www.ijarp.com ISSN 2456-9992 Page: 01-15

Research Article

International Journal Advanced Research Publications

Volume: 02
Issue: 10

IOT-BASED VEHICLE ACCIDENT DETECTION AND GPS-ENABLED EMERGENCY ALERT SYSTEM

[*1]S. Tamilarasan, [1]J. Selva Harish, [1]B. V. Rishikesh, [2]T. J. Saravanan

[1]Department of Information Technology, K.L.N. College of Engineering, Pottapalayam, Sivaganga, Tamil Nadu, India.

[2]Assistant Professor, Department of Information Technology, K.L.N. College of Engineering Pottapalayam, Sivaganga, Tamil Nadu, India.

Article Received: 07 September 2026, Article Revised: 27 September 2026, Published on: 17 October 2026

***Corresponding Author: S. Tamilarasan**

Department of Information Technology, K.L.N. College of Engineering, Pottapalayam, Sivaganga, Tamil Nadu, India.

Doi: https://doi-doi.org/101555/ijarp.5508

ABSTRACT

Road accident victims often do not receive timely help because the occupants may be injured or unconscious and unable to tell anyone where the accident happened. This paper presents the Accident Messaging System (AMS), a low-cost Internet of Things (IoT) prototype that detects a possible vehicle accident, warns the occupants, gives a short chance to cancel a false alert, and sends the vehicle location to a predefined contact. An ESP32 microcontroller coordinates an MPU6050 accelerometer, an SW-420 vibration sensor, a NEO-6M Global Positioning System (GPS) receiver, a SIM800L Global System for Mobile communications (GSM) module, a buzzer, a push button and a regulated power-supply stage. A possible accident is flagged when the total acceleration exceeds a threshold of 1.90 g or the vibration sensor is triggered. The buzzer then sounds and a five-second window allows the user to cancel the alert. If it is not cancelled, the system obtains the GPS coordinates, builds a Google Maps link and sends it as a Short Message Service (SMS) message through the SIM800L, which is powered from a dedicated regulated supply. The paper describes the architecture, wiring, power design, detection algorithm, calibration steps and test plan of the prototype, and discusses its limitations, such as the 100 ms polling interval, the GPS start-up delay and the dependence on 2G network coverage. The design is modular and can be extended with higher sampling rates, machine-learning-based severity classification and cloud connectivity.

INDEX TERMS: Accident Detection, Internet of Things, ESP32, GPS Tracking, GSM Emergency Messaging, Vehicle Safety.

The Accident Messaging System (AMS) supports Sustainable Development Goal 3 (Good Health and Well-being) by helping detect possible vehicle accidents and sending emergency SMS alerts with GPS location details, which can help emergency responders locate accident victims more quickly. It also supports SDG 9 (Industry, Innovation and Infrastructure) by using IoT sensors, GPS technology, and GSM communication to develop an innovative accident detection system. Additionally, it contributes to SDG 11 (Sustainable Cities and Communities) by promoting road safety and supporting safer communities through timely accident notifications.

I. INTRODUCTION

Road safety is one of the most crucial concerns for any country, since the time taken to inform someone after a crash strongly affects how quickly medical help arrives. The World Health Organization estimates that over one million people die on the world's roads every year [1]. However, in many accidents the occupants are injured or unconscious and cannot make a call or send a message, so the location of the accident reaches family members or emergency responders late or not at all.

Modern vehicles and vehicle prototypes can carry a great deal of sensing and communication hardware at low cost – accelerometers, vibration sensors, Global Positioning System (GPS) receivers and Global System for Mobile communication (GSM) modules, all controlled from a single microcontroller [2]–[5]. With embedded and Internet of Things (IoT) technologies, however, it is possible to do much more. The vehicle can detect a possible accident by itself, warn the occupants, and send its location to a chosen contact without any human action. Making this practical needs a sensing method that does not trigger too easily, a way to cancel false alarms, and a power supply that can support the GSM module.

Many accident alert designs have been reported. Most combine an accelerometer or vibration sensor with GPS and a GSM module, and some allow the driver to cancel the message with a switch, as reviewed in Section II. Fewer of the reported designs describe a defined cancellation window with a local warning, a map link that can be opened directly from the message, or the high peak current needed by the GSM module, which affects reliability in practice.

