IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Accident Forensic and Analysis System Using IoT-Based Vehicular Data Logging

Dr. LALITHA K, S NAVEEN KUMAR, S B NITHARSANA, S VIJAYAKRISHNA, MEGHA RAJESH

Department of ECE ,SRM Institute of Science and Technology,Ramapuram, Chennai, India

Doi: https://doi.org/10.56975/ijcrt.v14i7.309059

A. Problem Statement

The majority of automobiles and light trucks lack technology that records key data such as speed, crash severity, and weather conditions continuously. Therefore, one has to make assumptions or interpret existing digital data when trying to establish the details surrounding an accident. Such a problem may lead to inaccuracies since the person will only guess at what really occurred in the absence of concrete evidence. In addition, the current technology for handling emergency reporting requires an active human component, resulting in delays that are likely to occur in remote places where accidents happen. After the accident, there are safety concerns including leaking fuel, contaminants in the atmosphere, and possibility of fire outbreak, which are normally not considered. These issues could result in harm to victims and even personnel from the police department and ambulance service.

II. PROPOSED CIRCUIT

This system comprises three layers, including: First, the Sensor and Actuation Layer that consists of many devices like the ADXL345 accelerometer the SW-420 vibration sensor, the MQ-2 gas sensor, the DHT11 temperature sensor, the NEO-6M GPS module, the I2C LCD display, and the active buzzer. Second, the Processing and Communication Layer, which revolves around the ESP32 microcontroller. The ESP32 microcontroller operates alongside the SIM800L GSM module. Features inbuilt Wi- Fi. Third, the Cloud and Analytics Layer comprising ThingSpeak IoT platform and a pipeline for post-incident analytics using Python and Anaconda. The ESP32 monitors the sensors' readings continuously. It analyzes the level of motion detected by the ADXL345 accelerometer, which is calculated using the numerical data derived from the three axes of the ADXL345 accelerometers. If the motion exceeds a specified threshold, the system performs several actions concurrently. These include activating the buzzer, retrieving the GPS coordinates through analysis of GPS NMEA data, sending a text message to the user's mobile phone with the GPS coordinates, and transmitting all sensors' data to ThingSpeak over Wi-Fi.

A. Block Diagram

We have ADXL345, SW-420, MQ-2, and DHT11 sensors communicating with the ESP32 using I2C, GPIO, and ADC methods. The NEO-6M GPS communicates with the ESP32 using UART2 while SIM800L communicates with the ESP32 via UART1. An I2C LCD will display the time- based actions. In case the system detects an accident, the ESP32 triggers the buzzer by activating its GPIO-transistor interface. It will also send an alert via the GSM module. Accident details are stored on the cloud provided by ThingSpeak. Data obtained from the cloud will be retrieved via ThingSpeak REST API for detailed inspection using Python

B. System Requirements

a) ESP32 Microcontroller

The ESP32 manufactured by Espressif Systems is equipped with dual cores that have a maximum speed of 240 MHz.

In addition, it is equipped with 520 KB of RAM, 4 MB of flash memory, and 34 pins used to connect devices. The ESP32 integrates Wi-Fi and Bluetooth connectivity to enable communication between the device and other devices. In addition, it offers many features such as having a unique feature to allow converting sensor signals into digital signals for use by the microcontroller.

b) ADXL345

Analog Devices' ADXL345 is a sensor capable of measuring motion. It is capable of sensing motion along all three axes. It is highly sensitive. Communication between the ADXL345 and the ESP32 is made possible via I2C protocol. Its ability to sample data is very fast; it can sample data up to 3200 Hz. Upon detecting changes in motion, the sensor senses a crash. A threshold of 2.5 g above the normal range is set to differentiate an actual crash from a false alarm.

c) SW-420 Vibration Sensor

SW-420 is one more component in our system that will help us identify accidents. Upon getting a shock impulse, this sensor will transmit a signal to the ESP32 board. One can set the sensitivity threshold level for this sensor, making it trigger only on strong impacts. It is necessary to mention that this sensor acts as a safeguard against receiving any accidental impulses and serves as additional confirmation alongside the ADXL345 sensor that we are indeed dealing with an accident.

