IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

IMPLEMENTATION OF AN IOT-BASED INDUCTION MOTOR MONITORING AND PROTECTION SYSTEM USING ESP32

[1]Jyoti vitthal shinde, [2]Prof. Khanapure R P

[1]M.Tech Student, [2]Asst. Professor,

[12]Electronics & Telecommunication Engineering,

[12]M.S Bidve Engineering College, Latur, Maharashtra

Abstract: Three-phase induction motors form the backbone of industrial drives, yet they remain vulnerable to thermal overload, overcurrent and abnormal supply voltage, each of which shortens winding life and causes unplanned downtime. Conventional thermal overload relays and fuses act only after a fault has fully developed, and they retain no record of the operating conditions that preceded the trip, so continuous monitoring of the machine is required if abnormal conditions are to be detected early and the motor isolated before damage occurs. This paper reports the design, construction and operation of a prototype Internet of Things based monitoring and protection system built around an ESP32 microcontroller. Analogue temperature, current and voltage sensing channels are connected to the microcontroller, which digitises each quantity, converts the raw conversion result into engineering units using the scaling factors embedded in the firmware, and compares the result against threshold values held in the program. When a monitored quantity leaves its permitted band the microcontroller de-energises a relay through a Darlington driver stage and thereby disconnects the motor supply. The measured values are simultaneously written to a 16x2 liquid crystal display for local indication and uploaded over the on-board Wi-Fi interface to a ThingSpeak channel, where they are recorded as three separate fields and can be observed remotely from any browser. The complete circuit was fabricated on a single-sided printed circuit board using the screen-printing and etching route. The significance of the implementation lies in combining local threshold-based tripping with cloud-side visibility of the machine on a single low-cost embedded platform.

Index Terms **-** Induction motor protection, ESP32, Internet of Things, ThingSpeak, overcurrent protection, thermal protection, relay, voltage monitoring, embedded system.

I. INTRODUCTION

The three-phase squirrel-cage induction motor is the most widely deployed prime mover in industry. It is rugged, requires little maintenance and tolerates a wide range of duty cycles, and for these reasons it is used in pumps, compressors, conveyors, fans, machine tools and process drives of every capacity. Its robustness, however, is frequently mistaken for immunity. A motor that is mechanically sound will still fail if it is asked to operate outside the electrical and thermal envelope for which it was designed, and in practice the great majority of in-service failures are traceable to a small number of recurring supply-side and loading conditions rather than to the machine itself.

Three conditions dominate. The first is thermal overload. Insulation ageing is a strongly temperature-dependent process, and sustained operation above the rated winding temperature reduces insulation life considerably; updated thermal-model overload protection has been shown to extend motor life appreciably compared with fixed-setting devices [9]. The second is overcurrent, which arises from mechanical overloading, stalled rotors, bearing seizure or a jammed driven machine, and which converts directly into additional copper loss and therefore into additional heating. The third is abnormal supply voltage. Undervoltage forces the motor to draw a larger current for the same shaft power, overvoltage drives the

magnetic circuit towards saturation, and unbalanced or single-phased supply produces a negative-sequence current component that heats the rotor disproportionately [1], [4], [5], [8]. Field surveys of motor and generator windings confirm that these stresses, acting over time, are a principal cause of the winding failures encountered in service [10].

The protective devices most commonly fitted to small and medium industrial motors are the fuse, the bimetallic thermal overload relay and the motor circuit breaker. These devices are inexpensive and reliable, but they share two limitations that matter for the present work. First, they respond to the consequence of a fault rather than to its onset, so protection is inherently retrospective: the trip occurs after the machine has already been stressed. Second, they are silent. A bimetallic relay does not tell the maintenance engineer what the winding temperature was, how the line current behaved in the minutes before the trip, or whether the supply voltage had been drifting for hours. The information that would allow the operator to distinguish a genuine mechanical overload from a supply disturbance is simply not retained.

Continuous measurement of the motor's operating quantities addresses both limitations at once. If temperature, current and voltage are sampled repeatedly and compared against limits, an approaching abnormal condition becomes visible before it becomes destructive, and the same measurements, once digitised, can be transmitted elsewhere and stored. The emergence of low-cost microcontrollers with integrated wireless connectivity has made such an arrangement economically viable even for a single small machine. The ESP32 in particular combines a 32-bit dual-core processor operating at up to 240 MHz, on-chip analogue-to-digital conversion, and integrated 2.4 GHz Wi-Fi and Bluetooth interfaces in one inexpensive module, so that acquisition, decision-making and network transmission can all be carried out by a single device without an external communication gateway.

This paper describes the implementation of such a system. A prototype was constructed in which an ESP32 acquires temperature, current and voltage from an induction motor, evaluates the three measurements against thresholds fixed in the firmware, displays them on a 16x2 liquid crystal display for the operator standing at the machine, drives a relay through a Darlington array to disconnect the motor when a threshold is violated, and uploads the same three quantities to a ThingSpeak cloud channel over Wi-Fi so that the machine can be observed from any location with internet access. The emphasis throughout is on the implementation as actually built and programmed: the sensing interfaces, the power supply, the scaling arithmetic present in the firmware, the trip logic, the network transaction and the printed circuit board are described as they exist in the working prototype, and the points at which the implementation is incomplete or internally inconsistent are identified explicitly rather than smoothed over.

The remainder of the paper is organised as follows. Section II states the problem and the objectives. Section III reviews the related literature and identifies the gap that motivates the work. Section IV presents the proposed system architecture. Section V describes the hardware, software, cloud and printed circuit board implementation. Section VI explains the working methodology of each monitored quantity. Section VII sets out the protection algorithm together with a verification note on the firmware. Section VIII discusses the results obtained from the prototype, and Sections IX to XII cover advantages and applications, limitations, future scope and conclusions.

II. Problem Statement and Objectives

In a typical industrial installation a motor operates unattended for long periods. Its protection is delegated to devices that are set once during commissioning and then, in practice, rarely reviewed. The operator has no continuous indication of the machine's condition, and when a trip finally occurs there is no measurement record from which the cause can be established. Restoring service therefore becomes a matter of inspection and inference, and the same fault frequently recurs because its underlying cause, often a supply-side one, was never identified.

The problem addressed in this work is therefore twofold. The measurement problem is the absence of continuous, simultaneous acquisition of the thermal and electrical quantities that determine whether a motor is operating safely. The accessibility problem is that even where such measurements exist locally, they are confined to the machine and are unavailable to anyone not physically present at it. A protection scheme that solves only the first problem produces a smarter relay; a scheme that solves only the second produces a monitor with no authority to act. The system described here is intended to address both, by giving a single embedded controller the ability to measure, to decide, to display and to publish.

The objectives set for the implementation were as follows:

- To acquire the winding-region temperature, the line current and the supply voltage of an induction motor continuously using analogue sensing channels interfaced to an ESP32 microcontroller.
- To convert each raw analogue-to-digital conversion result into engineering units within the firmware using fixed scaling factors, so that the displayed and transmitted values are directly interpretable.
- To compare each measured quantity against a threshold held in the program and to generate a trip decision when the permitted operating band is left.
- To isolate the motor from the supply automatically by de-energising an electromagnetic relay through a Darlington driver stage when a trip decision is generated.
- To provide local indication of the three measured quantities and of the network status on a 16x2 liquid crystal display mounted with the controller.
- To transmit the three measured quantities over the on-board Wi-Fi interface to a ThingSpeak channel so that the machine can be monitored remotely.
- To realise the complete circuit as a single-sided printed circuit board suitable for a laboratory prototype, using a regulated supply derived from the 230 V mains.

