

INTRAVENOUS DRIP MONITER SYSTEM USING IOT

Submitted in partial fulfillment of the required for the award of

Masters of Computer Science

by

REJANSY BANU BB

REGISTERNO (42734016)



DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF SCIENCE AND HUMANITIES

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A++” by NAAC

JEPPIAR NAGAR, RAJIVGANDHI SALAI, CHENNAI - 600 119

APRIL 2024



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A++” by NAAC

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of **REJANSY BANU BB (42734016)** who carried out the project entitled **“INTRAVENOUS DRIP MONITER SYSTEM USING IOT”** under our supervision from JANUARY 2024 TO APRIL 2024.

Internal Guide

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

DEAN AND HEAD OF THE DEPARTMENT

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

Submitted for Viva voce examination held on

20.04.2024

Internal Examiner

External Examiner

DECLARATION

I am, **REJANSY BANU BB (42734016)** hereby declare that the project report entitled **“INTRAVENOUS DRIP MONITER SYSTEM USING IOT ”** done by me under the Guidance of **Dr. REKHA CHAKRAVARTHI M.E., Ph.D.**, Department of Computer Science at **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**, Jeppiaar Nagar, Rajiv Ghandhi Salai, Chennai-600119 is submitted in partial fulfillment of the requirements for the award of Master of Science degree in Computer Science.

DATE: 20.04.2024

PLACE: CHENNAI

REJANSY BANU BB
SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind Encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. REKHA CHAKRAVARTHI M.E.,Ph.D., Dean of Arts and Science** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide **MS. Dr. REKHA CHAKRAVARTHI M.E., Ph.D.,** Department of Computer Science for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff member of the Department of Computer Science who were helpful in many ways for the completion of the project.

REJANSY BANU BB

ABSTRACT

Intravenous (IV) therapy is a common medical procedure used to deliver fluids, medications, and nutrients directly into the bloodstream. However, managing IV drips requires constant monitoring to ensure correct dosage rates and prevent complications such as overhydration or medication errors. Traditional methods rely heavily on manual monitoring by healthcare professionals, which can be time-consuming and prone to human error. To address these challenges, an Intravenous Drip Monitoring and Alert System (IDMAS) is proposed. This system leverages sensor technology and data analytics to continuously monitor IV drips and provide real-time alerts to healthcare providers in case of deviations from prescribed parameters. The core components of the IDMAS include: Various sensors, such as flow rate sensors and pressure sensors, are integrated into the IV tubing to capture relevant data, including drip rate, fluid volume, and pressure. The data collected by sensors are processed and analyzed using algorithms designed to detect abnormalities or deviations from the desired parameters. Machine learning techniques may be employed to enhance the system's ability to recognize patterns and predict potential issues. When the system detects a deviation from the prescribed parameters or identifies a potential risk, it generates real-time alerts to notify healthcare providers. These alerts can be delivered through various channels, such as visual alarms, audible alerts, or notifications on a dedicated monitoring interface. The IDMAS may include remote monitoring capabilities, allowing healthcare providers to access real-time IV drip data and receive alerts from anywhere with an internet connection. This enables timely intervention and proactive management of IV therapy, even outside of traditional healthcare settings. To ensure seamless documentation and continuity of care, the IDMAS can be integrated with existing EHR systems. This integration enables automatic recording of IV drip data and alerts in patients' medical records, facilitating communication and collaboration among healthcare teams. Overall, the IDMAS aims to enhance patient safety, improve the efficiency of IV therapy management, and reduce the burden on healthcare professionals by automating monitoring and alerting processes. By leveraging advanced technology and data analytics, this system has the potential to revolutionize the delivery of IV therapy and improve patient outcomes.

INDEX

CHAPTER NO	TITLE	PAGE NUMBER
1	ABSTRACT	1
1.1	INTERNET OF THINGS	5
1.2	IOT IN HOSPITAL MANAGEMENT	6
1.3	DRIP MONITORING USING IOT	8
1.4	BACKGROUND STUDY	10
2	BACKGROUND STUDY	12
2.1	INTRAVENOUS DRIP MONITORING SYSTEM FOR SMART HOSPITAL USING IOT	12
2.2	A REMOTE DRIP INFUSION MONITORING SYSTEM EMPLOYING BLUETOOTH	12
2.3	A SMART INFUSION PUMP SYSTEM FOR REMOTE MANAGEMENT AND MONITORING OF INTRAVENOUS (IV)DRIPS	13
2.4	AND IOT BASED INTRAVENOUS DRIP RATE CONTROLLING AND MONITORING DEVICE	13
2.5	SMART COCOA NURSERY MONITORING SYSTEM USING IRT FOR AUTOMATIC DRIP IRRIGATION	13
2.6	A LOW POWER CIRCUIT FOR MEDICAL DRIP INFUSION MONITORING SYSTEM	14
2.7	AUTOMATIC CONTROL OF DRIP IRRIGATION ON HYDROPONIC AGRICULTURE: DANIELA TOMATO PRODUCTION	14
2.8	INTRAVENOUS DRIP METER AND CONTROLLER	15
2.9	SOLAR POWER AUTOMATIC DRIP IRRIGATION SYSTEM USING PARTICLE SWARM OPTIMIZATION	15
2.10	SMART DRIP INFUSION MONITORING SYSTEM FOR INSTANT ALERT THROUGH NRF24L01	15

3	METHODOLOGY	17
3.1	EXISTING SYSTEM	17
3.2	PROPOSED SYSTEM	18
3.3	BLOCK DIAGRAM	19
3.4	SYSTEM REQUIREMENTS	20
3.4.1	HARDWARE REQUIREMENTS	20
3.4.2	SOFTWARE REQUIREMENTS	21
3.5	USE CASE DIAGRAM	21
3.6	MODULES DESCRIPTION	22
3.6.1	ARDUINO CONNECTION	22
3.6.2	ULTRASONIC SENSOR	23
3.6.3	GSM INTER COMMUNICATION	24
3.6.4	LEVEL DEDUCTION	25
3.6.5	ALERT SYSTEM	26
4	SOFTWARE SPECIFICATION	28
4.1	ARDUINO	28
4.2	FEASIBILITY STUDY	31
4.3	SYSTEM TESTING	33
4.4	PRODUCT DESCRIPTION	34
4.4.1	ARDUINO UNO	34
4.4.2	ULTRASONIC SENSOR	36
4.4.3	GSM COMMUNICATION SETUP	38
4.5	POWER SUPPLY	41
5	CONCLUSION AND FUTURE ENHANCEMENT	44
5.1	CONCLUSION	44
5.2	FUTURE ENHANCEMENT	44

LIST OF FIGURES

FIGURE NUMBER	TITLE	PAGE NUMBER
1.1	INTERNET OF THINGS	6
1.2	IOT WITH HEALTHCARE SYSTEM	8
1.4	DRIP CONTROL SYSTEM	9
3.1	BLOCK DIAGRAM IMPLEMENTATION	20
3.2	USE CASE DIAGRAM	21
3.3	PIN CONNECTION DIAGRAM	23
4.1	ARDUINO SOFTWARE	28
4.2	ARDUINO UNO	36
4.3	ULTRASONIC SENSOR	37
4.4	GSM MODULE	40
4.5	POWER SUPPLY FOR ARDUINO	42

CHAPTER 1

INTRODUCTION

1.1 INTERNET OF THINGS

The Internet of Things (IoT) is a concept that refers to the network of physical objects or "things" embedded with sensors, software, and connectivity, enabling them to collect and exchange data with other devices and systems over the internet. These interconnected devices can range from everyday objects like household appliances and wearable devices to industrial machinery and infrastructure components.

Key aspects of the Internet of Things include:

- **Connectivity:** IoT devices are equipped with communication technologies such as Wi-Fi, Bluetooth, Zigbee, or cellular connections, allowing them to connect to the internet and to each other.
- **Sensing and Data Collection:** IoT devices are equipped with sensors that gather data about their environment, such as temperature, humidity, light, motion, or location. This data can be collected in real-time and used for various purposes, including monitoring, analysis, and control.
- **Data Processing and Analytics:** The data collected by IoT devices can be processed and analyzed to derive insights, detect patterns, and make decisions. Advanced analytics techniques, including machine learning and artificial intelligence, are often used to extract valuable information from the vast amounts of data generated by IoT systems.
- **Automation and Control:** IoT enables automation by allowing devices to communicate and coordinate with each other autonomously. This can lead to increased efficiency, improved productivity, and enhanced user experiences. For example, smart home devices can automatically adjust temperature settings based on occupancy and weather conditions.

- **Remote Monitoring and Management:** IoT enables remote monitoring and management of devices and systems from anywhere with an internet connection. This capability is particularly valuable for applications in industries such as healthcare, manufacturing, agriculture, and utilities, where remote monitoring can improve efficiency, safety, and reliability.
- **Scalability and Interoperability:** IoT solutions are designed to be scalable, allowing for the deployment of large-scale networks of interconnected devices. Interoperability standards and protocols ensure that devices from different manufacturers can communicate and work together seamlessly.



Fig 1.1 Internet of Things

Overall, the Internet of Things has the potential to transform industries, improve quality of life, and drive innovation across various sectors. However, it also raises challenges related to security, privacy, interoperability, and data management, which need to be addressed to fully realize its benefits.

1.2 IOT IN HOSPITAL MANAGEMENT

The implementation of the Internet of Things (IoT) in hospital management brings numerous benefits ranging from enhancing patient care to improving operational efficiency. Here are several ways IoT can be utilized in hospital management:

- **Asset Tracking and Management:** Hospitals can deploy IoT-enabled tracking devices on medical equipment, such as infusion pumps, wheelchairs, and defibrillators, to monitor their location, status, and usage in real-time. This helps in efficient inventory management, reduces equipment loss, and ensures timely maintenance.