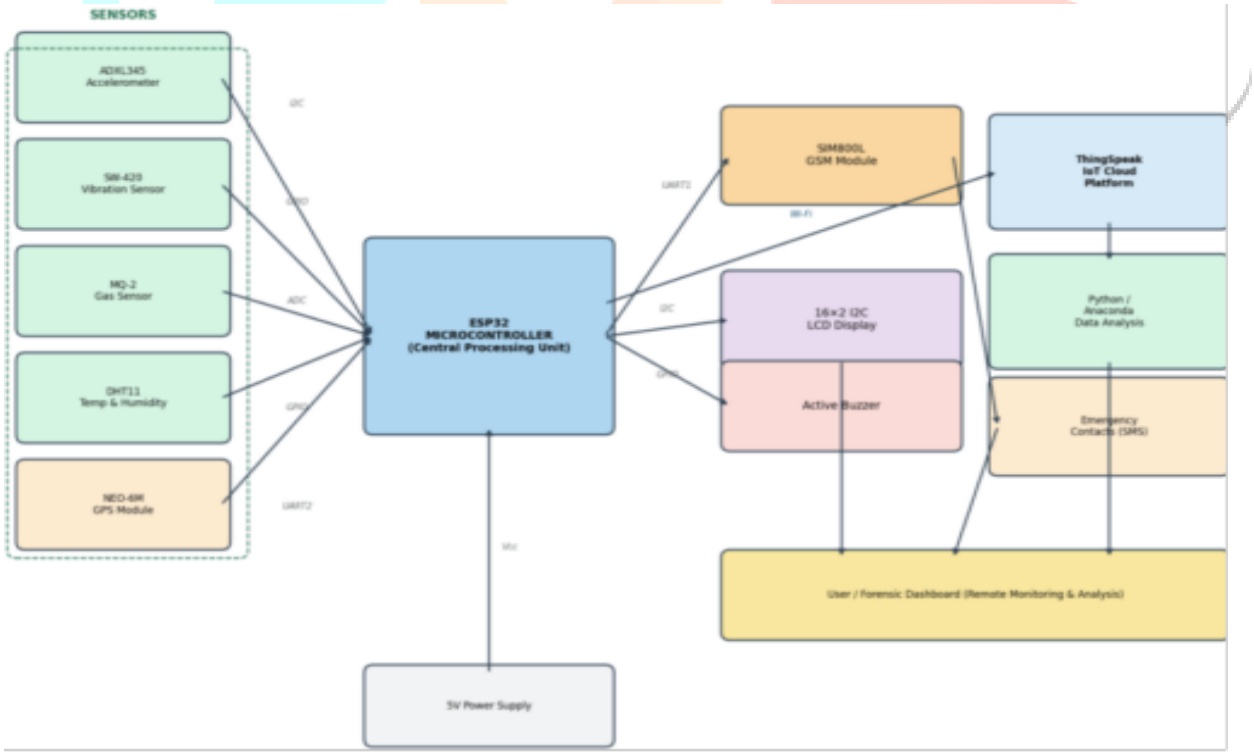

Fig 1 Block diagram of the Accident Forensic and Analysis System

d) MQ-2 Gas Sensor and DHT11 Temperature Sensor

The MQ-2 Gas Sensor senses gases such as LPG and methane within a range of 200 ppm to 10,000 ppm using an element. The DHT11 Temperature Sensor provides data on temperature and humidity in the atmosphere. It tells us the temperature readings between 0 to 50 degrees centigrade and humidity levels between 20% and 90%. This helps in finding out whether there is a fire or gas leak after an accident.

e) NEO-6M. Sim800l GSM Modules

The NEO-6M GPS module can locate its position with an accuracy of 2.5 meters. This operation will take less than 27 seconds. Communication between the devices is possible through a proprietary protocol with data transmission rate at the speed of 9600 bits per second. The SIM800L GSM module can send SMS notifications. In
order to accomplish this function, this module employs specific instructions. For its operation, it requires the current source capable of providing 2A.

f) Software: Arduino IDE, Embedded C and Python/Anaconda

The software that we will use to program the device is called Arduino Integrated Development Environment (IDE). The language we use in the programming is known

as Embedded C. If we wish to analyze the data provided by
our sensor system, then we use Python programming language. For analyzing purposes, we use software packages such as Pandas and Numpy. We can even create graphs using matplotlib package. Jupyter notebook is used in all this process. We can gather data from sensors such as MQ-2 Gas Sensor and DHT11 Temperature Sensor. The sensors we can use to gather and share the data are NEO- 6M GPS, and SIM800L GSM Module are near to the actual location. Satellite pictures confirmed this to be true. The error margin is 3.5 meters.

