TABLE OF CONTENTS

ABSTRACT

This project proposes an innovative solution to address the challenge of maintaining restroom cleanliness through the development of an IoT-based automatic restroom cleaning robot equipped with a gas sensor. The robot will be capable of autonomously navigating through restroom spaces, detecting and removing dirt and debris, and identifying and neutralizing unpleasant odours using the gas sensor. Through real-time data monitoring and analysis, facility managers can remotely supervise the cleaning process, receive alerts for maintenance needs, and track cleanliness metrics, thereby enhancing efficiency, hygiene, and overall user satisfaction. The development of smart homes and environments has led to an increased demand for autonomous cleaning robots. The robot will be programmed to navigate autonomously within a designated environment, utilizing sensors to detect obstacles and clean targeted areas. The integrated gas sensor will enable the robot to identify and respond to the presence of specific gases, potentially including harmful pollutants or unpleasant odours. This real-time data can be transmitted via IoT protocols to a central hub or smartphone application, allowing users to monitor air quality and initiate targeted cleaning tasks. The proposed system has the potential to revolutionize home cleaning by offering an intelligent and automated solution that promotes cleanliness and a healthier indoor environment.

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

LED	Light emitting Diode
LCD	Liquefied Crystal Display
GPS	Global positioning System
NIO	North Indian Ocean
RMS	Root Mean Square
Vi /Vo	Input Voltage / Output Voltage
PTAT	Proportional to Absolute Temperature
PWM	Pulse Width Modulation
SDA	Serial Data Access
SCL	Serial Clock Line
SPI	Serial Peripheral Interface
I2C	Inter Integrated Circuit
TX/RX	Transmitter / Receiver
R/W	Read / Write
MIMO	Multiple Input Multiple Output

CHAPTER 1

INTRODUCTION

1.1 CONTEXTUAL

For years, cleaning the toilet has remained a chore dreaded by many. But what if there was a little robot that could handle this task efficiently and autonomously. Introducing the Autonomous Toilet Cleaning Robot (ATCR), a technological marvel poised to transform bathroom sanitation. The ATCR is a self-contained robotic unit designed to fit comfortably within most standard toilets. Its sleek and discreet design allows for easy integration into existing bathrooms without sacrificing aesthetics.

For decades, robotic vacuum cleaners have revolutionized floor care, transforming tedious chores into automated routines. However, advancements in sensor technology are poised to usher in a new era of intelligent cleaning. This introduction explores the integration of gas sensors into cleaning robots, highlighting the potential for a more comprehensive, responsive, and user-friendly cleaning experience.

Traditional robotic cleaners primarily rely on obstacle detection sensors and dirt sensors to navigate and clean effectively. While these sensors are crucial for basic functionality, they cannot detect and address airborne contaminants and pollutants. This introduces a significant limitation, as dust particles and allergens can remain suspended in the air even after a cleaning cycle. Gas sensors bridge this gap by offering a new dimension of cleaning intelligence. Packed with cutting-edge technology, the ATCR boasts the following features:

- Advanced Sensor Suite: Equipped with various sensors, the ATCR can map its environment, detect the toilet bowl's presence and position, and navigate around obstacles. Additionally, occupancy sensors ensure the robot doesn't activate while the toilet is in use.
- Powerful Cleaning Mechanism: The robot utilizes a combination of high-pressure water jets, abrasive brushes, and disinfecting solutions to remove dirt, grime, and

bacteria effectively. Different cleaning modes can be programmed for varying levels of sanitation.

- Self-Cleaning and Disposal System: The ATCR features a built-in cleaning mechanism for its brushes and internal components. Collected waste can be stored internally in a disposable cartridge or directly flushed through the toilet, depending on the model.
- Long-lasting Battery: The robot operates on a rechargeable battery that provides enough power for multiple cleaning cycles before requiring a recharge.
- Smart Connectivity: High-end models may offer Wi-Fi or Bluetooth connectivity, allowing for scheduling cleaning cycles, monitoring cleaning status, and receiving alerts through a smartphone app.

These sensors can detect a wide range of gaseous compounds, including volatile organic compounds (VOCs), odors, and even harmful gases like carbon monoxide. By incorporating gas sensor technology, cleaning robots can evolve from simple floor cleaners into comprehensive air purifiers, creating a cleaner and healthier living environment.