III. Literature Review

Research on induction motor protection has developed along three broad lines: characterisation of the supply-side conditions that damage the machine, detection of those conditions from measurable quantities, and realisation of protection schemes in hardware. The literature drawn upon in the project report covers all three, and it is reviewed here in that order.

Kersting examined the causes and effects of single-phasing in induction motors and established the severity of the condition for the machine that continues to run on the remaining phases [1]. Sutherland and Short studied the effect of single-phase reclosing on industrial loads and showed that reclosing operations on the distribution system are themselves a source of stress on connected motors [2]. Together these two studies frame single-phasing not as a rare fault but as a routine consequence of normal distribution-system behaviour, which is precisely why a motor-side detection scheme is needed rather than reliance on upstream protection alone.

The consequences of unbalanced and out-of-range voltage have been quantified in several studies. Pillay, Hofmann and Manyage derived derating requirements for induction motors operating with combinations of unbalanced voltage and over- or undervoltage [4]. Faiz, Ebrahimpour and Pillay analysed the influence of unbalanced voltage on the steady-state performance of a three-phase squirrel-cage machine [5], and Lee reported the effects of unbalanced voltage on operating performance [8]. The common conclusion of this group of papers is that voltage quality translates directly into additional heating and reduced permissible loading, which justifies treating supply voltage as a protected quantity in its own right and not merely as a background condition.

Detection methods form the second line of work. Sudha and Anbalgan proposed a protection method directed specifically at faults arising from voltage unbalance and single phasing [3]. Javed and Izhar developed an improved method for detecting phase-failure faults in polyphase induction machines [6]. Chattopadhyay, Chattopadhyaya and Sengupta analysed the stator current of a traction induction motor under single phasing by extracting phase angle, symmetrical components, skewness, kurtosis and harmonic distortion in the Park plane [7]. These contributions demonstrate that the information required for reliable detection is present in the ordinary terminal quantities of the machine, although the signal-processing effort required rises steeply with the sophistication of the method.

On the thermal side, Ransom and Hamilton showed that overload protection based on an updated thermal model extends motor life relative to conventional fixed-characteristic devices [9], and Stone, Sasic, Dunn and Culbert documented the winding problems encountered in service on motors and generators [10]. Both papers support the position that direct temperature information, where it can be obtained, is more useful for protection than inference from current alone.

The third line of work concerns implementation. Cunkas, Akkaya and Ozturk demonstrated the protection of AC motors by means of microcontrollers [11], establishing that threshold-based protection logic can be executed satisfactorily by a small embedded processor. That result is directly relevant to the present work, since the protection decision here is likewise a comparison performed in firmware.

Work on remote monitoring of electrical quantities, as cited in the project report, has so far been directed mainly at the energy meter rather than at the machine. Das and Saikia reported a GSM-enabled smart energy

meter with automation of home appliances [12]. Edward described an energy meter reader with load-control capability using a password-protected relay circuit [13]. Cheng, Yang, Xiao and Hou addressed the running-state evaluation of electric energy meters [14]. These studies establish the feasibility of remote acquisition and remote switching of electrical loads, and the relay-based load-control arrangement in [13] is architecturally similar to the trip element used here, but the monitored object in each case is the metering point and not the motor.

Two observations follow from this survey and constitute the motivation for the present work. First, the studies that address the induction motor directly [1]-[11] are concerned with fault characterisation and detection, and where protection hardware is realised it is a local device with no provision for reporting the measured quantities beyond the machine. Second, the studies that do provide remote visibility [12]-[14] are concerned with metering and load control rather than with motor protection, so their measurements are not the quantities that determine motor health. The gap lies between the two groups: a low-cost implementation in which the same embedded controller both executes the protection decision locally and publishes the underlying measurements to a cloud platform for remote observation. The system described in the following sections is an implementation directed at that gap.

IV. Proposed System

The proposed system is organised as a single measurement-and-decision loop with two independent output paths, one local and one remote. The induction motor under protection is the monitored object. Three sensing elements are associated with it: a temperature sensor placed in the circuit to detect the operating temperature, a current sensor that senses the line current, and a voltage sensing channel that follows the supply voltage. The outputs of the three sensors are analogue signals and are taken to three separate analogue input channels of the ESP32 microcontroller, which forms the centre of the architecture.

The microcontroller performs four functions in sequence. It digitises each of the three analogue inputs; it converts the conversion results into engineering units using the scaling factors written into the firmware; it compares each converted value against the corresponding threshold and forms a protection decision; and it distributes the results. The protection decision drives a relay through a Darlington driver stage. The relay contact is in series with the motor supply, so that when the controller de-energises the relay coil the contact opens and the motor is disconnected from the 230 V AC line. This is the protective action of the system and it is taken entirely locally, without reference to the network.

The distribution of results follows two paths. On the local path the three measured quantities are written to a 16x2 liquid crystal display, which is mounted with the controller and gives an operator at the machine an immediate reading of temperature, current and voltage together with the status of the network connection. On the remote path the same three quantities are passed to the Wi-Fi interface of the ESP32, which opens a connection to the ThingSpeak server and writes the values into three separate channel fields. ThingSpeak aggregates and visualises the incoming stream, so the history of the three quantities is available in graphical form to any authorised user with internet access. The remote path is purely informational: loss of the network suppresses the upload but does not affect the trip decision, which continues to be evaluated locally.

Power for the system is derived from the mains. A 12 V transformer steps the 230 V supply down to 12 V, a centre-tap rectifier built with 1N4007 diodes and a 1000 uF capacitor filter produces an unregulated DC rail of 12 V to 14 V depending on the transformer rating, and an LM7805 three-terminal regulator derives the fixed 5 V rail required by the controller, the sensing circuits and the display. The relay and relay driver are operated from the 12 V rail. Decoupling capacitors of 0.1 uF are placed near the analogue, digital and microcontroller devices to suppress the spikes produced by the inductive load and by contact sparking, and a 1000 uF, 25 V capacitor at the regulator output counteracts the loading effect when a high-current source is being driven. The complete arrangement is shown in Fig. 1.