- Remote Patient Monitoring: IoT devices can monitor patients' vital signs, such as heart rate, blood pressure, and oxygen levels, remotely and in real-time. This enables healthcare providers to track patients' health status outside traditional clinical settings, reducing the need for frequent hospital visits and enabling early intervention in case of emergencies.
- Smart Hospital Rooms: IoT sensors and devices can be installed in hospital rooms to automate various functions, such as adjusting lighting, temperature, and blinds based on patients' preferences and clinical needs. Additionally, smart beds equipped with sensors can monitor patients' movements, alerting caregivers to potential falls or discomfort.
- Medication Management: IoT-enabled medication dispensing systems can automate medication administration, ensuring accurate dosing and reducing the risk of errors. These systems can also provide reminders to patients to take their medications on time and alert healthcare providers in case of missed doses or medication discrepancies.
- Infection Control: IoT sensors can monitor environmental conditions, such as air quality and temperature, to detect potential sources of infection or contamination in hospital facilities. Additionally, IoT-enabled hand hygiene monitoring systems can track healthcare workers' compliance with handwashing protocols, helping to prevent the spread of infections.
- Patient Flow Optimization: IoT devices, such as RFID tags and beacons, can track the movement of patients, staff, and equipment within the hospital premises. This data can be analyzed to identify bottlenecks, optimize workflow, and improve the efficiency of patient care delivery.
- Energy Management: IoT sensors can monitor energy consumption in hospital buildings and identify opportunities for energy conservation and cost reduction. For example, smart lighting systems can adjust lighting levels based on occupancy, while smart HVAC systems can optimize temperature settings for energy efficiency.
- Telemedicine and Telehealth: IoT-enabled telemedicine platforms allow healthcare providers to conduct remote consultations, monitor patients' health remotely, and deliver personalized care interventions. This enables hospitals to extend their reach beyond

physical boundaries and provide healthcare services to patients in remote or underserved areas.



Fig 1.2 IOT with health care system

By leveraging IoT technologies, hospitals can enhance patient safety, improve clinical outcomes, streamline operations, and reduce costs. However, it's essential to address challenges related to data privacy, security, interoperability, and regulatory compliance to ensure the successful implementation and adoption of IoT solutions in hospital management.

1.3 DRIP MONITORING USING IOT

Drip monitoring using IoT can greatly improve the accuracy and efficiency of intravenous (IV) therapy in healthcare settings. Here's how it works and some key benefits:

- **Sensor Integration:** IoT devices with specialized sensors are integrated into the IV tubing or infusion pump system. These sensors can measure parameters such as drip rate, fluid volume infused, and pressure within the IV line.
- **Real-time Monitoring:** The IoT system continuously monitors the IV drip parameters in real-time. Data from the sensors is transmitted wirelessly to a central monitoring system or cloud platform.
- **Alerts and Notifications:** The IoT system is programmed to detect deviations from the prescribed parameters or potential issues such as air bubbles, occlusions, or incorrect drip

rates. When anomalies are detected, the system generates alerts or notifications for healthcare providers.

- **Remote Monitoring:** Healthcare providers can remotely monitor IV drips using mobile devices or computers connected to the IoT system. This allows for continuous monitoring even when healthcare providers are not physically present in the patient's room.
- **Automated Documentation:** The IoT system can automatically record and document IV drip data, including infusion rates, volume infused, and timestamps. This eliminates the need for manual documentation by healthcare staff, reducing the risk of errors and improving efficiency.
- **Optimized IV Therapy:** By providing real-time feedback and alerts, the IoT system helps ensure that IV therapy is administered accurately and safely. Healthcare providers can promptly address any issues or adjust infusion parameters as needed to optimize patient care.



Fig .1.4 Drip Control System

- **Improved Patient Safety:** Drip monitoring using IoT enhances patient safety by reducing the risk of medication errors, fluid overload, or adverse reactions associated with IV therapy. Early detection of abnormalities allows for timely intervention, minimizing potential harm to patients.
- **Data Analytics and Insights:** The IoT system can analyze IV drip data over time to identify patterns, trends, and areas for improvement in IV therapy protocols. This data-driven approach enables healthcare facilities to continuously optimize their practices and enhance patient outcomes.

Overall, drip monitoring using IoT offers significant advantages in terms of accuracy, efficiency, and patient safety in IV therapy. By leveraging real-time monitoring, automated alerts, and remote access capabilities, healthcare providers can deliver high-quality care and improve clinical outcomes for patients receiving intravenous treatments.

1.4 BACKGROUND STUDY

A background study on drip monitoring using IoT would typically encompass several key areas:

Current State of IV Therapy Management: Provide an overview of the current practices and challenges in intravenous (IV) therapy management in healthcare settings. This could include manual monitoring methods, common issues such as medication errors or fluid overload, and the importance of accurate drip rate control.

Introduction to IoT in Healthcare: Introduce the concept of the Internet of Things (IoT) and its applications in healthcare. Explain how IoT technology can be leveraged to improve patient care, streamline operations, and enhance clinical outcomes.

IoT Sensors and Devices: Explore the types of sensors and devices used in drip monitoring using IoT. Discuss how these sensors are integrated into IV tubing or infusion pumps to measure parameters such as drip rate, fluid volume infused, and pressure.

Technical Aspects of IoT Implementation: Describe the technical architecture of an IoT-based drip monitoring system. This could include details on sensor integration, data transmission protocols, cloud-based data storage, and communication interfaces with healthcare IT systems.

Benefits of IoT-enabled Drip Monitoring: Discuss the advantages of using IoT for drip monitoring in healthcare settings. Highlight benefits such as real-time monitoring, automated alerts, remote access capabilities, improved patient safety, and enhanced efficiency in IV therapy management.

Challenges and Considerations: Identify potential challenges and considerations in implementing drip monitoring using IoT. This could include issues related to sensor accuracy, data security and privacy, interoperability with existing healthcare systems, regulatory compliance, and cost implications.

Case Studies and Use Cases: Provide examples of real-world applications of IoT-enabled drip monitoring in healthcare settings. Showcase case studies or pilot projects that demonstrate the effectiveness and impact of IoT technology in improving IV therapy management and patient outcomes.

Future Trends and Opportunities: Explore emerging trends and future opportunities in IoT-enabled healthcare, particularly in the context of drip monitoring and IV therapy management. Discuss potential advancements in sensor technology, data analytics, and artificial intelligence that could further enhance the capabilities of IoT systems in healthcare.

By conducting a comprehensive background study on drip monitoring using IoT, researchers and practitioners can gain a deeper understanding of the technology, its applications, benefits, challenges, and future directions in healthcare. This knowledge can inform the design, development, and implementation of IoT-based solutions to improve IV therapy management and patient care.

CHAPTER 2

BACKGROUND STUDY

2.1 INTRAVENOUS DRIP MONITORING SYSTEM FOR SMART HOSPITAL USING IOT

In our current medical care system, the monitoring of patients in a hospital throughout the day is a tiresome process. Sometimes Doctors or Nurses are too busy, so they can't monitor each patient. This causes many problems. The health related work should be properly done and that too with accurate manner. An example of such type of work in our hospital is injecting saline or Intravenous (IV) fluids in to the vein of patient. If the drip system is not monitored on time, it will causes problems like backflow of fluid, blood loss etc. In order to reduce the workload and overcome such critical situation in the area of an intravenous drip monitoring system, weproposed a system called Automated Intravenous Drip Monitoring System.

2.2 A REMOTE DRIP INFUSION MONITORING SYSTEM EMPLOYING BLUETOOTH

The current state of the art of intravenous (IV) fluid administration systems lacks remote monitoring and control capabilities. This study presents an intelligent infusion pump system for automatic and remote management and monitoring of IV drips. This work used Arduino based microcontroller for controlling drop counter, detecting tube blockage and monitoring the emptying of drip bag. This system used low power laser diodes and optical sensors for the aforementioned monitoring tasks. The flow rate (in drop per minute) and infusion-interruption problems were monitored remotely via transmission of data wirelessly to users' smartphones using Blynk mobile application and computer based applications. This study found no difference between the manual and automatic counting reading. In addition, the system is able to notify its users of empty bottle and line blockage. This work concluded that the developed prototype may be further enhanced (in its design and features) and tested for its effectiveness and consistencies in real clinical settings.

2.3 A SMART INFUSION PUMP SYSTEM FOR REMOTE MANAGEMENT AND MONITORING OF INTRAVENOUS (IV) DRIPS

We have developed a remote drip infusion monitoring system for use in hospitals. The system consists of several infusion monitoring devices and a central monitor. The infusion monitoring device employing a Bluetooth module can detect the drip infusion rate and an empty infusion solution bag, and then these data are sent to the central monitor placed at the nurses' station via the Bluetooth. The central monitor receives the data from several infusion monitoring devices and then displays graphically them. Therefore, the developed system can monitor intensively the drip infusion situation of the several patients at the nurses' station.

2.4 AN IOT BASED INTRAVENOUS DRIP RATE CONTROLLING AND MONITORING DEVICE

One of the most efficient means of drug administration in any healthcare setup is intravenously. Intravenous mode facilitates fastest delivery of medication and ensures fluid replacement all through the body. Existing intravenous drip set up for drug administering is manually controlled and is not accurate. Also, the inability of doctors to enter radioactive scan chambers to monitor the IV drip setup is an age-old problem associated with the existing system. The need for a device made to answer all these issues is identified, which can also solve another major problem i.e., patient anxiety. A systematic clinical immersion and survey was undertaken and observations made during this exercise led to the proposal of an alternate efficient drip rate controlling and monitoring device based on IoT. The proposed novel device intends to resolve the major issues such as air embolism and blood backflow. Apart from this, it can also provide efficient drug administration at precise intervals in neonates and pediatric subjects.

2.5 SMART COCOA NURSERY MONITORING SYSTEM USING IRT FOR AUTOMATIC DRIP IRRIGATION

Conventional agriculture systems normally ineffective and consume a lot of manpower and water resources. In this project, an automatic drip irrigation system based on IOT platform is designed. This system is implemented in cocoa nursery at Malaysian Cocoa (MCB) Bagan Datoh. Smart Cocoa Monitoring System Using IRT for Automatic Drip Irrigation is designed by creating optimal soil moisture conditions to facilitate healthy plant growth through proper irrigation frequencies. With a Raspberry Pi 3 Model B, the user can monitor the soil moisture in the cocoa plant nursery and the irrigation machine through the network. Therefore, to monitor

the environment for this agriculture, the system will collect the information of the soil moisture. Any changes detected by the built-in sensors of the system that can harm the cocoa plant nursery will be noted and the system will provide soil moisture information and can be monitored through ThinkSpeak.com application for further action to be taken. To monitor the efficiency of the system, the height of each cocoa plant (cm) and the number of leaves per cocoa plant is observed within 3 weeks. For comparison, this monitoring system that controlled by a Raspberry Pi3 model B with drip irrigation technique is recorded as Treatment 1 (T1) and the overhead sprinkler technique that has been applied in MCB, Bagan Datoh, Perak is recorded as Treatment 2 (T2). It is observed that the percentage height of coco plant based on T1 is increased by 26.82% while T2 only 16.04%. The percentage increment of leaves based on T1 is 36.31% while T2 is 20.70%.