These robots are compact and perform the assigned cleaning task without human intervention. For area coverage during cleaning, these robots follow the motion planning algorithms, including backtracking spiral motion, spiral motion, and boustrophedon motion (back and forth) and simple zig-zag motion patterns. These existing robots are of standard configuration (either circular or D shaped), and they fail to reach the corners, concave spaces, convex regions in the cleaning environment. For effective cleaning, the user has to manually rearrange the furniture setting to clear the navigation path of the robot. Even though a proper path planning and motion capabilities are added to a fixed configuration robots, these robots may need more time and energy to perform the complex task efficiently, compared to the reconfigurable counterparts. Therefore, the cleaning Autonomous Self-Reconfigurable Floor Cleaning Robot performance of the robots can be enhanced by developing a reconfigurable robot that adapt various configurations to reach inaccessible spaces in the cleaning workspace.

1.2 OVERVIEW OF PROJECT

Imagine a world where public restrooms are hygienically maintained without the need for constant manual cleaning. This vision can be realized with an autonomous toilet cleaning robot utilizing gas sensor technology and the power of IoT (Internet of Things). This innovative robot would be equipped with a gas sensor capable of detecting the presence and level of ammonia, a common indicator of restroom usage. Integrated with

IoT, the robot would be able to communicate real-time cleaning needs to a central hub. This allows for efficient cleaning schedules, ensuring restrooms remain fresh and sanitary throughout the day.

This autonomous system offers numerous benefits. Firstly, it reduces the burden on cleaning staff, freeing them for other tasks. Secondly, by automating cleaning based on actual usage, the robot ensures a consistently high level of hygiene, minimizing the risk of germ and odor transmission. Additionally, the IoT integration allows for remote monitoring and maintenance of the robot, optimizing its operation and ensuring longterm functionality. This fusion of robotics, gas sensor technology, and IoT paves the way for a cleaner and more pleasant public restroom experience.

The autonomous toilet cleaning robot will be a self-contained mobile unit equipped with various components:

Navigation System: Utilizing technologies like LiDAR or cameras, the robot will navigate the restroom environment autonomously, avoiding obstacles and reaching designated cleaning zones.

Cleaning Mechanism: The robot will be equipped with a cleaning apparatus that dispenses disinfectants and scrubs surfaces within the toilet bowl and surrounding areas.

Gas Sensor: A gas sensor will detect the presence and level of unpleasant odors, allowing the robot to prioritize cleaning in areas requiring immediate attention.

IoT Connectivity: The robot will be integrated with an IoT platform for remote monitoring, data collection, and control.

The system functionality was given as the robot will use its navigation system to move around the restroom, identifying and reaching designated toilet stalls. The gas sensor will continuously monitor for the presence of unpleasant odors. Based on sensor data and pre-programmed cleaning routines, the robot will prioritize cleaning areas with higher odor levels.

Upon reaching a stall, the robot will activate its cleaning mechanism, dispensing disinfectants and performing mechanical cleaning within the toilet bowl and surrounding areas. The robot will transmit data on its cleaning activity, gas sensor readings, and battery levels to the IoT platform. Facility managers can remotely monitor the robot's status, track cleaning progress, and even issue cleaning commands through the IoT platform.

1.3 CHALLENGES AND BENEFITS IN TOILET CLEANING PROCESS

Cleaning has been an essential and important part of our lives, which has evolved and improved over time. Today’s busy lifestyle has led to an increased demand for automated floor cleaning robots with fully automatic functions. In developing countries, autonomous floor cleaning robots are majorly used in smart homes, residential and office spaces to clean the floor. As per the global market study, there is a huge demand for the application these robots in domestics setting with fully automated features and the least human assistance and these robots still constitute a minute market share of the global vacuum cleaner market.

1.3.1 Benefits

- Improved Hygiene: An autonomous toilet cleaning robot can ensure consistent and thorough cleaning, reducing the risk of germs and bacteria spreading. This can be particularly beneficial in public restrooms with high traffic.
- Reduced Labor Costs: Automating toilet cleaning can free up staff for other tasks, leading to increased efficiency and potentially lower labor costs.
- Enhanced User Experience: Clean and odor-free restrooms create a more pleasant experience for users. Real-time data from the robot can also inform cleaning staff when a stall truly needs attention.

- Data-Driven Maintenance: IoT integration allows the robot to collect data on cleaning cycles, usage patterns, and potential issues. This data can be used to optimize cleaning schedules, predict maintenance needs, and identify areas for improvement.
- Improved Accessibility: The robot can clean toilets in areas that might be difficult or unsafe for humans to access, such as those in hospitals or assisted living facilities.