In this context, this paper presents the design of an Accident Messaging System (AMS), which brings together accident detection, local alerting, false-alert cancellation, GPS location tracking and GSM emergency messaging on one ESP32-based platform. The system uses an MPU6050 accelerometer and an SW-420 vibration sensor for detection, a buzzer and push button for a 5-second cancellation window, a NEO-6M receiver for location, and a SIM800L module for Short Message Service (SMS), powered from a dedicated regulated supply.

The system is designed with low cost, modularity and extensibility in mind, as a prototype that can be demonstrated on a small vehicle model and later adapted for real vehicles after suitable mechanical, electrical and power validation. It provides a base for further technologies such as machine-learning-based severity classification, cloud dashboards and automotive-grade communication.

II. LITERATURE REVIEW

Vehicle safety has benefited greatly from electronics: airbag controllers, event data recorders and, more recently, in-vehicle emergency call systems. In the European Union, for instance, eCall equipment is mandatory for new vehicle types [6]. Such factory-fitted systems work well, but they are tied to new and often costly vehicles and do little for older vehicles, two-wheelers and low-cost prototypes.

For low-cost systems, the ESP32 microcontroller provides general-purpose input/output (GPIO), I2C and UART interfaces on a single board [2]. Micro-electro-mechanical (MEMS) motion sensors such as the MPU6050 provide 3-axis acceleration and angular rate over I2C [3], GPS receivers such as the NEO-6M output position as NMEA sentences over a serial link [4], [7], and GSM modules such as the SIM800L send SMS through standard AT commands [5], [8]. Libraries such as TinyGPSPlus make GPS parsing simple on Arduino-compatible boards [9].

Accident detection in such systems is generally based on a threshold on acceleration or on a vibration switch. A single sensor or a low threshold reacts to potholes, speed breakers and hard braking, which produces false alerts, while a threshold that is too high can miss real events. For this reason, a combination of sensors and a way for the user to cancel the alert are considered useful.

Several GSM/GPS-based accident reporting systems have been published. Amin and Jalil [10] compare successive GPS speed readings to detect an accident and report the location to an alert centre. Prabha et al. [11] use GSM and GPS technology to send an immediate SMS alert with the vehicle position after an accident. Sarker et al. [12] combine an accelerometer

with ultrasonic sensors and send a Google Maps link by SMS to a nearby hospital and police station; they report detection, location and notification accuracies of 95%, 90% and 100% in tests at 20 locations. Alsayaydeh et al. [13] use an Arduino Uno with an ADXL335 accelerometer, a SIM900A, a NEO-6M, an LCD and a buzzer, and include a cancel button to stop false alerts. Sowah et al. [14] add a Wi-Fi link and a mobile application that shows the location on Google Maps, and report delivery by SMS and to the application in under 11 s. Among the works reviewed here, only [13] describes a user-cancel option. The present work builds on these designs and treats acceleration and vibration sensing, a short cancellation window, a Google Maps link and the peak-current demand of the SIM800L supply together on a low-cost ESP32 platform. In addition, GSM-based designs depend on mobile network availability, and modules such as the SIM800L need 2G coverage.

Sensor fusion, cancel switches and GPS/GSM alerts are therefore established ideas. The contribution of the present work is their integration on a low-cost ESP32 platform with a combined MPU6050 and SW-420 trigger, a timed 5 s cancellation window with a local buzzer, a Google Maps link that opens directly on a phone, and explicit attention to the supply needed by the SIM800L. Table I compares representative approaches with the proposed system.

Table I. Comparison of Existing Accident Alert Approaches.

