



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **D.OJASWINI (40130159)** who carried out the project entitled **AUTOMATION VENTILATOR USING WITH BLOOD OXYGEN SENSOR** under my supervision from November 2023 to April 2024.

INTERNAL GUIDE

Mr.V.KALIST,M.E

HEAD OF THE DEPARTMENT

Dr.T.RAVI,M.E,Ph.D.

Submitted for Viva voca Examination held on 18/04/2024

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

DECLARATION

I D.OJASWINI (40130159) hereby declare that the Project Report entitled **"AUTOMATION VENTILATOR USING WITH BLOOD OXYGEN SENSOR"** done by me under the guidance of **Mr.V.KALIST,M.E** is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in **DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

DATE : 18/4/24

PLACE: Chennai

Ojaswini D

SIGNATURE OF THE CANDIDATE

1. D.OJASWINI

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

I convey my thanks to **Dr. N. M. NANDHITHA, M.E., Ph.D., Dean, School of Electrical and Electronics Engineering** and **Dr. T. RAVI, M.E., Ph.D., Head of the Department, Department of Electronics and Communication Engineering** 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 **Mr.KALIST M.E** for his valuable guidance, suggestion and constant encouragement for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

This project describes overview of various research done on how human lungs used for respiration. They use push mechanism in each breath. Inhalation and exhalation process take place. The ventilator here we design is to help people during emergency Situation. It is very cheap and affordable. When people suffer from lungs or breathing problem this can be used for emergency situation. Motor mechanism is used to push the air bag. At the point when oxygen level counts are low this mechanism can be performed little screen is used to display. The oxygen levels. The entire system is driven by an Arduino microcontroller and a buzzer is fitted to detect any low levels of oxygen count.

Human lungs use the reverse pressure generated by contraction motion of the diaphragm to suck in air for breathing. A contradictory motion is used by a ventilator to inflate the lungs by pumping type motion. A ventilator mechanism must be able to deliver in the range of 10 – 30 breaths per minute, with the ability to adjust rising increments in sets of 2. Along with this the ventilator must have the ability to adjust the air volume pushed into lungs in each breath. The last but not the least is the setting to adjust the time duration for inhalation to exhalation ratio. Apart from this the ventilator must be able to monitor the patients blood oxygen level and exhaled lung pressure. The ventilator we here design and develop using arduino encompasses all these requirements to develop a reliable yet affordable ventilator to help in times of pandemic.

We here use a silicon ventilator bag coupled driven by DC motors with 2 side push mechanism to push the ventilator bag. We use toggle switch for switching and a variable pot to adjust the breath length and the BPM value for the patient. Our system makes use of blood oxygen sensor along with sensitive pressure sensor to monitor the necessary vitals of the patient and display on a mini screen. Also an emergency buzzer alert is fitted in the system to sound an alert as soon as any anomaly is detected. The entire system is driven by arduino controller to achieve desired results and to assist patients in emergency situations.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF ABBREVIATIONS	viii
	LIST OF FIGURES	ix
1	INTRODUCTION	1
	1.1 OVERVIEW	1
	1.2 KEY FEATURES	3
2	LITERATURE SURVEY	4
3	AIM AND SCOPE	9
	3.1 AIM OF PROJECT	9
	3.2 FUTURE SCOPE	10
4	METHODOLOGY & ALGORITHM USED	12
	4.1 RESEARCH METHODOLOGY	12
	4.2 EXISTING METHODOLOGY	13
	4.3 PROPOSED METHODOLOGY	15
	4.4 HARDWARE REQUIREMENT OF PROPOSED METHOD	16
	4.4.1 LCD DISPLAY	16
	4.4.2 DHT 11 SENSOR	17
	4.4.3 SERVO MOTOR	18
	4.4.4 AIR BAG	19
	4.4.5 VENTILATOR	20
	4.4.6 PULSE OXIMETER	21

	4.4.7 HEART BEAT SENSOR	22
	4.4.8 BUZZER	23
	4.5 SOFTWARE USED	24
	4.5.1 VISUAL STUDIO	24
	4.5.2 ARDUINO	26
	4.6 MECHANISM	27
	4.7 WORKING PRINCIPLE	29
	4.8 OPERATION ANALYSIS	34
	4.9 APPENDICES	36
5	RESULT AND DISCUSSION	42
	5.1 PERFORMANCE ANALYSIS	42
	5.2 OUTPUT OCCURANCE	44
	5.2.1 DIGITAL	44
	5.2.2 ANALOG	45
	5.3 RESULTS	46
6	SUMMARY AND CONCLUSION	50
	6.1 SUMMARY	50
	6.2 STANDARDS	51
	6.3 CONSTRAINTS	51
	6.4 TRADE – OFFS	52
	REFERENCE	53

LIST OF ABBREVIATIONS

PWM	PULSE WIDTH MODULATION
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
DTR	DEEP TENDON REFLEXES
AR	ANALOG REFERENCE
ICSP	IN CIRCUIT SERIAL PROGRAMMING
DHT	DIGITAL TEMPERATURE AND HUMIDITY
RPM	REVOLUTIONS PER MINUTE
MCU	MICROCONTROLLER UNIT
LCD	LIQUID CRYSTAL DISPLAY
RCA	ROOT CASE ANALYSIS
CRT	CATHODE RAY TECHNOLOGY
IPS	IN PLANE SWITCHING
VA	VERTICAL ALIGNMENT
AFFS	ADVANCED FRINGE FIELD SWITCHING
TFT	THIN FILM TRANSISTORS
CRO	CATHODE RAY TUBE
CRFL	COLD CATHODE FLUORESCENT LAMPS
LDR	LIGHT DEPENDENT RESISTOR

LIST OF FIGIURES

FIGURE NO	TITLE	PAGE NO
4.1	BLOCK DIAGRAM OF EXISTING SYSTEM	14
4.2	BLOCK DIAGRAM OF PROPOSED SYSTEM	16
4.3	LCD DISPLAY	17
4.4	DHT11	18
4.5	SERVO MOTOR	18
4.6	AIR BAG	19
4.7	VENTILATOR	20
4.8	PULSE OXIMETER	21
4.9	HEART BEAT SENSOR	23
4.10	BUZZER	24
4.11	ARDUINO UNO PERFOMANCE	33
4.12	ARDUINO MINI SETUP	33
5.1	HARDWARE SETUP	47
5.2	BLOOD OXYGEN SATURATION MEASUREMENT	48
5.3	MEASUREMENT BLOOD OXYGEN LEVEL PREDICTION AS NORMAL	49
5.4	BLOOD OXYGEN LEVEL PREDICTION AS ABNORMAL	49

CHAPTER - 1

INTRODUCTION

One significant gadget for which request has increase is ventilators for patients whoneed help with their breathing because of the respiratory issues. Essentially a ventilator is a machine that gives breathable air into and out of the lungs, to convey breathsto a patient who is actually incapable to inhale or breathing inadequately. A ventilator may not be effective as that of a medical grade ventilator yet it can go about as a decent substitute in the event that it has authority over the accompanying key boundaries. It is alsocombined with robust blood oxygen measuring sensor. Pulse oximetry might be utilized in both inpatient and outpatient settings. In some cases, your doctor may recommend that youhave a pulse oximeter for home use. During a pulse oximetry perusing, a little clamp like device is set on a finger, earlobe, or toe. Little light emissions go through the blood in the finger, estimating the measure of oxygen. It does this by estimating changes of light absorption in oxygenated or deoxygenated blood. This is a painless process. Yet this reliableand affordable ventilator during emergency times. After designing this model thesemodels are distributed and, on the web, so that others can also use it and design their ownventilator, even at the small scale.

Coronavirus disease 2019 (COVID-19), caused by a novel coronavirus (SARS-CoV-2), is inpart so dangerous because it threatens to overwhelm our medical infrastructure at the regional level, causing spikes in mortality rates. Within the medical infrastructure, there arecritical technologies that are generally available, but simply do not exist in a high enough density to handle the excessive volume of patients associated with pandemics. Thus, people die unnecessarily throughout the world because of a combination of COVID-19 infections and the lack of access to some of these technologies. Ventilators are an example of technologies that are currently in critical short supply.

1.1 OVERVIEW

In the wake of the COVID-19 pandemic, there has been a significant emphasis on developing innovative solutions to combat the shortage of medical equipment, particularly ventilators. Ventilators play a critical role in supporting patients with severe respiratory issues, including those caused by COVID-19. In this context, the integration of automation technology, particularly Arduino-based systems, offers a promising avenue for creating cost-effective and efficient ventilator solutions. One key aspect of ventilator functionality is monitoring the patient's blood oxygen levels. Blood oxygen saturation (SpO₂) is a crucial parameter that indicates the amount of oxygen carried in the blood. Monitoring SpO₂ levels helps healthcare professionals assess the patient's respiratory status and adjust ventilation parameters accordingly.

Arduino, an open-source electronics platform, provides a versatile and accessible framework for developing custom automation solutions. Its flexibility, affordability, and ease of programming make it an ideal choice for prototyping medical devices like ventilators. By leveraging Arduino, developers can design ventilator systems tailored to specific requirements while ensuring reliability and ease of deployment.

In this project, we propose an Arduino-based automated ventilator system equipped with a blood oxygen sensor. The system is designed to deliver precise and adjustable respiratory support to patients while continuously monitoring their SpO₂ levels. The integration of the blood oxygen sensor enables the ventilator to maintain optimal oxygenation levels, enhancing patient safety and treatment efficacy.

1.2 KEY FEATURES

- **RESPIRATORY SUPPORT:** The ventilator system delivers controlled ventilation to patients with respiratory insufficiency, ensuring adequate oxygenation and ventilation.
- **REAL-TIME MONITORING:** A blood oxygen sensor continuously monitors the patient's SpO2 levels, providing real-time feedback to the ventilator system.
- **ADJUSTABLE PARAMETERS:** The ventilator parameters, including respiratory rate, tidal volume, and oxygen concentration, are adjustable based on patient requirements and clinical guidelines.
- **ALARM SYSTEM:** The system includes an alarm mechanism to alert healthcare providers in case of abnormal SpO2 levels or equipment malfunctions.
- **DATA LOGGING:** Vital signs data, including SpO2 readings and ventilator settings, can be logged for monitoring patient progress and retrospective analysis.

CHAPTER - 2

LITERATURE SURVEY

TOPIC: DIY Ventilator using Arduino with Blood Oxygen Sensing for Covid Pandemic

YEAR OF PUBLICATION: 2022

AUTHOR: Rameshwar Bhaginath Kale, Shashank Kumar Singh

ABSTRACT : This paper portrays outline of different examination done. The human lungs are utilized for breaths. They use push system in every breath motivation and exhalation process happens. The DIY ventilator here we configuration is to help individuals during Covid pandemic. It is exceptionally modest and reasonable. At the point when patients experience the ill effects of lung or breathing issue this can be utilized in a patient basic condition. Stepper Motor component is utilized to push the ambu sack. While breathing heartbeat level identified are low this component can be performed. The LED screen is utilized to show the breathing heartbeat levels. Likewise, in a patient basic condition or breathing issue ringer is fitted in the framework to sound a ready when any irregularities are identified. Aside from this the ventilator should have the option to screen the patient's blood oxygen level and breathed out lung strain to keep away from over/under air tension at the same time.