1.3.2 Challenges

- Navigation and Maneuvering: Toilets have a complex geometry with tight spaces and obstacles. The robot needs to be able to navigate effectively within the stall, avoiding walls, fixtures, and potential hazards like dropped objects.
- Sensor Accuracy and Specificity: Gas sensors used for odor detection can be sensitive to various compounds, not just those associated with waste. Effectively differentiating between target odors and common bathroom cleaning products or air fresheners is crucial.
- Cleaning Mechanism Design: The robot's cleaning mechanism needs to be effective in removing waste and grime from various toilet surfaces, including the bowl, seat, and surrounding area.
- Water Management and Disposal: The robot's design needs to account for the efficient use and disposal of cleaning water. This includes minimizing water usage while ensuring a thorough clean and handling the potential for overflow or leaks.
- User Acceptance and Social Stigma: There may be initial resistance to the idea of an automated toilet cleaning robot due to concerns about hygiene, privacy, or potential malfunction.
- Cost and ROI: Developing, deploying, and maintaining a fleet of autonomous toilet cleaning robots can be expensive. The cost-effectiveness needs to be carefully evaluated to justify the investment.

Additional Considerations:

- Safety Features: The robot should be equipped with safety features to prevent collisions with users or objects, and should automatically shut down in case of malfunctions.
- Battery Life and Charging: The robot needs to have sufficient battery life to complete cleaning cycles efficiently. A reliable and automated charging system is also necessary.
- Data Security and Privacy: The data collected by the robot's IoT sensors needs to be secure to protect user privacy. Measures must be taken to prevent unauthorized access or data breaches.
- Waste Disposal: The robot's design should address the disposal of collected waste materials hygienically and efficiently.

By addressing these challenges and considerations, autonomous toilet cleaning robots with gas sensors and IoT have the potential to revolutionize bathroom sanitation, promoting improved hygiene, increased efficiency, and a more pleasant user experience.

CHAPTER 2

LITERATURE SURVEY

2.1 DIRTNET: VISUAL DIRT DETECTION FOR AUTONOMOUS CLEANING ROBOTS :

Richard Bormann, Xinjie Wang, Jiawen Xu, Joel Schmidt

Visual dirt detection is becoming an important capability of modern professional cleaning robots both for optimizing their wet cleaning results and for facilitating demand- oriented daily vacuum cleaning. This paper presents a robust, fast, and reliable dirt and office item detection system for these tasks based on an adapted YOLOv3 framework. Its superiority over state-of-the-art dirt detection systems is demonstrated in several experiments. The paper furthermore features a dataset generator for creating any number of realistic training images from a small set of real scene, dirt, and object examples.

2.2 AUTONOMOUS SELF-RECONFIGURABLE FLOOR CLEANING ROBOT :

Rizuwana Parween , (Member, Ieee), Manuel Vega Heredia, Madan Mohan Rayguru , Raihan Enjikalayil Abdulkader, And Mohan Rajesh Elara

The cleaning robots are going through some significant development in recent years, driven by the greater market penetration and the demand for better cleaning performance. However, most of the robots have problems covering the total cleaning are a given geometric limitation soft he platforms in relation to the cleaning stage, given furniture and architecture. In this paper, we describe the h-h-Trihex platform, which is a self-reconfigurable adaptive robot that attains three different configurations. The change in configuration gives rise to variation in the kinematic model, which in turn changes the behavior of robot locomotion. The change in configuration and the robot position is evaluated using the measurements from the on-board sensors like the encoders, LIDAR, and inertial measuring unit(IMU).These data are then utilized to develop a new cascade control strategy for motion control of h-Trihex. The control cascade comprises of a robust

backstepping controller in the outer loop for tackling the varying kinematic and a conventional PID in the inner loop for speed control. This cascade is appended by a PID controller, for correcting the wheel orientation after each configuration change. The proposed design assures asymptotic path tracking and satisfactory closed-loop performance, which are validated through numerical simulations and experiments.

2.3 OPERATION MODE DECISION OF INDOOR CLEANING ROBOT BASED ON CAUSAL REASONING AND ATTRIBUTE LEARNING Yapeng Li , Dongbo Zhang, Feng Yin, And Ying Zhang

At present, the cleaning robot son market generally have the defects of simple operation mode and weak intelligence. In order to improve the intelligent degree and operation ability of cleaning robots, this paper proposes a decision method for cleaning robot's operation mode. Firstly, use the hierarchical expression ability of deep network to obtain the attributes of garbage such as state, shape, distribution, size and soon. Then the causal relationship between the attributes and the operation modes can be built by using joint learning of association attributes with depth network model and causal inference. Based on this, a fuzzy inference network for operation mode decision is designed. With the help of causal analysis, the structure of the decision model is greatly simplified. Compared with conventional fuzzy neural networks, the total parameters of the model are reduced by 2 / 3. The method proposed in this paper imitates the way that human dispose of different types of garbage and has good interpretability. The experimental results verify the effectiveness of the proposed method.

