / phase 3 voltage "3 numbers of integer values / phase 3 current "3 numbers of Real values / temperature "4 numbers of Real value" / RPM "4 numbers of integer values).

Also, the code includes with connectivity mechanism to make sure about the connection between each node with the main XBee node. Figure 3 shows the output data form sent to Pc from Main Arduino Leonardo board.

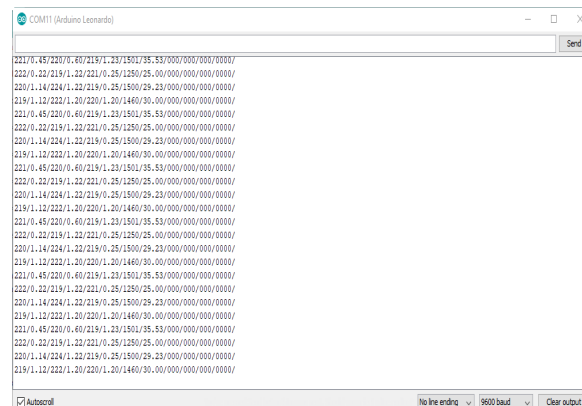


Figure 3: Output Data form Sent to PC from Main Arduino Leonardo Board.



Figure 2: Sample Code for Forming and Sending Sensors Data from Nodeto the Main Node.

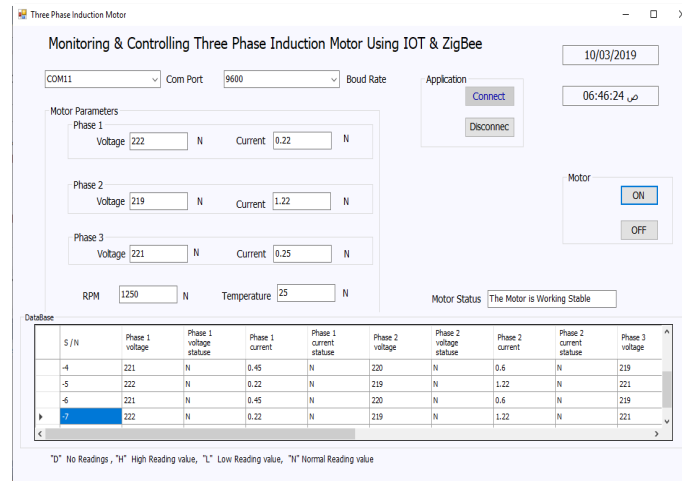


Figure 4: GUI Designed for Main PC.

High Level Programming

The High-Level Programming means the user interface used to monitor and control the system from different devices on real time manner.

As we mentioned before, all the data collected from the nodes will be sent to the main pc via COM port in string data form. We used visual studio 2017 community "Visual Basic. Net" for developing the graphic user interface, which is a free open source software[9].

We developed two GUI systems, the first GUI system developed to be installed in the local main pc which is connected directly with the main Arduino Leonardo board and configured to work as server who's responsible for providing the main services to the all designed system. The second GUI system is developed to be installed in remote PCs allowing their users for Real Time Remote monitoring and controlling "using IOT" for system administrators, or only real time remote monitoring for lower level users. Figure 4 shows the GUI Designed for Main PC.

The main PC designed GUI system can perform the following tasks:

- Configuring the serial port for receiving the string sent from Arduino Leonardo.
- Extracting the data string and performing it to the system on its original form.
- Viewing the extracted data in separate clear forms represents all motor parameters.
- Processing the extracted data by performing the necessary tests on it to show whether the received data in its normal values or not, before taking the appropriate action according to the test results.
- Allowing the user for starting or stopping the motor operation.
- Connecting the system with database file for saving the parameters of the motor for future use.
- Configuring the system to work with IOT application by connecting the system with the selected cloud to be ready for publishing/downloading the data to/from the cloud. Figure 5 shows the GUI Designed for Remote PC.