PROBLEM STATEMENT: Design and implement a low-cost, DIY ventilator system utilizing Arduino microcontroller platform with integrated blood oxygen sensing capability to aid in the management of COVID-19 patients, especially in resource-limited settings or emergency scenarios. Develop a user-friendly interface for clinicians or healthcare providers to monitor and adjust ventilator settings easily. This may include an LCD display or LED indicators for displaying vital parameters and controls for adjusting ventilation parameters.

TOPIC: Automation of Mechanical-Based Blood Oxygen Level Ventilator Using Arduino

YEAR OF PUBLICATION: 2022

AUTHOR: Gnanasekaran.S, Ashwinraj.M, Naveenkumar. S, Parveen Kumar. R, Prasannaa.V

ABSTRACT: This paper describes the development of a compact automated bag valve mask (BVM) compression system which could be used as a temporary emergency ventilator amid acute shortages and supply chain disruptions. The resuscitation system is built on an Arduino controller and a real-time operating system on a mainly parametric component-based structure. The materials are affordable, allowing manufacturers all around the world to create it. The device has a regulated breathing mode with tidal volumes ranging from 100 to 800 mL, breathing rates range from 5 to 40 breaths per minute, and an inspiratory-to-expiratory ratio range from 5 to 40:1. Because the object-oriented algorithmic approach, the system is designed for measurement circuit reliability and scalability through using serial peripheral interface. It has the capacity to link additional hardware. positive end-expiratory pressure, Peak inspiratory pressure , respiratory rate , tidal volume, proximal pressure, and lung pressure were tested on an artificial lung and indicated repeatability and accuracy in BVM-based manual ventilation that exceeded human capabilities. Further work is necessary to further develop and test the system when it can be used in clinical situations outside of emergencies such as COVID-19 pandemic. However, the design is that desired features are comparatively easy to add using protocol and parametric.

PROBLEM STATEMENT: Develop an automated mechanical-based ventilator system using Arduino microcontroller technology to provide precise control of airflow and integration of blood oxygen level monitoring for patients experiencing respiratory distress, particularly in the context of the COVID-19 pandemic.

TOPIC: Smart Ventilator Using Arduino with Blood Oxygen Sensing

YEAR OF PUBLICATION: 2021

AUTHOR: R.Senthilkumar , S.Sakthi , M.Aharish , K.Mahesh , T.Mari Bhagavathi

ABSTRACT: Human lungs use the reverse pressure generated by contraction motion of the diaphragm to suck in air for breathing. A contradictory motion is used by a ventilator to inflate the lungs by pumping type motion. A ventilator mechanism must be able to deliver in the range of 10 – 30 breaths per minute, with the ability to adjust rising increments in sets of Along with this the ventilator must have the ability to adjust the air volume pushed into lungs in each breath. The last but not the least is the setting to adjust the time duration for inhalation to exhalation ratio. Apart from this the ventilator must be able to monitor the patient's blood oxygen level and exhaled lung pressure to avoid over/under air pressure simultaneously. The ventilator we here with design and develop using Arduino encompasses all these requirements to develop a reliable yet affordable ventilator to help in times of pandemic.

PROBLEM STATEMENT: Design and implement a smart ventilator system using Arduino microcontroller technology with integrated blood oxygen sensing capability, providing precise ventilation support for patients, especially those with respiratory distress like COVID-19 cases. Key requirements include advanced ventilation modes, accurate oxygen monitoring, safety features, cost-effectiveness, user-friendly interface, connectivity for remote monitoring, compliance with medical standards, comprehensive documentation, and rigorous testing. The goal is to develop a reliable, affordable, and easy-to-use ventilator solution to aid healthcare professionals in managing respiratory conditions effectively.

TOPIC: Patient Ventilator Using Arduino with Blood Oxygen Sensing

YEAR OF PUBLICATION: 2020

AUTHOR: T.lakshmipriya, A.M.nirubama, A.parameswari, K.sasikala, A.suganya

ABSTRACT: Human lungs use the reverse pressure generated by contraction motion of the diaphragm to suck in air for breathing. A contradictory motion is used by a ventilator to inflate the lungs by pumping type motion. A ventilator mechanism must be able to deliver in the range of 10 – 30 breaths per minute, with the ability to adjust rising increments in sets of 2. Along with this the ventilator must have the ability to adjust the air volume pushed into lungs in each breath. The last but not the least is the setting to adjust the time duration for inhalation to exhalation ratio. Apart from this the ventilator must be able to monitor the patient's blood oxygen level and exhaled lung pressure to avoid over/under air pressure simultaneously. Our machine uses blood oxygen sensor alongside touchy heart Beat sensor to display the important vitals of the patient and show on a webpage the usage of IoT. To alter the time duration for inhalation the choice command given in the IoT application to set. The complete machine is driven by way of arduino controller to gain desired results and to assist patients in COVID pandemic and other emergency situations.

PROBLEM STATEMENT: Develop a patient ventilator system utilizing Arduino microcontroller technology with integrated blood oxygen sensing for effective respiratory support, particularly in cases of respiratory distress like COVID-19. Essential requirements include precise ventilation control, continuous blood oxygen level monitoring, safety mechanisms, cost-effectiveness, user-friendly interface, remote connectivity, compliance with medical standards, comprehensive documentation, and thorough testing. The objective is to create a reliable and accessible ventilator solution to assist healthcare providers in managing respiratory conditions efficiently and enhancing patient care during critical situations.

TOPIC: Ventilator Using Ardino

YEAR OF PUBLICATION: 2019

AUTHOR: Suhasini , Krishna saigal , Avinash G. B , Vivekanand. R , Shubha. S

ABSTRACT: In response to anticipated shortages of ventilators caused by the COVID-19 pandemic, mechanical ventilation is a life-saver in the development of modern ICUs. This project provides an idea of the origin of modern mechanical ventilators. Based on the reviewed literature, a simple, easy-to-use, and easy-to-build design of lowcost portable ventilator is proposed in this project. The proposed ventilator prototype here is assumed to have better working performance than already available around. This ventilator consists of oxygen sensor and a sensitive pressure monitoring system all at a very low cost. The ventilator we here design and develop using Arduino encompasses all the requirements to develop a reliable yet affordable whilst matching all the conditions and marks of an original mechanical ventilator along with its property of portability. This ventilator will help in a situation like COVID-19 when the whole world facing difficulties related to ventilators and thereby can also be used in other critical situations or mass casualties.

PROBLEM STATEMENT: Design and develop a ventilator system utilizing Arduino microcontroller technology to provide life-saving respiratory support for patients, especially in critical medical scenarios such as respiratory failure or during pandemics like COVID-19. Key objectives include precise control of airflow, monitoring of vital parameters, ensuring safety through fail-safe mechanisms, cost-effectiveness, user-friendly operation, compliance with medical standards, and scalability for various healthcare settings. The aim is to create a reliable and accessible ventilator solution to augment healthcare infrastructure and save lives during emergencies.

CHAPTER – 3

AIM AND SCOPE

3.1 AIM OF THE PROJECT

The aim of the project is to design and implement an automated ventilator system using Arduino microcontroller technology with integrated blood oxygen sensing capability.

This ventilator will provide precise and adjustable respiratory support for patients experiencing respiratory distress, with a particular focus on monitoring and maintaining optimal blood oxygen levels. Key objectives include:

- Develop a robust control system using Arduino microcontrollers to regulate ventilation parameters such as airflow rate, tidal volume, and respiratory rate.
- Integrate a blood oxygen sensor into the ventilator system to continuously monitor the patient's oxygen saturation levels.
- Implement safety mechanisms and fail-safe features to ensure the reliable and safe operation of the ventilator.
- Design a user-friendly interface for healthcare professionals to monitor vital parameters and adjust ventilation settings easily.
- Ensure cost-effectiveness by using readily available components and materials.
- Document the design, assembly instructions, and operating procedures comprehensively to facilitate replication and proper use of the ventilator.
- Test the ventilator system rigorously to validate its performance, reliability, and safety under various conditions, including simulated patient scenarios.

- Aim for regulatory compliance with relevant safety standards and guidelines for medical devices.
- Foster collaboration and community engagement to share knowledge and contribute to the collective effort in addressing respiratory distress effectively during critical situations, such as pandemics.

3.2 FUTURE SCOPE

The scope of the automated ventilator using Arduino with a blood oxygen sensor encompasses various aspects related to design, development, implementation, testing, and deployment. Here's an outline of the scope:

- **Hardware Design and Development** Selection of components including Arduino microcontroller, sensors, actuators, and other necessary hardware. Circuit design for controlling airflow, integrating blood oxygen sensor, safety mechanisms, and user interface components. Prototyping and testing of the hardware setup to ensure functionality and compatibility.
- **Software Development** Programming the Arduino microcontroller to control ventilation parameters such as airflow rate, tidal volume, and respiratory rate. Integration of the blood oxygen sensor to continuously monitor oxygen saturation levels. Implementation of safety features and alarms for abnormal conditions. User interface development for displaying vital parameters and enabling user interaction.

- **Testing and Validation** Rigorous testing of the ventilator system under various simulated conditions to validate performance, reliability, and safety. Verification of accurate ventilation control and blood oxygen level monitoring. Evaluation of response to different patient scenarios and adjustments in ventilation settings.
- **Documentation and Regulatory Compliance** Preparation of comprehensive documentation including schematics, source code, assembly instructions, and operating procedures. Ensuring compliance with relevant safety standards and regulatory requirements for medical devices. Obtaining necessary certifications and approvals for deployment in clinical settings.
- **Deployment and User Training:** Deployment of the ventilator system in healthcare facilities, including hospitals, clinics, and emergency response units. Training healthcare professionals on the proper use, maintenance, and troubleshooting of the ventilator. Providing ongoing support and updates as needed.
- **Community Engagement and Collaboration** Sharing the design, findings, and improvements with the maker community, healthcare professionals, and relevant stakeholders. Encouraging collaboration and knowledge-sharing to contribute to the collective effort in addressing respiratory distress, especially during critical situations like pandemics.

CHAPTER – 4

METHODOLOGY & ALGORITHM USED

4.1 RESEARCHED METHODOLOGY

The methodology for developing an automated ventilator using Arduino with a blood oxygen sensor involves a systematic approach encompassing several key stages, Begin by defining the functional requirements of the ventilator system, including ventilation parameters such as tidal volume, respiratory rate, and inspiration-expiration ratio. Specify the integration of a blood oxygen sensor for continuous monitoring of oxygen saturation levels. Identify safety features, user interface requirements, and regulatory compliance standards. Then for Hardware Design and Development Select appropriate hardware components including Arduino microcontrollers, sensors (such as pressure and flow sensors), actuators (such as motors or solenoid valves), and a blood oxygen sensor module. Design the circuitry to control airflow, integrate the blood oxygen sensor, and incorporate safety mechanisms such as pressure relief valves and alarm systems. Prototype the hardware setup to ensure compatibility and functionality.