2.4 AUTOMATION OF TRAIN CAB FRONT CLEANING WITH A ROBOT MANIPULATOR :

Joao Moura, William Mccoll, Gerard Taykaldiranian, Tetsuo Tomiyama, and Mustafa Suphi Erden

In this letter we present a control and trajectory tracking approach for wiping the train cab front panels, using a velocity controlled robotic manipulator and a force/torque sensor attached to its end effector, without using any surface model or vision-based surface detection. The control strategy consists in a simultaneous position and force

controller, adapted from the operational space formulation, that aligns the cleaning tool with the surface normal, maintaining a set-point normal force, while simultaneously moving along the surface. The trajectory tracking strategy consists in specifying and tracking at wo dimensional path that, when projected onto the train surface, corresponds to the de- sired pattern of motion. We first validated our approach using the Baxter robot to wipe a highly curved surface with both a spiral and a raster scan motion patterns.

Finally, we implemented the same approach in a scaled robot prototype, specifically designed byourselvestowipea1/8scale diversion of a train cab front, Using a raster scan pattern.

2.5 A SURVEY ON TECHNIQUES AND APPLICATIONS OF WINDOW-CLEANING ROBOTS :

Zhenjing Li, Qingsong Xu, (Senior Member, Ieee), And Lap Mou Tame

Maintaining a clean living and working environment is an important means to ensure people's quality of life. As the height of the building increases, the traditional method of cleaning windows by human workers not only takes a long time, but also brings the risk of falling off. It is necessary to replace human workers with robots for carrying out such heavy duty and dangerous work. Various window-cleaning robots have been applied in cleaning work for certain actual scenarios. In this paper, the stateof-the-art survey of recent developments on window-cleaning robots is presented. The applications of window-cleaning robots in two main domains of domestic use and highrise building environments are presented and the corresponding technical requirements are summarized. Afterward, the main techniques of window-cleaning robot development including locomotion mechanisms, adhesion mechanisms, cleaning mechanisms, and sensor and controller units are reviewed in detail. The survey provides a reference for the readers to design and develop a window-cleaning robot for specific application.

CHAPTER 3

PROPOSED SYSTEM

3.1 PROPOSED METHODOLOGY

An autonomous toilet cleaning robot can utilize gas sensors and Internet of Things (IoT) technology to enhance hygiene in public and private restrooms. The robot would be equipped with a gas sensor to detect the presence of methane or other gases indicative of waste buildup. This sensor data would be transmitted via IoT to a central hub for realtime monitoring and control. Upon detecting high gas levels, the robot would be programmed to navigate to the designated toilet stall and activate its cleaning mechanism. Once positioned, the robot would deploy its cleaning mechanism, equipped with disinfectant solutions, to automatically clean the toilet bowl and surrounding area. The robot would then communicate with the central hub to confirm completion and await further instructions. The entire process, including cleaning completion and status updates, could be monitored and controlled remotely through an IoT platform accessible via smartphone or tablet. The IoT integration allows for remote monitoring of cleaning schedules, sensor data analysis, and robot performance optimization. This system would not only promote hygiene but also free up time and effort typically spent on manual cleaning.

This proposed self-cleaning system offers several advantages. It reduces user interaction by automating dustbin emptying and brush cleaning. Additionally, it minimizes cleaning performance degradation caused by dirty brushes and promotes hygienic cleaning practices through mop sanitization. This system can be implemented in various ways, with features tailored to specific robot models and user needs. By incorporating self-cleaning capabilities, robotic vacuum cleaners can offer a more automated and user-friendly cleaning experience.