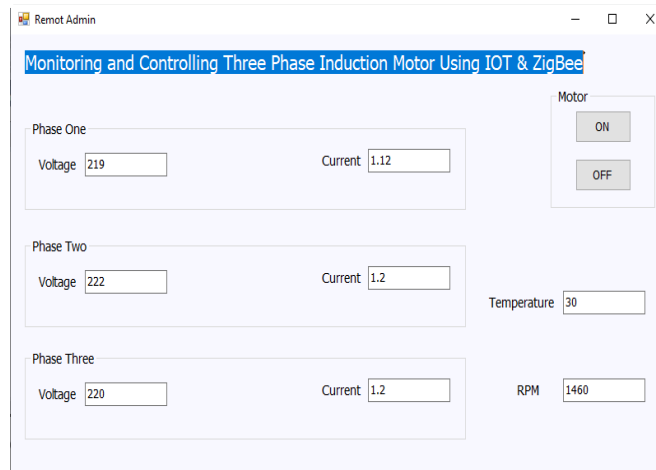


Figure 5: GUI Designed for Remote PC.

S#	Phase 1 vo	Phase 1 vo	Phase 1 cu	Phase 1 cu	Phase 2 vo	Phase 2 vo	Phase 2 cu	Phase 2 cu	Phase 3 vo	Phase 3 vo	Phase 3 cu	Phase 3 cu
3	219	N	1.12	N	222	N	1.20	N	220	N	1.20	N
4	220	N	1.14	N	224	N	1.22	N	219	N	1.23	N
5	221	N	0.45	N	220	N	0.60	N	219	N	1.23	N
6	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
7	221	N	0.45	N	220	N	0.60	N	219	N	1.23	N
8	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
9	220	N	1.14	N	224	N	1.22	N	219	N	0.25	N
10	221	N	0.45	N	220	N	0.60	N	219	N	1.23	N
11	220	N	1.14	N	224	N	1.22	N	219	N	0.25	N
12	219	N	1.12	N	222	N	1.20	N	220	N	1.20	N
13	221	N	0.45	N	220	N	0.60	N	219	N	1.23	N
14	220	N	1.14	N	224	N	1.22	N	219	N	0.25	N
15	219	N	1.12	N	222	N	1.20	N	220	N	1.20	N
16	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
17	220	N	1.14	N	224	N	1.22	N	219	N	0.25	N
18	221	N	0.45	N	220	N	0.60	N	219	N	1.23	N
19	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
20	219	N	1.12	N	222	N	1.20	N	220	N	1.20	N
21	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
22	220	N	1.14	N	224	N	1.22	N	219	N	0.25	N
23	221	N	0.45	N	220	N	0.60	N	219	N	1.23	N
24	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
25	219	N	1.12	N	222	N	1.20	N	220	N	1.20	N
26	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
27	220	N	1.14	N	224	N	1.22	N	219	N	0.25	N
28	221	N	0.45	N	220	N	0.60	N	219	N	1.23	N
29	222	N	0.22	N	219	N	1.22	N	221	N	0.25	N
30	219	N	1.12	N	222	N	1.20	N	220	N	1.20	N

Figure 6: Database File Designed using Microsoft Access.

We used Microsoft Access for developing a database file for saving a various readings of the motor parameters. figure 6 shows the database file designed using Microsoft Access.

Cloud Computing Services

Among the many types of cloud computing services delivered internally or by third party service providers, the most common are:

Software as a Service (SAAS)

Software runs on computers owned and managed by the SaaS provider versus installed and managed on user computers. The software is accessed over the public Internet and generally offered on a monthly or yearly subscription.

Infrastructure as a Service (IAAS)

Compute, storage, networking, and other elements (security, tools) are provided by the IaaS provider via public Internet, VPN, or dedicated network connection. Users own and manage operating systems, applications, and information running on the infrastructure and pay by usage.

Platform as a Service (PAAS)

All software and hardware required to build and operate cloud-based applications are provided by the PaaS provider via public Internet, VPN, or dedicated network connection. Users pay by use of the platform and control how applications are utilized throughout their lifecycle.

Cloud used in our IOT System

We used Thing Speak for our cloud computing which is a free web cloud service and provides Software as a Service (SaaS) cloud computing type, which enables the designers to collect and store their sensor's data in the cloud and build Internet of Things applications. Furthermore, its web service provides applications that let designers to analyze and visualize their data in MATLAB®, and then act on the data. Thing Speak can handle sensor data sent from computers, Arduino®, Raspberry Pi™, Beagle Bone Black, and other hardware[10].