For Software, Develop the software code for the Arduino microcontroller to regulate ventilation parameters based on predefined algorithms. Implement control loops to adjust airflow rates, tidal volumes, and respiratory rates in response to sensor feedback. Integrate the blood oxygen sensor data into the control system to monitor and maintain optimal oxygen saturation levels. Create a user interface to display vital parameters and enable user interaction for adjusting ventilation settings.

By Testing and Validation Conduct thorough testing of the ventilator system under various simulated conditions, including normal operation, alarm triggering scenarios, and sensor failure conditions. Verify the accuracy of ventilation control and blood oxygen level monitoring. Validate the responsiveness of the ventilator to different patient scenarios and adjustments in ventilation settings. For Documentation Prepare comprehensive

documentation including schematics, source code, assembly instructions, and operating procedures. Document the design rationale, implementation details, and testing results to facilitate replication and proper use of the ventilator system.

For Regulatory Compliance Ensure compliance with relevant safety standards and regulatory requirements for medical devices. Obtain necessary certifications and approvals before deploying the ventilator system in clinical settings.

Deployment and User Training Deploy the ventilator system in healthcare facilities, providing training to healthcare professionals on its proper use, maintenance, and troubleshooting. Establish protocols for monitoring and reporting ventilator usage and performance in clinical settings.

4.2 EXISTING METHODOLOGY

Before diving into building an automated ventilator, it's crucial to have a solid understanding of how mechanical ventilation works, including parameters such as tidal volume, respiratory rate, inspiratory/expiratory ratio, and positive end-expiratory pressure (PEEP). Determine the design of your ventilation system, including the type of ventilator (e.g., volume-cycled or pressure-cycled), the mechanism of delivering breaths, and safety features. Choose appropriate components for your project. This includes an Arduino microcontroller, motor or solenoid valve for controlling airflow, sensors (such as a blood oxygen sensor), display modules, power supply, and mechanical parts for building the ventilator itself. Once the ventilator is built and tested successfully, deploy it in the intended environment. Provide training and instructions for users, and establish protocols for maintenance and troubleshooting. Regularly monitor the performance of the ventilator, gather feedback from users, and make necessary improvements or updates to enhance its functionality and reliability over time.

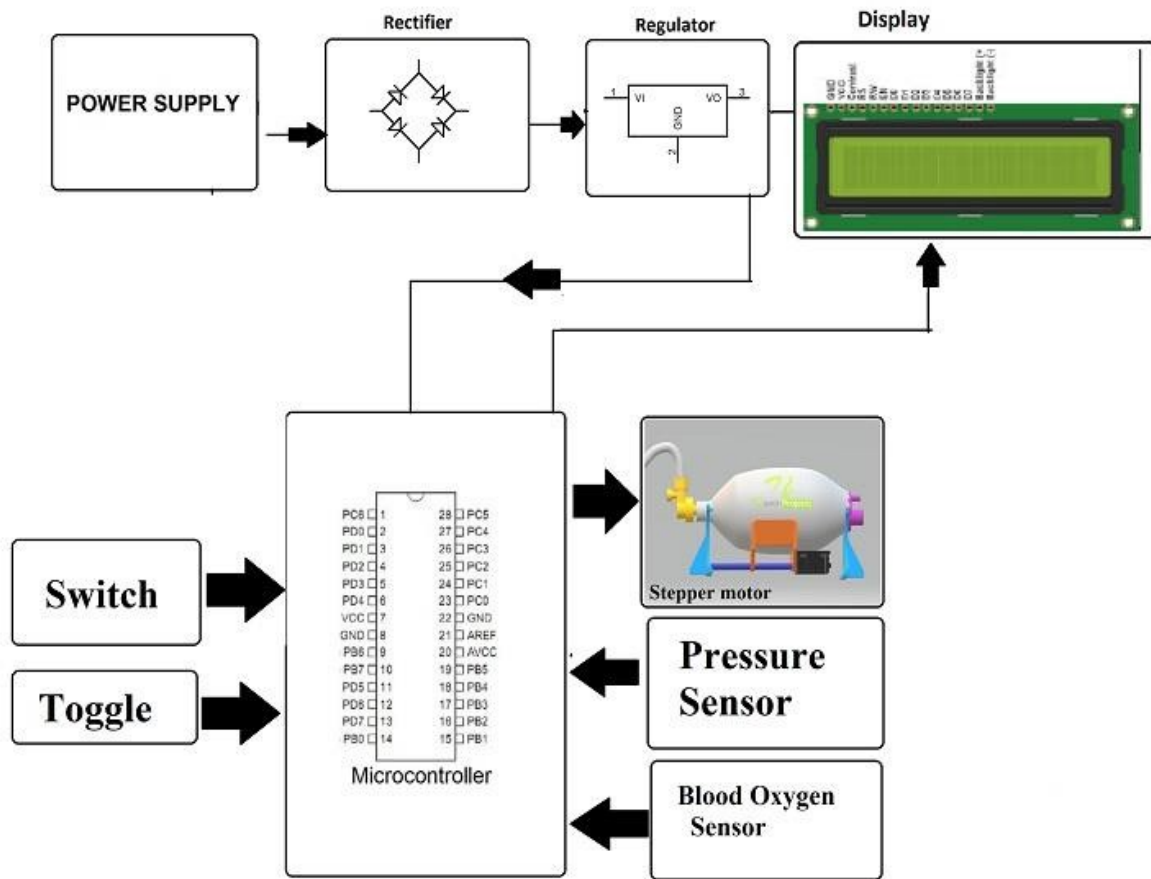


Fig: 4.1: Block Diagram of Existing System

One important device for which demand has ramped up is ventilators for patients who need assistance with their breathing due to the respiratory effects of COVID19. Basically a ventilator is a machine that provides breathable air into and out of the lungs, to deliver breaths to a patient who is physically unable to breathe, or breathing insufficiently. A DIY ventilator may not be efficient as that of a medical grade ventilator but it can act as a good substitute.

ADVANTAGES:

ventilators can be significantly cheaper to build compared to commercially available ventilators. This can be particularly beneficial in resource-constrained settings where healthcare facilities may have limited budgets. ventilators can often be constructed

quickly using readily available materials and basic tools. In emergencies such as natural disasters or pandemics, where there may be a sudden surge in the need for ventilators, solutions can help bridge the gap until more sophisticated equipment becomes available. Ventilators can be manufactured locally, reducing dependence on imported medical equipment. This can be advantageous in regions with limited access to commercial ventilator suppliers or during periods of supply chain disruption. Ventilator designs can be adapted to suit specific needs or circumstances. Engineers and healthcare professionals may modify designs to optimize performance, address local healthcare challenges, or accommodate available resources. Ventilators can provide valuable learning experiences for students, engineers, and healthcare professionals. It can foster innovation, problem-solving skills, and a deeper understanding of medical technology and respiratory care principles.

DISADVANTAGES:

Requires external oxygen supply with is biggest drawback of this methodology.

PROPOSED METHODOLOGY

The proposed methodology for automating a ventilator using Arduino with a blood oxygen sensor involves several key steps, including planning, development, testing, validation, documentation, and deployment. This is a research project and not a medical device. It should not be used for patient care. Prioritize safety features and alarm systems. The design should be modular and expandable for future improvements. User interface complexity should be balanced with ease of use for medical personnel. The system should comply with relevant safety standards. The size of the project is small and compact easily used for transportation purposes. Sensors used in our project are simple and cost efficient. It is an automatic pumping, so there is no necessary requirement for any Man power to access the project. Though, it is a prototype the value of the project is low.

4.4 HARDWARE REQUIREMENT OF PROPOSED METHODOLOGY

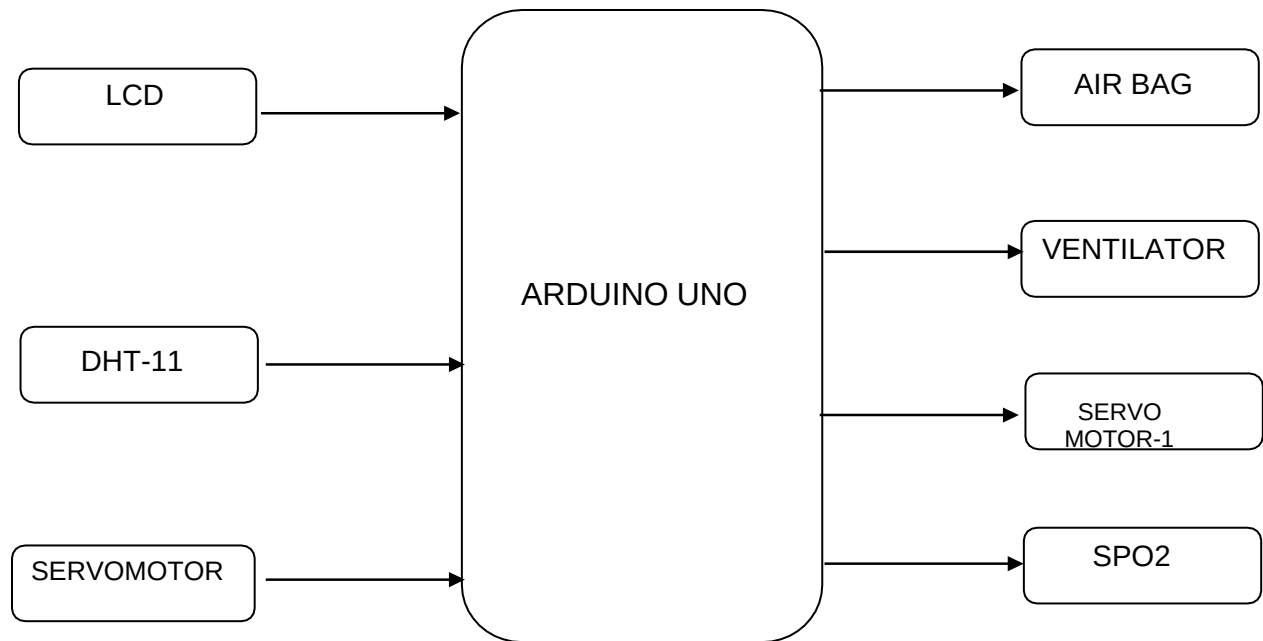


Fig: 4.2: Block Diagram of Proposed System

4.4.1 LCD DISPLAY

LCD (Liquid Crystal Display) is a type of flat panel display which uses liquid crystals in its primary form of operation. LEDs have a large and varying set of use cases for consumers and businesses, as they can be commonly found in smartphones, televisions, computer monitors and instrument panels. LCDs were a big leap in terms of the technology they replaced, which include light-emitting diode (LED) and gas-plasma displays



Fig:4.3: LCD DISPLAY

LCD (Liquid Crystal Display) is a type of flat panel display which uses liquid crystals in its primary form of operation. LEDs have a large and varying set of use cases for consumers and businesses, as they can be commonly found in smartphones, televisions, computer monitors and instrument panels. LCDs were a big leap in terms of the technology they replaced, which include light-emitting diode (LED) and gas-plasma displays. LCDs allowed displays to be much thinner than cathode ray tube (CRT) technology. LCDs consume much less power than LED and gas-display displays because they work on the principle of blocking light rather than emitting it. Where an LED emits light, the liquid crystals in an LCD produces an image using a backlight. As LCDs have replaced older display technologies, LCDs have begun being replaced by new display technologies such as OLEDs.