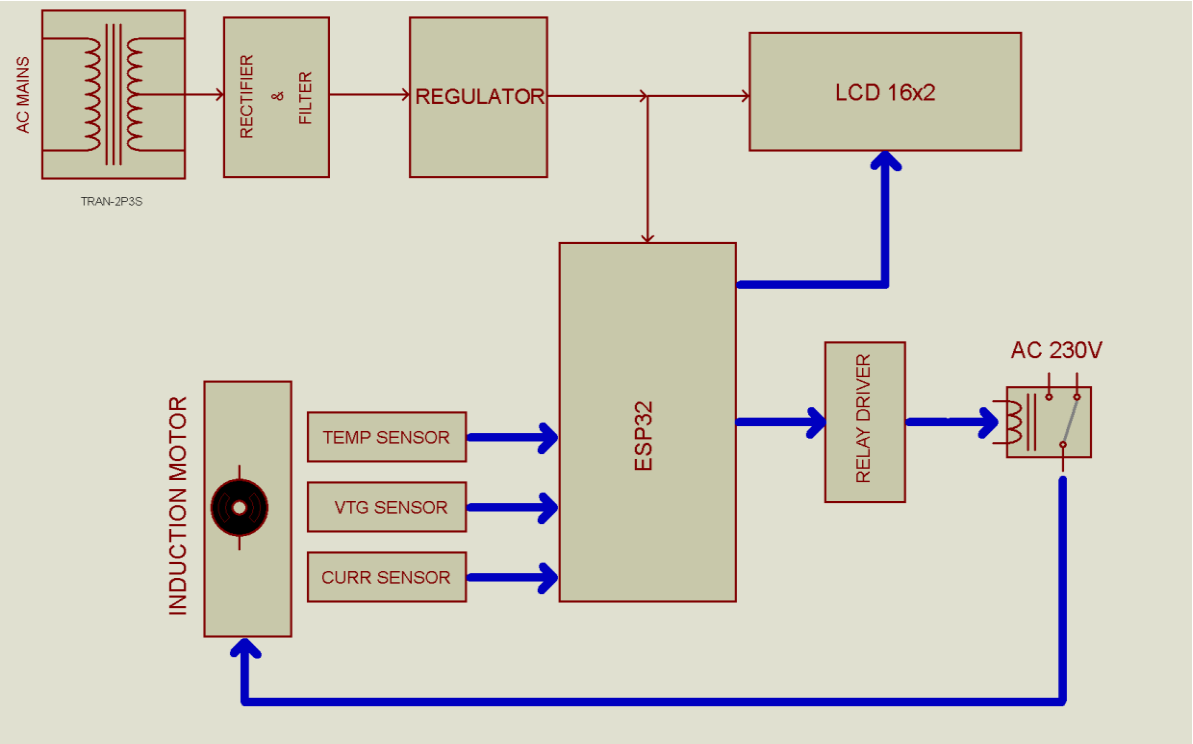

Fig. 1. Block diagram of the implemented IoT-based induction motor monitoring and protection system.

The architecture may therefore be summarised as three cascaded stages followed by a fork. The sensing stage converts physical quantities at the motor into analogue voltages. The processing stage, consisting of the ESP32 alone, converts those voltages into numbers, scales them and applies the protection rule. The actuation stage, consisting of the driver and the relay, translates the protection decision into a physical interruption of the motor supply. The fork is the pair of reporting paths, the LCD for local monitoring and Wi-Fi with ThingSpeak for remote monitoring, both of which are driven from the same set of measurements that produced the protection decision. Consistency between what the operator sees locally, what is visible on the cloud dashboard and what the protection logic acted upon is therefore structural rather than incidental.

V. System Implementation

This section describes the prototype as it was actually constructed and programmed. The circuit diagram of the implemented system is given in Fig. 2, and the components used are listed in Table I.

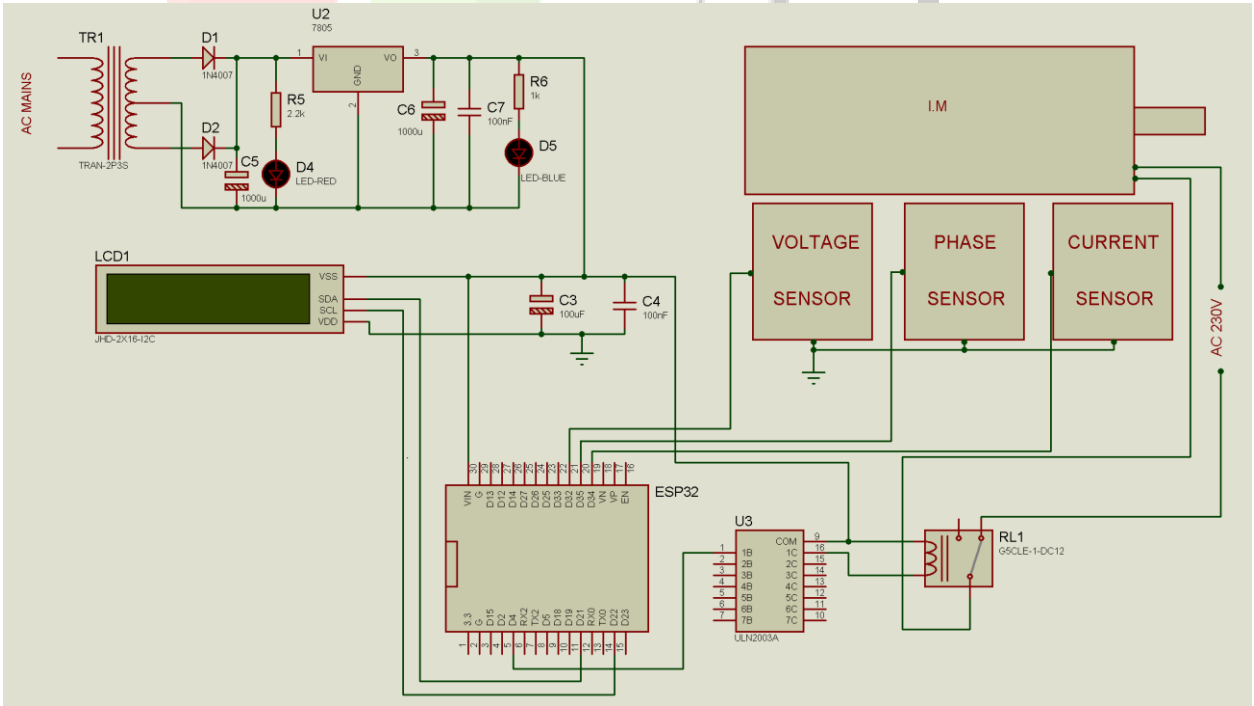

Fig. 2. Circuit diagram of the implemented system.

5.1 Hardware Implementation

ESP32 implementation. The ESP32 is a low-cost, low-power system-on-chip from Espressif Systems and is the successor to the ESP8266. It is built around a 32-bit Tensilica Xtensa LX6 processor, available in dual-core and single-core variants, operating at a clock frequency of up to 240 MHz and delivering processing performance of up to 600 DMIPS. The device provides 520 KB of internal SRAM and 448 KB of boot ROM, and typically supports 4 MB or 8 MB of external SPI flash for program storage. Its wireless subsystem comprises 2.4 GHz Wi-Fi to IEEE 802.11 b/g/n at rates up to 150 Mbit/s together with Bluetooth v4.2 BR/EDR and Bluetooth Low Energy. It operates from a supply in the range 2.2 V to 3.6 V and draws only 10 uA to 20 uA in deep sleep while keeping the internal real-time clock running. In this implementation the module carries the entire application: it owns the three analogue input channels, the display interface, the relay control output and the network stack. The integration of Wi-Fi on the same die as the processor is the property that removes the need for a separate communication gateway and is the principal reason for selecting the device. The board used is a 30-pin ESP32 module, as reflected in the printed circuit board layout of Fig. 3.