2.6 A low power circuit for medical drip infusion monitoring system

We developed a low power monitor for medical drip infusion monitoring system, it can operate for a year, and longer than other designs. The monitoring system consists of many drip infusion monitoring nodes and a monitoring center, and transmits the message by the Zigbee wireless sensor network. The capacitive sensor in the monitor node differentiates that air and water are different dielectric material, and has a feature of low power consumption. The status of the drip infusion tube is detected by the capacitive sensor, the monitor transmits the alarm message to the monitoring center when there is no water in the drip infusion tube, the nurses can handle quickly the drip infusion task.

2.7 Automatic control of drip irrigation on hydroponic agriculture: Daniela tomato production

The irrigation control and monitoring of the involved variables in the plant growth are actions that farmers must implement in their crops to save water resources and assessing the growth of the Daniela tomato plant. This work presents the control drip is applied to hydroponic farming in which is developed an interface between human and machine in a free software allowing continuous monitoring of moisture, pH, temperature and electrical conductivity of soil through the sensors housed in the crop root zone also, the controller performs the conditioning sensors, actuator control resource to irrigate water, and nutrient solution, and monitoring via web. The field implementation of the system is carried out in a tunnel-type greenhouse designed under the requirements of a hydroponic system. Finally to show the results we evaluate two

experiments: Experiment one: determining hydric resources, and nutritional requirements needed by the culturing based on monitoring of physical variables. Experiment 2; assess how media type tomato fruit have more weight and diameter.

2.8 Intravenous drip meter & controller

Intravenous (IV) drip is a crucial mode for delivery of fluids and other pharmacological substances directly into the blood circulation. IV drip is widely used because of its advantages. Though IV drip is a safe, effective and affordable tool, yet a number of complications can arise in its usage. In this paper, IV drip usage and its associated risks are reviewed and a solution is proposed to enable monitoring and control of IV drip based on sensing of drops falling through the drip chamber. Such a device will potentially reduce complication and provide peace of mind to users of IV drip system.

2.9 Solar Powered Automated Drip Irrigation System using Particle Swarm Optimization

There is a significant imbalance between energy demand and supply due to dwindling conventional energy resources and ever-increasing energy consumption. Researchers are now developing more ways to utilize green energy production, mainly renewable energy such as Solar PV and Wind energy. To increase the efficient use of water resources and decrease the dependency on conventional energy sources, researchers are finding ways to integrate renewable energy sources with irrigation systems to help the agricultural sector ease the cost of irrigation. The author used Particle Swarm optimization in their research to determine the optimal soil depth to be maintained at field capacity to minimize the cost of irrigation. A solar-powered drip irrigation system is then implemented to test and prove that drip irrigation systems are more efficient than traditional irrigation methods. This study also utilizes Internet-of-Thingstechnology to provide remote monitoring of the system. At the end of the study, it is found that the output solar-powered automated drip irrigation system is comparable to that of a traditional irrigation system while using significantly less water.

2.10 Smart drip infusion monitoring system for instant alert-through nRF24L01

This project work involves trickle implantation observing framework for use in hospitals. The framework comprises of a drip infusion, sugar level observing gadgets and a monitoring screen. The mixture observing gadget utilizing a pressure sensor (MPX10GP) technology module can identify the trickle implantation rate and a vacant imbueement arrangement sack, and after that, this information is sent to the monitoring screen put at the medical caretaker's station by means

of the radio frequency (nrf24L01). The monitoring screen gets the information from trickle implantation observing gadgets and after that shows graphically them. When pressure sensor value reaches the threshold value, control valve will close which stops immediately flow of fluid without any airflow in patient's vein. In this manner, the created framework can screen seriously the dribble implantation circumstance of the few patients at the medical caretakers' station.

CHAPTER 3

METHODOLOGY

3.1 EXISTING SYSTEM

The existing system for drip monitoring in healthcare facilities typically relies on manual methods and periodic checks by healthcare professionals. Here's an overview of the components and processes involved:

Manual Monitoring: Healthcare providers periodically check IV drips by visually inspecting the drip chamber and counting the number of drops per minute. This method is prone to human error and may not provide continuous monitoring of drip parameters.

Flow Rate Regulation: IV infusion pumps are commonly used to regulate the flow rate of IV fluids and medications. However, these pumps may not always provide accurate and consistent flow rates, leading to potential deviations from prescribed infusion rates.

Patient Observation: Healthcare providers monitor patients for signs of adverse reactions or complications related to IV therapy, such as fluid overload, infiltration, or air embolism. This typically involves regular assessments of vital signs and clinical symptoms.

Documentation: Nurses and other healthcare staff manually record IV drip parameters, infusion rates, fluid volumes infused, and other relevant information in patient charts or electronic health records (EHR). This documentation is essential for tracking patient progress and ensuring accountability.

Alarm Systems: Some infusion pumps are equipped with alarm systems that alert healthcare providers to certain conditions, such as occlusions, air bubbles, or low battery levels. However, these alarms may not always be reliable or timely in detecting problems.

Quality Control and Training: Healthcare facilities implement quality control measures and provide training to staff members responsible for IV therapy management. This includes proper handling of IV equipment, adherence to medication protocols, and prompt response to alarms or deviations from standard procedures.

Disadvantages of Existing System:

- **Manual Monitoring:** Reliance on manual monitoring methods is time-consuming, labor-intensive, and prone to errors.
- **Limited Visibility:** Healthcare providers may not have continuous visibility into IV drip parameters, especially outside of regular monitoring intervals.
- **Risk of Errors:** Human errors in regulating flow rates, documenting parameters, or responding to alarms can compromise patient safety and quality of care.
- **Lack of Real-time Alerts:** The existing system may not provide real-time alerts or notifications to healthcare providers in case of deviations from prescribed parameters or adverse events.
- **Potential for Improvements:** There is a need for technological advancements and innovative solutions to improve the accuracy, efficiency, and safety of IV therapy management.

3.2 PROPOSED SYSTEM

The proposed system using Arduino, a buzzer, an ultrasonic sensor, and GSM technology aims to provide a cost-effective and efficient solution for drip monitoring in healthcare settings. Here's how the system could be designed and implemented:

Arduino Microcontroller: The Arduino serves as the main control unit of the system. It interfaces with the other components, processes sensor data, and triggers actions based on predefined conditions.

Ultrasonic Sensor: The ultrasonic sensor is used to measure the fluid level in the IV drip chamber. By sending ultrasonic pulses and measuring the time taken for the echoes to return, the sensor can calculate the distance to the liquid surface, indicating the fluid level.

Buzzer: The buzzer is used as an audible alert mechanism to notify healthcare providers in case of critical conditions, such as low fluid level or occlusions in the IV drip line.

GSM Module: The GSM module enables the system to send SMS alerts to designated recipients, such as nurses or doctors, in addition to the audible alerts from the buzzer. This allows for

remote monitoring and timely intervention, even when healthcare providers are not physically present.

Here's an overview of how the system would operate:

The ultrasonic sensor continuously monitors the fluid level in the IV drip chamber.

The Arduino compares the measured fluid level to predefined thresholds. If the fluid level falls below a certain threshold, indicating a low fluid level, or if an obstruction is detected in the drip line, the Arduino triggers the alarm sequence.

The buzzer emits an audible alert to notify nearby healthcare providers of the issue.

Simultaneously, the Arduino sends a command to the GSM module to send an SMS alert to pre-programmed phone numbers, notifying healthcare providers of the issue.

Healthcare providers can then take appropriate action to address the problem, such as refilling the IV bag, clearing obstructions, or adjusting the infusion rate.

Advantages of Proposed System

- **Real-time Monitoring:** Continuous monitoring of the IV drip ensures timely detection of issues.
- **Audible and Remote Alerts:** Healthcare providers are alerted both audibly and remotely via SMS, allowing for quick response even if they are not in close proximity to the patient.
- **Cost-effective Solution:** The use of Arduino, ultrasonic sensor, buzzer, and GSM module provides a relatively low-cost solution compared to commercial drip monitoring systems.
- **Customizable and Expandable:** The system can be customized and expanded based on specific requirements and can be integrated with other healthcare monitoring systems if needed.

3.3 BLOCK DIAGRAM

Here's a block diagram illustrating the proposed system using Arduino, a buzzer, an ultrasonic sensor, and GSM technology for drip monitoring in healthcare settings:

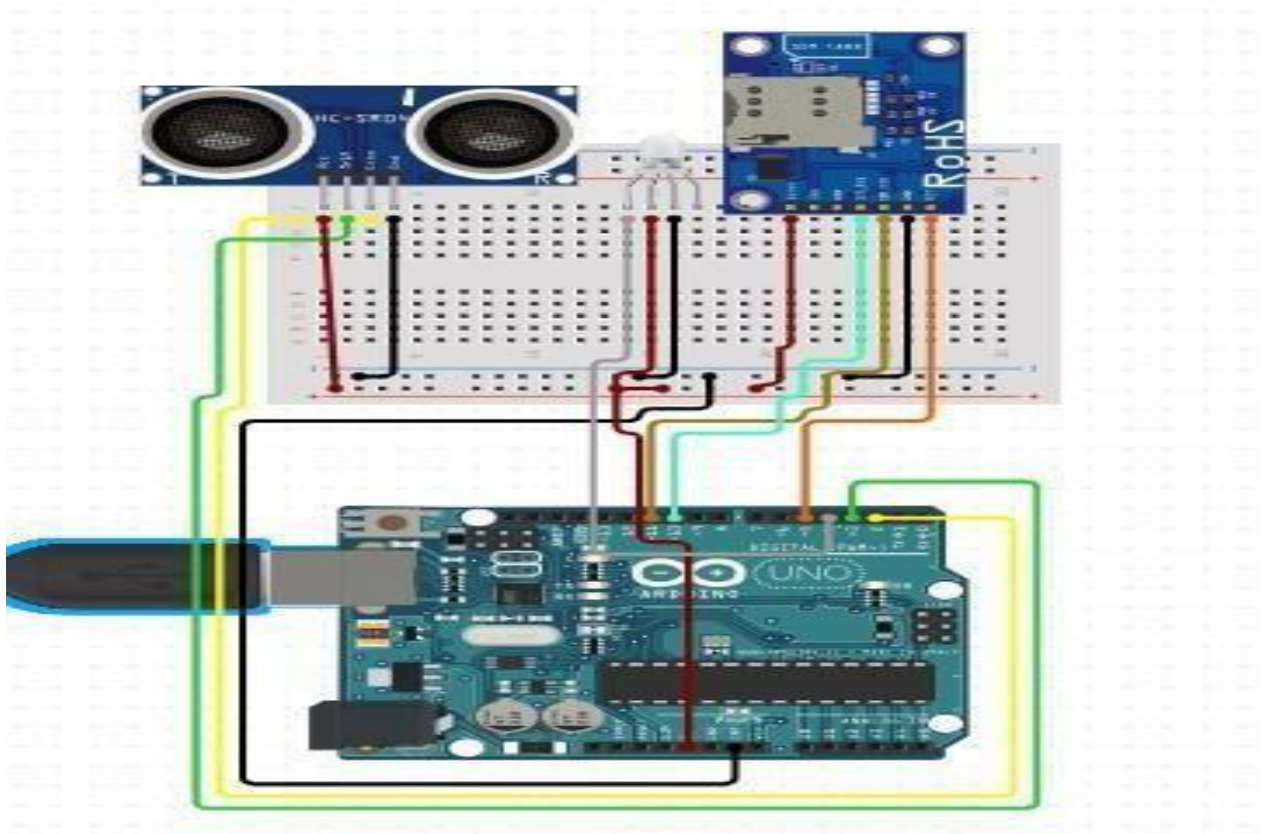


Fig 3.1 Block Diagram Implementation

The Arduino UNO is the microcontroller where it is connected with the Ultrasonic sensor on analog supply and earth connection supply. The LED light connected with the bread board with low level glucose monitoring. GSM is connected with the Arduino where a digital connection and analog connection with power supply has been given to forward a communication system.