4.4.2 DHT11 SENSOR

This sensor is used here to monitor the humidity variation of the environment where the crops are cultivated. This is a digital sensor and measures the humidity value in percentage format. DHT11 humidity and temperature sensor is available as a sensor and as a module. The difference between this sensor and module is the pull-up resistor and a power-on LED. DHT11 is a relative humidity sensor. To measure the surrounding air this sensor uses a thermistor and a capacitive humidity sensor.

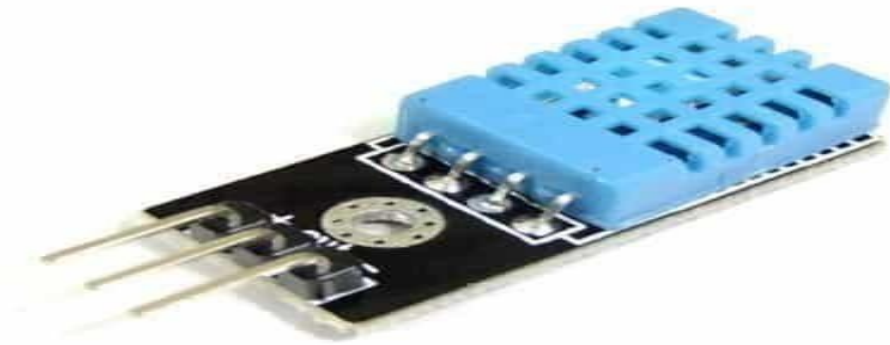


Fig:4.4:DHT11

4.4.3 SERVOMOTOR

A servo motor is defined as an electric motor that allows for precise control of angular or linear position, speed, and torque. It consists of a suitable motor coupled to a sensor for position feedback and a controller that regulates the motor's movement according to a desired setpoint. Servo motors are widely used in industrial applications such as robotics, CNC machinery, and automated manufacturing, where high accuracy, fast response, and smooth motion are required.



Fig: 4.5: Servomotor

A servo motor is a type of motor that operates based on feedback control. It consists of a small DC motor, a gear train, and a control circuitry, all housed within a compact casing.

Servo motors are commonly used in applications requiring precise control of angular position, such as robotics, remote-controlled vehicles, and automation systems. They provide accurate and repeatable movement over a specific range of motion, typically up to 180 degrees. The control circuitry of a servo motor interprets signals from an external source, such as a microcontroller like Arduino, and adjusts the motor's position accordingly. This feedback mechanism enables precise positioning and control, making servo motors suitable for tasks where accuracy and reliability are critical. Additionally, servo motors are available in various sizes and torque ratings, allowing for versatility in different applications.

4.4.4 AIR BAG

An airbag is a safety device installed in vehicles to protect occupants during collisions. It consists of a flexible fabric bag that rapidly inflates with gas upon impact, providing a cushioning barrier between occupants and the vehicle's interior

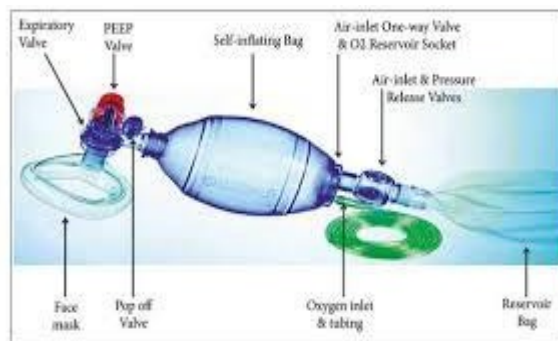


Fig:4.6: Air bag

Typically, airbags deploy from the steering wheel, dashboard, side doors, and other locations. They work in conjunction with seat belts to reduce the risk of injury by distributing the force of impact more evenly across the body. Advanced sensors detect collision forces and trigger the airbag deployment. Proper use of airbags can significantly decrease the severity of injuries sustained in automobile accidents.

4.4.5 VENTILATOR

A ventilator is a medical device designed to assist or replace spontaneous breathing in patients who are unable to breathe adequately on their own. It delivers a controlled flow of air or oxygen into the lungs, helping to maintain proper oxygenation and carbon dioxide removal. Ventilators are crucial in various medical settings, including intensive care units, operating rooms, and emergency departments.



Fig:4.7: Ventilator

A ventilator is a medical device designed to assist or replace spontaneous breathing in patients who are unable to breathe adequately on their own. It delivers a controlled flow of air or oxygen into the lungs, helping to maintain proper oxygenation and carbon dioxide removal. Ventilators are crucial in various medical settings, including intensive care units, operating rooms, and emergency departments. They come in different types, including invasive ventilators that require intubation and non-invasive ventilators that use masks or nasal prongs. Ventilators feature various settings allowing customization to meet individual patient needs, including respiratory rate, tidal volume, and positive end-expiratory pressure (PEEP). Advanced ventilators may incorporate additional features such as pressure support, volume targeting, and modes for specialized respiratory conditions. Effective ventilation can improve patient outcomes in critical situations by supporting respiratory function and providing vital oxygenation.

4.4.6 PULSE OXIMETER

A pulse oximeter is a non-invasive medical device used to measure the oxygen saturation level of arterial blood and monitor heart rate. It operates by emitting light wavelengths through a sensor placed on a thin part of the body, such as the fingertip, toe, or earlobe. The device detects the amount of oxygenated and deoxygenated hemoglobin in the blood based on the light absorption pattern, providing a SpO₂ reading as a percentage. Additionally, it measures the heart rate by analyzing the pulsatile blood flow



Fig:4.8:Pulse oximeter

Pulse oximeters are compact, portable, and easy to use, making them invaluable tools in various healthcare settings, including hospitals, clinics, ambulances, and home care environments. They play a crucial role in assessing respiratory function, monitoring patients with respiratory conditions, evaluating oxygen therapy effectiveness, and detecting hypoxemia or low oxygen levels promptly. Continuous monitoring with pulse oximeters helps ensure patient safety and enables timely interventions to optimize oxygenation and prevent complications. SpO₂, or peripheral capillary oxygen saturation, is a vital physiological parameter that indicates the percentage of oxygenated hemoglobin in the blood. It is measured non-invasively using a pulse oximeter, typically attached to a fingertip, earlobe, or toe. SpO₂ levels provide essential information about the oxygenation

status of a patient's blood. Normal SpO2 levels typically range between 95% and 100%, indicating sufficient oxygenation. However, levels below 90% may indicate hypoxemia, a condition where the blood has inadequate oxygen levels. A pulse oximeter measures oxygen saturation and heart rate non-invasively, essential for monitoring respiratory function and oxygen therapy effectiveness. Monitoring SpO2 is critical in various medical settings, including intensive care units, emergency departments, and during anesthesia. It helps healthcare professionals assess respiratory function, identify hypoxemia, and adjust oxygen therapy or ventilation as necessary. Continuous SpO2 monitoring plays a crucial role in the early detection of respiratory compromise or deterioration, enabling timely interventions to improve patient outcomes and prevent complications related to oxygen deficiency.

4.4.7 HEARTBEAT SENSOR

A heart rate sensor, also known as a heart rate monitor or pulse sensor, is a device that measures the heart rate in real-time. It typically uses photoplethysmography (PPG) technology to detect changes in blood volume beneath the skin caused by cardiac activity. This technology often involves an optical sensor placed on the fingertip, wrist, or chest, which emits light and measures the amount of light absorbed by the blood vessels. By analyzing the variations in light absorption, the sensor can accurately determine the heart rate. Heart rate sensors are widely used in fitness trackers, smartwatches, medical devices, and sports equipment to monitor exercise intensity, track fitness levels, assess cardiovascular health, and provide feedback during workouts. Continuous monitoring of heart rate enables individuals to optimize their exercise routines, track progress, and maintain a healthy lifestyle by staying within their target heart rate zones.



Fig:4.9: Heartbeat sensor

A heart rate sensor is a device that detects and measures the heart's pulsations, typically through photoplethysmography (PPG) technology. It employs an optical sensor to illuminate the skin and detect changes in light absorption caused by blood flow. This data is then processed to calculate the heart rate in beats per minute (BPM). Heart rate sensors are commonly integrated into wearable fitness trackers, smartwatches, and medical monitors to provide real-time heart rate monitoring during physical activity, exercise, or rest.

4.4.8 BUZZER

A buzzer is an electromechanical device designed to produce a buzzing sound, typically used as a signaling or alarm mechanism. It consists of an electrically conductive coil wrapped around a magnetic core, with a metal diaphragm attached to one end. When an electric current is applied to the coil, it generates a magnetic field that moves the diaphragm rapidly, causing it to vibrate and produce a buzzing noise. Buzzer technology has been utilized in various applications across industries, including electronics, automotive, security systems, and household appliances. In electronics, buzzers are often incorporated into circuit designs to provide auditory feedback, such as indicating errors or confirming user inputs.



Fig:4.10: Buzzer

In automobiles, buzzers can serve as warning signals for seatbelt reminders or indicating low fuel levels. Security systems commonly employ buzzers as audible alarms to alert individuals of potential threats or breaches. Additionally, buzzers are frequently found in household devices like timers, doorbells, and kitchen appliances to indicate the completion of a task or signal an event. With their simple yet effective design, buzzers play a crucial role in enhancing safety, communication, and user experience in a wide range of devices and applications.

4.5 SOFTWARE USED

Providing ventilatory support to an ICU patient. Transporting a patient to and from the operating room. Helping patients breathe during certain operations. On running the Arduino takes a glance at the code, the motor executes cycles of dextrorotatory or anticlockwise rotations so you'll make sure that the mechanism is running swimmingly.

4.5.1 VISUAL STUDIO

Microsoft Visual Studio is used as an IDE in this project. Visual Studio Code incorporates

the ease of use of a source code editor with advanced developer features such as IntelliSense code completion and debugging.