3.2 BLOCK DIAGRAM

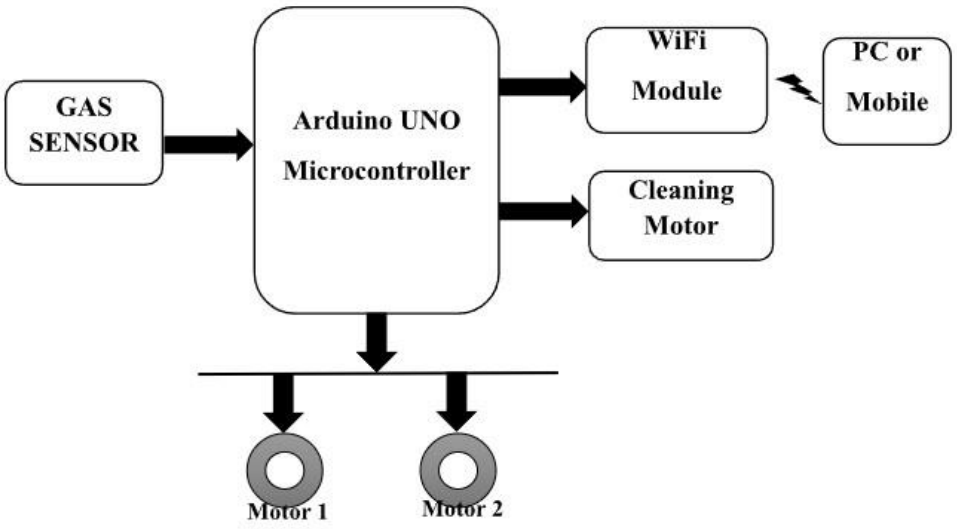

Fig 3.1 Proposed Block Diagram

The above Fig 3.1 (proposed block diagram) is the schematic diagram of a system that includes an Arduino UNO Microcontroller, Gas sensor, WiFi module, Personal Computer or Smart phone and Cleaning motor.

3.3 HARDWARE REQUIREMENTS

For the proposed method the hardware required for the autonomous toilet cleaning robot has,

- Arduino UNO Microcontroller
- Gas Sensor
- Robo car chassis
- ESP 8266 Module

3.4 ARDUINO UNO CONTROLLER

A micro-controller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/ output peripherals The important part for us is that a micro-controller contains the processor (which all computers have) and

memory, and some input/output pins that can be control. (often called GPIO - General Purpose Input Output Pins). "Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino

Software (IDE) were the reference versions of Arduino, now evolved to newer releases. We will be using the Arduino Uno board. This combines a micro-controller along with all of the extras to make it easy to build and debug in projects. The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

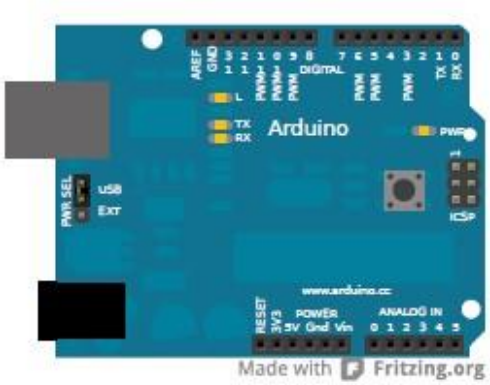

Fig 3.2 Arduino UNO Pin Diagram

The above Fig 3.2 describes the pin details of Arduino UNO board and the below Fig 3.3 describes the Arduino UNO in microcontroller board.