Approach / Model	Primary Technique	Alert / Output	Remark
Amin and Jalil [10]	GPS speed comparison; GSM/GPRS	Location sent to an alert centre	Trigger is a speed change, not an acceleration threshold
Sarker et al. [12]	Accelerometer + ultrasonic sensors, GPS, GSM	SMS with Google Maps link to hospital and police	Ultrasonic sensor can give false collision alerts [12]
Alsayaydeh et al. [13]	Arduino Uno, ADXL335, SIM900A, NEO-6M	SMS/call with location; LCD, buzzer	Has a cancel button; accelerometer threshold trigger
Sowah et al. [14]	Accelerometer, Wi-Fi, GPS, GSM, mobile app	SMS and app alert, Google Maps	SMS and app delivery in under 11 s reported
Factory eCall / telematics [6]	Dedicated crash sensors with cellular unit	Detection, location, voice call	Higher cost; limited to fitted new vehicles
Proposed AMS	MPU6050 + SW-420 trigger, 5 s cancel window, GPS, GSM	Buzzer, cancel, GPS link, SMS	Needs threshold calibration and a dedicated GSM supply

III. METHODOLOGY

This section gives the method used in the development of an ESP32-based platform which detects a possible vehicle accident, gives a chance to cancel false alerts, finds the vehicle location and sends an emergency SMS. The methodology combines embedded programming, MEMS sensing, GPS and GSM communication, and a regulated power supply.

Unlike simple alert devices that depend on a single sensor, this system monitors both acceleration and vibration, adds a 5-second cancellation window, and sends a Google Maps link which can be opened directly on any phone. The ESP32 coordinates all modules, and a dedicated regulated supply stage powers the SIM800L. The firmware is written in Embedded C/C++ in the Arduino IDE. The design can be segmented into six key phases:

1. Sensing and accident detection
2. Local alert and false-alert cancellation
3. GPS location acquisition
4. Google Maps link generation
5. GSM emergency messaging
6. Power supply management

A. System Architecture and Overall Methodology

The ESP32 is the central controller. The MPU6050, SW-420 and push button provide inputs, the NEO-6M supplies location data, the buzzer gives a local warning, and the SIM800L sends the emergency SMS. A regulated power-supply stage provides about 4.0 V for the SIM800L, and all modules share a common ground. Fig. 1 shows the system block diagram and Fig. 2 shows the process flow. The workflow of the detection and alert sequence is summarized in Fig. 3.

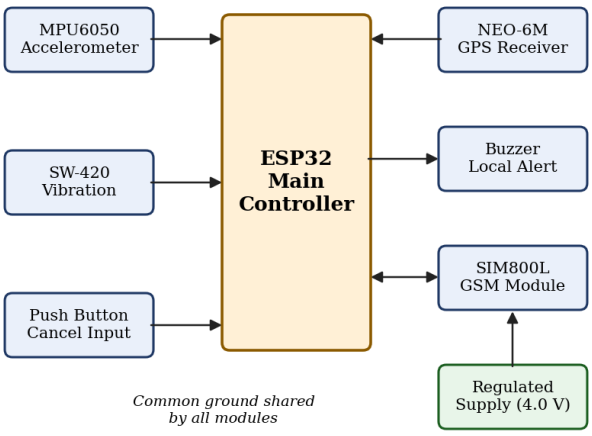


Fig. 1. System block diagram of the Accident Messaging System.

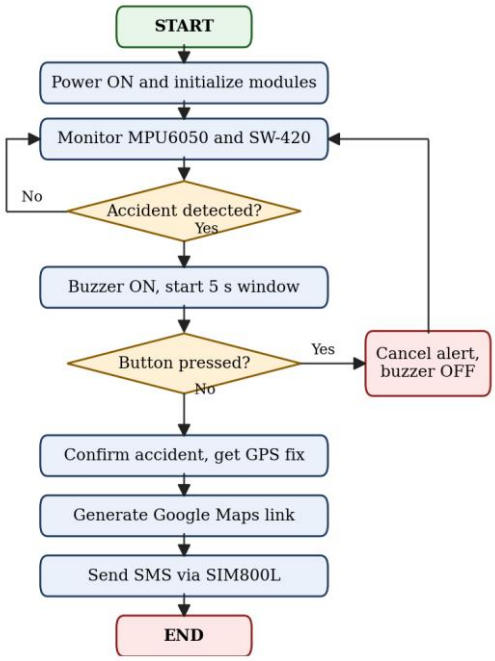


Fig. 2. Process flow of the Accident Messaging System.

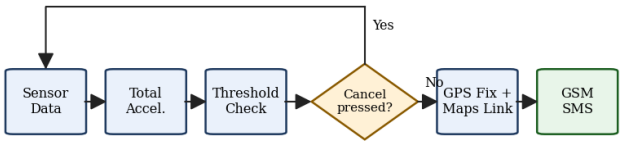


Fig. 3. Accident detection and emergency alert workflow.