In thing speak you can act with your sensor data using IOT by creating channel or more. Every channel can be used for presenting and acting with maximum upto eight sensors data "Eight fields". Each field is specialized for single sensor data.

In free account you can publish up to 3 million messages to/from thing speak cloud, when you arrive to your limit you must pay for cloud services in thing speak cloud.

To use Thing speak cloud you must create account in <https://www.mathworks.com>, after signing in thing speak cloud with your math works account you have to create a channel. After creating your channel, you will have the following information to use for publishing and reading data to/from your fields channel:

- **Channel ID:** It's a unique number assigned for you from Thinkspeak.com used for acting with your thing speak field's channel.
- **Read API key:** This key, preceded with channel ID, followed with field number, is used for reading data from thing speak fields of your channel.
- **Write API key:** This key, preceded with channel ID, followed with field number, is used for writing data from sensor to thing speak fields of your channel.
- **MQTT API key:** This key, preceded with channel ID, followed with field number, is used for publish/subscribe data to/from thing speak fields of your channel.



Name	Created	Updated
 Monitoring three phase induction motor <small>Induction motor, zigbee, IOT</small> Private Public Settings Sharing API Keys Data Import / Export	2018-12-26	2019-03-04 07:57
 motor status Private Public Settings Sharing API Keys Data Import / Export	2019-03-04	2019-03-05 05:09

Figure 7: Channels Created in Thingspeak.com.

You can also configure your channel viewing to be shared with everyone, shared with selected users or not shared "private view".

We created a two free channel in Thing speak for our IOT developed system their channel id is: **662590** (contains Eight fields named with "Monitoring three phase induction motor") and **718912** (contains Two fields named with "motor status"). Figure 7 shows the channels created in Thingspeak.com

Monitoring Three Phase Induction Motor Channel

This channel is created for posting, saving and monitoring real time motor parameters.

We activated eight fields on this channel. Every field assigned for single sensor data as following:

- Field 1 assigned for phase one voltage fed to the Induction motor.
- Field 2 assigned for phase one current fed to the Induction motor.
- Field 3 assigned for phase two voltage fed to the Induction motor.
- Field 4 assigned for phase two current fed to the Induction motor.
- Field 5 assigned for phase three voltage fed to the Induction motor.
- Field 6 assigned for phase three current fed to the Induction motor.
- Field 7 assigned for the Induction motor's stator Temperature.
- Field 8 assigned for the Induction motor's number of Rotates per minute.

The Data that presented in proposed channel which is created in Thingspeak.com is shown in figure 8.

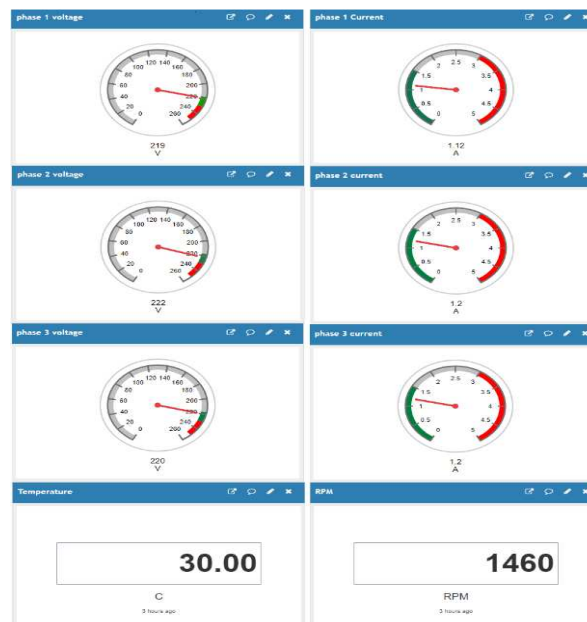


Figure 8: Data Presented in the Channel Created in Thingspeak.com.