It is, first and foremost, an editor who stays out of your way. Visual Studio Code is available for macOS, Linux, and Windows, so you can get started right away. Visual Studio Code's core feature is a lightning-fast source code editor that's ideal for day-to-day use. Syntax highlighting, bracketmatching, auto-indentation, box-selection, snippets, and more are all available in VS Code, which supports hundreds of languages. You can navigate your code with ease thanks to intuitive keyboard shortcuts, simple customization, and community-contributed keyboard shortcut mappings. When it comes to serious coding, you'll also benefit from resources that understand more code than just blocks of text. IntelliSense code completion, rich semantic code comprehension and navigation, and code refactoring are all integrated into Visual Studio Code. When the coding becomes difficult, the tough turn to debugging. We made it happen because debugging is always the one element that developers lack the most in a leaner coding experience. You can walk through source code, inspect variables, view call stacks, and run commands in the console with Visual Studio Code's interactive debugger. VS Code also integrates with build and scripting software to help you complete basic tasks faster. With Git support in VS Code, you can deal with source control without having to leave the editor, including displaying pending changes diffs. Customize every feature to your taste and add as many third-party plugins as you like. Although most scenarios work right out of the box, VS Code develops with you, and we allow you to customize your experience to fit your particular needs. VS Code comes with 20 enhanced built-in support for Node.js development, including JavaScript and TypeScript, and is powered by the same underlying technologies as Visual Studio. Web technologies such as JSX/React, HTML, CSS, SCSS, Less, and JSON are all well-represented in VS Code. Visual Studio Code incorporates the best of internet, native, and language specific technology in its architecture. VS Code uses Electron to combine web technologies like JavaScript and Node.js with native app speed and versatility. VS Code makes use of a newer, faster version of the same industrial-strength HTML-based editor that powers the "Monaco" cloud editor, Internet Explorer's F12 Tools, and a variety of other projects. VS Code also

uses a tools service architecture that allows it to integrate with many of the same technologies that power Visual Studio, such as Roslyn for.NET, TypeScript, the Visual Studio debugging engine, and others. A public extensibility model in Visual Studio Code enables developers to create and use extensions, as well as customize their edit-build-debug experience.

4.5.2 ARDUINO

The Arduino software is used in this project. Beginners will find the Arduino programme simple to use, but experienced users will find it versatile. It's compatible with Mac, Windows, and Linux. It is used by teachers and students to create low-cost science instruments, to demonstrate chemistry and physics concepts, and to begin learning programming and robotics. Designers and developers create immersive prototypes, while musicians and artists use it to create installations and try out new instruments. Makers, for example, use it to create many of the projects on display at the Maker Faire. Arduino is a valuable platform for learning new skills. Anyone - children, hobbyists, artists, programmers - can get started tinkering by following the step-by-step instructions in a kit or sharing ideas with other Arduino members online. For physical computing, there are a range of 21 other microcontrollers and microcontroller platforms to choose from. Similar features can be found in Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handy board, and many other programmes. Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than \$50. Both of these programmes condense the complicated aspects of microcontroller programming into an easy-to-use bundle. While Arduino simplifies the process of working with microcontrollers.

- Cross-platform – The Arduino Program (IDE) is compatible with Windows, Macintosh OSX, and Linux. The majority of microcontroller systems are only compatible with Windows.

- Easy-to-use programming environment - The Arduino Software (IDE) is simple to use for beginners while still being versatile enough for advanced users. It's based on the Processing programming environment, which is convenient for teachers since students learning to programme in that environment would be familiar with how the Arduino IDE works.
- Open source and extensible software - The Arduino software is available as open source tools that professional programmers can extend. People who want to learn more about the language should use C++ libraries, and those who want to learn more about the technical specifics should move from Arduino to the AVR C programming language. Similarly, if you want to, you can use AVR-C code directly in your Arduino programmes.
- Open source and extensible hardware - The Arduino boards' plans are made available under a Creative Commons license, enabling professional circuit designers to build their own version of the module, expanding and enhancing it. The breadboard version of the module can be built by even beginner users to learn how it works and save money.

4.6 MECHANISM

The mechanism behind an automated ventilator system utilizing Arduino with a blood oxygen sensor is a sophisticated interplay of mechanical, electronic, and physiological processes aimed at providing effective respiratory support to patients in need. Here's an overview of the mechanism involved in such a system:

- Ventilation Control Algorithm: At the heart of the system lies a ventilation control algorithm programmed into the Arduino microcontroller. This algorithm governs the

ventilation parameters such as tidal volume, respiratory rate, inspiratory and expiratory flow rates, and the ratio of inspiration to expiration (I:E ratio). The algorithm receives input from the user interface and adjusts these parameters accordingly.

- **Sensor Integration:** The system incorporates a blood oxygen sensor based on pulse oximetry principles. This sensor typically consists of a light emitter and a photodetector, which are placed on the patient's fingertip or earlobe. The Arduino reads the sensor data, which includes the absorbance of light by oxygenated and deoxygenated hemoglobin, and calculates the oxygen saturation (SpO₂) level in the patient's blood.
- **Feedback Loop:** The SpO₂ reading obtained from the blood oxygen sensor serves as feedback to the ventilation control algorithm. If the SpO₂ level falls below a certain threshold, indicating inadequate oxygenation, the algorithm adjusts the ventilation parameters to increase oxygen delivery to the patient. This feedback loop ensures that the ventilator responds dynamically to changes in the patient's respiratory status, optimizing oxygenation while avoiding overventilation.
- **Actuation Mechanism:** The mechanical component of the ventilator system involves an actuation mechanism responsible for generating airflow. This mechanism may utilize a motor-driven piston, a bellows, or a turbine, depending on the design of the ventilator. The Arduino controls the actuation mechanism to produce the desired tidal volume and flow rates during inspiration and expiration.
- **Safety Features:** The system incorporates various safety features to ensure patient well-being. These may include alarms for low SpO₂ levels, disconnection of the breathing circuit, or mechanical failure of the ventilator components. Additionally, manual override controls allow healthcare providers to intervene and adjust ventilation settings manually if needed.

- **User Interface:** A user interface provides healthcare providers with the means to interact with the ventilator system. This interface may consist of a touchscreen display, physical buttons, or a graphical user interface (GUI) on a computer or mobile device. Through the user interface, clinicians can set ventilation parameters, monitor patient status, and receive alerts and alarms.
- **Power Supply and Backup:** The ventilator system requires a reliable power supply to operate effectively. This may involve mains power, battery backup, or a combination of both to ensure uninterrupted ventilation, especially in emergency situations or areas with unreliable electricity supply.

In summary, the mechanism of an automated ventilator system using Arduino with a blood oxygen sensor involves a complex integration of control algorithms, sensor feedback, mechanical actuation, safety features, and user interface components. By combining these elements, the system delivers precise and adaptive respiratory support tailored to the individual needs of the patient, thereby improving clinical outcomes and enhancing patient care.

4.7 WORKING PRINCIPLE

The working principle of an automated ventilator system using Arduino with a blood oxygen sensor is a complex interplay of mechanical, electronic, and physiological processes aimed at providing effective respiratory support to patients in need. This system integrates advanced control algorithms, real-time feedback from a blood oxygen sensor, and safety mechanisms to ensure optimal oxygenation levels while adapting to the patient's respiratory needs. In this detailed explanation, we will delve into each component's functionality and how they work together to deliver ventilatory support.

At the core of the system lies the mechanical ventilation component, responsible for delivering airflow to the patient's lungs. This is achieved through an actuation mechanism,

which could be a motor-driven piston, a bellows, or another suitable mechanism. The actuation mechanism generates positive pressure during inspiration, forcing air into the patient's airways, and allows passive exhalation during expiration. The mechanical ventilation parameters include tidal volume (the volume of air delivered per breath), respiratory rate (the number of breaths per minute), inspiratory and expiratory flow rates, and the ratio of inspiration to expiration (I:E ratio). The Arduino microcontroller serves as the brain of the system, controlling and coordinating the operation of the mechanical ventilation component and other electronic modules. It runs a ventilation control algorithm programmed to regulate the ventilation parameters based on user input and feedback from the blood oxygen sensor. The Arduino processes sensor data, calculates oxygen saturation levels, and adjusts ventilation settings in real-time to maintain optimal oxygenation.

The blood oxygen sensor employs pulse oximetry principles to measure the oxygen saturation (SpO₂) level in the patient's blood. It consists of a light emitter and a photodetector placed on the patient's fingertip or earlobe. The sensor emits light at specific wavelengths into the tissue, and the photodetector measures the amount of light absorbed by oxygenated and deoxygenated hemoglobin. The Arduino reads the sensor data and computes the SpO₂ level, providing crucial feedback on the patient's oxygenation status. The feedback loop is a fundamental aspect of the system, enabling dynamic adjustment of ventilation parameters based on real-time SpO₂ measurements. If the measured SpO₂ level falls below a predetermined threshold, indicating hypoxemia (low oxygen levels), the Arduino adjusts ventilation settings to increase oxygen delivery to the patient. This closed-loop feedback mechanism ensures that the ventilator responds promptly to changes in the patient's respiratory status, optimizing oxygenation while preventing hypoxia.

Safety features are integrated into the system to ensure patient well-being and prevent adverse events. These features include alarms for low oxygen saturation, disconnection of the breathing circuit, or abnormal pressure levels. The Arduino monitors these safety parameters continuously and triggers alarms or safety mechanisms if any anomaly is

detected. Additionally, manual override controls allow healthcare providers to intervene and adjust ventilation settings manually if needed, providing an extra layer of safety and control. A user interface provides healthcare providers with the means to interact with the ventilator system, set ventilation parameters, and monitor patient status. This interface may consist of a touchscreen display, physical buttons, or a graphical user interface (GUI) on a computer or mobile device. Through the user interface, clinicians can input ventilation settings, view real-time SpO₂ readings, and receive alerts and alarms regarding the patient's respiratory status.

The ventilator system requires a reliable power supply to operate effectively. This may involve mains power, battery backup, or a combination of both to ensure uninterrupted ventilation, especially in emergency situations or areas with unreliable electricity supply. Battery backup ensures the system remains operational during power outages or when mobility is required, providing continuous respiratory support to the patient. In conclusion, the working principle of an automated ventilator system using Arduino with a blood oxygen sensor involves the integration of mechanical ventilation, electronic control, real-time feedback from a blood oxygen sensor, safety mechanisms, user interface, and power supply. This sophisticated system delivers precise and adaptive respiratory support tailored to the patient's needs, ensuring optimal oxygenation levels while minimizing the risk of adverse events. By combining advanced technology with physiological monitoring, this ventilator system contributes to improved patient outcomes and enhanced quality of care in various healthcare settings. An automated ventilator system utilizing an Arduino microcontroller and a blood oxygen sensor is a crucial innovation in medical technology, especially in contexts where timely respiratory support is essential, such as in hospitals and emergency medical services. This system combines the precise control capabilities of Arduino with the vital feedback provided by a blood oxygen sensor to efficiently regulate ventilation parameters, ensuring optimal patient care. Let's delve into the working of such a system. The heart of the system is the Arduino microcontroller, which serves as the central processing unit. It executes the programmed instructions to control the various components of the ventilator, including the motor, valves, and display interface. Arduino's versatility allows for customization and adaptation to specific patient requirements and

medical settings.

The blood oxygen sensor plays a pivotal role in monitoring the patient's oxygen saturation levels. It typically employs a photodetector to measure the absorption of light by oxygenated and deoxygenated blood. This data is transmitted to the Arduino, providing real-time feedback on the patient's respiratory status. Based on the feedback from the blood oxygen sensor and predefined parameters, the Arduino regulates the ventilation process. It controls the speed and duration of the ventilator's motor to adjust the airflow rate and volume delivered to the patient's lungs. Additionally, the Arduino may manage the opening and closing of valves to control the inhalation and exhalation phases of ventilation. To ensure patient safety, the ventilator system includes an alarm mechanism triggered by abnormal readings from the blood oxygen sensor or malfunctions in the ventilation process. The Arduino continuously monitors sensor data and system performance, activating alarms when necessary to alert healthcare providers of potential issues.