B. Hardware Components and Power Supply

The ESP32 board provides the GPIO, I2C and UART interfaces needed by all modules [2]. The MPU6050 contains a 3-axis accelerometer and gyroscope and is read over I2C [3]. The SW-420 gives a digital output when vibration or impact is sensed. The NEO-6M outputs NMEA data over UART at 9600 baud [4], and the SIM800L is a GSM/GPRS module used for SMS [5]. A push button cancels false alerts and a 3-pin buzzer module gives the audible warning. Table II organizes the hardware used in each module.

Table II. Module-Wise Hardware, Interface and Supply.

Module	Hardware	Interface / Supply
Control	ESP32 board	GPIO, I2C, UART; 5 V USB, 3.3 V logic
Accident detection	MPU6050, SW-420	I2C and digital input; 3.3 V
Cancellation	Push button	Digital input to GND
Alert indication	3-pin buzzer	Digital output; 3.3 V
Location tracking	NEO-6M	UART, 9600 baud; supply as per board rating

Module	Hardware	Interface / Supply
Emergency alert	SIM800L	UART, AT commands; about 3.4–4.4 V dedicated supply
Power	Regulated supply stage	Battery/USB in; regulated ~4.0 V, 2 A peak out

The SIM800L draws much higher instantaneous current during transmission bursts than the sensors, so it must not be powered from the ESP32 3.3 V or 5 V pin. It needs its own supply that stays within the range given in its datasheet [5] (about 3.4–4.4 V, with a target of about 4.0 V) and can deliver peaks of up to about 2 A. A single-cell Li-ion battery (about 3.0–4.2 V) either powers the module directly, if the cell and wiring can supply the peak current, or feeds a boost or buck-boost converter that holds the output near 4.0 V as the cell discharges. A buck converter only steps voltage down, so it is suitable only when the source always stays above the required output; it cannot hold 4.0 V from a discharging single cell. The output must be set and checked with a multimeter, with the SIM800L disconnected, before the module is connected. A bulk capacitor of several hundred microfarads is placed close to the SIM800L supply pins to absorb the current bursts. During bench testing the ESP32 is powered through a USB cable, while the final portable prototype should use a stable battery with proper power distribution.

C. Circuit Connections

Fig. 4 shows the wiring between the ESP32 and the modules and Table III lists the pin-to-pin connections. The NEO-6M TX line goes to ESP32 RX2 (GPIO 16) and its RX line to TX2 (GPIO 17), and the SIM800L uses a separate hardware UART. All grounds are common. The exact supply voltage of the NEO-6M and of the SIM800L board must be confirmed from the board in use.

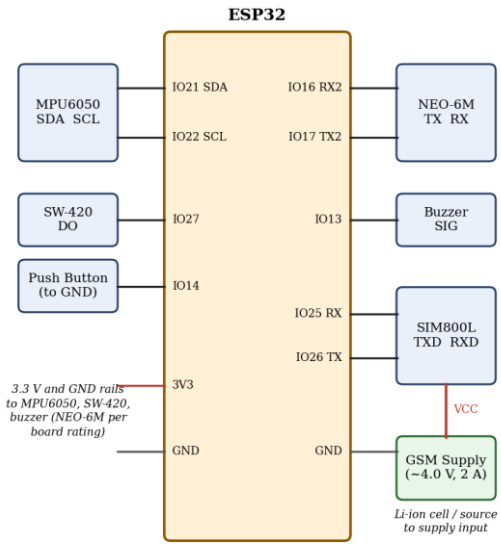


Fig. 4. Wiring Diagram of the Prototype.

Table III. Component Pin Connections.