III.RESULTS AND DISCUSSION

The system underwent testing cycles to evaluate its functionality. This was done by impacting the prototype with a hammer whose force was consistent in each cycle. The prototype was also tested by exposing gas around the MQ-2 sensor. The prototype was also subjected to tests for consistency in the location of the GPS in the open environment.

Fig 3 ThingSpeak Cloud Dashboard - Live Sensor Data Visualisation (Temperature, Gas and Acceleration fields)

TABLE I. PERFORMANCE EVALUATION RESULTS

Performance Metric	Result	Target
Accident Detection Accuracy	100%	>90%
False Positive Rate	0%	<5%
Avg. SMS Alert Latency	6.3 s	<10 s
GPS Position Accuracy	<3.5 m	<5 m
MQ-2 Gas Response Time	~8 s	<15 s
ThingSpeak Upload Success	100%	>95%
DHT11 Temp. Accuracy	±1.8 C	±2 C

ThingSpeak successfully captured the ten accident events. This was achieved within fifteen seconds. In

addition, all required data regarding each event were recorded correctly. After analyzing the accident data in our Python software, we could see that all accident instances were captured. The accident detection procedure, which does not rely on MOSFET, has its merits. These include the capability to adjust its sensitivity without necessarily changing the hardware. Moreover, there is no limitation as regards what
one can do with sensor data collected by this system, for instance alerting and uploading data into the cloud. All this is done in software. The accelerometer, GSM module, and GPS modules operate smoothly as expected.

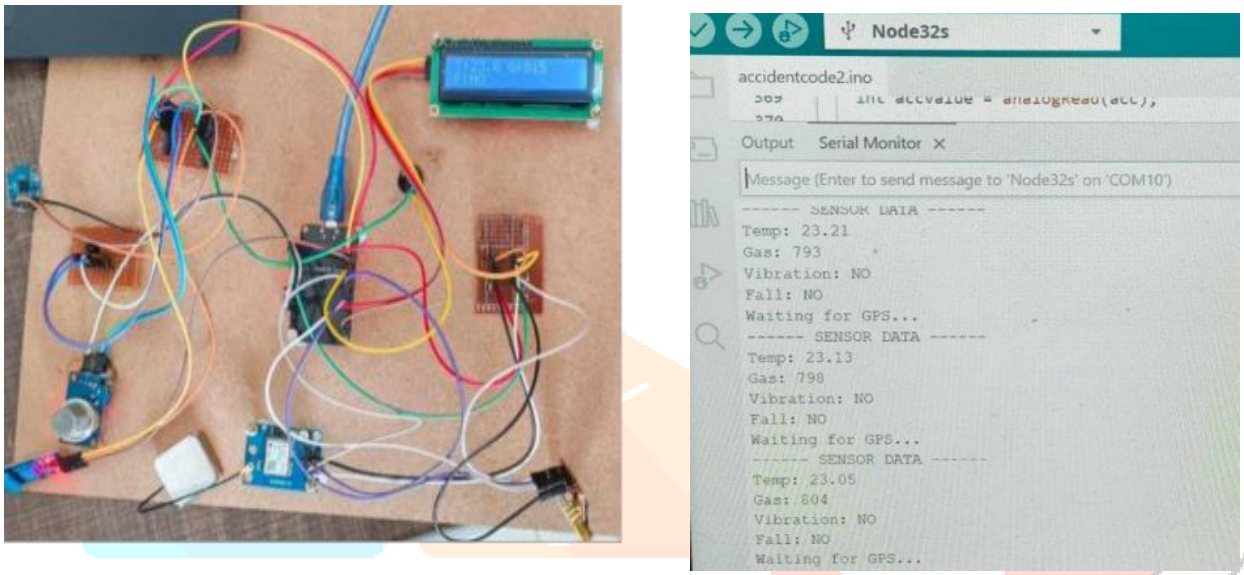

Fig 2 Final Hardware Prototype - Complete Assembled System with ESP32, Sensors, GPS, GSM Module

