IOT BASED INDUCTION MOTOR MONITORING SYSTEM

Submitted by

GIRISHWAR G	**19E650**
KABILAN G	**20E903**
MANOJKUMAR V	**20E905**
VINETH K.S	**20E908**

Dissertation submitted in partial fulfilment of the requirements of the degree of

BACHELOR OF ENGINEERING

ELECTRICAL AND ELECTRONICS ENGINEERING (SANDWICH)

May 2022

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

PSG COLLEGE OF TECHNOLOGY

(Autonomous Institution)

PSG INDUSTRIAL INSTITUITE

COIMBATORE – 641 004

PSG COLLEGE OF TECHNOLOGY

(Autonomous Institution)

COIMBATORE – 641 004

IOT BASED INDUCTION MOTOR MONITORING SYSTEM

Bonafide record of work done by

GIRISHWAR G	**19E650**
KABILAN G	**20E903**
MANOJKUMAR V	**20E905**
VINETH K.S	**20E908**

Dissertation submitted in partial fulfilment of the requirements of the degree of

BACHELOR OF ENGINEERING

ELECTRICAL AND ELECTRONICS ENGINEERING (SANDWICH)

…….	……
Dr.S. Karthik	**Dr.A. Soundarajan**
Faculty guide	Programme Coordinator

Certified that the candidate was examined in the viva voice examination held on

---

……	…..……..
(Internal Examiner)	(External Examiner)

ACKNOWLEDGEMENT

The completion of this project would not have been possible without the guidance and support of many people. We would like to express our sincere gratitude to **Dr.K.Prakasam**, the Principal, PSG College of Technology for providing us the necessary facilities for the project work.

We express our heartfelt gratitude to **Dr.J.Kanakaraj,** the Head of Department, Department of Electrical and Electronics Engineering, (sandwich) PSG College of Technology, for providing us the opportunity to carry out this project work and for her constant support and encouragement throughout the project.

We would like to thank our **Dr.D.Muralidhar**, Training Manager, PSG Industrial Institute, for his great support, motivation, and guidance.

We would like to thank **Dr.A.Soundarajan**, Professor, Programme Coordinator, Department of Electrical and Electronics Engineering, PSG College of Technology, for providing us with an opportunity to do this project and make use of the facilities.

Our sincere thanks to our tutor **Ms.K.Latha Maheswari**, Assistant Professor, Department of Electronics and Electronics Engineering, PSG College of Technology, for his constant support and guidance.

We extend our sincere thanks to **Dr.S.Karthik**, Assistant Professor, Department of Electrical and Electronics Engineering, PSG College of Technology, for sharing their knowledge and expertise and for guiding us throughout the project work.

We like to express our thanks to all teaching and non-teaching staff members in the Department of Electrical and Engineering, for their direct and indirect support

ABSTRACT

Rotating electrical machines are widely used in every manufacturing industry. Maintenance schedule and repair of AC motors are of utmost importance for industrial sectors. There has been considerable growth in methods of condition monitoring for motors and its predictive maintenance. In this paper recent technologies will be discussed where all the parameters like temperature, current, vibration & others are monitored wirelessly with the help of internet connectivity. This paper presents the review of various IOT based system used for data acquisition from sensors and its storage in cloud. The real time monitoring of motors is also done with graphical interface available in web server and API'S. The data stored in cloud as history can be used for making mathematical models which can predict the future faults in motors and in conjunction to that maintenance schedule can be generated. The review of various methods will help researchers in analysing available IOT & wireless based system in condition monitoring and failure prediction of AC rotating electrical machine

TABLE OF CONTENTS

LIST OF FIGURES

Page No

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Industrial IoT (IIoT) refers to the application of IoT technology in industrial settings, especially with respect to instrumentation and control of sensors and devices that engage cloud technologies. Recently, industries have used machine-to-machine communication to achieve wireless automation and control. But with the emergence of cloud and allied technologies (such as analytics and machine learning), industries can achieve a new automation layer and with it create new revenue and business models. IIoT is sometimes called the fourth wave of the industrial revolution, or Industry 4.0. Equipment maintenance is a critical aspect in industry. Traditional reactive maintenance only carries out maintenance activities after failure detection. Widespread preventive maintenance implies periodic maintenance activities based on previous experience about the periodicity of failure. Finally, predictive maintenance has arisen as an ideal approach for saving costs and preventing equipment failure in industry. In the Industry 4.0, failures are predicted based on real-time information received from sensors in industrial equipment. Predictive maintenance of industrial equipment has become a critical aspect in the Industry 4.0. Here the design, implementation and testing of an Industrial Internet of Things (IIoT) system designed to monitor electric motors in real-time. This system will be the basis for detection of operating anomalies and a future predictive maintenance system. The system has been designed using low-cost hardware components (wireless multi-sensor modules and single-board computer as gateway), open-source software and a free version of an IoT analytics service in the cloud, where all the relevant information is stored. The module gathers real-time data about the voltage and temperature of an electric motor. This approach is also the springboard to take advantage of edge and fog computing as a complement to cloud computing. The prototype has been tested in a laboratory. Real-time monitoring is one of the bases of the Industry4.0, and many systems have been developed to monitor currents, pressures, temperatures and other variables in industrial plants.

With the advances in micro-electromechanical systems, it is possible to deploy myriads of low-cost sensors capable of sensing, computing and communicating wirelessly to gather information for environment and equipment monitoring. These sensors are connected using wireless sensor networks. They send data to the cloud for storage or further processing using IoT protocols and technologies. Many of the public cloud service providers offer IoT services using standard protocols for real-time storage and extract the data. This makes it possible to use historical data to predict future failures of equipment. On occasions, the amount of data to be sent to the cloud or the latency of sending data to the cloud and back to the sensors/actuators is excessive. In these cases, moving part of the computation close to the sensors may alleviate the resources consumed in the network and the cloud.

Induction motors are complicated electro-mechanical machines that convert electrical energy into mechanical energy in most industrial applications. It acts as the work horse of manufacturing applications all over the world. These motors are tough devices that can be used in a variety of situations, including harsh conditions and dangerous places. Pumps, manufacturing equipment, centrifugal machines, machine tools, conveyors, presses and elevators are all popular induction motor applications. Petrochemical as well as natural gas plants, on the other hand, use induction motors in dangerous areas, while coal plant machinery, grain elevators and shredders use induction motors in harsh environments. Induction motors are also extremely dependable, minimal-maintenance, and possess a relatively high performance. Furthermore, induction motors' broad power range meets the output requirements of several commercial functions. The faults in motor are because of electrical as well as mechanical issues. Overloads and sudden load changes cause mechanical loads, which can lead to bearing failure and rotor bar rupture. Electrical strains, on the other hand, are normally linked to the supply of power. Induction motors can be powered by variable speed ac drives or power supplies of constant sinusoidal frequency. Induction motors, on the other hand, are more prone to failure when driven by ac drives. This occurs because of ac drives' increased voltage tension on windings of the stator, stator current elements of increased frequency and stimulated bearing currents. Furthermore, due to the cable connection length from the motor to AC drive, motor over voltage occur. Reflected wave transient voltages trigger this last effect. These electrical stresses can cause short circuits in the windings of stator, resulting in a complete motor failure.

The main goal is to improve the motor application's reliability by using current technological advancements. This project ensures that induction motors used in a number of industrial fields are continuously monitored and regulated. By maintaining system reliability, irregular conditions can be detected and corrected quickly. Since induction machines are utilized in virtually every industry, monitoring of economic data is essential. Industry efficiency can be improved by performing regular maintenance on induction machines.

This monitoring system is made up of various sensors as well as circuits that continuously track the motor's parameters. These sensors must be Arduino compliant, which means their resultant range of voltage is between 0- and 5-volts DC.

1.2 Details of Induction Motor Monitoring

The overall structure of the system hardware is shown in Fig. 1.1. This section presents overview about the parameter monitoring arrangement of three phase induction motor. A general block diagram of parameter monitoring system using IoT has been presented in this project. The whole arrangement is divided into following parts transmitter and receiver. Transmitter system consist of sensors, transducers and microcontroller which are used to acquire the parameters such as temperature, voltage, current, flow, float sensors for single phase induction motor located at distant location. Acquired parameters are then send to Arduino for display through LCD (Liquid Crystal Display) and transmitting end is IoT platform. Measured values are then compared with set values through Arduino cloud program and if in any case the measured value exceeds the set value of any parameter controlling signal will be generated by microcontroller to take proper control action such as to stop the motor, depending on the measured value of the parameter.