3.4 SYSTEM REQUIREMENTS

3.4.1 Hardware Requirements

- Arduino UNO
- Ultrasonic Sensor
- GSM
- Buzzer Alert
- Power Supply
- Bread Board

- LED Light

3.4.2 Software Requirements

- Language : Embedded C
- Software : Arduino

3.5 USE CASE DIAGRAM

Here's a simplified use case diagram illustrating the interactions and relationships between the actors and the proposed drip monitoring system using Arduino, a buzzer, an ultrasonic sensor, and GSM technology in a healthcare setting:

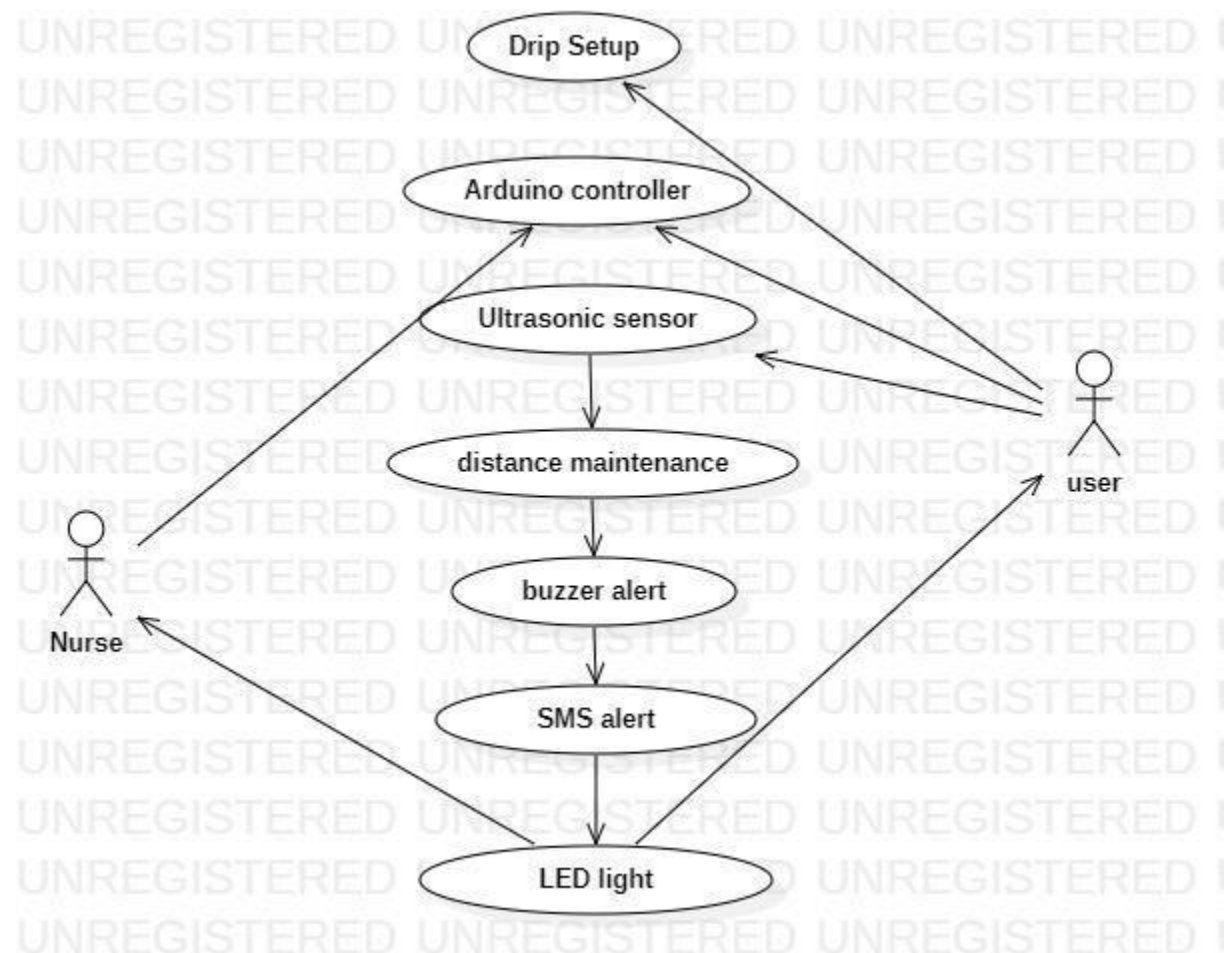


Fig 3.2 Use case Diagram

3.6 MODULES DESCRIPTION

- Arduino connection
- Ultrasonic sensor setup
- GSM inter communication
- Level detection
- Alert

3.6.1 Arduino Connection

To implement the proposed drip monitoring system using Arduino, a buzzer, an ultrasonic sensor, and GSM technology, you'll need to connect the components to the Arduino microcontroller. Here's how you can establish the connections:

Ultrasonic Sensor (HC-SR04):

- VCC (Power): Connect to the 5V pin on the Arduino.
- GND (Ground): Connect to any GND pin on the Arduino.
- Trig (Trigger): Connect to a digital pin (e.g., pin 7) on the Arduino.
- Echo (Echo): Connect to another digital pin (e.g., pin 8) on the Arduino.

Buzzer:

- Positive (+): Connect to a digital pin (e.g., pin 9) on the Arduino.
- Negative (-): Connect to the GND pin on the Arduino.
- GSM Module:
 - VCC (Power): Connect to the 5V pin on the Arduino.
 - GND (Ground): Connect to any GND pin on the Arduino.
 - TX (Transmit): Connect to a digital pin (e.g., pin 10) on the Arduino.

- RX (Receive): Connect to another digital pin (e.g., pin 11) on the Arduino.

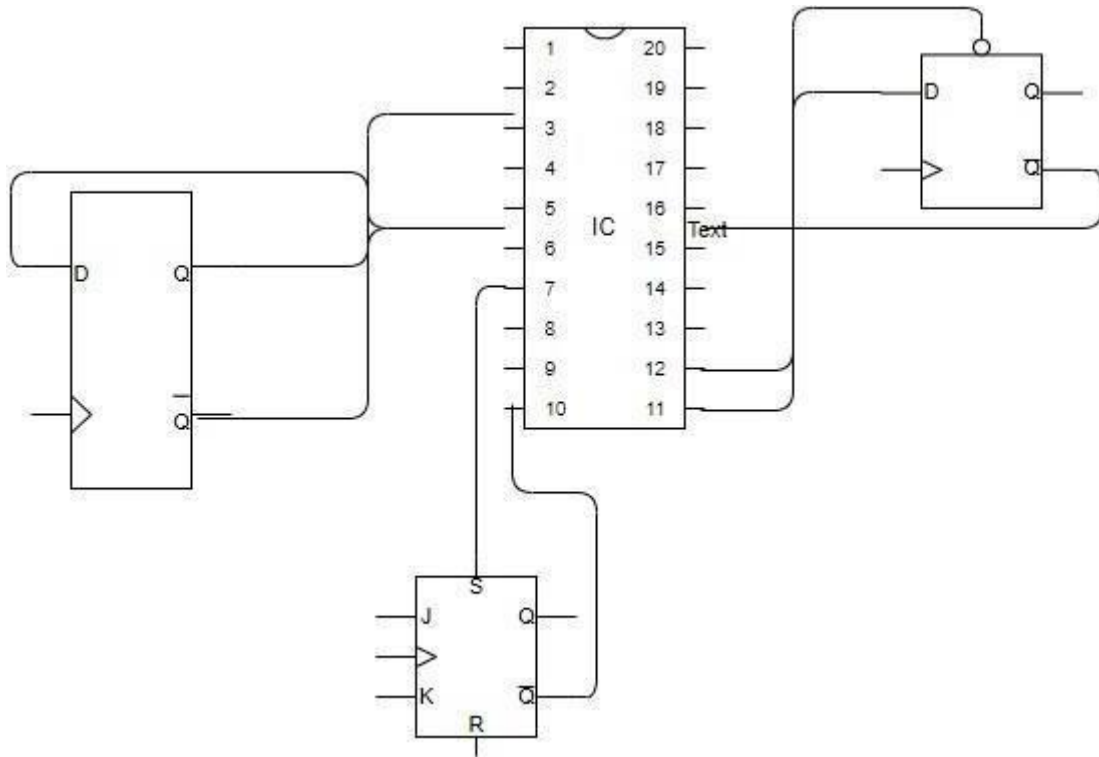


Fig 3.3 PIN Connection Diagram

3.6.2 Ultrasonic Sensor

Setting up the ultrasonic sensor (HC-SR04) involves connecting it to the Arduino and writing code to read distance measurements from the sensor. Here's a step-by-step guide to setting up the ultrasonic sensor:

Hardware Setup:

- Connect VCC (Power):

- Connect the VCC pin of the ultrasonic sensor to the 5V pin on the Arduino.
- Connect GND (Ground):
- Connect the GND pin of the ultrasonic sensor to any GND pin on the Arduino.
- Connect Trig (Trigger):
- Connect the Trig pin of the ultrasonic sensor to a digital pin (e.g., pin 7) on the Arduino.
- Connect Echo (Echo):
- Connect the Echo pin of the ultrasonic sensor to another digital pin (e.g., pin 8) on the Arduino.

Software Setup:

Install Arduino IDE: If you haven't already, download and install the Arduino IDE from the official Arduino website:
<https://www.arduino.cc/en/software>

Include Libraries: In the Arduino IDE, go to Sketch -> Include Library -> Manage Libraries. Search for "NewPing" and install the library by Tim Eckel. This library simplifies interfacing with ultrasonic sensors.