The ADXL345 accelerometer has been found to be excellent in detecting impacts more than 2.5g. The maximum acceleration recorded when tested using normal movements was 1.2g, showing no false triggering by the accelerometer. The SIM800L GSM module successfully sent SMS notifications whenever required. This was achieved in ten tests conducted on the module. The time taken to do this varied from 4.8 seconds to 9.1 seconds. The variation is attributed to the GSM network delay. The NEO-6M GPS module returned location coordinates that*Fig 4 Arduino IDE Serial Monitor– Real-Time Sensor Readings (Temperature, Gas, Vibration, Fall Status and GPS)*

IV. CONCLUSION

This paper talks about an accident system with the capability of the internet of things technology. This kind
of system can detect accidents accurately. The system can provide alert services within six and a half seconds of accident detection. Besides, all the activities taking place within the system will be recorded on the cloud. Accidents are detected by the system through the use of sensors. Location of the vehicle involved in the accident can be tracked through the use of GPS. Similarly, emergency alerts could also be sent using GSM technology. Data collected from all these activities will be stored at the ThingSpeak cloud storage facility. It is a cost-effective process. The accident detection system proves the fact that
Internet of things can be used for creating black box systems. We could use affordable and accessible components as well as open-source tools for creating the system. There are some improvements planned in the future for the system. Intelligence-based anomaly detection technology will be integrated into the system using TensorFlow lite micro. Additionally, we plan on using the fourth generation long-term evolution network to increase coverage area. Alerts will be able to reach
national emergency dispatch centers while the system will also be able to capture videos of accidents. The Internet of Things accident system will have a wide variety of applications.

V. REFERENCES

[1] R. Prabhu, J. V. Ananthi, and C. Kavitha, 'IoT-Based Vehicle Accident Detection and Alert

System,' Int. J. Recent Technol. Eng., vol. 8, no. 2, pp. 1580-1585, Jul. 2019. DOI: 10.35940/ijrte.B2085.078219.

[2] B. Santhosh Kumar, R. Karthik, and D. Priyanka, 'Real- Time Vehicle Accident Detection and Reporting System Using IoT,' in Proc. IEEE ICCSP, Chennai, 2020, pp. 1012- 1016. DOI: 10.1109/ICCSP48568.2020.9182046.

[3] N. Nalini and M. Geethanjali, 'Black Box System for Vehicles Using IoT,' J. Phys.: Conf. Ser., vol. 1979, no. 1, p. 012068, 2021. DOI: 10.1088/1742-6596/1979/1/012068.

[4] M. Sudhan, K. Tamilarasi, and G. Vinothkumar, 'Accident Alert and Ambulance Rescue System Using GSM and GPS,' Int. J. Eng. Res. Technol., vol. 9, no. 5, pp. 431-435, 2020. DOI: 10.17577/IJERTV9IS050431.

[5] G. Ramesh, S. Varshini, and K. Krishnamurthy, 'Smart Vehicle Accident Detection Using ML and IoT,' in Proc. IEEE ICISS, 2021, pp. 1487-1492. DOI: 10.1109/ICISS53185.2021.9633989.

[6] A. Arulmurugan and C. Saravanan, 'IoT-Based Monitoring for Hazardous Gas Detection in Vehicles,' Mater. Today: Proc., vol. 45, pp. 2229-2233, 2021. DOI: 10.1016/j.matpr.2020.10.572.

[7] R. Balamurugan, M. Vijayalakshmi, and E. Sathiyamoorthy, 'Cloud-Based Data Logging Using ThingSpeak,' Proc. Comput. Sci., vol. 171, pp. 1039-1047, 2020. DOI: 10.1016/j.procs.2020.04.111.

[8] T. Goyal, A. Sharma, and R. Bhardwaj, 'ESP32-Based IoT Systems for Real-Time Sensor Data Acquisition,' J. Ambient Intell. Humanized Comput., vol. 13, pp. 5187- 5199, 2022. DOI: 10.1007/s12652-021-03642-3.

[9] World Health Organisation, 'Global Status Report oRoad Safety 2023,' WHO Press, Geneva, 2023.

[10] Espressif Systems, 'ESP32 Technical Reference Manual,' v4.5, 2022. [Online]. Available: https://www.espressif.com.