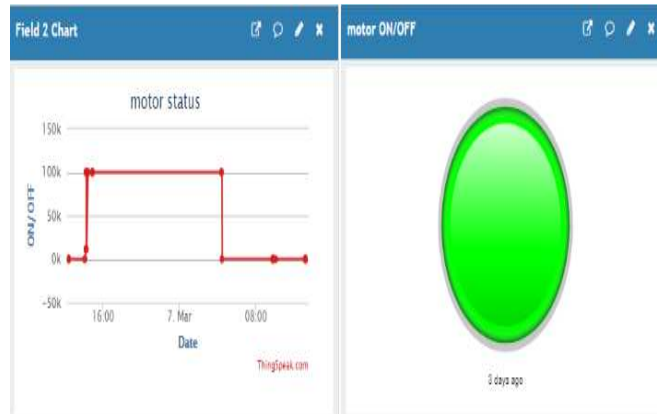


Figure 9: Data Presented in the Motor Status Channel Created in Thingspeak.com.

We activate two fields of the second channel assigned for wither the motor is OFF or ON.

The data that presented in proposed motor status Channel which created in Thingspeak.com is illustrated in figure 9.

For inserting the sensor data from the system to the cloud we used our API write key which is provided by thing speak cloud. And for downloading data saved in thing speak cloud to the system we used our API Read key which is also provided by thing speak cloud.

CONCLUSIONS

The proposed study is divided into two parts hardware and software. In the hardware part the requirements to implement the proposed system includes group of sensors like (speed sensor and temperature sensor) for reading motor temperature and it's velocity, current and power transformers for current and voltage measurement and control, and contactors and relays for turns the motor on and off. All data are retrieved from the sensors and the transformers have fed to Embedded Hardware-Arduino Leonardo and sent wirelessly to the main node via ZigBee protocol. In the Software part a group of low-level codes is used for handling data from deferent sensors to main XBee node and a high level program is designed for extracting, processing, saving, getting, requesting and publishing data about motor parameters to/from thingspeak.com cloud is designed. The Thingspeak, API which is an "Application Programming Interface" (API) and web service for the "Internet of Things" (IOT) for pushing data to cloud from the main Node is used.

REFERENCES

1. R. Buyya and A. V. Dastjerdi, *Internet of Things: Principles and paradigms*. Elsevier, 2016. Wandhare, M. K., & Porate, K. B. *IOT Based Induction Motor Parameter Monitoring And Controlling*.
2. "Internet of Things (IoT) History," *Postscapes*, Edited Aug, vol. <https://www.postscapes.com/internet-of-things-history>, Dec. 2018.
3. M. Bodkhe and K. Pawar, "Monitoring and Control System for Three Phase Induction Motor Using Poly Phase Multifunction Energy Metering IC ADE7758 and Zigbee Protocol."

4. M. F. Işık, M. R. Haboğlu, and B. Yartaşı, "Smart Phone Based Energy Monitoring System for 3 Phase Induction Motors," 2017.
5. M. P. Bodkhe and K. Pawar, "Parameter Monitoring Using Zigbee Protocol for Three Phase Induction Motor," *International Journal of Emerging Technology and Advanced Engineering*, vol. 4, no. 1, pp. 73–77, 2014.
6. P. Geetha and V. Saravanan, "Online Parameter Monitoring Of Induction Motor Using Wireless Network," *International Journal on Advanced Computer Theory and Engineering (IJACTE)*, vol. 1, no. 2, pp. 1–8, 2012.
7. V. Rekha and K. S. Ravi, "Induction Motor Condition Monitoring and Controlling Based on IoT," *International Journal of Electronics, Electrical and Computational System*, vol. 6, no. 9, pp. 74–89, 2015.
8. A. ARDUINO 1.xx The open-source Arduino Software (IDE), "<https://www.arduino.cc/en/Main/Software>," Jun. 2018.
9. S. 2018, "Visual Studio 2017 Community free download, Available:<https://visualstudio.microsoft.com/downloads/>, Sep. 2018.
10. A. get started with Thing speak, "<https://www.mathworks.com/help/thingspeak>," Nov. 2018.

AUTHOR PROFILE



Ali Husein Benhusein, was born in Tripoli-Libya 1971. He finished his Master of Science degree in mechatronics engineering Tripoli university. He started the Ph.D degree in Material Science and Engineering in Kastamonu University Turkey. His current researches lies on Zigbee, and real time systems.



Muhammed Fatih Kiliçaslan is associated professor in Kastamonu University. He has many scientific papers in Journals and conferences.