Temperature sensor implementation. A temperature sensor is placed in the circuit to detect the operating temperature of the machine. Its output is an analogue voltage proportional to temperature and is taken to the first analogue input channel of the controller. The firmware reads this channel and converts the result to degrees Celsius by multiplying the conversion result by 500 and dividing by 1024, which is the scaling appropriate to a linear analogue temperature sensor referred to a 5 V full-scale range on a 10-bit converter. The converted value is the quantity that is displayed, compared against the thermal threshold and uploaded as the first cloud field.

Current sensor implementation. The current sensor senses the line current drawn by the motor and is referred to in the project report as measuring current at the output by means of a current transformer. Its analogue output is connected to the second analogue channel. Because the current waveform is alternating and the sampled value therefore fluctuates from one conversion to the next, the firmware does not use a single reading. Ten successive conversions are taken with a delay of 10 ms between them, each conversion result is multiplied by a fixed scaling factor of 1.5, the ten scaled values are accumulated, and the accumulator is divided by ten. The resulting arithmetic mean is the current value used for display, comparison and upload. This ten-sample averaging is the only signal conditioning applied in software and it serves to suppress the sample-to-sample variation of the rectified sensor output.

Voltage sensing implementation. The supply voltage is brought to the third analogue channel through a voltage sensing network. The firmware reads this channel inside a ten-iteration loop and applies a two-stage conversion to each reading: the conversion result is first multiplied by 3.37 and by 4.3 and divided by 1024, which recovers the voltage present at the sensing node, and the result is then multiplied by 24.1, which is the divider ratio that refers the sensed value back to the line side. The value obtained is expressed in volts and is compared against the two voltage limits held in the program.

Relay implementation. The protective element is an electromagnetic relay whose contact is connected in series with the 230 V motor supply. The relay used in the implemented circuit is a 12 V DC coil type. Because the coil current exceeds what a microcontroller pin can supply, and because the coil is an inductive load, the relay is not driven directly. A ULN Darlington transistor array is interposed between the controller output and the coil. Each channel of the array is a Darlington pair with a common emitter and an integral suppression diode for inductive loads, rated for a peak load current of 600 mA and 500 mA continuous, and able to withstand at least 50 V in the off state. The suppression diodes internal to the array absorb the reverse voltage generated when the coil is de-energised, which is what makes the array suitable for driving a relay from a logic-level output. The relay contact is a changeover type with normally open, normally closed and common terminals, so that either fail-safe or fail-on wiring is possible; in this implementation the motor supply is taken through the contact in such a way that de-energising the coil removes the supply.

LCD implementation. Local indication uses a 16x2 alphanumeric liquid crystal display module, that is, a display of sixteen characters per line on two lines, each character being formed in a 5x7 pixel matrix. The module has a command register, which receives instructions such as initialisation, clearing, cursor positioning and display control, and a data register, which receives the ASCII code of the character to be shown. The module is preferred over seven-segment and multi-segment LED displays because it is inexpensive, easily programmed and able to display special and custom characters without restriction. In the implemented circuit the display is an I2C-interfaced 16x2 module, which reduces the connection to the controller to two signal lines in addition to power and ground.

Power supply implementation. The supply arrangement follows the description already given in Section IV: a 12 V transformer, a centre-tap rectifier using 1N4007 diodes, a 1000 uF filter capacitor, and an LM7805 regulator producing the 5 V rail. The LM78XX family used here provides an output current of up to 1 A with thermal overload protection, short-circuit protection and output transistor safe-operating-area protection, and is supplied in a TO-220 package. The regulator is fed from the unregulated 12 V to 14 V rail, which also supplies the relay and the driver array separately, so that the switching transients of the relay coil do not appear on the regulated rail used by the controller and the sensing circuits.

Table I. Components used in the implemented system

S. No.	Component	Quantity	Function
1	ESP32 module	1	Main controller; digitises the three analogue channels, applies the protection rule, drives the display and relay, and provides the Wi-Fi interface
2	LM7805	2	Three-terminal positive regulator producing the fixed 5 V rail for the controller, sensing circuits and display
3	Terminal block	2	Screw terminals for the incoming mains supply and the switched output to the motor
4	1N4007	2	Rectifier diodes of the centre-tap rectifier following the transformer secondary
5	Resistors 1K, 10K, 240R, 560R	12	Biasing, current limiting and potential division in the sensing and indicator circuits
6	16 MHz crystal	2	Listed in the report; not required by the ESP32 circuit of Fig. 2 (see Section 7.2)
7	Pulse sensor	4	Listed in the report; not read by the firmware and not present in the circuit of Fig. 2 (see Section 7.2)
8	Current sensor	2	Senses the line current drawn by the motor and presents an analogue output to the controller
9	Battery 12 V	1	Auxiliary 12 V source for the relay and driver rail
10	LED indicators	2	Visual indication of the presence of the unregulated and regulated supply rails
11	Wi-Fi	1	Wireless connectivity for the ThingSpeak upload; provided on-chip by the ESP32 in the circuit of Fig. 2
12	33 pF capacitor	2	Oscillator loading capacitors associated with the crystal entry above
13	28-pin IC base	1	Listed in the report; belongs to an eight-bit controller implementation (see Section 7.2)
14	Burg strip	2	Header connectors for the display and sensor interconnections on the board
15	Mains cord	1	230 V AC input lead to the transformer primary
16	Connectors	4	Board-to-wire connections for the sensor leads and the relay output
17	IRF540	5	Listed in the report; no function identifiable in the circuit of Fig. 2 (see Section 7.2)
18	PCB, 4 in x 6 in	1	Single-sided board carrying the controller, supply and interface circuitry
19	Other hardware	–	Standoffs, fasteners, sleeving and interconnecting wire
20	PCB fabrication	–	Screen printing, etching and drilling of the board as described in Section 5.4

5.2 Software Implementation

The application firmware is a single program with the conventional embedded structure of an initialisation routine executed once, a main loop executed indefinitely, and one subroutine that handles the network transaction. It is written in C for the Arduino toolchain and is compiled and uploaded to the board over the USB serial interface. The following description explains the algorithm of the program as it exists in the project report; the listing itself is not reproduced.

Initialisation. The initialisation routine configures the display for sixteen characters by two lines, opens the hardware serial port and the software serial port used for the network interface at 9600 baud, and declares the relay pin and the display reset pin as outputs. It then writes an identification banner to the display for two seconds so that the operator has visual confirmation that the controller has started.

Sensor reading and unit conversion. The main loop begins by resetting the display, then reads the three measured quantities in a fixed order. The temperature channel is read once and converted as described in Section 5.1. The current channel is read ten times in a counted loop, each result being scaled and accumulated, and the accumulator is then divided by ten to produce the averaged value. The voltage channel is read inside a similar ten-iteration loop, each reading being converted to volts by the two-stage scaling described earlier. The three converted values are the only state that the remainder of the loop uses.

Display. After each quantity is converted the loop clears the display, positions the cursor at the start of the first line, writes a short label and the numeric value, and holds the reading for two seconds before proceeding. The three quantities are therefore presented in sequence rather than simultaneously, and the complete display cycle for one pass of the loop occupies approximately six seconds.

Threshold comparison and relay control. The three converted values are then tested against the thresholds fixed in the program. The outcome of the test determines the state written to the relay pin: a logic high energises the relay and keeps the motor connected, while a logic low de-energises it and disconnects the motor. The exact form of the comparison, together with the threshold values, is set out in Section VII, and a verification note in Section 7.2 records a defect found in the logic as written.