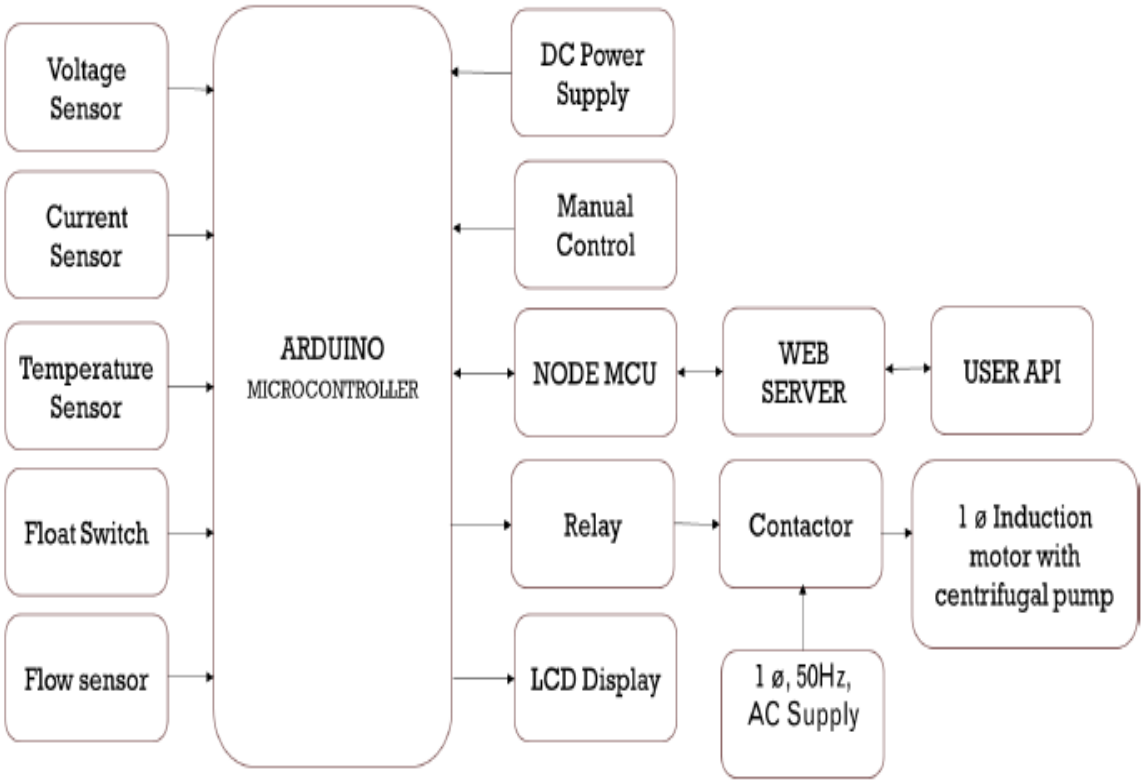


Figure 1.1 Block diagram

1.3 IoT Communication Architecture and Protocols

IoT protocols are an essential element of the IoT technology stack and without them, the hardware connected to it is useless. Because the IoT protocols only tell data exchange in a structured and meaningful way. Whatever IoT is today is due to its essential characteristics like communication between sensors, equipment, servers, gateways, and end user's applications. Although the IoT protocols enable this all smart stuff to talk and interact. Though the availability of present Internet communications is free and the biggest constraint for any IoT device even today is that it is too heavy and a power-consuming process for any *IoT* use case. The application layer works as an association between the user and sensor that is needed for the application. "Constrained Application Protocol (CoAP) was designed to translate the HTTP model so that it could be used in restrictive devices and network environments. CoAP relies on the User Datagram Protocol (UDP) for establishing secure communication between endpoints. Apart from transferring IoT data, CoAP leverages Datagram Transport Layer Security (DTLS) for the secure exchange of messages in the transport layer". CoAP is more efficient when used in arrangement with LoWPAN, a lighter version of the traditional HTTP protocol. "Message Queuing Telemetry Transport-(MQTT)" is a lightweight publication/subscription type messaging protocol. MQTT's design is simple and lightweight, and it is specially designed for battery-powered devices, providing small power consumption for devices. It works over TCP, and it cannot be used with all types of IoT applications. Moreover, it uses text for topic names, which increases its overhead. Once there is physical connectivity done, then with a unique method, all the devices can be separated from operating in their range. The network address has a critical role to identify each computer that is associated with the same router. Each alliance in IoT has its network address; for occurrence, ZigBee is one alliance with its network addresses. Like this, "BLE and Z-Wave" are having their network addresses according to their environments. "6LoWPAN (IPv6 Low Power Wireless Personal Area Network)" devices also function on IEEE 802.15.4, but their network stack is with IP connectivity (IPv6). With effective physical layer standards, we can achieve all this like more devices in a single environment and low power consumption, longer battery life, lower bandwidth consumption, the ability to connect smaller and lighter devices. "Near Field Communication (NFC), IEEE 802.15.1 (Bluetooth Low Energy (BLE) – Bluetooth 4.0), IEEE 802.15.4 (ZigBee, 6LoWPAN, Wireless HART, Mi-Wi)," etc., are the standards set by definite bodies such as proprietary vendors (Z-Wave by SIGMA DESIGNS) and IEEE (Institute of Electrical and Electronics Engineers).

IoT architecture is divided into five layers. The responsibilities of different layers in IoT architecture are:

Perception layer

The perception layer is like a physical layer, or we can also call it the device layer. On this layer, all sensors like RFID, infrared sensors. barcode is scanned and collect data from the sensor node to forward it to the network layer further.

Network layer

In IoT architecture, the network layer is also called as transmission layer. It works by sending the data received from the perception layer to the processing unit securely. The transmission standard can be wired and wireless to send data securely and this layer use 3G, wi-fi, Bluetooth, and Zigbee technology. Thus, the data are then forwarded to the middleware layer.

Middleware layer

One device of IoT connects to the other device only when both provide the same type of services. The middleware layer has the responsibility to connect it to the database by keeping in mind the service management. Further, computation and decision-making are also done by the same middleware layer.

Application layer

This layer comes up with the global management service. As per the device applications like it can be smart farming, smart health, smart home, intelligent transportation, smart city, etc.

Business layer

The business layer helps in creating a business model, graph, flowchart, etc., for different business models from the output received from this layer, which is useful in determining the further business strategies**.**

CHAPTER 2

HARDWARE DESCRIPTION

2.1 MICROCONTROLLER

A microcontroller (MCU for microcontroller unit) is a small computer on a single metal-oxide-semiconductor (MOS) integrated circuit chip. In modern terminology, it is similar to, but less sophisticated than, a system on a chip (SoC); a SoC may include a microcontroller as one of its components. A microcontroller contains one or more CPUs (processor cores) along with memory and programmable input/output peripherals. Program memory in the form of ferroelectric RAM, NOR flash or OTP ROM is also often included on chip, as well as a small amount of RAM. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general-purpose applications consisting of various discrete chips.

Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. By reducing the size and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes. Mixed signal microcontrollers are common, integrating analog components needed to control non-digital electronic systems. In the context of the internet of things, microcontrollers are an economical and popular means of data collection, sensing and actuating the physical world as edge devices.

Some microcontrollers may use four-bit words and operate at frequencies as low as 4 kHz, for low power consumption (single-digit milliwatts or microwatts). They generally have the ability to retain functionality while waiting for an event such as a button press or other interrupt; power consumption while sleeping (CPU clock and most peripherals off) may be just nanowatts, making many of them well suited for long lasting battery applications. Other microcontrollers may serve performance-critical roles, where they may need to act more like a digital signal processor (DSP), with higher clock speeds and power consumption.

2.2.1 ARDUINO UNO

Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller.

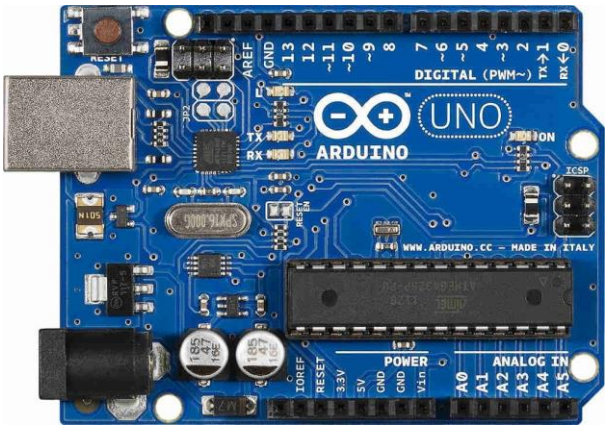


Figure 2.1 Arduino Uno Microcontroller

Summary:

Microcontroller	ATmega328P	
USB connector	USB-B	
Pins	Built-in LED Pin	13
	Digital I/O Pins	14
	Analog input pins	6
	PWM pins	6
Communication	UART	Yes
	I2C	Yes
	SPI	Yes
Power	I/O Voltage	5V
	Input voltage (nominal)	7-12V
	DC Current per I/O Pin	20 mA
	Power Supply Connector	Barrel Plug
Clock speed	Main Processor	ATmega328P 16 MHz
	USB-Serial Processor	ATmega16U2 16 MHz
Memory	ATmega328P	2KB SRAM, 32KB FLASH, 1KB EEPROM

Table No:1 Summary of Arduino UNO

General pin functions:

LED: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it is off.