Component / Pin	ESP32	Purpose
MPU6050 SDA	GPIO 21	I2C data
MPU6050 SCL	GPIO 22	I2C clock
MPU6050 VCC / GND	3.3 V / GND	Power
SW-420 DO	GPIO 27	Vibration input
NEO-6M TX	GPIO 16	GPS data to ESP32
NEO-6M RX	GPIO 17	ESP32 data to GPS
Push button	GPIO 14 / GND	Cancel input
Buzzer SIG	GPIO 13	Buzzer control (active-low setting)
SIM800L TXD	UART RX GPIO 25	GSM data to ESP32
SIM800L RXD	UART TX GPIO 26	ESP32 data to GSM
SIM800L VCC	Regulated 4.0 V supply	Regulated GSM supply
All GND	Common GND	Common reference

D. Accident Detection

The ESP32 reads the X, Y and Z acceleration values from the MPU6050 and computes the total acceleration magnitude:

$$A_{total} = \sqrt{(a_x^2 + a_y^2 + a_z^2)} \quad (1)$$

The SW-420 output V_{SW} is read as a digital input. A possible accident is flagged when either condition is satisfied:

$$E = 1, \text{ if } A_{total} > A_{th} \text{ or } V_{SW} = HIGH \quad (2)$$
$$E = 0, \text{ otherwise}$$

where A_{th} is the acceleration threshold, set to 1.90 g in the firmware, and both sensors are read every 100 ms (10 Hz). The MPU6050 is configured for a ±8 g range with a 21 Hz digital low-pass filter. Either condition alone starts the cancellation window, and the trigger reason (acceleration, vibration or both) is printed to the serial monitor. Because a toy-car model and a real vehicle produce different acceleration and vibration patterns, the threshold and the SW-420 sensitivity must be calibrated on the final vehicle structure.

E. False-Alert Cancellation and Local Alert

When E = 1, the buzzer is switched on and a cancellation window of T_c = 5 s begins. If the push button is pressed within the window, the alert is cancelled, the buzzer is turned off and monitoring resumes. Otherwise the accident is confirmed, as expressed by:

$$Z = 1, \text{ if } E = 1 \text{ and no button press within } T_c \quad (3)$$
$$Z = 0, \text{ otherwise}$$

where Z = 1 starts the emergency messaging sequence. This prevents unnecessary messages caused by bumps or sudden non-accident movement.

F. Location Acquisition and Emergency Messaging

After confirmation the ESP32 reads NMEA data from the NEO-6M [7] and parses it with the TinyGPSPlus library [9] for up to 30 s until a valid fix that is not older than 5 s gives non-zero latitude and longitude. If no fix is obtained, the SMS is still sent and states that the location is unavailable. These are formatted into a link of the form https://maps.google.com/?q=<lat>,<lon>. The ESP32 then drives the SIM800L with standard AT commands [8]: the firmware first confirms that the module answers AT and is registered on the network (AT+CREG? polled for up to 30 s), then AT+CMGF=1 selects text mode, AT+CMGS opens a message to the stored contact, the message body carries the accident status and map link, and the Ctrl+Z character (0x1A) sends the SMS. The SMS is counted as sent only if the module returns +CMGS and OK within 20 s. After a confirmed accident the firmware stops accident monitoring and reports the result once, so the board must be reset to resume monitoring.

G. Hardware Implementation and Calibration

The prototype is assembled on a breadboard or perfboard and fixed to the vehicle model. The following steps are used for assembly and calibration:

- Connect all grounds together first, then wire the modules as in Table III, keeping the I2C wires short.
- Configure the push button as an active-low input with the internal pull-up.
- Check the buzzer module polarity (active-high or active-low) before writing the ON/OFF logic.

Set the GSM supply output to about 4.0 V with a multimeter while the SIM800L is disconnected, confirm that it can deliver the peak current, and add the bulk capacitor at the module supply pins.

- Verify the serial logic levels between the 3.3 V ESP32 and the SIM800L, and use a divider or level shifter if required.
- Record A_{total} at rest (about 1 g) to check the MPU6050 offset.
- Tune the SW-420 trimmer so that its output stays LOW at rest and goes HIGH on a tap or impact.
- Choose A_{th} from shake and impact tests, then repeat on the final mounting.

H. Firmware Configuration

The firmware is written for the Arduino-ESP32 core and uses the Adafruit MPU6050 and TinyGPSPlus libraries. The main settings are as follows.