Network transaction. Whichever branch of the comparison is taken, the loop then calls the network subroutine, so that data are uploaded on every pass regardless of whether the motor is running or has been tripped. The subroutine is described in Section 5.3. On return, control passes to the top of the loop and the cycle repeats indefinitely.

5.3 IoT and ThingSpeak Integration

ThingSpeak is a platform that aggregates, visualises and analyses live data streams in the cloud. It allows devices to be configured to send data using common IoT protocols, visualises sensor data in real time, aggregates data on demand from third-party sources, supports analysis of the collected data with MATLAB, runs analytics automatically on a schedule or in response to events, and allows IoT systems to be prototyped without setting up servers or developing web software. These properties make it appropriate for a prototype in which the requirement is remote visibility of a small number of scalar quantities rather than a bespoke dashboard.

In the implemented system the upload is performed by a dedicated subroutine that is called once per pass of the main loop. The subroutine carries out four steps. It first configures the network interface for a single connection and checks the response, writing either a connection-established or a connection-error message to the second line of the display so that the operator can see the link state. It then opens a TCP connection to the ThingSpeak server on port 80 and again reports the outcome on the display. It then composes the update request, which carries the channel write API key and three field parameters, and sends first the length of the request and then the request itself. Finally it waits for the server acknowledgement, displays either a data-sent or an error message, and closes the connection.

The mapping between the measured quantities and the channel fields is fixed in the request string and is given in Table III. Because the three fields are written in a single transaction, the values plotted on the ThingSpeak channel are always mutually consistent, that is, they belong to the same pass of the measurement loop. The connection is opened and closed on every pass rather than being held open, which keeps the interface simple at the cost of the connection setup overhead being incurred repeatedly.

The channel write API key is embedded as a literal in the request string in the source listing contained in the project report. For a deployed system this key would be treated as a credential and kept out of the source, but in the prototype it is present in the program text.

Table III. Mapping of measured quantities to ThingSpeak channel fields

ThingSpeak field	Quantity uploaded	Firmware variable
field1	Temperature of the machine, in degrees Celsius after scaling	temp
field2	Averaged line current in the scaled units produced by the current channel	currsample
field3	Supply voltage referred to the line side, in volts	vtg

5.4 PCB Implementation

The circuit was realised as a single-sided printed circuit board. Two methods of board making were considered, photo printing and screen printing; photo printing is accurate but expensive, and the screen-printing route was adopted. In this method a resist ink is forced through the openings of a stencil onto the copper surface of the blank board, producing a positive image of the circuit which is then allowed to dry before etching.

The stencil is prepared photographically. A pre-sensitised film is exposed under the artwork for about eight minutes using a number-two photoflood lamp positioned approximately 18 inches above the glass, developed for about ninety seconds in a solution held between 40 C and 46 C, washed in running warm water to remove the unexposed emulsion, dried and transferred to a fine mesh screen stretched over a wooden frame. The frame is hinged to the base of the printing board, and registration guides are placed on three sides of the blank board so that every board is positioned identically. The resist is then drawn across the screen with a rubber squeegee in a single pass, and the printed board is set aside to dry.

Etching removes the unprotected copper to leave the required conductor pattern. Ferric chloride is used as the etchant, being inexpensive, comparatively safe and readily available; ammonium persulphate, chromic acid and cupric chloride are the recognised alternatives. The solution is prepared by dissolving 500 g of ferric chloride crystals to make one litre, and tray rocking is used as the agitation method. Etching is carried out with the solution held between 60 C and 70 C so that the etching time remains consistent from board to board.

The board is then drilled to accept the component leads. Hole size and position must be held to tolerance and edge deformation minimised, and the drill speed is selected accordingly: high-speed steel at 8000 rpm or less for paper laminates, and tungsten carbide at about 15000 rpm for epoxy-glass laminates. The completed layout is shown in Fig. 3. It provides the footprint for the 30-pin ESP32 module, a header for the display, three sensor connectors, the regulator and rectifier components, screw terminals for the incoming supply and the switched output, and mounting holes at the four corners.

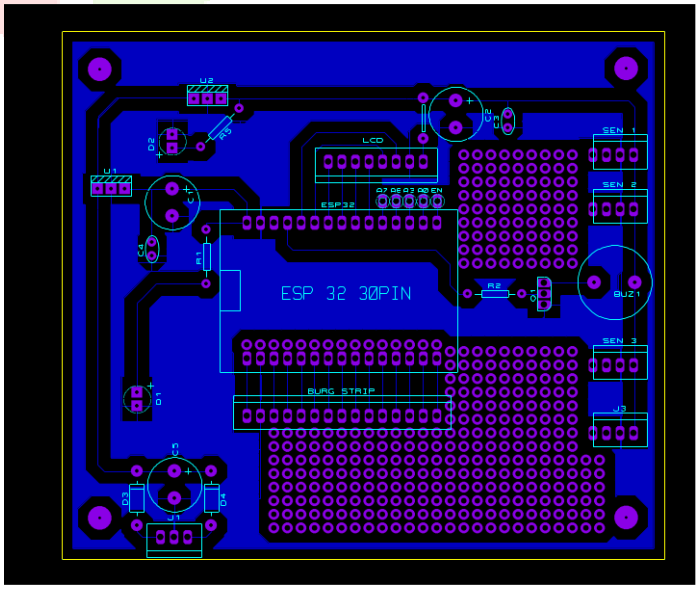

Fig. 3. Printed circuit board layout of the implemented system.

VI. WORKING METHODOLOGY

6.1 Temperature Monitoring

The temperature sensing channel provides the thermal input to the protection decision. The sensor is positioned in the circuit so as to follow the operating temperature of the machine, and it produces a voltage that rises linearly with temperature. On each pass of the main loop the controller performs a single conversion on this channel and applies the scaling given in Section 5.1, which yields a value in degrees Celsius. The value is written to the display, held for two seconds, compared against the thermal threshold and uploaded as the first ThingSpeak field. Because a single conversion is used, no averaging is applied to the temperature channel; this is acceptable in that the thermal time constant of a motor is very long compared with the loop period, so successive conversions differ little.

6.2 Current Monitoring

Current monitoring detects mechanical overloading of the machine. The sensor output is sampled ten times in succession at 10 ms intervals, each sample is scaled by the fixed factor of 1.5 written in the firmware, and the mean of the ten scaled samples is taken. Averaging is necessary here, unlike the temperature channel, because the measured quantity is derived from an alternating waveform and a single conversion would capture an arbitrary point on that waveform. The averaged value is displayed, compared against the current threshold and uploaded as the second ThingSpeak field. An increase in the averaged value above the threshold is interpreted as an overload condition and causes the motor to be disconnected.

6.3 Voltage Monitoring

Voltage monitoring protects the machine against undervoltage and overvoltage of the supply. The sensing node is sampled ten times, and each sample is converted to a line-referred value in volts by the two-stage scaling described in Section 5.1. Unlike the current channel, the accumulated sum is not divided by ten, so the value carried forward to the display, the comparison and the upload is the value obtained from the last iteration of the loop rather than an average; this is noted again in Section 7.2. The voltage test is a band test rather than a single-sided limit, since both a supply that is too low and a supply that is too high are damaging: the former raises the current drawn for a given shaft load and the latter drives the magnetic circuit towards saturation.