VIN: The input voltage to the Arduino/Genuino board when it is using an external power source.

5V: This pin outputs a regulated 5V from the regulator on the board.

3V3: A 3.3-volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND: Ground pins.

IOREF: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates.

Reset: Typically used to add a reset button to shields that block the one on the board

Special pin functions

Serial / UART: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL serial chip.

External interrupts: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.

PWM (pulse-width modulation): pins 3, 5, 6, 9, 10, and 11. Can provide 8-bit PWM output with the analog Write () function.

SPI (Serial Peripheral Interface): pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). These pins support SPI communication using the SPI library.

TWI (two-wire interface) / I²C: pin SDA (A4) and pin SCL (A5). Support TWI communication using the Wire library.

AREF (analog reference): Reference voltage for the analog

Communication:

- The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX).
- The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required.
- Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board.
- The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1
- A Software Serial library allows serial communication on any of the Uno's digital pins.

Programming

The Arduino UNO can be programmed with the Arduino software. The ATmega-328P on the Arduino UNO comes pre-burned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header.

2.2 NODEMCU-ESP8266

NodeMCU is an open-source Lua based firmware and development board specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

The NodeMCU ESP8266 development board comes with the ESP-12E module containing the ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This microprocessor supports RTOS and operates at 80MHz to 160 MHz adjustable clock frequency. NodeMCU has 128 KB RAM and 4MB of Flash memory to store data and programs. Its high processing power with in-built Wi-Fi / Bluetooth and Deep Sleep Operating features make it ideal for IoT projects.

NodeMCU can be powered using a Micro USB jack and VIN pin (External Supply Pin). It supports UART, SPI, and I2C interface.

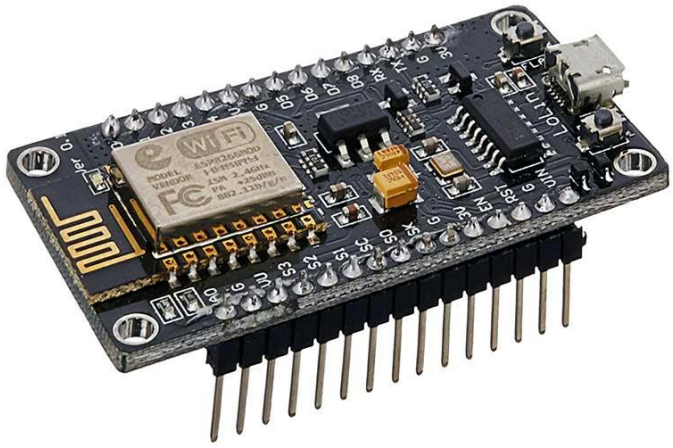


Figure 2.2 Esp8266 Nodemcu

NODEMCU ESP8266 Specifications & Features

Microcontroller: Ten silica 32-bit RISC CPU Xtensa LX106

Operating Voltage: 3.3V

Input Voltage: 7-12V

Digital I/O Pins (DIO): 16

Analog Input Pins (ADC): 1

UARTs: 1

SPIs: 1

I2Cs: 1

Flash Memory: 4 MB

SRAM: 64 KB

Clock Speed: 80 MHz

USB-TTL based on CP2102 is included onboard, Enabling Plug n Play

PCB Antenna

Small Sized module to fit smartly inside your IoT projects

Pin Descriptions

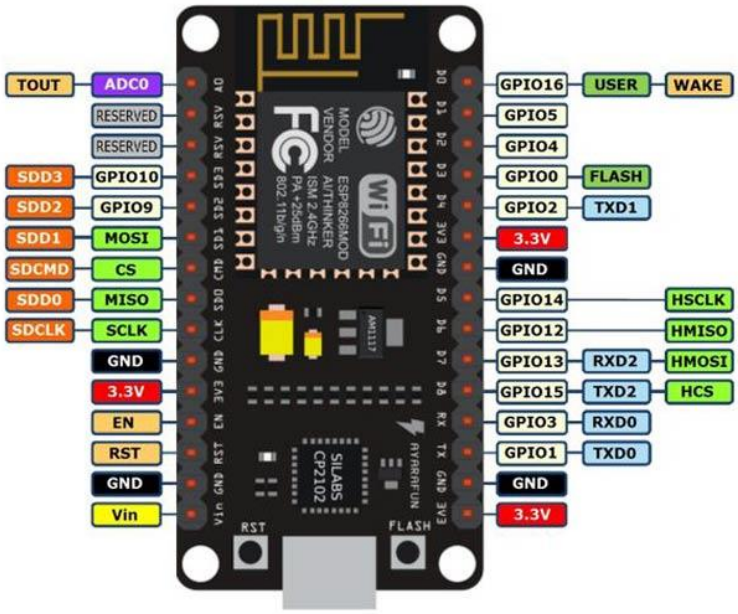


Figure 2.3 Esp8266 Nodemcu Pinout

Pin Category	Name	Description
Power	Micro-USB, 3.3V, GND, Vin	Micro-USB: NodeMCU can be powered through the USB port
		3.3V: Regulated 3.3V can be supplied to this pin to power the board
		GND: Ground pins
		Vin: External Power Supply
Control Pins	EN, RST	The pin and the button reset the microcontroller
Analog Pin	A0	Used to measure analog voltage in the range of 0-3.3V
GPIO Pins	GPIO1 to GPIO16	NodeMCU has 16 general purpose input-output pins on its board
SPI Pins	SD1, CMD, SD0, CLK	NodeMCU has four pins available for SPI communication.
UART Pins	TXD0, RXD0, TXD2, RXD2	NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program.
I2C Pins		NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C.

Table No:2 Pin Description Of NODEMCU

2.3 SENSORS:

2.3.1 Voltage Sensor - ZMPT101B

Description

ZMPT101B AC Single Phase voltage sensor module is based on a high precision ZMPT101B voltage Transformer. ZMPT101B AC Voltage Sensor is the best for the purpose of the DIY project, where we need to measure the accurate AC voltage with a voltage transformer. This is an ideal choice to measure the AC voltage using Arduino/ESP8266/Raspberry Pi like an opensource platform. In many electrical projects, engineer directly deals with measurements with few basic requirements like High galvanic isolation, Wide Range, High accuracy, Good Consistency.

Onboard precision miniature voltage transformer, The active phase AC output voltage transformer module. Onboard precision op-amp circuit, the signal sampling and appropriate compensation for precise functions. Modules can be measured within 250V AC voltage, the corresponding analog output can be adjusted. It is brand new, good quality high performance.

Figure 2.4 ZMPT101B Voltage Sensor Module Pinout

FEATURES of ZMPT101B AC Single Phase Voltage Sensor Module: -

- Voltage up to 250 volts can be measured
- Light weight with on-board micro-precision voltage transformer
- High precision on-board op-amp circuit
- Operating temperature: 40ºC ~ + 70ºC
- Supply voltage 5 volts to 30 volts

ADVANTAGES of ZMPT101B AC Single Phase Voltage Sensor Module: -

- Analog output corresponding quantity can be adjusted.
- Pcb board size: 49.5 (mm) x19.4 (mm)
- Good consistency, for voltage and power measurement
- Very efficient and accuracy

2.3.2 Current Sensor - SCT013-030

Description:

This is SCT-013-030 Non-invasive AC Current Sensor Clamp Sensor 30A. SCT-013-000 is a Non-Invasive AC current sensor i.e. it is a current transformer that can be used to measure AC current up to 30 amperes.

This non-invasive current sensor clamped around the supply line can measure a load up to 30 Amps, and allow you to calculate how much current pass through it. It can be useful for building your own energy monitor or for building an over-current protection device for an AC load. This current clamp can be used to detect a current of up to 30A. Simply clip it around the current source that you wish to measure and it will produce a (very) small AC voltage proportional to the current. The cable is terminated on one end with a standard 3.5mm jack (like a headphone jack).