- Detection: A_{th} = 1.90 g; MPU6050 range ±8 g, gyroscope ±500 °/s, 21 Hz low-pass filter; MPU6050 and SW-420 are read every 100 ms.
- Cancellation: 5 s window with the buzzer on; the push button on GPIO 14 uses the internal pull-up and cancels when it is pulled LOW.
- GPS: NEO-6M on UART2 (GPIO 16 and 17) at 9600 baud; a fix is accepted when the coordinates are non-zero and not older than 5 s, with a maximum wait of 30 s.
- GSM: SIM800L on UART1 (GPIO 25 and 26) at 9600 baud; the module must answer AT and register on the network (maximum 30 s) before the SMS is sent, and the SMS is accepted as sent only after +CMGS and OK are received (maximum 20 s).
- Monitoring: the serial monitor (115200 baud) reports acceleration, vibration state and satellite count every 2 s. If the MPU6050 is not found at address 0x68 or 0x69, only the SW-420 input stays active. After a confirmed accident the result is reported once and monitoring stops.

I. Algorithm of the Proposed System

- Step 1: Power on the system and initialize I2C, UART, sensors, buzzer, button, GPS and GSM.
- Step 2: Read MPU6050 and SW-420 data every 100 ms and compute A_{total}.
- Step 3: Check the accident condition of Eq. (2); if false, continue monitoring.
- Step 4: If an accident is detected, turn the buzzer ON and start the 5-second cancellation window.
- Step 5: If the button is pressed, cancel the alert, turn the buzzer OFF and return to Step 2.
- Step 6: If not cancelled, confirm the accident and wait up to 30 s for a valid GPS fix.
- Step 7: Generate the Google Maps link from the latitude and longitude; if there is no fix, state that the location is unavailable.
- Step 8: After the network registration check, send the emergency SMS through the SIM800L to the predefined contact, then stop monitoring and report the result once.

IV. RESULTS AND DISCUSSION

A. Test Procedure and Expected Results

Each module is tested individually before integrated testing, as summarized in Table IV, which has a column for the observed result of each test. A satellite count of zero with invalid coordinates only means that no fix has been obtained yet and does not indicate a wiring fault, as NMEA characters are still received. Outdoor testing with a clear view of the sky is recommended. In the bench run recorded so far, the system started normally and read about 1.02–1.04 g at rest with 7–8 GPS satellites reported. A test accident event was detected through acceleration and vibration together, a GPS fix and a Google Maps link were obtained, and the SIM800L registered on the network and confirmed that the SMS was sent. The acceleration threshold is chosen from the calibration record in Table V.

Table IV. Test Cases and Expected Outcomes.

Test	Expected outcome	Observed result
ESP32 startup	Board initializes and begins sensor monitoring	Initialized; "System Ready" printed
MPU6050 connection	Acceleration values are read over I2C	Detected; about 1.02–1.04 g at rest
SW-420 test	Digital output changes when vibration is applied	DO LOW at rest; vibration detected at the accident trigger
Power test	SIM800L supply stays stable during transmission	4.0V
Buzzer / button test	Buzzer toggles; button cancels the alert in the window	Detected and test passed
GPS UART test	NMEA characters received at 9600 baud	NMEA data parsed; 7–8 satellites reported
GPS fix test	Valid satellite count and coordinates appear	Valid fix obtained with 7–8 satellites
Maps link test	Valid coordinates inserted into a Google Maps URL	Link generated in the form maps.google.com/?q=lat,lon
SIM800L test	SMS sent when supply and network are adequate	Network registered; module confirmed SMS sent
Accident simulation	Acceleration or vibration event starts buzzer and window	Detected (acceleration + vibration); cancel window started
False-alert test	Button press within 5 s cancels the sequence	Button Function works properly

Table V. Threshold calibration record.

Test condition	Peak total acceleration (g)	SW-420 output	Alert triggered
Vehicle model at rest	1.02–1.04	LOW	No
Light tap on the board	1.10-1.30	HIGH briefly	Yes
Pothole / speed-breaker simulation	1.30-1.70	HIGH	Yes
Hard shake	1.7.-2.20	HIGH	Yes
Impact (collision) test	2.20-.50	HIGH	Yes

B. Illustrative System Behavior

To complement the test plan, Figs. 5–7 present three indicative visualizations: the accident decision regions of Eq. (2), the approximate typical current drawn by each module, and the alert timeline. They are illustrative and are not measured data.