6.4 Protection and Relay Control

The protection and relay control element converts the outcome of the three comparisons into a physical action. The controller writes a logic level to the relay control pin, and that level is presented to the input of the Darlington array. When the input is high the corresponding Darlington pair conducts, the relay coil is energised from the 12 V rail, the contact closes and the motor remains connected to the 230 V supply. When the input is low the pair turns off, the coil is de-energised, the contact opens and the motor supply is interrupted. The suppression diode integral to the array clamps the inductive transient generated at turn-off. The action is latching only in the sense that the state persists until the next pass of the loop re-evaluates the comparison; the system does not implement a separate lock-out that would require manual reset, so if the measured quantities return within limits the relay is re-energised automatically on the following pass.

6.5 LCD Monitoring

Local monitoring is provided by the 16x2 display, which performs two distinct functions. On the first line it presents the measured quantities in sequence, each with a short label, so that an operator at the machine can read the present temperature, current and voltage without any additional instrument. On the second line the network subroutine writes status text describing the progress of the upload, namely whether the interface accepted the connection configuration, whether the TCP connection to the server was established, and whether the data were sent successfully or an error occurred. The display therefore serves both as the measurement indicator and as the diagnostic channel for the network link, which is the only means by which a communication failure can be observed at the machine.

6.6 Cloud Monitoring

Cloud monitoring is provided through the ThingSpeak channel. Once per pass of the loop the three measured values are written into fields one, two and three of the channel, and ThingSpeak plots each field against time. The result is that the operating history of the motor, expressed in the three quantities that determine its health, is retained on the server and can be reviewed remotely in graphical form. The cloud path is strictly one-way in this implementation: the system publishes measurements but accepts no commands from the server, so remote operation cannot alter the state of the relay. This is a deliberate consequence of

the architecture rather than an omission, since it guarantees that the protection function remains available and unaltered when the network is unavailable.

VII. Protection Algorithm

7.1 Threshold Logic Implemented in the Firmware

The protection algorithm is a threshold comparison evaluated once per pass of the main loop, after all three quantities have been measured and converted. The thresholds are constants in the source code and are not adjustable at run time. The values, taken directly from the source listing in the project report, are a temperature limit of 75, a current limit of 600 in the scaled units produced by the current channel, and a voltage band bounded by 180 V and 270 V. The protection conditions are summarised in Table II, and the sequence of operations is shown in the flowchart of Fig. 4, which has been drawn from the source listing.

Table II. Protection conditions implemented in the system

Parameter	Normal condition	Fault condition	Protection action
Temperature	temp < 75	temp ≥ 75	Relay de-energised; motor supply disconnected
Current (averaged, scaled units)	currsample < 600	currsample ≥ 600	Relay de-energised; motor supply disconnected
Supply voltage	180 V < vtg < 270 V	vtg ≤ 180 V or vtg ≥ 270 V (see Section 7.2)	Relay de-energised; motor supply disconnected
All quantities within limits	All three conditions satisfied	–	Relay energised; motor supply maintained
Data reporting	Upload executed on every pass	Upload executed on every pass	Three values written to ThingSpeak fields 1 to 3; link status shown on LCD

The intended behaviour is straightforward. While the temperature is below its limit, the averaged current is below its limit and the voltage lies within the band, the relay is held energised and the motor runs. If any one of the three quantities leaves its permitted region the relay is de-energised and the motor is disconnected. The upload subroutine is called in both cases, so the cloud record continues during a trip condition and the trip itself is therefore visible remotely as a change in the plotted quantities.

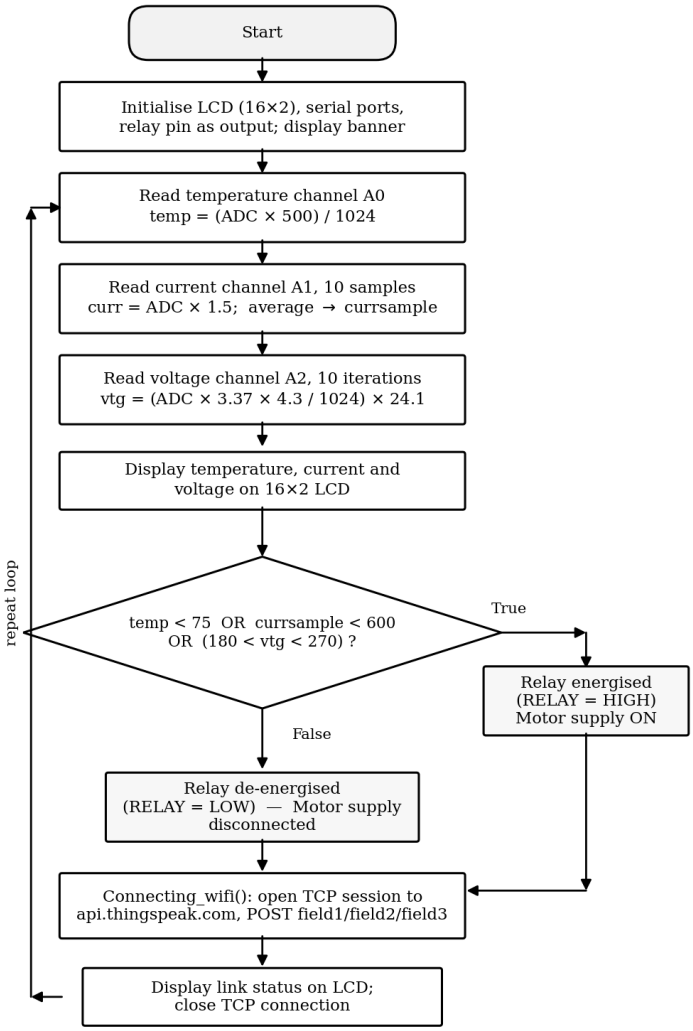

Fig. 4. Flowchart of the protection algorithm as implemented in the firmware.

7.2 Verification Note on the Implemented Logic

Comparison of the source listing with the block diagram, the circuit diagram and the bill of materials in the project report reveals several points that the implementation does not resolve. They are recorded here rather than corrected silently, because correcting them would change the system being reported.

First, the two threshold tests in the listing are written with the logical OR operator rather than the logical AND operator. The condition that energises the relay is satisfied when the temperature is below its limit, or the current is below its limit, or the voltage lies within the band. Because a healthy value of any single quantity is sufficient to satisfy the condition, the relay can be energised while another quantity is in its fault region. The behaviour intended by the design, and described in the project report, requires all three conditions to hold simultaneously and therefore requires the conjunction.

Second, the fault-side test is written as a conjunction of the voltage being less than or equal to 180 and greater than or equal to 270. No value satisfies both, so this part of the test can never evaluate true, and the voltage limits can contribute to a trip only through the first test. The complement of the intended in-band condition is a disjunction, not a conjunction.

Third, the two tests are written as independent conditional statements rather than as the two branches of a single decision. With the operators as written, both conditions can be satisfied on the same pass, in which case the relay is first energised and then immediately de-energised within one iteration.

Fourth, the voltage channel accumulates ten scaled samples but the accumulator is never divided by the number of samples, and the accumulated variable is not the variable used for display, comparison or upload. The value acted upon is that of the final iteration, so the averaging intended for this channel is not realised. The current channel, by contrast, does perform the division correctly.