Figure 2.5 SCT0013-030 Current Sensor

Features:

- Non-invasive current transformer
- Suitable for lighting equipment, AC motors, air compressors, monitoring, current measurement, and protection
- Meet UL94-V0 flame retardant properties
- Two forms of output current, voltage (voltage output built-in the sampling resistor)
- Non-linearity ±3%(10%-120% rated input current)

Specifications :-

Core Material	Ferrite
External Material	ABS
Input Current	0-30A
Opening Size(mm)	1313
Dielectric Strength (VAC/1min)	6000
Operating Temperature (C)	-25 to 85
Output Mode	0-1V @ 30mA
Length (mm)	57
Width (mm)	36
Height (mm)	21
Weight (gm)	65
Cable Length (Meter)	1

Applications:

1. Suitable for the current measuring.
2. Monitoring and protection of AC motor
3. Lighting equipment
4. Air compressor.

2.3.3 Temperature Sensor – DS18B20:

Description:

The DS18B20 is a 1-wire programmable Temperature sensor from maxim integrated. It is widely used to measure temperature in hard environments like in chemical solutions, mines or soil etc. The constriction of the sensor is rugged and also can be purchased with a waterproof option making the mounting process easy. It can measure a wide range of temperature from -55°C to +125° with a decent accuracy of ±5°C. Each sensor has a unique address and requires only one pin of the MCU to transfer data so it a very good choice for measuring temperature at multiple points without compromising much of your digital pins on the microcontroller**.**

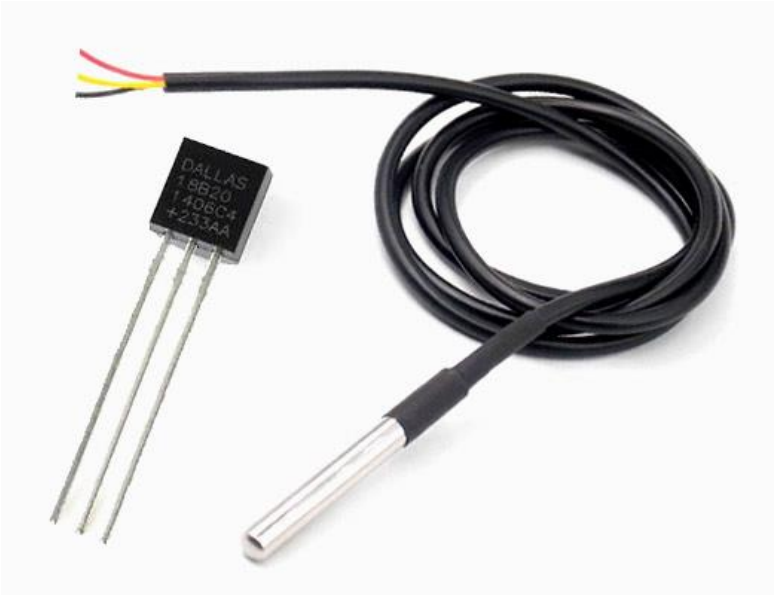


Figure 2.6 DS18B20 Temperature Sensor

DS18B20 Sensor Specifications

- Programmable Digital Temperature Sensor
- Communicates using 1-Wire method
- Operating voltage: 3V to 5V
- Temperature Range: -55°C to +125°C
- Accuracy: ±0.5°C
- Output Resolution: 9-bit to 12-bit (programmable)
- Unique 64-bit address enables multiplexing
- Conversion time: 750ms at 12-bit
- Programmable alarm options
- Available as To-92, SOP and even as a waterproof sensor

2.3.4 Water flow sensor – YFS 402:

Description:

This water flow sensor is basically a small turbine whose output signal is a series of digital pulses. The frequency of the pulses is proportional to the flow rate of the liquid passing through the sensor. That digital signal, whose frequency is in the range between 0Hz and 100Hz, can be read directly through one of the digital input/output pins of a microcontroller**.**

Figure 2.7 YF S-402 Flow Sensor

Specification:

- Mini. Working Voltage: DC 4.5V
- Max. Working Voltage: 15mA (DC 5V)
- Working Voltage: DC 5V~24V
- Flow Rate Range:1~5L/min
- Load Capacity: ≤10mA (DC 5V)
- Operating Temperature: ≤80C
- Liquid Temperature: ≤120C
- Operating Humidity: 35%~90%RH
- Water Pressure: 0.35MPa
- Storage Temperature: -25~+80C
- Storage Humidity: 25%~95%RH
- Insulation resistance > 100M
- Pipe Gauge: 1/4 inch outer diameter
- Error: +/-2L/min;

- Flow pulse characteristic f = (73 * Q) ± 2% Q = L / MIN
- Output pulse duty cycle: 50% ± 10%
- Dimensions: 2.28 in x 1.38 in x 1.06 in (5.8 cm x 3.5 cm x 2.7 cm)
- Weight: 1.02 oz (29 g)
- Thread Size: G 1/4″

Features:

- Lightweight, Small in size, Easy to install;
- With stainless steel axis in the wheel, abrasion resistant;
- Sealing ring design to prevent water leaks;
- Compliant with RoHS standard

Applications:

- Suitable for water heater, POS terminal, automatic water dispenser etc.

2.3.5 Water float switch:

Description:

A float switch is a device used to sense the level of liquid within a tank, it may actuate a pump, an indicator, an alarm, or other device

The purpose of a float switch is to open or close a circuit as the level of a liquid rises or falls. Most float switches are "normally closed," meaning the two wires coming from the top of the switch complete a circuit when the float is at its low point, resting on its bottom clip or stop (for example, when a tank is dry).

Most float switches utilize a magnetic reed switch to open or close the circuit. The reed is encased in a glass tube, which is cemented into a plastic or stainless-steel stem with epoxy. The illustration to the left demonstrates how a magnet can be used to open or close a circuit by moving it closer to or farther away from a reed switch. When the magnet comes close to the two contacts, they draw together and touch, allowing current to pass through. When the magnet is moved away, the contacts demagnetize and separate, breaking the circuit. In a float switch, the magnetic reed switch is hermetically sealed in a stem, most often made from plastic or stainless steel. The float encases a sealed magnet, which moves up and down the length of the stem as a fluid level rises and falls.

As the magnet passes by the contacts in the encased reed switch, they draw together and complete a circuit between the two lead wires. The operation shown can usually be reversed by removing the bottom clip from the switch, inverting the float and replacing the clip. When this change is made, the switch circuit will be open when the float is resting on the bottom clip and closed when the float rises. Properly used, float switches can deliver millions of on/off cycles, for years of dependable operation. Failures are normally due to overloading, frequently caused by spiking voltage.

Figure 2.8 Float Switch

Specifications:-

- Cable Length: 30.5(cm)
- Maximum Load: 50 W
- Max Switching Voltage: 100V DC
- Minimum Voltage: 250V DC
- Maximum Switching Current: 0.5 A
- Max Load Current: 1.0 A
- Max Contact Resistance: 0.4 ΩTemp Rating: -20~ 80 degree

Applications:-

- House hold Equipment such as overhead water tanks, Water purifiers.
- Use them with hydroponics, saltwater tank, freshwater tank, gardening, aquariums for power head control, pet bowls, fish tanks, filtration, heating, or whatever your project may be. Up float switch Contains no mercury Product.

2.4 LCD Display:

LCD modules are very commonly used in most embedded projects, the reason being its cheap price, availability and programmer friendly. Most of us would have come across these displays in our day to day life, either at PCO's or calculators.

16×2 LCD is named so because; it has 16 Columns and 2 Rows. There are a lot of combinations available like, 8×1, 8×2, 10×2, 16×1, etc. but the most used one is the 16×2 LCD. So, it will have (16×2=32) 32 characters in total and each character will be made of 5×8 Pixel Dots.

Now, we know that each character has (5×8=40) 40 Pixels and for 32 Characters we will have (32×40) 1280 Pixels. Further, the LCD should also be instructed about the Position of the Pixels. Hence it will be a hectic task to handle everything with the help of MCU, hence an Interface IC like HD44780is used, which is mounted on the backside of the LCD Module itself. The function of this IC is to get the Commands and Data from the MCU and process them to display meaningful information onto our LCD Screen.

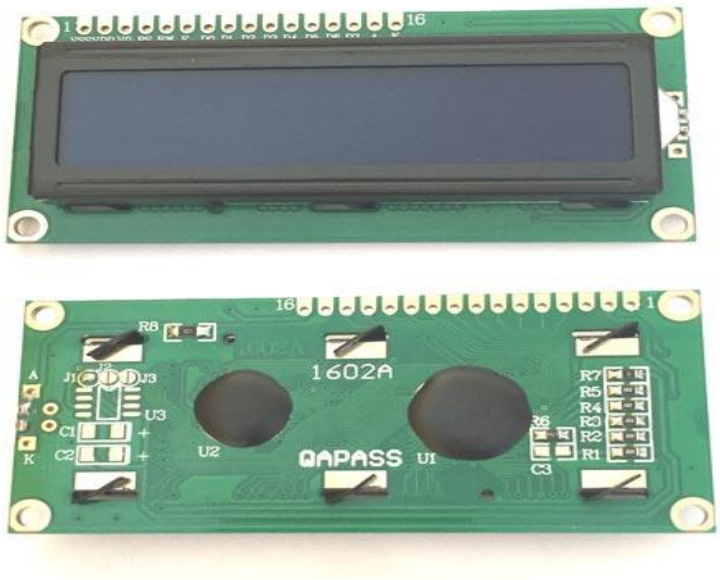


Figure 2.9 LCD Display

Features of 16×2 LCD module:

- Operating Voltage is 4.7V to 5.3V
- Current consumption is 1mA without backlight
- Alphanumeric LCD display module, meaning can display alphabets and numbers
- Consists of two rows and each row can print 16 characters.
- Each character is built by a 5×8-pixel box
- Can work on both 8-bit and 4-bit mode
- It can also display any custom generated characters
- Available in Green and Blue Backlight

Pin Configuration

Pin No:	Pin Name:	Description
1	Vss (Ground)	Ground pin connected to system ground
2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contrast.
4	Register Select	Connected to Microcontroller to shift between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement
7	Data Pin 0,1 2,3,4,5,6,7	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free.
8	LED Positive	Backlight LED pin positive terminal
9	LED Negative	Backlight LED pin negative terminal

Table No.3 Pin Description Of LCD Display

I2C 1602 Serial LCD Module

C/I2C Interface Adapter Module is used for 16×2 LCD Display. It uses the PCF8574T IC chip which converts I2C serial data to parallel data for the LCD display. Also this interface module simplifies connecting an Arduino to a 16×2 Liquid Crystal display using only 4 wires.