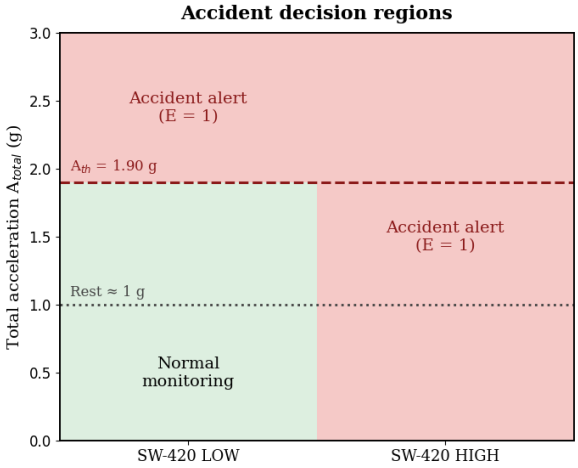


Fig. 5. Illustrative accident decision regions.

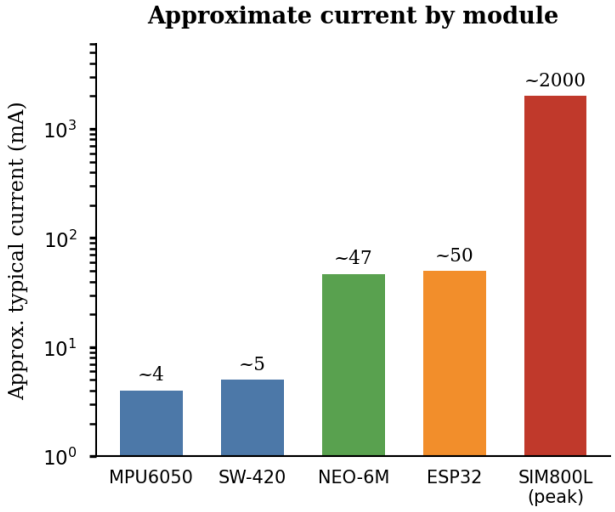


Fig. 6. Approximate typical current drawn by each module.

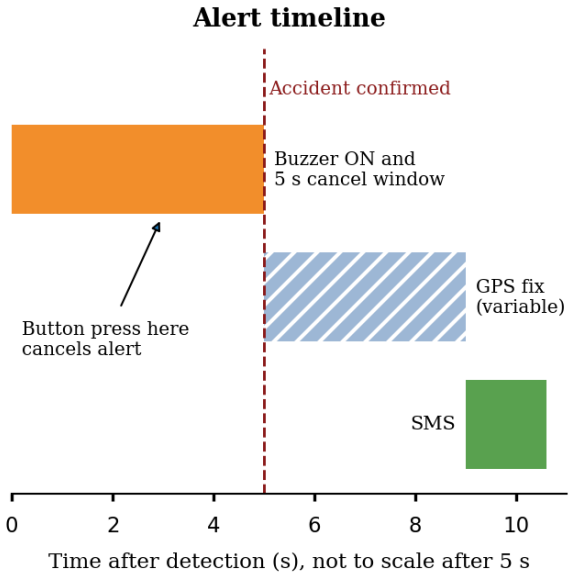


Fig. 7. Alert timeline after detection.

C. DISCUSSION

The threshold of about 1.9 g is only about 0.9 g above the resting value of 1 g, so a pothole or hard braking can approach it. Because either the acceleration threshold or the SW-420 output can start the alert, the system is sensitive, and the 5-second window is the main protection against false triggers; requiring both conditions is a simple option if false alerts are frequent. The threshold and the SW-420 sensitivity must still be calibrated on the final vehicle because a toy-car model and a real vehicle behave differently.

The power design is as important as the sensing. The SIM800L can draw peaks of about 2 A, which is far above the other modules (Fig. 6), so a dedicated regulated supply and a bulk capacitor are needed to avoid supply drops and module resets during SMS transmission. GPS start-up can also take time indoors or under obstructed sky, so outdoor testing is recommended, and SMS delivery depends on SIM service and 2G network coverage.

The main advantages of the proposed system are automatic accident detection, two independent sensing inputs, a cancellation button that reduces false alerts, a location link that is easy to open in Google Maps, SMS delivery without a smartphone application, and a modular design in which each module can be tested or replaced independently. Its main limitations are:

- GPS may take time to obtain a fix and location cannot be guaranteed without satellite signals.
- A simple acceleration threshold may not separate every real crash from a strong bump, and a vibration pulse alone can start the cancellation window.