An intuitive user interface is essential for healthcare professionals to monitor and configure the ventilator settings easily. The Arduino may drive a graphical display or interface with external devices such as smartphones or tablets to provide a user-friendly interface for adjusting ventilation parameters, viewing sensor readings, and responding to alarms. The system incorporates various safety features to prevent adverse events and ensure patient well-being. These may include fail-safe mechanisms to halt ventilation in the event of sensor failure or abnormal readings, as well as backup power sources to maintain operation during power outages or other emergencies. Optionally, the Arduino can log ventilation parameters and sensor data for later analysis or integration with electronic health record systems. It may also support connectivity options such as Wi-Fi or Bluetooth for remote monitoring and control, enabling healthcare providers to supervise multiple ventilator units from a centralized location.

Arduino Uno

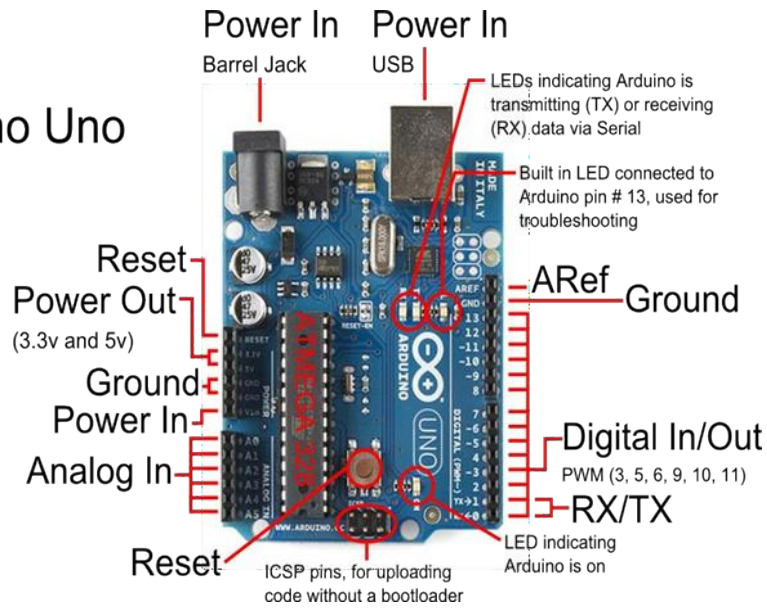


Fig 4.11 Arduino UNO Performance

These boards below use the identical micro-controller, just in a very different package. The Lilypad is meant to be used with conductive thread rather than wire and therefore the Arduino Mini is solely a smaller package without the USB, Barrel Jack and Power.

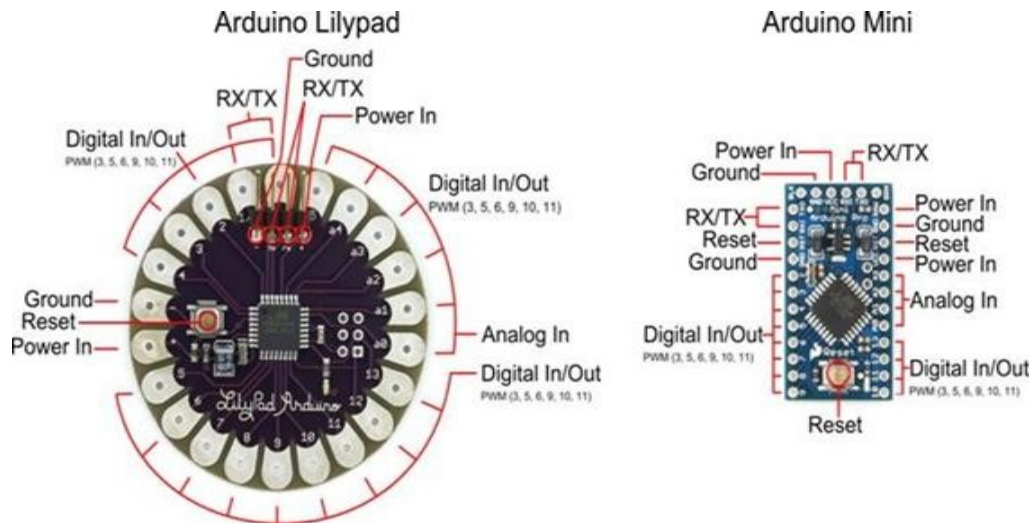


Fig 4.12 Arduino Mini setup

It relies upon what you might want to attempt to do with it truly. There are two unique purposes laid out above for the resistor, we are going to reexamine both. On the off chance that you need to utilize the obstruction as a sensor perusing gadget first you might want to get a handle on the most voltage permitted by the simple data sources you're utilizing to peruse the sign. On an Arduino this is frequently 5V. In this way, as of now we as a whole know the most worth we need for V_{out} . The V_{in} is exclusively the amount of voltage effectively present on the circuit before it arrives at the essential resistor. you should be prepared to locate the most extreme voltage your sensor outputs by relying on the Datasheet, this can be the most measure of voltage your sensor will let through given the voltage in of your circuit. Presently we've precisely one variable left, the value of the subsequent resistor. Explain for R_2 and you'll have all the parts of your obstruction sorted out! We tackle for R_1 's most noteworthy worth in light of the fact that a more modest resistor will essentially give us a more modest sign which can be clear by our simple sources of info. Controlling a simple Reference is actually equivalent to perusing a sensor aside from you need to figure for the Voltage Out worth you need to use as the simple Reference. The entirety of the electrical signs that the Arduino works with are either Analog or Digital. it's critical to get a handle on the distinction between these two sorts of sign and the best approach to control the information these signs speak.

4.8 OPERATION ANALYSIS

The integration of a blood oxygen sensor in a ventilator system is a critical component that provides essential feedback for optimizing respiratory support and ensuring patient safety. Here's an analysis of the role and implications of blood oxygen sensors in ventilators:

Blood oxygen sensors continuously monitor the oxygen saturation levels (SpO_2) in the patient's blood. This real-time data enables healthcare providers to assess the effectiveness of ventilation and make adjustments as necessary to maintain optimal

oxygenation. By providing immediate feedback on the patient's respiratory status, blood oxygen sensors enhance the precision and responsiveness of the ventilator system. Incorporating blood oxygen sensors into ventilators allows for closed-loop control of ventilation parameters. The sensor feedback can be used to automatically adjust ventilation settings, such as tidal volume and respiratory rate, to achieve and maintain target oxygen saturation levels. This closed-loop control mechanism minimizes the need for manual intervention and ensures that ventilation is precisely tailored to the patient's needs, optimizing therapeutic outcomes. Blood oxygen sensors serve as early warning systems for hypoxemia, a condition characterized by low oxygen levels in the blood. By continuously monitoring SpO₂ levels, these sensors can detect decreases in oxygen saturation before clinical symptoms become apparent.

Blood oxygen sensors play a crucial role in ensuring patient safety during mechanical ventilation. Ventilator systems equipped with oxygen sensors can trigger alarms or alerts when oxygen saturation levels fall below or exceed predefined thresholds. These safety features prompt healthcare providers to intervene promptly, preventing potential adverse events such as hypoxia or hyperoxia. Additionally, sensors can detect disconnection or malfunction of the oxygen delivery system, further enhancing patient safety. The data provided by blood oxygen sensors enables healthcare providers to personalize ventilation therapy based on the patient's individual physiology and response to treatment. By monitoring changes in oxygen saturation levels over time, clinicians can adjust ventilator settings to optimize oxygen delivery while minimizing the risk of oxygen toxicity or other complications. This individualized approach to therapy enhances patient comfort and improves overall clinical outcomes. Blood oxygen sensor data can be logged and analyzed to evaluate the effectiveness of ventilation strategies, identify trends or patterns in patient responses, and inform quality improvement initiatives. By tracking SpO₂ levels and ventilator parameters over time, healthcare facilities can identify areas for optimization, refine clinical protocols, and enhance the delivery of respiratory care.

While blood oxygen sensors offer significant benefits in ventilator management, it's essential to ensure regulatory compliance and adherence to safety standards. Medical devices incorporating oxygen sensors must undergo rigorous testing and validation to

demonstrate their accuracy, reliability, and safety for clinical use. Additionally, proper calibration and maintenance of sensors are essential to ensure accurate and consistent measurement of oxygen saturation levels. In conclusion, blood oxygen sensors play a crucial role in optimizing respiratory support, enhancing patient safety, and facilitating individualized therapy in ventilator management. By providing real-time feedback on oxygen saturation levels, these sensors enable closed-loop control of ventilation parameters, early detection of hypoxemia, and prompt intervention to prevent adverse events. However, it's essential to address regulatory considerations and ensure proper calibration and maintenance of sensors to ensure their reliable performance in clinical practice.

4.9 APPENDICES

```
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <LiquidCrystal.h>
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
#define REPORTING_PERIOD_MS 1000
PulseOximeter pox;
uint32_t tsLastReport = 0;
unsigned int heart=0, oxidation=0;
int flag=0,timer=0;
const int interruptPin = 2;
const int solenoid_valve = 7; // Connect your button to pin #13
volatile unsigned int Pulse_State_1 = 0; //,Pulse_State_2=0;
volatile unsigned long int Consumption_Pulse_Count_1 = 0,
Live_Usage_Count_1 = 0;
volatile float Pulse_Count_1 = 0; //,Pulse_Count_2=0;
volatile float Count_For_LPM = 2.84;
volatile float Count_Per_Liter = 170;
volatile float Live_Flow_1 = 0;
```

```

volatile double Consumption_1 = 0, cost=0;
volatile unsigned int Consumption_1_convert_int = 0, cost_convert_int = 0,
con_convert = 0;
volatile unsigned int read_Consumption_1 = 0, read_cost_1 = 0;
volatile int write_Consumption_1_1 = 0, write_Consumption_1_2 = 0,
write_cost_1_1 = 0,
write_cost_1_2 = 0;
volatile int read_Consumption_1_1 = 0, read_Consumption_1_2 = 0,
read_cost_1_1 = 0,
read_cost_1_2 = 0, read_date = 0;
volatile unsigned int Live_Flow_Timer = 0, Consumption_Timer = 0,
Live_Usage_Timer = 0,
Write_Date_Timer = 0;
volatile int Leakage_Timer = 0, No_Flow_Timer = 0, Open_Tap_Timer = 0,
water_flow = 0,
Leakage_detected = 0, Open_Tap_detected = 0, No_Flow_detected = 0;
volatile int Valid_Live_Flow_Flag_1 = 0, Valid_Live_Usage_Flow_Flag_1 = 0;
volatile int lpm_int=0;
volatile unsigned int con_int=0;
char oxi[10];
void onBeatDetected()
{
// Serial.println ("Beat!");
}
void setup ()
{
Serial.begin (9600);
Serial.print("Initializing pulse oximeter.");
// Initialize the PulseOximeter instance
// Failures are generally due to an improper I2C wiring, missing power supply
// or wrong target chip