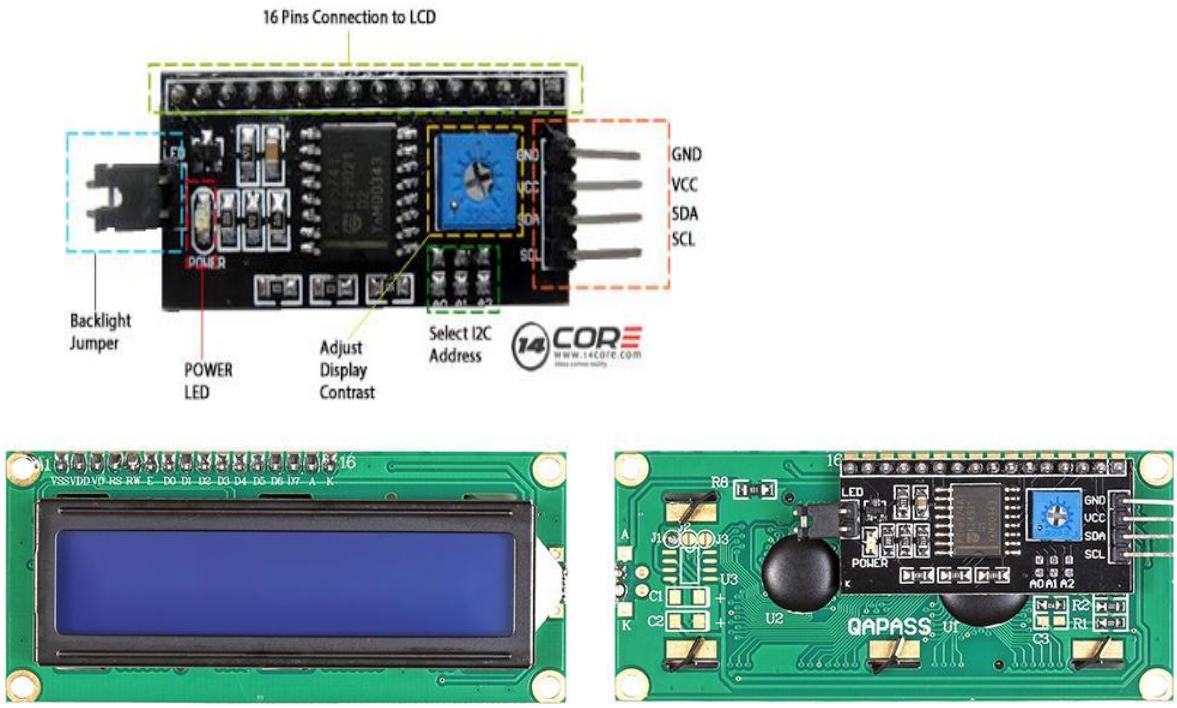


Figure 2.10 LCD Display with I2c Interface

Specifications

- Supply Voltage: 5V
- Interface: I2C
- Compatible for 16×2 LCD
- Brightness and Contrast can be adjusted by the Potentiometer

2.5 Relays:

Description:

A relay is an electrically operated switch. It consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals. The switch may have any number of contacts in multiple contact forms, such as make contacts, break contacts, or combinations thereof.

Relays are used where it is necessary to control a circuit by an independent low-power signal, or where several circuits must be controlled by one signal. Relays were first used in long-distance telegraph circuits as signal repeaters: they refresh the signal coming in from one circuit by transmitting it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

The traditional form of a relay uses an electromagnet to close or open the contacts, but other operating principles have been invented, such as in solid-state relays which use semiconductor properties for control without relying on moving parts. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called protective relays.

Latching relays require only a single pulse of control power to operate the switch persistently. Another pulse applied to a second set of control terminals, or a pulse with opposite polarity, resets the switch, while repeated pulses of the same kind have no effects. Magnetic latching relays are useful in applications when interrupted power should not affect the circuits that the relay is controlling.

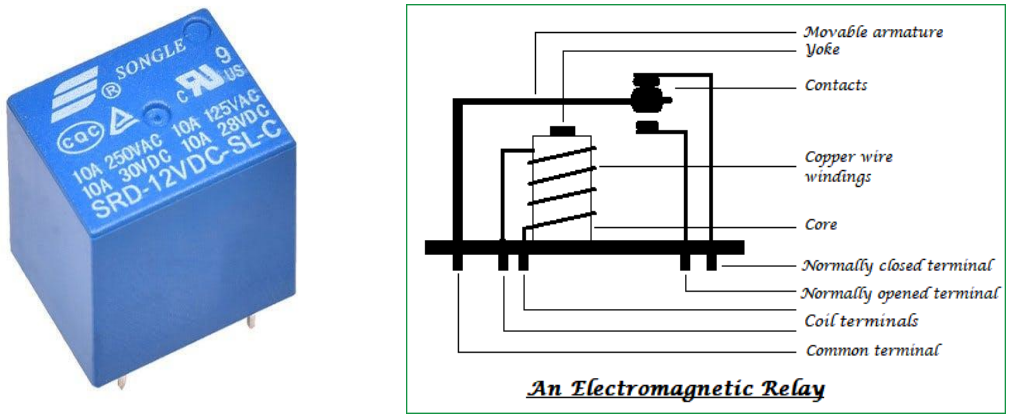


Figure 2.11 Relay and Its Cross-Sectional View

Schematic:

VCC and RY-VCC are also the power supply of the relay module. When you need to drive a large power load, you can take the jumper cap off and connect an extra power to RY-VCC to supply the relay; connect VCC to 5V of the MCU board to supply input signals. NOTES: If you want complete optical isolation, connect "Vcc" to Arduino +5 volts but do NOT connect Arduino Ground. Remove the Vcc to JD-Vcc jumper. Connect a separate +5 supply to "JD-Vcc" and board Gnd. This will supply power to the transistor drivers and relay coils. If relay isolation is enough for your application, connect Arduino +5 and Gnd, and leave Vcc to JD-Vcc jumper in place.

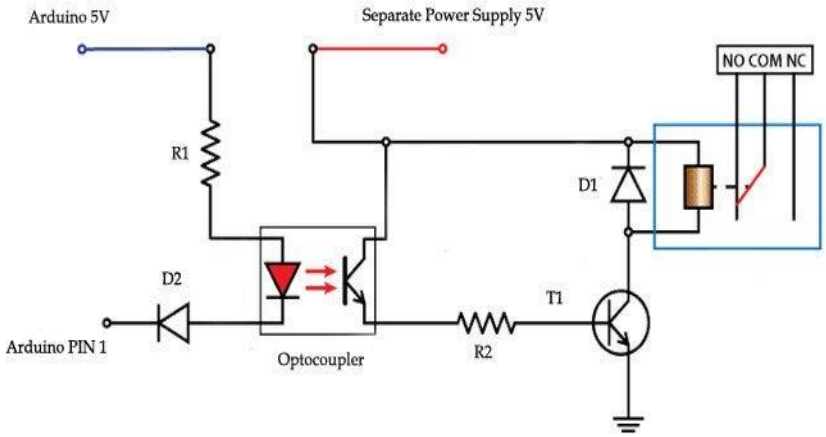


Figure 2.12 Schematic of Single Channel Relay

It is sometimes possible to use this relay boards with 3.3V signals, if the JD-VCC (Relay Power) is provided from a +5V supply and the VCC to JD-VCC jumper is removed. That 5V relay supply could be totally isolated from the 3.3V device, or have a common ground if opto-isolation is not needed. If used with isolated 3.3V signals, VCC (To the input of the opto-isolator, next to the IN pins) should be connected to the 3.3V device's +3.3V supply.

Application:

Motor control, automotive applications such as an electrical fuel pump, industrial applications where control of high voltages and currents is intended, controlling large power loads, and so on.