Fifth, the temperature variable is declared as an integer type while the conversion produces a fractional result, so the displayed and transmitted temperature is truncated to whole degrees.

Sixth, the current threshold of 600 is expressed in the scaled units produced by multiplying the raw conversion result by 1.5. The project report does not state the sensor's transfer characteristic, so this threshold cannot be referred to amperes from the information available, and the value uploaded as the second field is likewise not in amperes.

Seventh, the peripheral interfaces assumed by the source listing do not correspond to the circuit diagram. The listing drives the display through a six-line parallel interface and reaches the network through a software serial port using AT commands, which is the arrangement appropriate to an eight-bit Arduino board with a separate Wi-Fi module, whereas the circuit diagram shows an ESP32 with an I2C display and no separate Wi-Fi module. The listing also names analogue channels A0, A1, A2 and A5, which are Arduino channel designations. The listing therefore appears to be an earlier version carried forward, and it does not compile against the hardware shown in the circuit diagram without modification of the display driver, the analogue channel names and the network interface.

Eighth, the circuit diagram includes a phase sensor block that is not present in the block diagram, is not read by the firmware and is not listed in the bill of materials. The printed circuit board layout provides a buzzer position that is likewise not referred to anywhere else in the report.

Ninth, the bill of materials reproduced in Table I is the list given in the project report. It contains entries that cannot be reconciled with the ESP32 circuit, notably a 16 MHz crystal and a 28-pin IC base, which belong to an eight-bit microcontroller implementation, together with a pulse sensor and IRF540 devices that have no function in the circuit shown. It also omits several components that the circuit requires, namely the temperature sensor, the voltage sensor, the relay, the Darlington driver array, the transformer and the display.

These points do not prevent the prototype from demonstrating the measurement, display, tripping and upload functions, since each of those functions can be exercised individually. They do mean that the combined protection behaviour of the system as programmed differs from the behaviour that the design intends, and any reproduction of this work should begin by resolving them.

VIII. Results and Discussion

A working prototype of the system was constructed and operated. The project report documents the outcome qualitatively: the system employs multiple sensors for temperature, voltage and current; on detection of the respective condition the microcontroller takes action and shows the result on the LCD display; and the system updates data to the ThingSpeak server using an internet connection. The results reported below follow that documentation. No quantitative experimental measurements, calibration records, trip-time measurements or repeated-trial data are contained in the project report, and none are presented here.

Sensor monitoring. The three sensing channels were exercised and produced varying readings that tracked the corresponding physical quantities. The temperature channel returned a value in degrees Celsius after scaling, the current channel returned an averaged value derived from ten successive conversions, and the voltage channel returned a line-referred value in volts. The measurement loop ran continuously without manual intervention, which confirms that the acquisition, scaling and sequencing described in Section V operate as programmed.

LCD output. The display presented the three measured quantities in sequence on the upper line, each preceded by its label and each held for approximately two seconds, and presented the network status on the lower line. The identification banner written during initialisation appeared on power-up, confirming that the controller had started and that the display interface was functional before measurement began. Local indication therefore performed its intended function of allowing an operator at the machine to read the present state without additional instruments.

Relay operation and protection response. The relay responded to the state written by the controller: a logic high on the control pin energised the coil through the Darlington array and maintained the motor supply, and a logic low de-energised the coil and interrupted it. Transitions occurred on the pass of the loop following the change in the measured quantity, so the response of the system is bounded by the loop period, which is dominated by the display hold delays and the network timeouts rather than by the measurement or the comparison. The protection response is thus of the order of seconds rather than milliseconds, which is appropriate to thermal and sustained-overload protection but not to fault clearing, a function that remains with the upstream switchgear.

ThingSpeak communication and remote monitoring. The upload subroutine executed on every pass of the loop and reported its progress on the display, first the configuration of the interface for a single connection, then the establishment of the TCP connection to the server, and finally the acknowledgement of the data. Where a step failed, the corresponding error text appeared in place of the success text, so failures were observable locally. On successful transmission the three quantities appeared in the three fields of the ThingSpeak channel, from which the platform generated the graphical presentation of the data described in the project report. Remote monitoring of the machine was therefore achieved in the sense intended, namely that the operating quantities of the motor could be observed from a browser without physical access to the installation.

Prototype implementation. The circuit was fabricated on a single-sided printed circuit board by the screen-printing and etching route described in Section 5.4, drilled, populated and tested. The board accommodates the 30-pin ESP32 module, the display header, three sensor connectors, the power supply components and the output terminals, and it is compact enough that the report identifies small space requirement as one of the advantages of the design.

Discussion. Taken together, these observations establish that the four functional blocks of the architecture, namely acquisition, local decision, local indication and cloud publication, were each realised and each operated. What they do not establish is the accuracy of the measurements or the correctness of the composite protection behaviour. Accuracy cannot be assessed because the project report contains no comparison of the system's readings against a reference instrument and does not state the transfer characteristics of the sensors; the scaling constants in the firmware are asserted rather than derived from a calibration. Composite protection behaviour cannot be assessed because, as recorded in Section 7.2, the logical structure of the threshold test as programmed does not implement the intended rule. The evaluation of this implementation is accordingly qualitative, being based on the documented operation of the prototype, and a quantitative evaluation would require a calibration campaign against reference instruments together with repeated trip trials at controlled values of each monitored quantity.

IX. Advantages and Applications

The advantages claimed for the implemented system in the project report are the following. The system measures the temperature and current of the induction motor and monitors them automatically, without operator intervention. It provides long-distance communication over Wi-Fi, so that the machine can be observed from outside the plant. The circuit is compact and therefore requires little mounting space. It provides automatic control for overload protection, so that the disconnection of the motor following an overload requires no human action.

Two further advantages follow from the architecture rather than being stated separately. The first is that the protection decision is local, so a network outage suppresses only the reporting function and leaves the protective function intact. The second is that the same measurements drive the local display, the trip decision and the cloud record, so the three views of the machine cannot disagree.

The applications identified for the system are the measurement of temperature and current with overload protection, which is the primary use; implementation as a power-theft monitoring arrangement, since the same measurement and reporting chain applies to any monitored electrical load; and use within an automation system to control electrical equipment from a long distance, using the relay output as the controlled element.

X. Limitations

The limitations of the implementation as reported are the following.

- Protection is threshold-based. The system compares each quantity against a fixed constant and takes a binary decision. It implements no inverse-time characteristic, no thermal model of the machine and no trip delay, so it cannot distinguish a starting inrush from a sustained overload other than by the coarse effect of the loop period.
- The thresholds are compiled into the firmware. Changing a limit requires the program to be edited, recompiled and re-uploaded, since no run-time configuration interface is provided.
- Cloud monitoring depends on Wi-Fi. Loss of the wireless link or of internet connectivity suspends the upload, and although the protection function is unaffected, the remote view of the machine is lost and no local buffering of the missed samples is performed.
- The set of monitored parameters is limited. Temperature, current and voltage are measured on a single channel each. Vibration, speed, power factor, phase sequence and per-phase quantities are not measured, so faults that manifest primarily in those quantities are not detected.