```

```

If (!pox.begin()) {
  Serial.println("FAILED");
  for(;;);
} else {
  Serial.println("SUCCESS");
}
pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
// Register a callback for the beat detection
pox.setOnBeatDetectedCallback(onBeatDetected);
lcd.begin(16, 2);
delay(1000);
lcd.setCursor(0, 0);
lcd.print(" OXYGEN FLOW ");
lcd.setCursor(0, 1);
lcd.print(" CONTROL SYSTEM ");
delay(1000);
lcd.setCursor(0, 0);
lcd.print("");
lcd.setCursor(0, 1);
lcd.print("");
lcd.setCursor(0, 0);
lcd.print("OXYGEN LEVEL:");
lcd.setCursor(0, 1);
lcd.print("OXYGEN FLOW :");
pinMode(interruptPin, INPUT_PULLUP);
lcd.setCursor(13, 0);
lcd.print("90 ");
delay(100);
pinMode(solenoid_valve, OUTPUT);
delay(100);
attachInterrupt(digitalPinToInterrupt(interruptPin), interrupt, RISING);

```

```

TCCR1A = 0;// set entire TCCR1A register to 0
TCCR1B = 0;// same for TCCR1B
TCNT1 = 0;//initialize counter value to 0
// set compare match register for 1hz increments
OCR1A = 1562;// = (16*10^6) / (1*1024) - 1 (must be <65536)//100ms
timer(15624 for 1
second)
// turn on CTC mode
TCCR1B |= (1 << WGM12);
// Set CS10 and CS12 bits for 1024 prescaler
TCCR1B |= (1 << CS12) | (1 << CS10);
// enable timer compare interrupt
TIMSK1 |= (1 << OCIE1A);
sei();//allow interrupts
}
void loop()
{
// Make sure to call update as fast as possible
pox.update();
if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
// Serial.print("Heart rate:");
// Serial.print(pox.getHeartRate());
Serial.print("bpm / SpO2:");
oxidation = pox.getSpO2();
// Serial.print(oxidation);
// Serial.println("%");
tsLastReport = millis();
}
// itoa(oxidation,oxi,2);
// lcd.setCursor(13, 0);
// lcd.print(oxi);

```

```

timer++;
if(timer >= 10)
{
timer = 0;
if (oxidation >= 80)
{
flag = 0;
}
else if (oxidation < 79)
{
flag = 1;
}
}
if((flag == 0))
{
digitalWrite(solenoid_valve, LOW);
Serial.println("Normal oxidation detected");
lcd.setCursor(13, 1);
lcd.print("OFF ");
delay(2000);
}
else if((flag == 1))
{
digitalWrite (solenoid_valve, HIGH);
Serial.println("Abormaloxidation detected");
Serial.println("Switch on the oxidation supply");
lcd.setCursor(13, 0);
lcd.print("0 ");
lcd.setCursor(13, 1);
lcd.print("ON ");
delay(5000);
}

```

```

}
}
ISR(TIMER1_COMPA_vect)
{
    Live_Flow_Timer++;
    if (Live_Flow_Timer >= 1)
    {
        Live_Flow_Timer = 0;
        Live_Flow_1 = (float)(Pulse_Count_1 / Count_For_LPM);
        Pulse_Count_1 = 0;
        if (Live_Flow_1 > 0)
        {
            Valid_Live_Usage_Flow_Flag_1 = 1;
        }
        // lcd.setCursor(1, 14);
        // lcd.print(Live_Flow_1);
    }
}

void interrupt()
{
    if (Pulse_State_1 == 0)
    35
    {
        Pulse_State_1 = 1;
        Pulse_Count_1 += 1;
        Consumption_Pulse_Count_1++;
    }
    else
        Pulse_State_1 = 0;
}

```

CHAPTER - 5

RESULT AND DISCUSSION

5.1 PERFORMANCE ANALYSIS

Performance analysis of an automated ventilator utilizing Arduino with a blood oxygen sensor involves evaluating various aspects, including accuracy, reliability, responsiveness, safety, and usability. Here's a breakdown of each component of the performance analysis:

The accuracy of the blood oxygen sensor is critical for reliable monitoring of oxygen saturation levels (SpO₂). Performance analysis involves comparing sensor readings with reference standards or established methods to assess its accuracy under different conditions. Calibration procedures should be conducted regularly to ensure accurate measurement of SpO₂ levels over time. Reliability refers to the consistency and dependability of the ventilator system in delivering accurate ventilation and oxygenation support. Performance analysis evaluates the system's ability to operate continuously without failure or deviation from intended performance specifications. Reliability testing may involve prolonged operation under various conditions to identify potential failure points or weak spots in the system.

Responsiveness measures the system's ability to detect changes in patient condition and adjust ventilation parameters promptly. Performance analysis assesses how quickly the ventilator responds to changes in blood oxygen levels detected by the sensor. Short response times are essential for ensuring optimal oxygenation and preventing hypoxemia or hyperoxia. Safety is paramount in ventilator systems, especially those incorporating automated features. Performance analysis evaluates the effectiveness of safety mechanisms, such as alarms, fail-safes, and backup systems, in detecting and preventing adverse events. The system should undergo rigorous testing to ensure compliance with regulatory standards and mitigate potential risks to patient safety.

Usability encompasses the ease of use, user interface design, and overall user experience of the ventilator system. Performance analysis evaluates the clarity, intuitiveness, and efficiency of the interface for adjusting ventilation settings, monitoring sensor readings, and responding to alarms. Usability testing may involve feedback from healthcare providers to identify areas for improvement and optimize the user interface design. Stability refers to the ability of the ventilator system to maintain consistent performance over time and under varying conditions. Performance analysis assesses the system's stability by monitoring key parameters, such as motor speed, airflow rate, and oxygen saturation levels, over prolonged periods of operation. Stability testing helps identify potential drift or degradation in performance and ensures the long-term reliability of the system.

Interoperability refers to the ability of the ventilator system to communicate with other devices or systems, such as electronic health records or monitoring platforms. Performance analysis evaluates the system's compatibility with existing healthcare infrastructure and its ability to exchange data seamlessly with external systems. Integration testing helps ensure interoperability and facilitates seamless integration into clinical workflows. In summary, performance analysis of an automated ventilator using Arduino with a blood oxygen sensor involves comprehensive evaluation of accuracy, reliability, responsiveness, safety, usability, stability, and interoperability. By assessing these key aspects, healthcare providers can ensure that the ventilator system meets the highest standards of performance, safety, and efficacy in clinical practice.

5.2 OUTPUT OCCURANCE

5.2.1 DIGITAL

setup(): A capacity present in each Arduino sketch. Run once before the loop() function. Frequently used to set pinmode to info or yield. The setup() function resembles like:

```
void setup() {  
  
    //code goes here  
  
}
```

Input: A pin mode that admissions data.

Output: A pin mode that sends data.

HIGH: Electrical sign present (5V for Uno). Additionally ON or True in boolean rationale.

LOW: No electrical sign present (0V). Additionally OFF or False in boolean rationale.

DigitalRead: Get a HIGH or LOW perusing from a pin previously proclaimed as an info.

DigitalWrite: Assign a HIGH or LOW an incentive to a pin previously pronounced asanoutput .

AnalogRead: Get an incentive between or including 0 (LOW) and 1023 (HIGH). Thispermits you to get readings from simple sensors or interfaces that have multiple states.

AnalogWrite: Assign an incentive between or including 0 (LOW) and 255 (HIGH). Thisallows you to set output to a PWM esteem rather than simply HIGH or LOW.

PWM: Stands for Pulse-Width Modulation, a strategy for copying a simple sign through anadvanced pin. An incentive between or including 0 and 255.

5.2.2 ANALOG

People see the world in simple. All that we see and hear is a ceaseless transmission of data to our faculties. The temperatures we see are rarely 100% hot or 100% cold, they are continually changing between our scopes of satisfactory temperatures. (What's more, in the event that they are out of our scope of adequate temperatures, at that point what are we doing there?) This ceaseless stream is the thing that characterizes simple information. Advanced data, the correlative idea to Analog, gauges simple information utilizing just ones and zeros. In the realm of Arduino an Analog sign is essentially a sign that can be HIGH(on), LOW (off) or anything in the middle of these two states. This implies an Analog sign has a voltage esteem that can be anything somewhere in the range of 0V and 5V (exceptif you play with the Analog Reference pin). Simple permits you to send yield or get contribution about gadgets that run at rates just as here and there. The Arduino does this by inspecting the voltage signal shipped off these pins and contrasting it with a voltage reference signal (5V). Contingent upon the voltage of the Analog sign when contrasted with the Analog Reference signal the Arduino then doles out a mathematical incentive to the sign somewhere close to 0 (0%) and 1023 (100%). The computerized arrangement of the Arduino would then be able to utilize this number in figurings and portrayals.

To get Analog Input the Arduino utilizes Analog pins # 0 - # 5. These pins are intended for use with segments that yield Analog data and can be utilized for Analog Input. There is no arrangement fundamental, and to peruse them utilize the order. where pinNumber is the Analog In pin to which the Analog part is associated. The analogRead order will restore a number including or somewhere in the range of 0 and 1023. The Arduino additionally has the ability to yield a computerized signal that goes about as an Analog sign, this sign is called Pulse Width Modulation (PWM). Advanced Pins. The Arduino additionally has the ability to yield a computerized signal that goes about as an Analog sign, this sign is called Pulse Width Modulation (PWM). Advanced Pins # 3, # 5, # 6, # 9, # 10 and #11

have PWM capacities. To yield a PWM signal utilize the order:

```
analogWrite(pinNumber, value);
```

where pinNumber is a Digital Pin with PWM abilities and value is a number between 0 (0%) and 255 (100%). On the Arduino UNO PWM pins are meant by a ~ sign. For more data on PWM see the PWM worksheets or S.I.K. circuit 12.

5.3 RESULTS

1. In this fig 5.1 we are using arduino software to detect whether the blood oxygen saturation level is normal or critical and to control the oxygen flow meter valve by using a servo motor to deliver necessary oxygen supply to the patient. The measured parameters are displayed on LCD display. If an individual gets a respiratory issue. He really wants to be taken to clinic quickly while going in rescue vehicle or in the mishap area he wanted of ventilator to breathe so around then our venture is little, helpful and to work which can save a daily existence. Since the cost of our project is reasonable it is simple purchase by a destitute group to rich individuals. Amid the global crisis caused by the corona virus pandemic, hospitals and healthcare facilities are reporting shortages of vital equipments. As makers it's our responsibility to combat the shortage by constructing makeshift-open-source substitute devices. Our country might be in a lock down but our ingenuity isn't One important device for which demand has ramped up is ventilators for patients who need assistance with their breathing due to the respiratory effects of COVID-19. Basically a ventilator is a machine that provides breathable air into and out of the lungs, to deliver breaths to a patient who is physically unable to breathe, or breathing insufficiently. A DIY ventilator may not be efficient as that of a medical grade ventilator but it can act as a good substitute if it has control over the following key parameters Tidal volume: It's the volume of air delivered to the lungs with each breath

by the ventilator - typically 500ml at rest. BPM(Breaths per minute): This is the set rate for delivering breaths. Inspiratory Expiratory ratio (IE Ratio): refers to the ratio of inspiratory time: expiratory time. Flow rate: is the maximum flow at which a set tidal volume breath is delivered by the ventilator. Peep (Positive end expiratory pressure): It is the pressure in the lungs above atmospheric pressure that exists at the end of expiration. My design is based on the automation of the manual BVM (Ambu-bag), which you can find in any medical supply store. It is a hand-held device commonly used to provide positive pressure ventilation.