2.6 Single Phase Two Pole Contactor:

Description:

A contactor is an electrically-controlled switch used for switching an electrical power circuit. A contactor is typically controlled by a circuit which has a much lower power level than the switched circuit, such as a 24-volt coil electromagnet controlling a 230-volt motor switch.

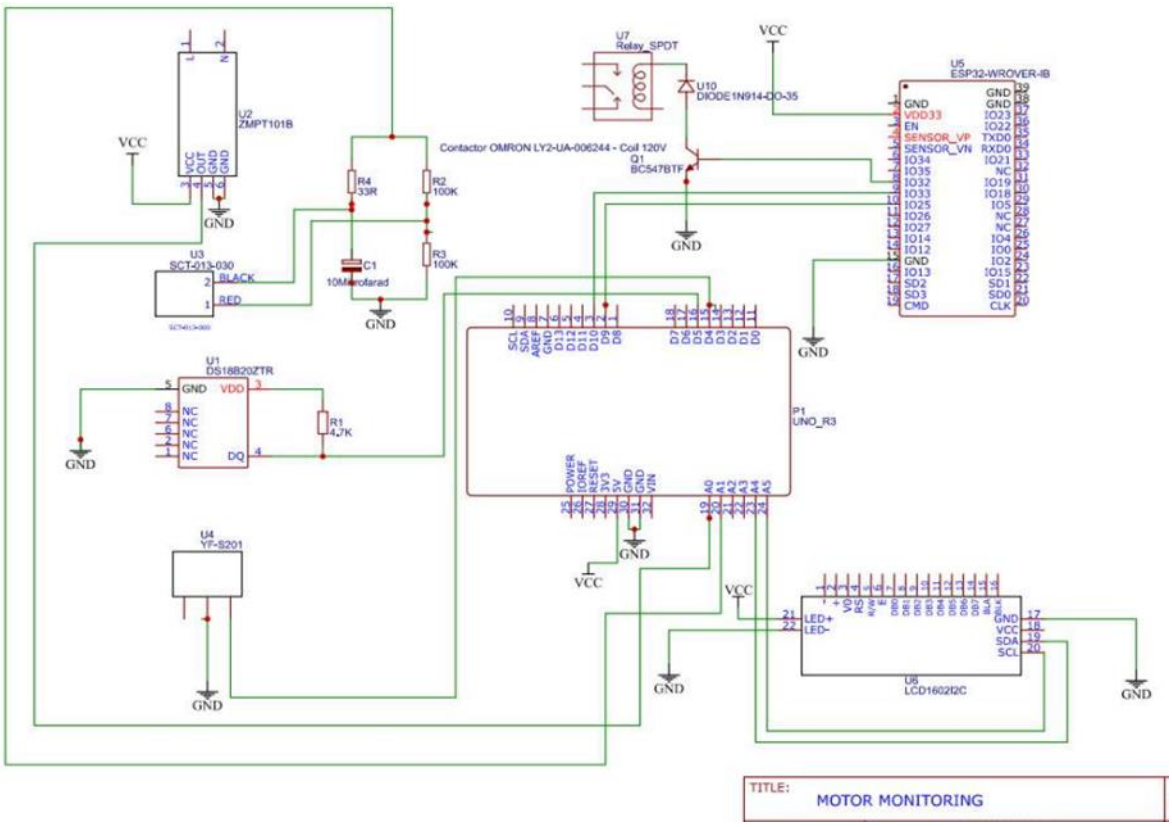
Unlike general-purpose relays, contactors are designed to be directly connected to high-current load devices. Relays tend to be of lower capacity and are usually designed for both normally closed and normally open applications. Devices switching more than 15 amperes or in circuits rated more than a few kilowatts are usually called contactors. Apart from optional auxiliary low-current contacts, contactors are almost exclusively fitted with normally open ("form A") contacts. Unlike relays, contactors are designed with features to control and suppress the arc produced when interrupting heavy motor currents.

Contactors come in many forms with varying capacities and features. Unlike a circuit breaker, a contactor is not intended to interrupt a short circuit current. Contactors range from those having a breaking current of several amperes to thousands of amperes and 24 V DC to many kilovolts. The physical size of contactors ranges from a device small enough to pick up with one hand, to large devices approximately a meter (yard) on a side.

Figure 2.13 Single Phase Two Pole Contactor

CHAPTER 3

CIRCUIT DIAGRAM



CHAPTER 4

PROGRAM

4.1 ARDUINO CODE:

```
#include <SoftwareSerial.h>
#include <LiquidCrystal_I2C.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <ArduinoJson.h>
#include "EmonLib.h"  // Include Emon Library
SoftwareSerial nodemcu(9, 10);//Initialise Arduino to NodeMCU (9=Rx & 10=Tx)
float vol;
float rms;
float flows;
float floats;
float temps;
//////////////////////////////////////////////////////////////////////////////
LiquidCrystal_I2C lcd(0x27,16,2);  // set the LCD address to 0x3F for a 16 chars and 2 line display
//////////////////////////////////////////////////////////////////////////////
int FloatSensor=5;
int buttonState = 1; //reads pushbutton status
//////////////////////////////////////////////////////////////////////////////
EnergyMonitor emon1;                   // Create an instance
//////////////////////////////////////////////////////////////////////////////
#define ONE_WIRE_BUS 4// Data wire is plugged into digital pin 2 on the Arduino
DallasTemperature sensors(&oneWire);// Pass oneWire reference to DallasTemperature library
//////////////////////////////////////////////////////////////////////////////
double sensorValue1 = 0;
double sensorValue2 = 0;
int crosscount = 0;
int climb_flag = 0;
int val[100];
int max_v = 0;
double VmaxD = 0;
```

```
double VeffD = 0;
double Veff = 0;
////////////////////////////////////////////////////////////////////////////
volatile int flow_frequency; // Measures flow sensor pulses
unsigned char flowsensor = 2; // Sensor Input
unsigned long currentTime;
unsigned long cloopTime;
////////////////////////////////////////////////////////////////////////////
void flow () // Interrupt function
{
  flow_frequency++;
}
////////////////////////////////////////////////////////////////////////////
void setup()
{
   Serial.begin(9600);
   nodemcu.begin(9600);
   delay(1000);
   sensors.begin();  // Start up the library
   pinMode(flowsensor, INPUT);
   attachInterrupt(0, flow, RISING); // Setup Interrupt
   sei(); // Enable interrupts
   currentTime = millis();
   cloopTime = currentTime;
   emon1.current(1, 111.1);// Current: input pin, calibration.
   pinMode(FloatSensor, INPUT_PULLUP); //Arduino Internal Resistor 10K
   lcd.init();
   lcd.clear();
   lcd.backlight();
}
////////////////////////////////////////////////////////////////////////////
void loop(void)
{
  for ( int i = 0; i < 100; i++ ) {
```

```
   sensorValue1 = analogRead(A0);
   if (analogRead(A0) > 511) {
     val[i] = sensorValue1;
   }
   else {
     val[i] = 0;
   }
   delay(1);
 }
 max_v = 0;
 for ( int i = 0; i < 100; i++ )
 {
   if ( val[i] > max_v )
   {
     max_v = val[i];
   }
   val[i] = 0;
 }
 if (max_v != 0)
 {
   VmaxD = max_v;
   VeffD = VmaxD / sqrt(2);
   Veff = (((VeffD - 420.76) / -90.24) * -210.2) + 210.2;
 }
 else {
   Veff = 0;
 }
 VmaxD = 0;
 delay(1000);
//////////////////////////////////////////////////////////////////
 int Irms = emon1.calcIrms(10);  // Calculate Irms only
delay(1000);
```

```
////////////////////////////////////////////////////////////////////
  // Send the command to get temperatures
  sensors.requestTemperatures();
  //print the temperature in Celsius
  delay(1000);
////////////////////////////////////////////////////////////////////
  currentTime = millis();
   // Every second, calculate and print litres/hour
   if(currentTime >= (cloopTime + 1000))
   {
      // Pulse frequency (Hz) = 7.5Q, Q is flow rate in L/min.
      l_hour = (flow_frequency * 60 / 7.5); // (Pulse frequency x 60 min) / 7.5Q = flowrate in L/hour
      flow_frequency = 0; // Reset Counter
   }
   delay(1000);
////////////////////////////////////////////////////////////////////
buttonState = digitalRead(FloatSensor);
delay(1000);
////////////////////////////////////////////////////////////////////
lcd.setCursor(0,0);   //Set cursor to character 2 on line 0
  lcd.print("Voltage =");
  lcd.print(Veff);
  lcd.print("V");
  lcd.setCursor(0,1);   //Move cursor to character 2 on line 1
  lcd.print("Current = ");
  lcd.print(Irms);
  lcd.print("A");
  delay(5000);
////////////////////////////////////////////////////////////////////
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Temp = ");
  lcd.print(sensors.getTempCByIndex(0));
  lcd.print((char)223);
```

```
  lcd.setCursor(0,1);
  lcd.print("Flow = ");
  lcd.print(l_hour,DEC);
  lcd.print("L/Hr");
  delay(5000);
////////////////////////////////////////////////////////////////////
  lcd.clear();
  lcd.setCursor(0,0);
  if (buttonState == LOW)
  {
    lcd.print( "W.Level = Low");
  }
  else
  {
    lcd.print( "W.Level = High" );
  }
  delay(5000);
////////////////////////////////////////////////////////////////////
  vol=(Veff);
  rms=(Irms);
  temps = (sensors.getTempCByIndex(0));
  flows = (l_hour);
  floats = (buttonState);
  StaticJsonBuffer<1000> jsonBuffer;
  JsonObject& data = jsonBuffer.createObject();
  data["voltage"] = vol;
  data["current"] = rms;
  data["tempre"] = temps;
  data["flow"] = flows;
  data["floats"] = floats;
  data.printTo(nodemcu);
  jsonBuffer.clear();
 }
```