- Single-phase quantities are measured on what is a three-phase machine, so unbalance between phases and single-phasing conditions cannot be detected directly from the acquired data.
- Quantitative experimental validation is limited. The project report documents the operation of the prototype qualitatively and contains no calibration data, accuracy figures, trip-time measurements or records of repeated trials, so the performance of the system has not been characterised numerically.
- There is no predictive fault diagnosis. The system reports and reacts to the present state of the machine but performs no trend analysis, no residual-life estimation and no classification of the fault type; the interpretation of the uploaded history remains a task for the human observer.
- The firmware defects recorded in Section 7.2 constitute a limitation in their own right, since the composite protection rule as programmed does not correspond to the rule the design intends.

XI. Future Scope

The following extensions are proposed for future work. None of them is implemented in the prototype described in this paper.

- A dedicated mobile application, so that the operating quantities and the trip status can be viewed on a handset without a browser, and so that acknowledgements and settings could be handled from the field.
- Automatic alerting by SMS and electronic mail on a trip or on a threshold approach, so that the responsible engineer is informed without having to consult the dashboard.
- Extension of the measurement to full three-phase monitoring, with per-phase current and voltage channels, which would permit direct detection of unbalance, phase failure and single phasing rather than inference from a single channel.
- Addition of vibration monitoring using an accelerometer, which would extend coverage to bearing degradation, misalignment and mechanical looseness, none of which is reliably visible in the electrical quantities at an early stage.
- Replacement of the fixed thresholds by a machine learning model trained on the accumulated operating history, so that the decision boundary reflects the behaviour of the particular machine rather than a constant chosen in advance.
- Development of a predictive maintenance function that estimates remaining useful life from trends in the monitored quantities and schedules intervention before failure.
- Historical analytics over the stored channel data, including duty-cycle summaries, overload histograms and correlation of trips with supply-side events.
- An advanced cloud dashboard with multi-machine views, configurable alarm limits, user roles and export of records, in place of the standard channel plots used in the prototype.

XII. Conclusion

This paper has described the implementation of an Internet of Things based monitoring and protection system for an induction motor, built around an ESP32 microcontroller. A working prototype was constructed in which temperature, current and supply voltage are acquired through analogue channels, converted to engineering units by scaling factors embedded in the firmware, compared against thresholds held in the program, and used to control an electromagnetic relay through a Darlington driver array so that the motor is disconnected from the 230 V supply when a monitored quantity leaves its permitted region. The same measurements are written to a 16x2 liquid crystal display for local indication and uploaded over the integrated Wi-Fi interface to a ThingSpeak channel, where they are recorded as three separate fields and can be observed remotely. The circuit was fabricated on a single-sided printed circuit board using the screen-printing and etching route, and the supply is derived from the mains through a transformer, a centre-tap rectifier and an LM7805 regulator.

The contribution of the work lies in placing acquisition, protection decision, local indication and cloud publication on a single low-cost embedded platform, so that the protective function remains entirely local and therefore available during a network outage, while the measurements that produced the decision remain visible remotely. The prototype demonstrated each of these functions in operation.

The evaluation reported here is qualitative, being based on the documented operation of the prototype, and the paper has recorded explicitly, in Section 7.2 and Section X, the firmware defects, the component ambiguities and the absence of calibration and trip-timing data that limit what can be claimed. Resolving the logical structure of the threshold test, calibrating the sensing channels against reference instruments,

extending the measurement to all three phases and characterising the trip behaviour quantitatively are the immediate steps required to develop the prototype into a system suitable for industrial service.

ACKNOWLEDGMENT

The authors express their sincere gratitude to the Head of the Department and to the faculty members of the Department of Electrical Engineering for the guidance and the laboratory facilities extended during the design, fabrication and testing of the prototype. The authors also thank the technical staff for their assistance with the fabrication of the printed circuit board and with the assembly and testing of the hardware.

REFERENCES

[1] W. H. Kersting, "Causes and effects of single-phasing induction motors," IEEE Transactions on Industry Applications, vol. 41, no. 6, pp. 1499–1505, 2005.

[2] P. E. Sutherland and T. A. Short, "Effect of single-phase reclosing on industrial loads," in Conference Record of the 2006 IEEE Industry Applications Conference, Forty-First IAS Annual Meeting, vol. 5, Tampa, FL, USA, Oct. 8–12, 2006, pp. 2636–2644.

[3] M. Sudha and P. Anbalgan, "A novel protecting method for induction motor against faults due to voltage unbalance and single phasing," in Proc. IECON 2007, 33rd Annual Conference of the IEEE Industrial Electronics Society, Taipei, Taiwan, 2007, pp. 1144–1148.

[4] P. Pillay, P. Hofmann, and M. Manyage, "Derating of induction motors operating with a combination of unbalanced voltages and over or undervoltages," IEEE Transactions on Energy Conversion, vol. 17, no. 4, pp. 485–491, 2002.

[5] J. Faiz, H. Ebrahimpour, and P. Pillay, "Influence of unbalanced voltage on the steady-state performance of a three-phase squirrel-cage induction motor," IEEE Transactions on Energy Conversion, vol. 19, no. 4, pp. 657–662, 2004.

[6] A. Javed and T. Izhar, "An improved method for the detection of phase failure faults in polyphase induction machines," in Proc. 2009 Third International Conference on Electrical Engineering, Lahore, Pakistan, Apr. 9–11, 2009, pp. 1–6.

[7] S. Chattopadhyay, A. Chattopadhyaya, and S. Sengupta, "Analysis of stator current of induction motor used in transport system at single phasing by measuring phase angle, symmetrical components, skewness, kurtosis and harmonic distortion in Park plane," IET Electrical Systems in Transportation, vol. 4, no. 1, pp. 1–8, 2014.

[8] C.-Y. Lee, "Effects of unbalanced voltage on the operation performance of a three-phase induction motor," IEEE Transactions on Energy Conversion, vol. 14, no. 2, pp. 202–208, 1999.

[9] D. L. Ransom and R. Hamilton, "Extending motor life with updated thermal model overload protection," IEEE Transactions on Industry Applications, vol. 49, no. 6, pp. 2471–2477, 2013.

[10] G. C. Stone, M. Sasic, D. Dunn, and I. Culbert, "Recent problems experienced with motor and generator windings," in Record of Conference Papers, Industry Applications Society 56th Annual Petroleum and Chemical Industry Conference, Anaheim, CA, USA, Sep. 14–16, 2009, pp. 1–9.

[11] M. Cunkas, R. Akkaya, and A. Ozturk, "Protection of AC motors by means of microcontrollers," in Proc. 10th Mediterranean Electrotechnical Conference (MELECON 2000), vol. 3, Lemesos, Cyprus, May 2000, pp. 1093–1096.

[12] H. Das and L. C. Saikia, "GSM enabled smart energy meter and automation of home appliances," IEEE, 2018, ISBN 978-1-4678-6503-1.

[13] O. O. Edward, "An energy meter reader with load control capacity and secure switching using a password based relay circuit," in Proc. Annual Global Online Conference on Information and Computer Technology, IEEE, 2018, ISBN 978-1-4799-8311-7.

[14] Y. Cheng, H. Yang, J. Xiao, and X. Hou, "Running state evaluation of electric energy meter," in Proc. Workshop on Electronics, Computer and Applications, IEEE, 2019, ISBN 978-1-4799-4565-8.