Fig: 5.1: Hardware Set Up

2. The below fig 5.2 is the captured image of working prototype of blood oxygen saturation measurement and automated control of oxygen supply based on oxygen saturation level. This indicates the oxygen level of an individual person. For example , Inflammation

in the lungs and respiratory tract can reduce the flow of oxygenated blood throughout the body, causing a patient to gasp for air. Normal oxygen saturation levels range between 94%-99%. When SpO_2 levels fall below 93% it is a sign that oxygen therapy is required.

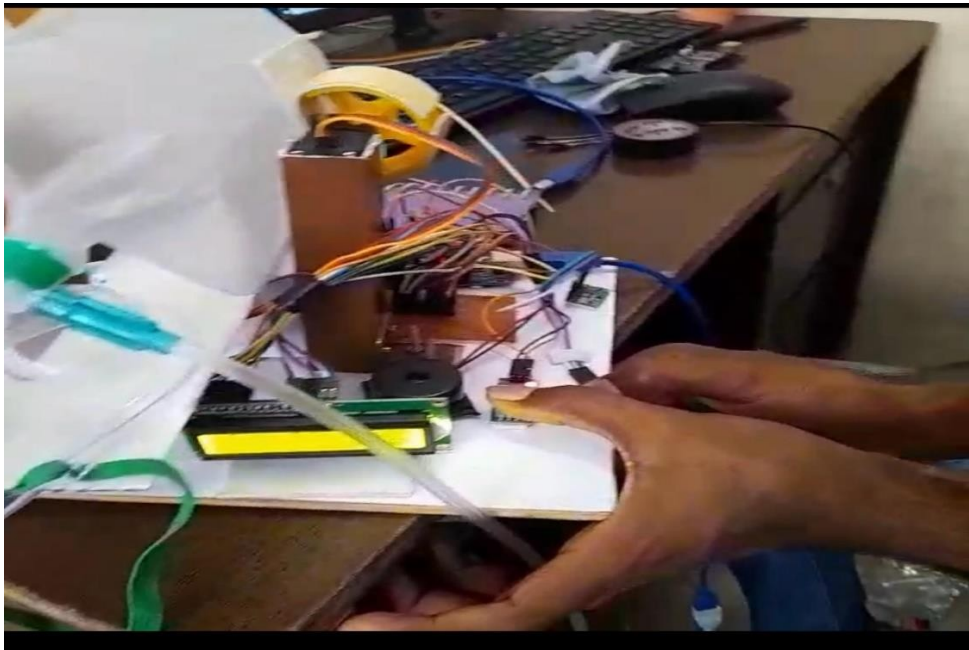


Fig: 5.2: Blood Oxygen Saturation Measurement

3. .The below fig 5.3 represents the measured oxygen saturation level as normal (Oxygen Level: 97%) and status of oxygen supply control (Oxygen Flow: OFF) A normal level of oxygen is usually 95% or higher. Some people with chronic lung disease or sleep apnea can have normal levels around 90%. The “ SpO_2 ” reading on a pulse oximeter shows the percentage of oxygen in someone's blood. If your home SpO_2 reading is lower than 95%, call your health care provider. This indicates for normal human oxygen level such as 97% so therefore oxygen supply for the ventilator is off.



Fig: 5.3: Blood Oxygen Level Predicted As Normal

4. The above fig 5.4 represents the measured oxygen saturation level as abnormal (Oxygen Level: 93%) and status of oxygen supply control (Oxygen Flow: ON). SpO₂ values below 95% (90.1%-94.9%) are considered to be abnormal, and caution should be taken at these values. Patients with an SpO₂ reading of less than 90% are said to be hypoxemic. Patients with an SpO₂ reading of less than 85% would be severely hypoxemic. Hence below fig represents the abnormal oxygen level where person is in danger the oxygen flow is on and O₂ supply is provided.



Fig: 5.4: Blood Oxygen Level Predicted As Abnormal

CHAPTER 6

6.1 SUMMARY

In conclusion, the automation of a ventilator system utilizing Arduino with a blood oxygen sensor represents a significant advancement in medical technology, offering precise and responsive respiratory support for patients in need. By integrating advanced control capabilities with real-time physiological feedback, this innovative technology enhances patient care, improves clinical outcomes, and ensures safety in critical care settings.

The incorporation of a blood oxygen sensor enables continuous monitoring of oxygen saturation levels, allowing for early detection of hypoxemia and timely adjustments to ventilation parameters to optimize oxygenation. Closed-loop control mechanisms ensure that ventilation is precisely tailored to the patient's needs, minimizing the risk of complications and improving therapeutic outcomes.

Furthermore, the inclusion of safety features such as alarms and fail-safes enhances patient safety by alerting healthcare providers to abnormal sensor readings or system malfunctions, prompting prompt intervention to prevent adverse events.

Usability and interoperability considerations ensure that the ventilator system is user-friendly and seamlessly integrates into existing healthcare infrastructure, facilitating efficient clinical workflows and enhancing overall patient care.

In summary, automation ventilators using Arduino with blood oxygen sensors offer a comprehensive solution for respiratory support, combining advanced technology with real-time feedback to deliver optimal ventilation therapy while ensuring patient safety and comfort. Continued research, development, and validation efforts are essential to further refine and optimize these systems for broader clinical adoption and improved patient outcomes.

6.2 STANDARDS

Creating a ventilator using Arduino and integrating a blood oxygen sensor requires careful consideration of both hardware and software standards to ensure safety, accuracy, and reliability. Follow safety standards for medical devices such as ISO 13485, which specifies requirements for quality management systems for medical devices to relevant electrical safety standards such as IEC 60601-1 for medical electrical equipment. Ensure proper insulation, grounding, and electrical isolation to prevent electrical hazards. Performance Standards is Validate the accuracy and reliability of the blood oxygen sensor. Common standards for pulse oximeters include ISO 80601-2-61. Ensure the ventilator meets performance requirements for tidal volume, respiratory rate, and inspiratory/expiratory ratios. Regulatory Compliance Depending on your country, you may need to comply with regulatory bodies such as the FDA (Food and Drug Administration) in the United States marking in Europe Compliance with relevant standards such as IEC 62304 for software development in medical devices is crucial. User Interface Standards Design an intuitive user interface that complies with standards for medical device user interfaces, including ISO 9241 for ergonomic design. Ensure clear visual and audible alarms for critical events or malfunctions. Documentation and Traceability is Maintain thorough documentation throughout the development process, including design specifications, risk management documentation, and testing/validation reports. Implement a traceability system to track components, software versions.

6.3 CONSTRAINS

The size of the project is small and compact easily used for transportation purposes. Sensors used in our project are simple and cost efficient. It is an automatic pumping, so there is no necessary requirement for any Man power to access the project. Though, it is a prototype the value of the project is low. the improvement of low-cost, opensource automatic ventilators. This text also shows the numerical method for monitoring patients' pulmonary conditions. With the assistance of a pressure sensor, we will classify whether the patients are healthy or unhealthy lungs.

This work is a reasonable strategy potential for crisis. It is an opensource ventilator configuration created utilizing disseminated fabricating. This paper is an itemized clarification of delivering minimal expense, opensource mechanical ventilators for patients. This is at the beginning phases of plan required further turns. Sure this work will acquire more noteworthy consideration. There is a great deal of future work to be moved up to make it clinical grade equipment. It is a major hotspot for both the ongoing pandemic circumstance and crisis purposes and in any event, for regular use in low asset settings.

6.4 TRADE -OFFS

Processing Power: Arduino boards typically have limited processing power compared to dedicated medical devices. This limitation may affect the complexity of algorithms that can be implemented for controlling the ventilator and processing data from the oxygen sensor

Real-Time Operation: Ventilators require precise timing and real-time operation to deliver breaths at the correct intervals and volumes. Arduino's multitasking capabilities may be limited, and ensuring real-time operation could be challenging.

Sensor Accuracy and Reliability: The accuracy and reliability of the blood oxygen sensor are crucial for patient safety. Integrating a sensor with sufficient accuracy and reliability while maintaining compatibility with Arduino may be difficult.

Safety: Safety is paramount in medical device design. Ensuring that the ventilator operates safely under all conditions, with fail-safe mechanisms in place, requires careful attention to detail and testing.

Cost Constraints: While Arduino boards and components are relatively low-cost, the overall cost of developing a medical-grade ventilator, including sensors, actuators, and regulatory compliance, can add up. Balancing cost constraints while maintaining quality and safety is a significant challenge.

REFERENCE

- [1] Balamurugan, C. R., Kasthuri, A., Malathi, E., Dharanidharan, S., Hariharan, D., Kishore, B. V., & Venkadesh, T. (2021). Design of Ventilator Using Arduino for CovidPandemic.
- [2] Acho, L., Vargas, A. N., & PujolVázquez, G. (2020, September). LowCost, Open-SourceMechanical Ventilator with Pulmonary Monitoring for COVID-19 Patients. In Actuators (Vol.9, No. 3, p. 84). Multidisciplinary Digital Publishing Institute
- [3] Petsiuk, A., Tanikella, N. G., Dertinger, S., Pringle, A., Oberloier, S., & Pearce, J. M. (2020). Partially RepRapable automated open source bag valve mask-based ventilator.HardwareX, 8, e00131.
- [4] World Health Organization: Critical preparedness, readiness and response actions for COVID-19: interim guidance, 7 March 2020. World Health Organization.2020.
- [5] The Lancet: COVID-19 too little, too late Lancet. 2020;395(10226):755. 10.1016/S0140-6736(20)30522-5.
- [6] Fisher D, Heymann D: Q&A: The novel coronavirus out causing COVID-19. BMCMed.2020;18(1):57.101186/s12916-020-01533.
- [7] Ramesy L: Hospitals could be overwhelmed with patients and run out of beds and ventilators as the coronavirus pushes the US healthcare system to its limits. BusinessInsider. 2020.
- [8] Zhang X, Melzer MI, WortleyPM: FluSurge—a tool estimate demand for hospital services durning the next pandemic influenza. Med Decis Making 2006;26(6):617-623.
- [9] Miller J: Germany Italy rush to buy life-saving as manufactures warn ofshortages.Reuters.2020.

RE-2022-228076-plag-report

ORIGINALITY REPORT

16%	10%	7%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.elprocus.com Internet Source	2%
2	circuitdigest.com Internet Source	1%
3	kanchiuniv.ac.in Internet Source	<1%
4	www.dnatechindia.com Internet Source	<1%
5	docplayer.net Internet Source	<1%
6	www.kapcsolasok.hu Internet Source	<1%
7	www.ijrat.org Internet Source	<1%
8	electronicwings.com Internet Source	1%
9	f1000research.com Internet Source	2%