3.6.3 GSM Inter Communication

To enable GSM communication with the Arduino for sending SMS alerts, you'll need to connect a GSM module to the Arduino and write code to send messages through the module. Here's a step-by-step guide:

Hardware Setup:

- Connect VCC (Power):
- Connect the VCC pin of the GSM module to the 5V pin on the Arduino.

- Connect GND (Ground):
- Connect the GND pin of the GSM module to any GND pin on the Arduino.
- Connect TX (Transmit):
- Connect the TX pin of the GSM module to a digital pin (e.g., pin 10) on the Arduino.
- Connect RX (Receive):
- Connect the RX pin of the GSM module to another digital pin (e.g., pin 11) on the Arduino.

Insert SIM Card:

- Insert a valid SIM card into the GSM module, ensuring it is properly activated and has sufficient credit for sending SMS messages.

Software Setup:

- Install Arduino IDE: If you haven't already, download and install the Arduino IDE from the official Arduino website: <https://www.arduino.cc/en/software>
- Include Libraries: In the Arduino IDE, go to Sketch -> Include Library -> Manage Libraries. Search for "GSM" and install the library by Arduino.

3.6.4 Level Detection

To detect the fluid level in the IV drip chamber using an ultrasonic sensor, you can use the following approach:

Setup Ultrasonic Sensor: Connect the ultrasonic sensor to the Arduino as described earlier.

Read Distance Measurements: Use the ultrasonic sensor to measure the distance from the sensor to the liquid surface in the IV drip chamber. The sensor will emit ultrasonic pulses and measure the time it takes for the pulses to bounce back. This time can be converted into a distance measurement.

Calibrate: Calibrate the sensor to determine the distance readings corresponding to different fluid levels in the IV drip chamber. You may need to perform this calibration step experimentally by measuring the distance readings for various levels of fluid in the chamber.

Thresholds: Define threshold values for different fluid levels. For example, you could set a threshold for a low fluid level to trigger an alert when the distance reading falls below a certain value, indicating that the fluid level is critically low.

Alert Mechanism: Implement an alert mechanism in your Arduino code to trigger an alert (e.g., activate a buzzer or send an SMS) when the distance reading from the ultrasonic sensor crosses a predefined threshold, indicating a low fluid level in the IV drip chamber.

3.6.5 Alert System

To implement an alert system for detecting a low fluid level in the IV drip chamber using an ultrasonic sensor and Arduino, you can incorporate both visual and auditory alerts. Here's how you can set it up:

Hardware Setup:

Connect the ultrasonic sensor (HC-SR04) to the Arduino as described earlier.

Connect a buzzer to a digital pin on the Arduino for audible alerts.

Software Setup:

Write Arduino code to read distance measurements from the ultrasonic sensor and trigger alerts when the fluid level falls below a certain threshold. When the distance reading falls below the predefined threshold (`FLUID_THRESHOLD`), indicating a low fluid level, the Arduino triggers an alert. It activates the buzzer for 1 second to provide an audible alert.

You can customize the alert system further by adding additional features, such as sending SMS alerts using a GSM module or integrating with a display to provide visual alerts. Additionally, you may want to implement mechanisms to reset the alert once the fluid level is restored to normal.

CHAPTER 4

SOFTWARE SPECIFICATION

4.1 ARDUINO



Fig 4.1 Arduino Software

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the .ino extension on save.

IDE VERIFY File

Verify Checks your code for errors compiling it.

IDE UPLOAD File

Upload Compiles your code and uploads it to the configured board. See uploading below for details.

IDE NEW File

New Creates a new sketch.

IDE OPEN File

Open Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

IDE SAVE File

Save Saves your sketch.

IDE SERMON File

Serial Monitor Opens the serial monitor.

Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

File

- New Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
- Open Allows to load a sketch file browsing through the computer drives and folders.
- Open Recent Provides a short list of the most recent sketches, ready to be opened.
- Sketchbook Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.

Upload Using Programmer This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.

Export Compiled Binary Saves a .hex file that may be kept as archive or sent to the board using other tools. **Include Library** Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.

Add File... Adds a supplemental file to the sketch (it will be copied from its current location). The file is saved to the data subfolder of the sketch, which is intended for assets such as documentation. The contents of the data folder are not compiled, so they do not become part of the sketch program.

Tools

Auto Format This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.

Archive Sketch Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.

Fix Encoding & Reload Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.

Serial Monitor Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.

Board Select the board that you're using. See below for descriptions of the various boards.

Port This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.

Programmer For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're burning a bootloader to a new microcontroller, you will use this.

Burn Bootloader The items in this menu allow you to burn a bootloader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino board but is useful if you purchase a new ATmega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader on the target board. This command also sets the right fuses.

4.2 FEASIBILITY STUDY

A feasibility study for an Arduino-based alert system for detecting low fluid levels in an IV drip chamber involves assessing various aspects to determine the viability and practicality of the project. Here's how you can conduct a feasibility study for such a system:

Technical Feasibility:

Evaluate the technical feasibility of implementing the alert system using Arduino and other components (e.g., ultrasonic sensor, buzzer). Assess the compatibility of the chosen hardware components and ensure that they can interface seamlessly with the Arduino board. Determine if the required functionalities, such as distance measurement using the ultrasonic sensor and triggering alerts, can be implemented using Arduino programming.

Resource Feasibility:

Determine the availability of necessary resources for developing and deploying the alert system, including Arduino boards, sensors, and other hardware components. Assess the availability of skilled personnel with expertise in Arduino programming, electronics, and sensor interfacing. Evaluate the cost of acquiring hardware components and any additional resources needed for the project.

Time Feasibility:

Estimate the time required for developing the alert system, including hardware setup, software development, testing, and validation. Consider any potential delays or challenges that may arise during the development process and adjust the project timeline accordingly. Determine if the project can be completed within the desired timeframe and whether any constraints may affect the project schedule.

Operational Feasibility:

Evaluate the operational feasibility of the alert system in a real-world healthcare environment. Assess the ease of use, reliability, and effectiveness of the system in detecting low fluid levels in the IV drip chamber and triggering timely alerts. Consider factors such as user acceptance, maintenance requirements, and integration with existing healthcare systems.

Financial Feasibility:

Conduct a cost-benefit analysis to determine the financial feasibility of the project. Estimate the total project cost, including hardware, software development, testing, and deployment expenses. Compare the projected costs with the potential benefits, such as improved patient safety, reduced medication errors, and operational efficiencies, to assess the return on investment (ROI).

Regulatory and Compliance Feasibility:

Identify any regulatory requirements or compliance standards applicable to the alert system, especially if it will be used in a healthcare setting. Ensure that the system complies with relevant regulations, standards, and guidelines for medical devices, data privacy, and safety.

Risk Assessment:

Identify potential risks and challenges associated with the project, such as technical difficulties, supply chain disruptions, or regulatory hurdles. Develop mitigation strategies to address identified risks and minimize their impact on the project's success.

By conducting a comprehensive feasibility study, you can assess the viability of developing an Arduino-based alert system for detecting low fluid levels in an IV drip chamber and make informed decisions regarding the project's initiation and execution.

4.3 SYSTEM TESTING

System testing for an Arduino-based alert system for detecting low fluid levels in an IV drip chamber involves validating the functionality, performance, and reliability of the system. Here's how you can conduct system testing for such a project:

Functional Testing:

Verify that the system performs the intended functions correctly, such as reading distance measurements from the ultrasonic sensor and triggering alerts when the fluid level falls below a certain threshold. Test different scenarios, including normal operation and various levels of fluid depletion, to ensure that the system responds appropriately to different conditions.

Integration Testing:

Test the integration of hardware components, including the Arduino board, ultrasonic sensor, buzzer, and any other modules (e.g., GSM module). Ensure that the components communicate effectively with each other and that data flows seamlessly between them.

Alert Mechanism Testing:

Test the alert mechanism implemented in the system, such as activating the buzzer for audible alerts or sending SMS alerts using the GSM module. Verify that alerts are triggered accurately when the fluid level falls below the predefined threshold and that they are delivered promptly to the intended recipients.

Accuracy Testing:

Validate the accuracy of distance measurements obtained from the ultrasonic sensor by comparing them with known distances or physical measurements of fluid levels in the IV drip chamber. Ensure that the system provides reliable and consistent distance readings under various environmental conditions.

Stress Testing:

Subject the system to stress tests to evaluate its performance under extreme conditions, such as rapid changes in fluid levels, high noise environments, or sensor malfunctions. Assess how the system handles stress situations and identify any potential weaknesses or failure points that need to be addressed.

User Acceptance Testing (UAT):

Involve end-users, such as healthcare providers or medical staff, in testing the system to ensure that it meets their requirements and expectations. Gather feedback from users regarding the usability, effectiveness, and overall satisfaction with the system.

Reliability and Durability Testing:

Conduct reliability testing to assess the system's ability to operate consistently over time without failures or malfunctions. Evaluate the durability of hardware components and their resistance to wear and tear in a real-world healthcare environment.

By conducting thorough system testing, you can ensure that the Arduino-based alert system for detecting low fluid levels in an IV drip chamber meets the desired quality standards, performs reliably, and delivers the intended functionality to enhance patient safety in healthcare settings.

4.4 PRODUCT DESCRIPTION

4.4.1 Arduino UNO

The Arduino Uno is one of the most popular and widely used development boards in the Arduino ecosystem. It's a microcontroller board based on the ATmega328P microcontroller and is designed for beginners, hobbyists, and professionals alike. Here's a description of the Arduino Uno:

Microcontroller: The Arduino Uno is powered by the Atmel ATmega328P microcontroller, which operates at a clock speed of 16 MHz. The ATmega328P has 32 KB of flash memory for storing code (of which 0.5 KB is used by the bootloader), 2 KB of SRAM, and 1 KB of EEPROM.

Digital and Analog Pins: The Arduino Uno features 14 digital input/output (I/O) pins, of which 6 can be used as PWM (Pulse Width Modulation) outputs. These pins can be used to interface with various external devices, such as sensors, actuators, LEDs, and displays. Additionally, there are 6 analog input pins, which can be used to read analog voltage levels from sensors.

Power Supply: The Arduino Uno can be powered via USB connection, external DC power supply (7-12V), or a battery. It has a built-in voltage regulator that allows it to operate within a voltage range of 5V-20V.

Communication Interfaces: The Arduino Uno supports various communication interfaces, including:

USB: Used for programming the board and serial communication with a computer.

UART (Universal Asynchronous Receiver-Transmitter): Used for serial communication with external devices.

SPI (Serial Peripheral Interface) and I2C (Inter-Integrated Circuit): Used for communication with SPI and I2C compatible devices, such as sensors, displays, and EEPROMs.