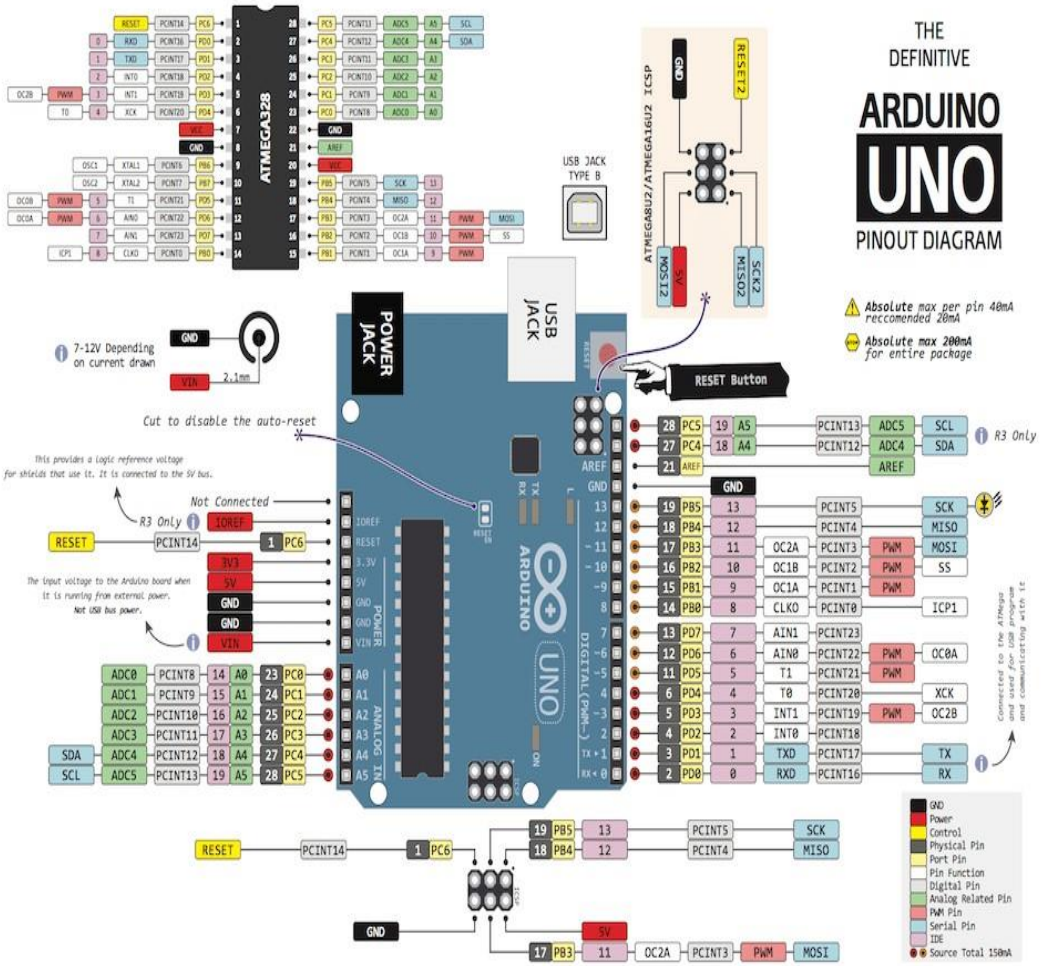

Fig 3.3 Arduino UNO Pin out

This is a relatively easy way to make circuits quickly. Breadboards are made for doing quick experiments. They are not known for keeping circuits together for a long time.

The columns on the edges do not have letters or numbers. The columns on the edges are connected from top to bottom inside of the breadboard to make it easy to supply power and ground. Inside of the breadboard, the holes in each row are connected up to the break in the middle of the board.

For Example: a1, b1, c1, d1, e1 all have a wire inside of the breadboard to connect them. Then f1, g1, h1, i1, and j1 are all connected. But a1 is not connected to f1. This may sound confusing now, but it will quickly come to make sense as we wire up circuits.

Table 3.1 AVR family Microcontroller

Microcontroller	**ATmega328P – 8bit AVR family microcontroller**
Operating Voltage	5V
Recommended Input Voltage	7-12V
Input voltage Limits	6-20V
Analog Input Pins	6 (A0 – A5)
Digital I/O Pins	14 (Out of which 6 provide PWM output)
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
Flash Memory	32 KB (0.5 KB is used for Bootloader)
SRAM	2 KB
EEPROM	1 KB
Frequency (Clock Speed)	16 MHz

3.5 ARDUINO UNO TECHNICAL SPECIFICATION

The 14 digital input/output pins can be used as input or output pins by using pinMode(), digital Read() and digital Write() functions in arduino programming. Each pin

operate at 5V and can provide or receive a maximum of 40mA current, and has an internal pull-up resistor of 20-50 KOhms which are disconnected by default. Out of these 14 pins, some pins have specific functions as listed below:

- Serial Pins 0 (Rx) and 1 (Tx): Rx and Tx pins are used to receive and transmit TTL serial data. They are connected with the corresponding ATmega328P USB to TTL serial chip.
- External Interrupt Pins 2 and 3: These pins can be conFigd to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- PWM Pins 3, 5, 6, 9 and 11: These pins provide an 8-bit PWM output by using analogWrite() function.
- SPI Pins 10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK): These pins are used for SPI communication.
- In-built LED Pin 13: This pin is connected with an built-in LED, when pin 13 is HIGH – LED is on and when pin 13 is LOW, its off.

Along with 14 Digital pins, there are 6 analog input pins, each of which provide 10 bits of resolution, i.e. 1024 different values. They measure from 0 to 5 volts but this limit can be increased by using AREF pin with analog Reference() function. Analog pin 4 (SDA) and pin 5 (SCA) also used for TWI communication using Wire library.

Arduino Uno has a couple of other pins as explained below:

- AREF: Used to provide reference voltage for analog inputs with analogReference() function.
- Reset Pin: Making this pin LOW, resets the microcontroller.

Applications

- Prototyping of Electronics Products and Systems
- Multiple DIY Projects.
- Easy to use for beginner level DIYers and makers.
- Projects requiring Multiple I/O interfaces and communications.

3.6 PROGRAMMING IN ARDUINO IDE

The Uno can be programmed with the Arduino Software (IDE). Select "Arduino/Genuino Uno" from the Tools > Board menu (according to the microcontroller on the board). For details, see the reference and tutorials. The ATmega328 on the Uno comes preprogrammed with a bootloader that allows to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

Bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

It can use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or the ISP header with an external programmer (overwriting the DFU bootloader) can be used.