4.2 NODEMCU CODE:

```
#include "arduino_secrets.h"
#include "thingProperties.h"
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
//D6 = Rx & D5 = Tx
int LED=13;
SoftwareSerial nodemcu(D6, D5);
void setup() {
  Serial.begin(9600);
  delay(1500);
  initProperties();
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
  nodemcu.begin(9600);
  while (!Serial) continue;
  pinMode(LED, OUTPUT);
}

void loop() {
  ArduinoCloud.update();
  StaticJsonBuffer<1000> jsonBuffer;
  JsonObject& data = jsonBuffer.parseObject(nodemcu);
   if (data == JsonObject::invalid()) {
    //Serial.println("Invalid Json Object");
    jsonBuffer.clear();
    return;

  }
  float vol = data["voltage"];
  float rms=data["current"];
  float temps = data["tempre"];
  float flow = data["flow"];
```

```
  float floats = data["floats"];
  Serial.println(vol);
  Serial.println(rms);
  Serial.println(temps);
  Serial.println(flow);
  Serial.println(floats);
motortemperature = (temps);
motorvoltage = (vol);
waterflow = (flow);
motorcurrent = (rms);
delay(5000);
}
void onMotorswitchChange()
{
 if(motorswitch)
 {
    digitalWrite(LED, LOW);
  }
  else
  {
    digitalWrite(LED, HIGH);
  }
}
```

CHAPTER 5

BENEFITS AND CONCLUSION

5.1 Benefits of the Project:

An induction motor can be examined carried out immediate action to avoid motor downtime so it can be saved time and cost. The advantages of monitoring induction motor by using IoT are notification for fault alert and historical data for predictive maintenance.

Temperature Monitoring

Temperature of stator winding can be measured through Dallas DS1820 Direct to digital temperature sensor. The Dallas Direct–to–Digital Temperature Sensors measure temperature through the use of an onboard proprietary temperature measurement technique.

Process Monitoring

The temperature data logger is used extensively in industries such as food processing, manufacturing, printing, metallurgy, etc. where critical process variables (temperature, pressure, etc.) may adversely affect the results of the process. Continuous monitoring records the process, and user-defined alarm limits immediately alert the operator of extreme conditions that require attention. In the event of power interruptions, the system automatically resumes monitoring the process at the present conditions. Our research objective is to design and deploy the framework which is worth useful to improve the existing scenario in field of controlling and making the systems automated, simplified on the other hand also helps in maintain security aspects also.

5.2 Conclusion:

IoT is a paradigm which makes each object as an intelligent object. Intelligent objects have the identification, sensing, communication and processing features which makes them capable to communicate with other objects, software and services running on the internet. Intelligent objects forms the backbone of the IoT and helps to improve the living standard in a city. A smart city has many IoT based applications running in different areas across the city. In this thesis, we have applied IoT to study the issues related to the power consumption, irrigation system, and traffic flow and security system in a city to make it a smart city. We have proposed sub module at three distinct levels carrying out the purpose of automation using IoT. These modules monitor and control of the Induction Motor required solving the aforementioned problems. Proposed System is found capable to solve the various issues, importantly issues related to the power consumption, its automation and billing process successfully**.**