Programming Environment: The Arduino Uno is programmed using the Arduino Integrated Development Environment (IDE), which is based on the Processing programming language. The IDE provides a user-friendly interface for writing, compiling, and uploading code to the Arduino board. It also includes a rich set of libraries and examples to simplify the development process.

Bootloader: The Arduino Uno comes with a pre-installed bootloader, which allows it to be programmed over a USB connection without the need for an external programmer. The bootloader enables easy firmware updates and makes the Arduino Uno suitable for rapid prototyping and development.

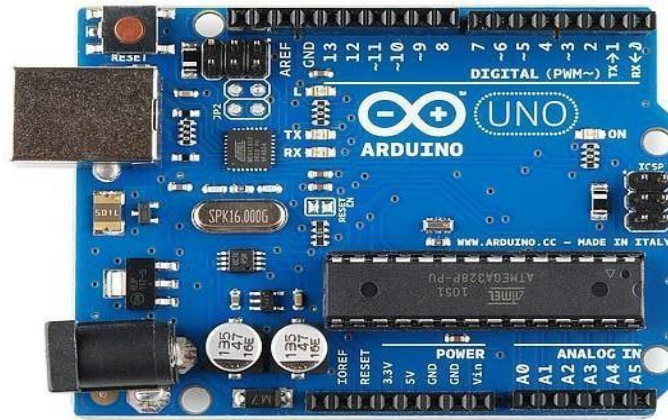


Fig 4.2 Arduino UNO

Open-Source Hardware and Software: The Arduino Uno is part of the Arduino open-source hardware and software platform, which means that its schematics, PCB layout, and software are freely available for modification and redistribution. This open approach encourages collaboration, innovation, and community-driven development within the Arduino ecosystem.

Overall, the Arduino Uno is a versatile and beginner-friendly development board that provides an accessible platform for building a wide range of projects, from simple LED blinkers to more complex IoT applications and robotics projects. Its ease of use, extensive documentation, and large community make it an ideal choice for makers, educators, and professionals looking to explore electronics and programming.

4.4.2 Ultrasonic Sensor

An ultrasonic sensor is a device that uses sound waves at frequencies higher than the human audible range to detect the distance of an object. Here's an overview of how ultrasonic sensors work and their applications:

Ultrasonic sensors typically consist of a transmitter and a receiver. The transmitter emits ultrasonic waves, which travel through the air and bounce off objects in their path. The receiver detects the reflected waves.

By measuring the time it takes for the ultrasonic waves to travel to the object and back, the sensor can calculate the distance between the sensor and the object using the speed of sound in air.

The distance measurement is typically proportional to the time taken for the sound waves to travel to the object and back. This time is often converted into a distance measurement using the speed of sound in air (approximately 343 meters per second at room temperature).

Applications of Ultrasonic Sensors:

Distance Measurement: Ultrasonic sensors are commonly used for measuring distances to objects without physical contact. They are widely used in robotics, automation, and industrial applications for obstacle detection and collision avoidance.

Proximity Sensing: Ultrasonic sensors can detect the presence of objects within a certain range. They are often used in security systems, automatic door openers, and smart appliances for proximity sensing.



Fig 4.3 Ultrasonic sensor

Level Sensing: Ultrasonic sensors can measure the level of liquids or solids in containers, tanks, or vessels. They are used in industrial processes, agricultural equipment, and automotive systems for level monitoring and control.

Flow Measurement: Ultrasonic sensors can measure the flow rate of fluids, such as water or gas, by detecting the velocity of sound waves traveling through the fluid. They are used in flow meters for applications such as water management, fuel monitoring, and HVAC systems.

Object Detection: Ultrasonic sensors can detect the presence or absence of objects within their detection range. They are used in automation, robotics, and IoT applications for object detection, counting, and sorting.

Advantages of Ultrasonic Sensors:

Non-contact measurement: Ultrasonic sensors can measure distances without physical contact with the object, making them suitable for applications where contact is impractical or undesirable.

Wide operating range: Ultrasonic sensors can detect objects at distances ranging from a few centimeters to several meters, depending on the sensor's specifications.

Environmental robustness: Ultrasonic sensors are not affected by factors such as color, transparency, or surface texture of the object, making them suitable for various environments and conditions.

Limitations of Ultrasonic Sensors:

Limited accuracy: Ultrasonic sensors may have limitations in accuracy, especially at longer distances or in challenging environmental conditions (e.g., temperature variations, humidity).

Sensitivity to air disturbances: Ultrasonic sensors may be affected by air currents, wind, or acoustic noise, which can interfere with distance measurements.

Blind spots: Ultrasonic sensors may have blind spots or dead zones where objects cannot be detected, especially at close range or in confined spaces.

Overall, ultrasonic sensors are versatile devices with a wide range of applications in various industries, offering non-contact distance measurement, proximity sensing, and object detection capabilities. They are commonly used in robotics, automation, automotive, industrial, and consumer electronics applications for their reliability, ease of use, and affordability.

4.4.3 GSM Communication Setup

GSM (Global System for Mobile Communications) communication enables devices to communicate over cellular networks, allowing them to send and receive data, including text

messages (SMS), voice calls, and internet connectivity. Here's an overview of GSM communication and its applications:

SIM Card: GSM-enabled devices, such as mobile phones or GSM modules, require a SIM (Subscriber Identity Module) card to connect to the cellular network. The SIM card stores the user's subscriber information, including phone number and authentication data.

Cellular Network: GSM operates on a cellular network consisting of base stations (cell towers) that provide coverage over a specific geographic area. Each base station is connected to the core network, allowing devices to communicate with each other and with external networks.

Handshake: When a GSM-enabled device is powered on and registers with the cellular network, it performs a handshake process to authenticate and establish a connection. This involves exchanging encrypted data with the network to verify the device's identity and authorize network access.

Data Transmission: Once the connection is established, the device can send and receive data over the cellular network. This includes voice data for phone calls, packet data for internet access, and SMS messages for text communication.

Protocols: GSM uses various protocols for communication, including signaling protocols (e.g., GSM MAP), radio protocols (e.g., GSM, GPRS), and application protocols (e.g., SMS, USSD). These protocols define how data is transmitted, routed, and processed within the cellular network.

Applications of GSM Communication:

Mobile Phones: GSM communication is widely used in mobile phones for voice calls, text messaging (SMS), and internet access (GPRS/3G/4G).

IoT Connectivity: GSM modules are commonly used in IoT (Internet of Things) devices for remote monitoring, control, and data logging. They enable devices to communicate over cellular networks without relying on Wi-Fi or wired connections.

Telemetry and Tracking: GSM communication is used in telemetry systems for remote monitoring of assets, such as vehicles, equipment, and environmental sensors. It allows real-time tracking and data transmission over long distances.

Alarm and Security Systems: GSM modules are integrated into alarm and security systems for sending alerts, notifications, and status updates to users or monitoring centers via SMS or voice calls.

Healthcare Monitoring: GSM-enabled devices are used in healthcare applications for remote patient monitoring, telemedicine, and emergency alerting. They enable healthcare providers to track patients' vital signs, medication adherence, and health status remotely.



Fig 4.4 GSM Module

Advantages of GSM Communication:

Wide Coverage: GSM networks provide extensive coverage in urban, suburban, and rural areas, making them suitable for applications requiring long-range communication.

Global Compatibility: GSM is an international standard adopted by networks worldwide, ensuring interoperability and compatibility between devices and networks in different countries.

Security: GSM communication incorporates encryption and authentication mechanisms to protect data privacy and prevent unauthorized access.

Reliability: GSM networks are designed for high reliability and availability, with redundant infrastructure and failover mechanisms to ensure continuous service.

Limitations of GSM Communication:

Coverage Limitations: GSM coverage may be limited in remote or sparsely populated areas, where cellular infrastructure is less developed or unavailable.

Bandwidth Constraints: GSM networks have limited bandwidth compared to broadband technologies, which may result in slower data transfer rates for internet access.

Latency: GSM communication introduces latency due to signal propagation delays, network routing, and processing overhead, which may impact real-time applications.

Overall, GSM communication offers a reliable and widely adopted technology for wireless connectivity, enabling a wide range of applications in telecommunications, IoT, healthcare, security, and beyond. Its ubiquity, compatibility, and robustness make it a preferred choice for many wireless communication needs.

4.5 POWER SUPPLY

The power supply for an Arduino-based system, including the ultrasonic sensor and GSM module, is crucial for reliable operation. Here are some considerations for selecting and configuring the power supply:

Voltage Requirement: Arduino Uno operates at 5V DC. Ensure that the power supply provides a stable 5V output to the Arduino board. Many USB chargers or adapters provide 5V DC output and can be used as a power source.

Current Rating: The total current requirement depends on the power consumption of all components, including the Arduino board, ultrasonic sensor, GSM module, and any other peripherals. Make sure the power supply can deliver sufficient current to meet the maximum demand of the system.

Power Input: Arduino Uno can be powered through the USB port, DC barrel jack, or VIN pin. Choose the appropriate input based on the power supply you're using. If using a USB charger, connect it to the USB port. If using an external DC power supply, connect it to the DC barrel jack or VIN pin (make sure the voltage does not exceed 12V to avoid damaging the board).

Stability and Regulation: Ensure that the power supply provides stable and regulated output voltage to prevent fluctuations or voltage spikes that could damage the components. Using a regulated power supply with low ripple and noise is recommended for reliable operation.

Battery Backup: Consider adding a battery backup or uninterruptible power supply (UPS) to ensure continued operation in case of mains power failure. This is particularly important for critical applications where uninterrupted operation is essential, such as alarm systems or remote monitoring.

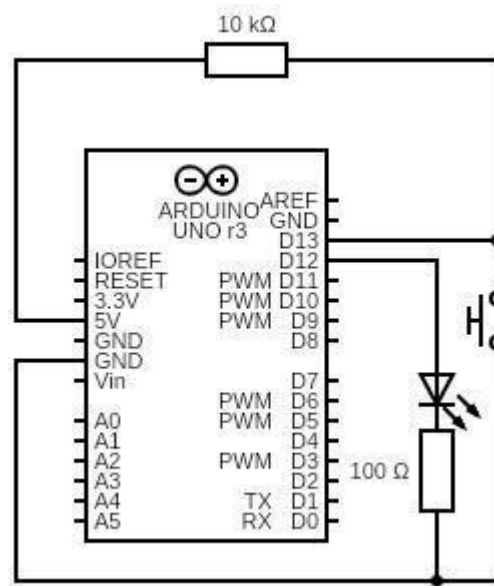


Fig 4.5 Power Supply for arduino

Power Consumption Optimization: Optimize the power consumption of the system by minimizing the use of power-hungry components, such as LEDs or high-power sensors, when not needed. Use low-power modes or sleep modes where applicable to reduce overall power consumption and extend battery life in battery-powered applications.