3.6.1 Warnings

The Uno has a resettable polyfuse that protects the computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

3.6.2 Power

The Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- 5V- This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board. We don't advise it.
- 3V3 - A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- Vin - The input voltage to the Uno board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). the supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- GND - Ground pins.
- IOREF - This pin on the Uno board provides the voltage reference with which the microcontroller operates. A properly conFigd shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

3.6.3 Memory

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

3.6.4 Input and Output

Realize the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical. Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 2050k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be conFigd to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labelled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

There are a couple of other pins on the board:

- AREF - Reference voltage for the analog inputs. Used with analogReference().
- Reset - Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

3.6.5 Communication

The UNO has a number of facilities for communicating with a computer, another Uno board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed.

However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1. A Software Serial library allows serial communication on any of the UNO's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

3.6.6 Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Uno board is designed in a way that allows it to be reset by software running on a

connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allows to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labelled "RESETEN". It is also possible to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

3.6.7 Revisions

Revision 3 of the board has the following new features:

- pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V and with the Arduino Due that operates with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

3.6.8 USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects the computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

- **Physical Characteristics**

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

- **Software Tips**

When bootloading Atmega8 chip with Arduino 0010, there is a command (-i800) that makes bootloader delay 10 minutes. So, if need to use bootloader, use command line instead of IDE, removing "–i800" command and adding "–F" command, or use Arduino 0007 IDE. To upload sketches Arduino 0010 works fine.

3.6.9 Arduino new features

- Full compatible with Shield Boards (Version 2 is the only Arduino Board not compatible with Shield Boards because of ICSP header wrong position, and tall components);
- AVcc LP filter to reduce noise level on ADC;
- Auto reset feature;
- Auto reset enable/disable jumper, to avoid not desired resetting;
- Arduino Diecimila compatible reset pin;

- Pin13 onboard led, with current limiter resistor;
- TX and RX onboard leds;
- Power led with appropriate current limiter resistor (less 20mA of consumption);
- Jumper to disable serial communication and to enable RX external pull down resistor, to avoid "RX floating error". This feature allows to use digital pin0 and pin1 as a normal pin, when serial communication is not needed;
- All similar components (diodes, transistors, leds, capacitors) has the same board orientation (to makes easier to mount with less mistakes);
- No wires between pads, more space between wires, larger wires, larger pads (better for etching, soldering and drilling, with no short circuits, soldering bridges or open wires in corrosion);
- Only 3 wire bridges; electrolitic capacitor (in serial to TTL circuit) changed to bipolar type (to avoid inverted voltage problem when serial cable is not connected), All jumpers are right angle type, to allow Shield Boards use.

3.7 GAS SENSOR

A device that is used to detect or measure or monitor the gases like ammonia, benzene, sulfur, carbon dioxide, smoke, and other harmful gases are called as an air quality gas sensor. The MQ135 sensor, which belongs to the series of MQ gas sensors, is widely used to detect harmful gases, and smoke in the fresh air as shown in the Fig 3.4. This article gives a brief description of how to measure and detect gases by using an MQ135 sensor.

The MQ135 family of sensors has a minor warmer inside alongside with an electrochemical sensor. They respond for range gases at the room temperature. An analog signal and might be read with an analog input of the microcontroller like the yield of each sensor. The fundamental objective of the in general framework will be should identify

“poisonous gas and radiation leakage”. In case any radiation or poison gases available in industrial zones, that radiation or gases generally influenced by the industries nearby living humans. Few gasses continuous breathing means to kill the people begin, and nature mixed this gas or radiation contaminated nature condition. If the gases are unscented will a chance to be uncovered to it for a long time that might cause severe health issues. Gases like CO (carbon monoxide) are scentless that with a focus above 350ppm cause fainting and confusion, over it will confidently kill separately. Every gas has its own physical and chemical assets that make them problematic to examine without any instrument.

Fig 3.4 Gas Sensor

Dangerous gases are existing in different levels relying upon the density and concentration of it. Gas sensor is working gas atom to absorb IR light each gas atom absorption having specific wavelength. Wavelength-based distinguished gases. Radiation sensor working it measures the number of counts striking per every minute distinguished towards the Geiber tube. The temperature sensor is sensing temperatures situation, this all sensor gathering information send to board.

A microcontroller officially programmed that system that system operation dependent upon gases and radiation observing level. Assume getting sensor value level is high means designated the closest fire station; this evidence-based save industries surrounding human life. Gadget placed area is having an LCD display it shows any leakage happening time indication display. Wi-Fi module utilizing transmitting information speed rate is high compared to ZigBee module.