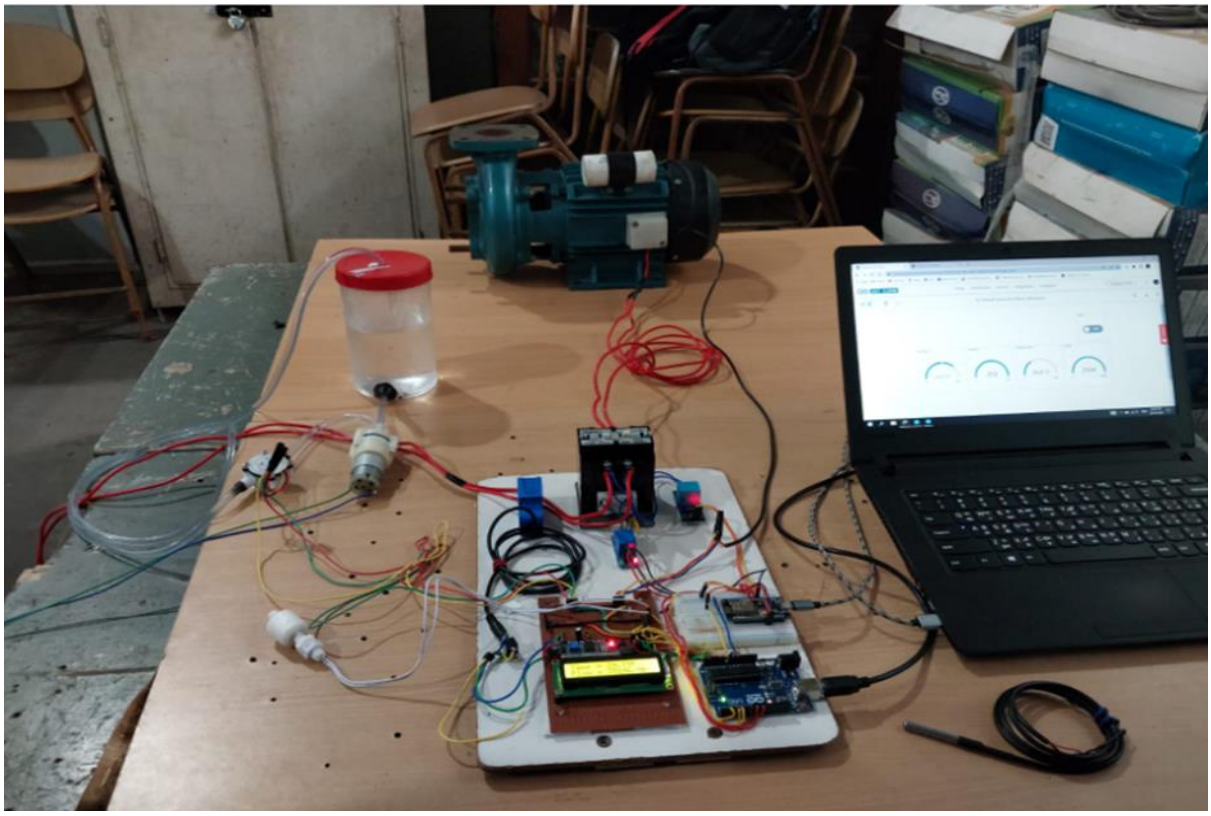
CHAPTER 6

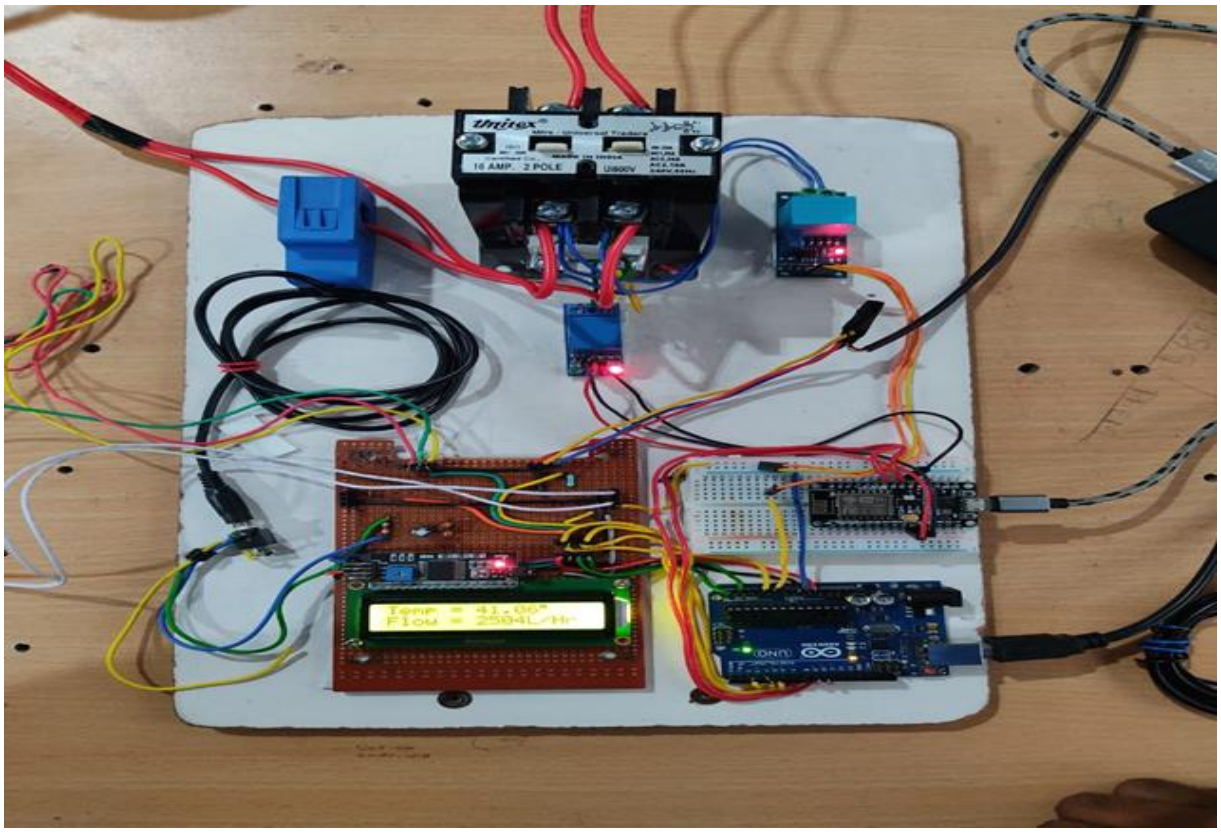
APPENDIX

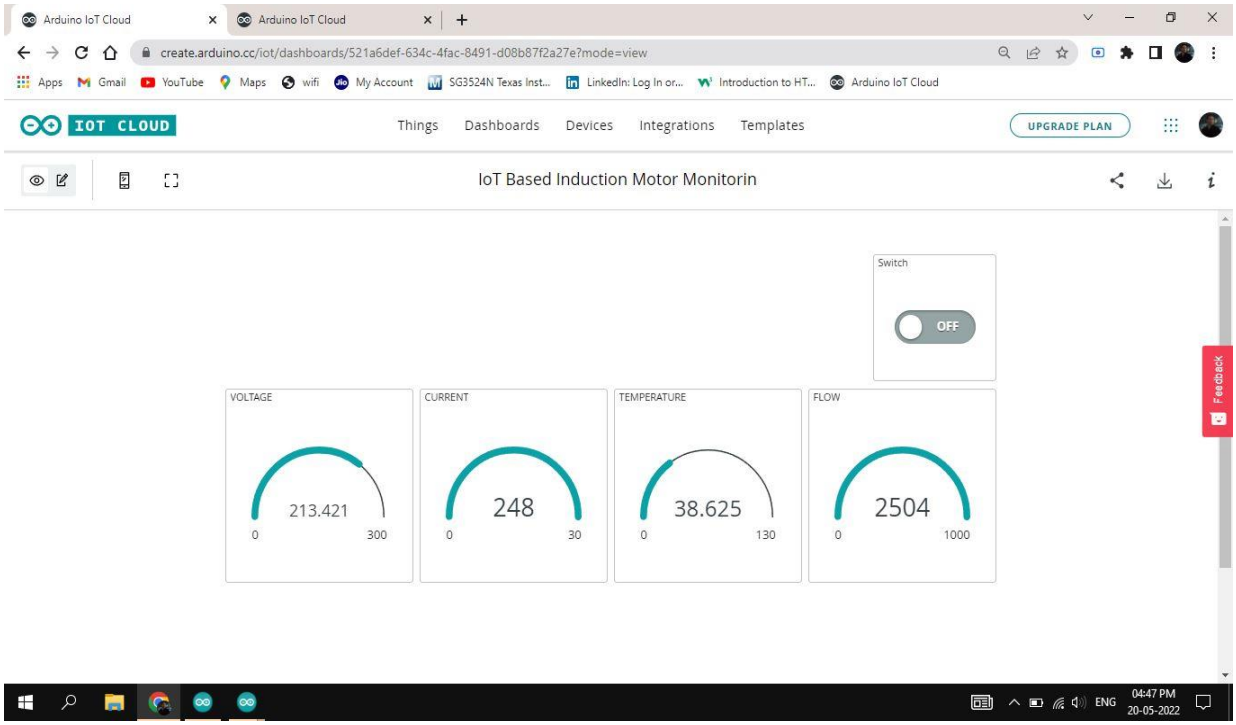
6.1 Bill of Materials:

S.NO	NAME OF THE MATERIALS	COST IN RUPEES
1	CURRENT SENSOR (SCT 013-030)	570.00
2	WATER FLOW SENSOR (YF-S201)	340.00
3	VOLTAGE SENSOR (ZMPT10B)	120.00
4	FLOAT SWITCH	150.00
5	TEMPERATURE SENSOR (DS18B0)	100.00
6	RELAY (2 CHANNEL 5V)	120.00
7	DOTTED PCB BOARD	40.00
8	LCD DISPLAY (1602 LCD WITH I2C	150.00
9	NODE MCU	200.00
10	RESISTOR SETS	50.00
11	CAPACITOR	2.00
12	FEMALE HEADER PINS	20.00
13	2.5 SQUARE MM WIRE	150.00
14	16 AMPS PLUG	90.00
15	ARDUINO UNO	700.00
16	BREAD BOARD	60.00
17	SINGLE STAND WIRE	35.00
18	MULTI COLOUR WIRE	35.00
19	SINGLE PHASE INDUCTION MOTOR	-
	TOTAL	**2957.00**

6.2 Photography:






Arduino IoT Cloud
Arduino IoT Cloud
create.arduino.cc/iot/dashboards/521a6def-634c-4fac-8491-d08b87f2a27e?mode=view
Apps
Gmail
YouTube
Maps
wifi
My Account
SG3524N Texas Inst...
LinkedIn: Log In or...
Introduction to HT...
Arduino IoT Cloud
IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
IoT Based Induction Motor Monitorin
Switch
OFF
VOLTAGE
213.421
0
300
CURRENT
248
0
30
TEMPERATURE
38.625
0
130
FLOW
2504
0
1000
Feedback
ENG
04:47 PM
20-05-2022

6.3 Bibliography:

[1] IoT platform for condition monitoring of industrial motor :Published in 2nd International Conference on Communication and Electronics Systems (ICCES 2017) IEEE Explore Compliant - Part Number:CFP17AWO-ART, ISBN:978-1-5090-5013-0 by Shyamala.D .

[2] IoT-based traction motor drive condition monitoring in electric vehicles: Part1: published in Power Electronics and Drive Systems (PEDS), 2017 IEEE 12th International Conference on. IEEE, 2017.

3] Smart Shut-Down and Recovery Mechanism for Industrial Machines Using Internet of Things : Published in 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, 2018 by Prakash, Chetna, and Sanjeev Thakur.

[4]IoT-based wireless induction motor monitoring: Published in Scientific Conference Electronics (ET), 2017 XXVI International. IEEE, 2017 ,by Şen, Mehmet, and Basri Kul.

[5] The application of wireless sensor networks for condition monitoring in three-phase induction motors: Published in Electrical Insulation Conference and Electrical Manufacturing Expo, 2007. IEEE, 2007 by, Xue, Xin, V. Sundararajan, and Wallace P. Brithinee

[6] D. Shyamala, D. Swathi, J. L. Prasanna and A. Ajitha, "IoT platform for condition monitoring of industrial motors," 2017 2nd International Conference on Communication and Electronics Systems (ICCES), 2017, pp. 260-265

[7] J. Kunthong et al., "IoT-based traction motor drive condition monitoring in electric vehicles: Part 1," 2017 IEEE 12th International Conference on Power Electronics and Drive Systems (PEDS), 2017, pp. 1,184-1,188.

[8] C. Prakash and S. Thakur, "Smart Shut-Down and Recovery Mechanism for Industrial Machines Using Internet of Things," 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2018, pp. 824-828.

[9] M. Şen and B. Kul, "IoT-based wireless induction motor monitoring," 2017 XXVI International Scientific Conference Electronics (ET), 2017, pp.

[10] Xin Xue, V. Sundararajan and W. P. Brithinee, "The application of wireless sensor networks for condition monitoring in three-phase induction motors," 2007 Electrical Insulation Conference and Electrical Manufacturing Expo, 2007, pp. 445-448.

[11]C. Woodford, “Induction Motors”, [Online]. Available: http://www.explainthatstuff.com/induction-motors.html. [Accessed: 4- November-2015].