- The sensors are polled every 100 ms, so a shorter impact or vibration pulse can be missed; a final design should use higher-rate sampling or a hardware interrupt.
- The SIM800L needs sufficient peak current, a working SIM and available 2G network coverage.
- A real vehicle installation needs proper mounting and electrical protection.

V. CONCLUSION AND FUTURE SCOPE

In this study, a hardware prototype named the Accident Messaging System (AMS) has been developed to tackle the delay in reporting road accidents. Automatic, location-aware emergency notification has been achieved by integrating accident detection, a cancellation window, GPS tracking and GSM messaging on one ESP32-based platform, with a dedicated regulated supply powering the SIM800L. The architecture is compact, low cost and modular, and it can be applied to passenger vehicles, two-wheelers, fleet monitoring and research prototypes.

Future work includes sampling the accelerometer at a much higher rate with peak detection, using the MPU6050 gyroscope and GPS speed to detect rollovers and sudden speed drops, and training machine-learning models to classify accident severity. Automatic SMS retry after a failure, multiple emergency contacts, a cloud dashboard, a mobile application and event logging would make the alert more dependable. On the hardware side, battery monitoring, an automotive-grade GPS and GSM/LTE module and a compact custom PCB would prepare the system for permanent installation in a vehicle.

REFERENCES

1. World Health Organization, *Global Status Report on Road Safety 2023*. Geneva, Switzerland: WHO, 2023.
2. Espressif Systems, *ESP32 Series Datasheet*. Shanghai, China: Espressif Systems.
3. InvenSense Inc., *MPU-6000 and MPU-6050 Product Specification*. San Jose, CA, USA: InvenSense.
4. U-blox AG, *NEO-6 Series u-blox 6 GPS Modules Data Sheet*. Thalwil, Switzerland: u-blox.
5. SIMCom Wireless Solutions, *SIM800 Series Hardware Design and AT Command Manual*. Shanghai, China: SIMCom.

6. Regulation (EU) 2015/758 of the European Parliament and of the Council of 29 April 2015 concerning type-approval requirements for the deployment of the eCall in-vehicle system based on the 112 service, *Official Journal of the European Union*, 2015.
7. National Marine Electronics Association, *NMEA 0183 Interface Standard*. Severna Park, MD, USA: NMEA.
8. 3GPP, *Use of Data Terminal Equipment – Data Circuit terminating Equipment (DTE–DCE) interface for Short Message Service (SMS)*, 3GPP TS 27.005.
9. M. Hart, "TinyGPSPlus: A compact Arduino NMEA parsing library." [Online]. Available: https://github.com/mikalhart/TinyGPSPlus
10. M. S. Amin and J. Jalil, "Accident detection and reporting system using GPS, GPRS and GSM technology," in *Proc. ICIEV*, IEEE, 2012, pp. 640–643.
11. C. Prabha, R. Sunitha and R. Anitha, "Automatic vehicle accident detection and messaging system using GSM and GPS modem," *Int. J. Adv. Res. Electr. Electron. Instrum. Eng.*, vol. 3, no. 7, pp. 10723–10727, 2014, doi: 10.15662/ijareeie.2014.0307062.
12. S. Sarker, M. S. Rahman and M. N. Sakib, "An approach towards intelligent accident detection, location tracking and notification system," *arXiv preprint arXiv:2001.00453*, 2020.
13. J. A. J. Alsayaydeh, M. F. Yusof, M. A. A. Abdillah, A. J. A. Al-Gburi, S. G. Herawan and A. Oliinyk, "Enhancing vehicle safety: A comprehensive accident detection and alert system," *Int. J. Adv. Comput. Sci. Appl. (IJACSA)*, vol. 14, no. 11, 2023.
14. R. A. Sowah, M. Shahid, N. L. Sowah and G. Buah, "Design and development of IoT-based automatic vehicle accident detection system," in *Proc. IEEE 9th Int. Conf. Adaptive Science and Technology (ICAST)*, Accra, Ghana, 2024, doi: 10.1109/ICAST61769.2024.10856503.