Overcurrent and Overvoltage Protection: Incorporate overcurrent and overvoltage protection measures to safeguard the components from damage due to excessive current or voltage spikes. This can be achieved using fuses, voltage regulators, or protective circuitry.

Heat Dissipation: Ensure proper heat dissipation to prevent overheating of the components, especially voltage regulators or power supply modules. Adequate ventilation and heat sinks may be required in enclosed or high-temperature environments.

By carefully selecting and configuring the power supply for your Arduino-based system, you can ensure reliable and stable operation while protecting the components from potential damage due to overvoltage, overcurrent, or other electrical anomalies.

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENT

5.1 CONCLUSION

In conclusion, designing an Arduino-based alert system for detecting low fluid levels in an IV drip chamber involves several key components and considerations. By integrating an ultrasonic sensor, GSM module, and buzzer with an Arduino Uno board, you can create a versatile and effective solution for monitoring fluid levels and triggering timely alerts.

The ultrasonic sensor provides accurate distance measurements, allowing the system to detect the fluid level in the IV drip chamber. When the fluid level falls below a predefined threshold, the Arduino triggers an alert mechanism, such as activating a buzzer for audible alerts and sending SMS messages through the GSM module for remote notification.

Throughout the development process, it's essential to conduct feasibility studies, system testing, and power supply selection to ensure the system's viability, functionality, and reliability. By addressing technical, resource, operational, financial, and regulatory aspects, you can design a robust and efficient alert system suitable for various healthcare, industrial, and IoT applications.

In summary, the Arduino-based alert system offers a cost-effective, customizable, and scalable solution for enhancing patient safety, improving process efficiency, and enabling remote monitoring in healthcare environments. With proper design, testing, and implementation, the system can provide valuable insights, real-time alerts, and proactive intervention to healthcare providers, ensuring optimal care delivery and patient outcomes.

5.2 FUTURE ENHANCEMENT

For future enhancement of the Arduino-based alert system for detecting low fluid levels in an IV drip chamber, several possibilities can be explored to improve functionality, performance, and usability. Here are some ideas:

Data Logging and Analytics: Integrate data logging capabilities to record fluid level measurements over time. Implement analytics algorithms to analyze trends, detect patterns, and generate insights into fluid consumption, usage patterns, and potential issues.

Remote Configuration and Monitoring: Develop a web or mobile interface to remotely configure system parameters, set alert thresholds, and monitor real-time status and alerts. Enable healthcare providers to access the system remotely and take proactive actions as needed.

Integration with Electronic Health Records (EHR): Integrate the alert system with electronic health record systems to capture fluid level data and alert notifications directly into patient records. Enhance care coordination, documentation, and decision-making processes for healthcare providers.

Machine Learning and Predictive Maintenance: Implement machine learning algorithms to predict potential equipment failures or maintenance needs based on historical data and sensor readings. Enable predictive maintenance scheduling to minimize downtime and ensure equipment reliability.

APPENDIX

SOURCECODE

Arduino Connection

```
#include <NewPing.h>

#define TRIGGER_PIN 7

#define ECHO_PIN 8

#define MAX_DISTANCE 200 // Maximum distance in centimeters

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // Create a NewPing object

void setup() {

    Serial.begin(9600); // Initialize serial communication

}

void loop() {

    delay(50); // Wait for 50 milliseconds between measurements

    unsigned int distance = sonar.ping_cm(); // Send a ping and get the distance in centimeters

    Serial.print("Distance: ");

    Serial.print(distance);

    Serial.println(" cm");

}
```

GSM Communication

```
#include <GSM.h>
```

```

#define PINNUMBER "" // Leave empty if your SIM card doesn't have a PIN

#define GSM_TX_PIN 10

#define GSM_RX_PIN 11


GSM gsmAccess; // Create a GSM object

GSM_SMS sms; // Create an SMS object


void setup() {

    Serial.begin(9600); // Initialize serial communication

    while (!Serial) {

        ; // Wait for serial port to connect

    }


    // Initialize GSM module with appropriate pins

    gsmAccess.begin(PINNUMBER, true, false, GSM_TX_PIN, GSM_RX_PIN);

}


void loop() {

    Serial.println("Sending SMS...");

    sms.beginSMS("+1234567890"); // Replace with recipient's phone number

```

```

sms.print("Alert: Low fluid level in IV drip chamber!"); // SMS message content

sms.endSMS();

Serial.println("SMS sent!");

delay(5000); // Wait for 5 seconds before sending another SMS

}

```

Ultrasonic Setup

```

#include <NewPing.h>

#define TRIGGER_PIN 7

#define ECHO_PIN 8

#define MAX_DISTANCE 200 // Maximum distance in centimeters

#define FLUID_THRESHOLD 10 // Threshold for low fluid level in centimeters


NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // Create a
NewPing object


void setup() {

  Serial.begin(9600); // Initialize serial communication

}


void loop() {

```

```

delay(50); // Wait for 50 milliseconds between measurements

unsigned int distance = sonar.ping_cm(); // Send a ping and get the distance in
centimeters

Serial.print("Distance: ");

Serial.print(distance);

Serial.println(" cm");


// Check if distance reading is below threshold
if (distance < FLUID_THRESHOLD) {

    // Trigger alert (e.g., activate buzzer or send SMS)

    Serial.println("Alert: Low fluid level detected!");

    // Add code here to activate buzzer or send SMS

}

}

```

Buzzer Setup

```

#include <NewPing.h>


#define TRIGGER_PIN 7

#define ECHO_PIN 8

#define MAX_DISTANCE 200 // Maximum distance in centimeters

```

```

#define FLUID_THRESHOLD 10 // Threshold for low fluid level in centimeters

#define BUZZER_PIN 9 // Digital pin for the buzzer


NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // Create a
NewPing object


void setup() {

    Serial.begin(9600); // Initialize serial communication

    pinMode(BUZZER_PIN, OUTPUT); // Set buzzer pin as output
}


void loop() {

    delay(50); // Wait for 50 milliseconds between measurements

    unsigned int distance = sonar.ping_cm(); // Send a ping and get the distance in
centimeters

    Serial.print("Distance: ");

    Serial.print(distance);

    Serial.println(" cm");

    // Check if distance reading is below threshold

```

```

if (distance < FLUID_THRESHOLD) {

    // Trigger alert

    Serial.println("Alert: Low fluid level detected!");

    // Activate buzzer

    digitalWrite(BUZZER_PIN, HIGH); // Turn buzzer on

    delay(1000); // Buzzer on for 1 second

    digitalWrite(BUZZER_PIN, LOW); // Turn buzzer off

    delay(1000); // Wait for 1 second

}

}

```

Integration Code

```

#include <ArduinoIoTCloud.h>

#include <Arduino_ConnectionHandler.h>

const char THING_ID[] = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx"; //
Replace with your Thing ID

const char USERNAME[] = "arduino"; // Replace with your Arduino IoT Cloud
username

```

```
const char DEVICE_ID[] = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx"; //
```

Replace with your device ID

```
const char DEVICE_SECRET[] = "your-device-secret"; // Replace with your  
device secret
```

```
void setup() {
```

```
    // Initialize serial communication
```

```
    Serial.begin(9600);
```

```
    // Initialize Arduino IoT Cloud
```

```
    ArduinoCloud.begin(ArduinoIoTPreferredConnection, USERNAME,  
DEVICE_ID, DEVICE_SECRET);
```

```
    // Add variables to Arduino IoT Cloud
```

```
    setDebugMessageLevel(2);
```

```
    ArduinoCloud.addDevice(THING_ID, "fluid_level", "numeric", NULL, NULL,  
NULL);
```

```
}
```

```
void loop() {
```

```
    // Read fluid level from ultrasonic sensor
```

```

float fluidLevel = readFluidLevel();

// Update IoT Cloud variable with fluid level data
ArduinoCloud.update();

// Delay before next iteration
delay(1000);
}

float readFluidLevel() {
    // Code to read fluid level from ultrasonic sensor

    // Replace with your actual code to read distance measurements from the
    ultrasonic sensor and calculate fluid level

    // Example:

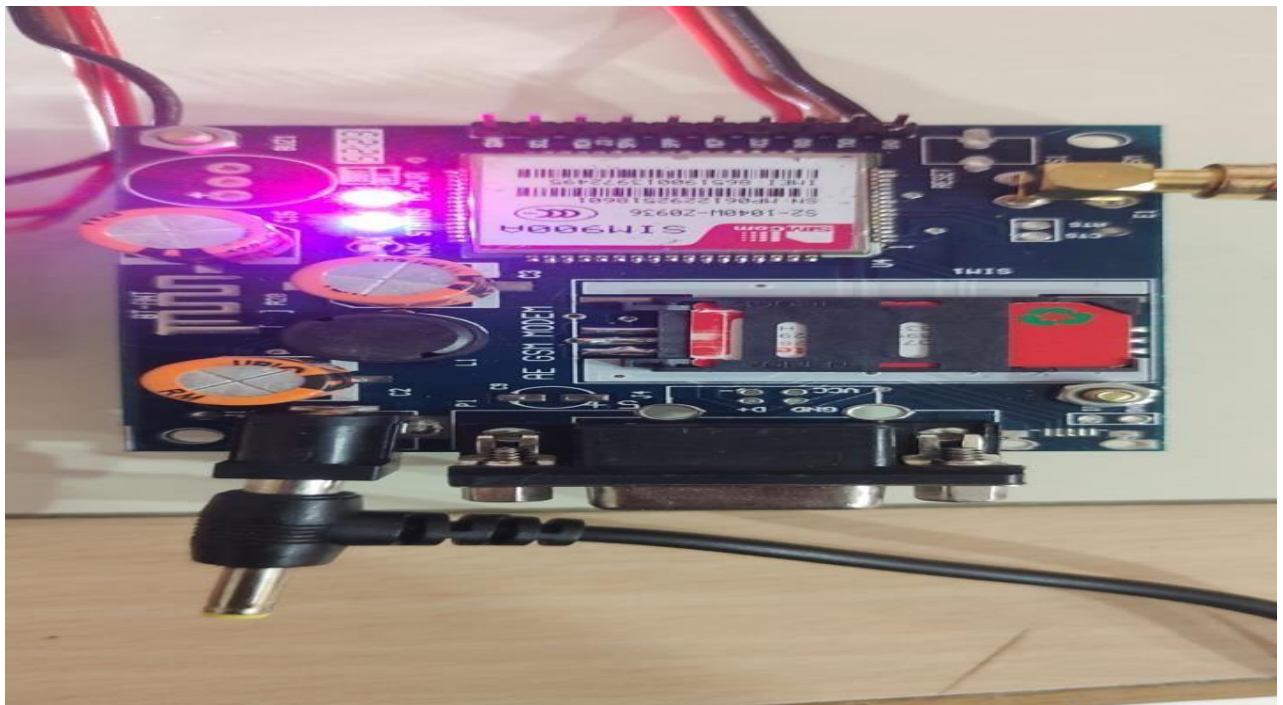
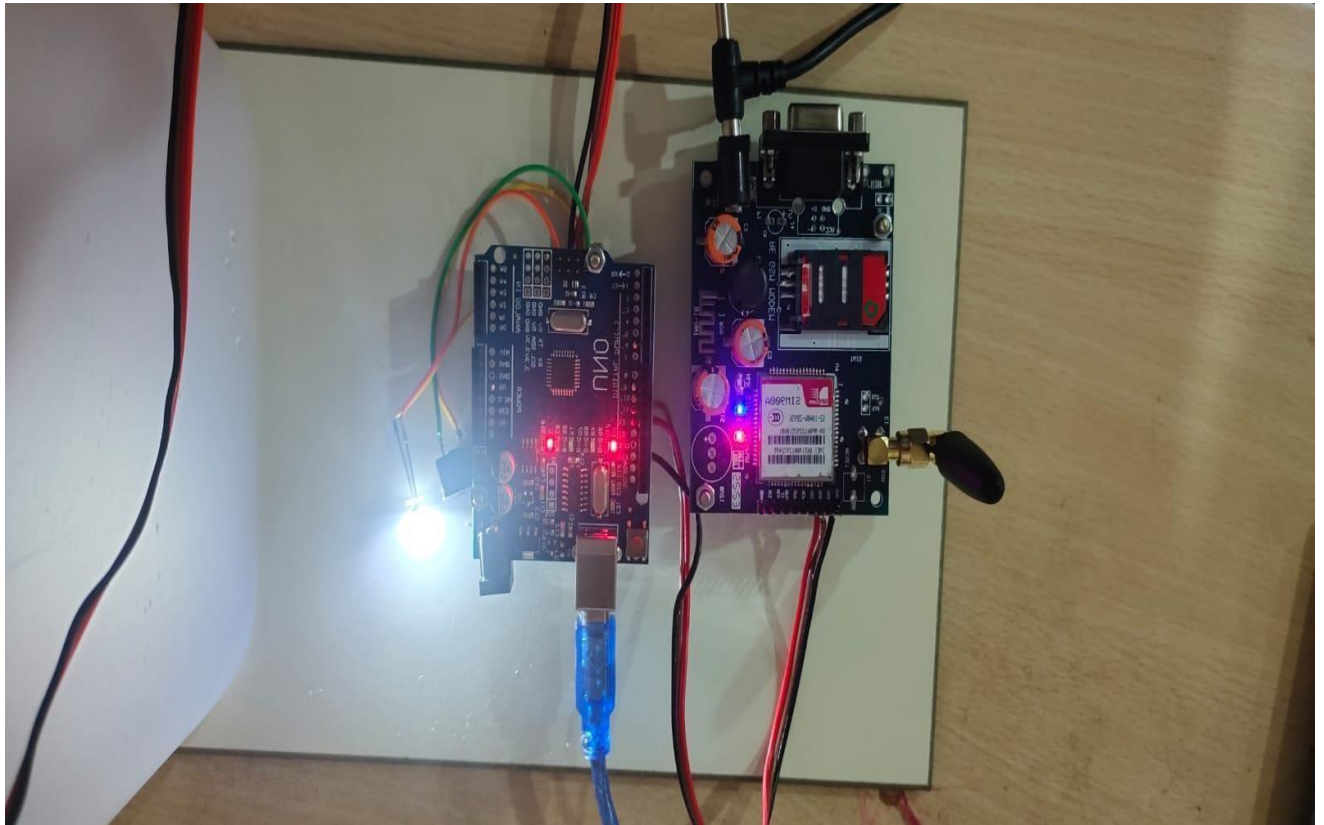
    // float distance = readUltrasonicSensor();

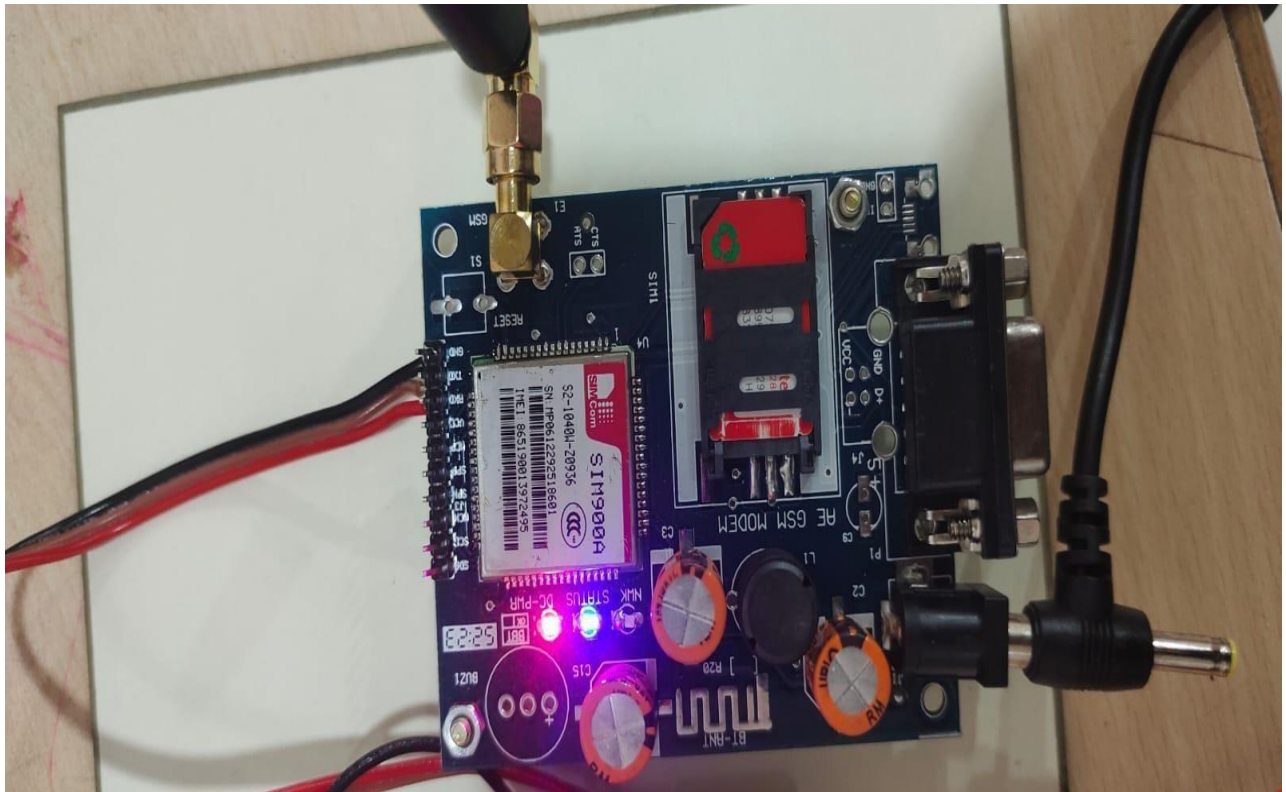
    // float fluidLevel = map(distance, 0, maxDistance, 0, 100); // Map distance to
    fluid level percentage

    // return fluidLevel;
}

```

SCREENSHOTS







SS

REFERENCES

1. IV Fluid Monitoring by Lalitha krithika. K, Khyati. S. Davey, T. Kaliselvi.
2. Hung, Orlando R., Peter H. Gregson, and David C. Roach. "Fluid monitoring device." U.S. Patent No. 7,327,273. 5 Feb. 2008.
3. Rader, R. Scott, et al. "Liquid level sensing and monitoring system for medical fluid infusion systems." U.S. Patent No. 5,563,584. 8 Oct. 1996.
4. Barney, George M. "Instrument for measuring and computing heart beat, body temperature and other physiological and exercise-related parameters." U.S. Patent No. 4,312,358. 26 Jan. 1982.
5. Issachar, David. "Liquid level sensor." U.S. Patent No. 6,028,521. 22 Feb. 2000.
6. Sukanesh, R., S. Vijayprasath, and P. Subathra. "GSM based ECG tele-alert system." Innovative Computing Technologies (ICICT), 2010 International Conference on. IEEE, 2010.
7. Lv,Ziyu, et al. "iCare: a mobile health monitoring system for the elderly." Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing. IEEE Computer Society, 2010.
8. Aziz, Kahtan, et al. "Smart real-time healthcare monitoring and tracking system using GSM/GPS technologies." Big Data and Smart City (ICBDSC), 2016 3rd MEC International Conference on. IEEE, 2016.
9. Pramanik, Aniket, et al. "GSM based Smart home and digital notice board." Computational Techniques in Information and Communication Technologies (ICCTICT), 2016 International Conference on. IEEE, 2016.
10. Lassen, K. "Intravenous fluid therapy." British Journal of Surgery 96.2 (2009).
11. Bahrudin, Md Saifudaullah Bin, Rosni Abu Kassim, and Norlida Buniyamin. "Development of Fire Alarm System using Raspberry Pi and Arduino Uno." Electrical, Electronics and System Engineering (ICEESE), 2013 International Conference on. IEEE, 2013.

12. R.Vasuki, C.Dennis, HemPriya Changer, “An portable monitoring device of measuring drips rate by using an Intravenous (IV) set”, International Journal of Biotechnology Trends and Technology Vol. 1, Issue 3, No.4 2011.
13. C.C.Gavimath, Krishnamurthy Bhat, C.C.Chayalakshmi, R.S.Hooli, B.E.Ravishankera “Design and Development of versatile saline flow rate measuring system and GSM based remote monitoring device”, International Journal Of Pharmaceutical Applications ISSN 0976-2639.
14. R.Aravind, Syed Mustak Ahmed “Design of family health monitoring system using wireless communication”, International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 9, September 2013
15. V.Ramya, B.Palaniappan, Anuradha Kumari “Embedded patient monitoring system” International Journal of Embedded Systems and Applications (IJESA) Vol.1, No.2, December 2011.
16. D.Janani, J.Prathibanandhi, P.Meenakshi Vidya, K.S.Sujatha “Wireless Saline Bottle Level Indicator for Hospitals”, Compo soft an International Journal of Advanced computerTechnology.