3.7.1 MQ135 Pin Configuration

- Pin 1: VCC: This pin refers to a positive power supply of 5V that power up the MQ135 sensor module.
- Pin 2: GND (Ground): This is a reference potential pin, which connects the MQ135 sensor module to the ground.
- Pin 3: Digital Out (Do): This pin refers to the digital output pin that gives the digital output by adjusting the threshold value with the help of a potentiometer. This pin is used to detect and measure any one particular gas and makes the MQ135 sensor work without a microcontroller.
- Pin 4: Analog Out (Ao): This pin generates the analog output signal of 0V to 5V and it depends on the gas intensity. This analog output signal is proportional to the gas vapor concentration, which is measured by the MQ135 sensor module. This pin is used to measure the gases in PPM. It is driven by TTL logic, operates with 5V, and is mostly interfaced with microcontrollers.

3.7.2 Specifications and Features

- It has a wide detection scope. High sensitivity and faster response. Long life and stability.
- The operating voltage: +5V.
- Measures and detects NH3, alcohol, NOx, Benzene, CO2, smoke etc.
- Range of analog output voltage: 0V-5V.
- Range of digital output voltage: 0V-5V (TTL logic).
- Duration of preheating: 20 seconds.
- Used as an analog or digital sensor.
- The potentiometer is used to vary the sensitivity of the digital pin.
- Operating temperature: -10°C to -45°C.

3.7.3 Working of MQ135 Sensor

To measure or detect the gases, use analog pins or digital pins. Just apply 5V to the module and can observe that the module's power LED turns ON (glows) and the output LED turns OFF when no gas is detected by the module. This means that the output of the digital pin is 0V. Note that the sensor must be kept for preheating time for 20seconds (as mentioned in the specifications) before the actual operation.

Now, once when the MQ135 sensor is operated to detect, then the LED output goes high along with the digital output pin. Otherwise, use the potentiometer until the output increases. Whenever the sensor detects a certain gas concentration, the digital pin goes high (5V), otherwise it stays low (0V).

We can also use analog pins to get the same result. The output analog values (05V) are read from the microcontroller. This value is directly proportional to the gas concentration detected by the sensor. By the experimental values, we can observe the working and reaction of the MQ135 sensor with different gas concentrations and the programming developed accordingly.

3.8 INTERNET OF THINGS (IOT)

The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

The Internet of Things (IoT) refers to the ever-growing network of physical objects that feature an IP address for internet connectivity, and the communication that occurs between these objects and other Internet-enabled devices and systems. IoT systems allow users to achieve deeper automation, analysis, and integration within a system. They improve the reach of these areas and their accuracy. IoT utilizes existing and emerging technology for sensing, networking, and robotics. IoT exploits recent advances in software, falling hardware prices, and modern attitudes towards technology. Its new and advanced elements bring major changes in the delivery of products, goods, and services;

and the social, economic, and political impact of those changes. The IoT applications of computing devices as shown in the Fig 3.5

Fig 3.5 IoT applications

3.8.1 History of IoT

Kevin Ashton, co-founder of the Auto-ID Center at MIT, first mentioned the internet of things in a presentation he made to Procter & Gamble (P&G) in 1999. Wanting to bring radio frequency ID (RFID) to the attention of P&G's senior management, Ashton called his presentation "Internet of Things" to incorporate the cool new trend of 1999: the internet. MIT professor Neil Gershenfeld's book, When Things Start to Think, also appearing in 1999, didn't use the exact term but provided a clear vision of where IoT was headed.

IoT has evolved from the convergence of wireless technologies, microelectromechanical systems (MEMS), microservices and the internet. The convergence has helped tear down the silos between operational technology (OT) and information technology (IT), enabling unstructured machine-generated data to be analyzed for insights to drive improvements. Although Ashton's was the first mention of the internet of things, the idea of connected devices has been around since the 1970s, under the monikers embedded internet and pervasive computing.

The first internet appliance, for example, was a Coke machine at Carnegie Mellon University in the early 1980s. Using the web, programmers could check the status of the machine and determine whether there would be a cold drink awaiting them, should they decide to make the trip to the machine.

IoT evolved from machine-to-machine (M2M) communication, i.e., machines connecting to each other via a network without human interaction. M2M refers to connecting a device to the cloud, managing it and collecting data. Taking M2M to the next level, IoT is a sensor network of billions of smart devices that connect people, systems and other applications to collect and share data. As its foundation, M2M offers the connectivity that enables IoT.

The internet of things is also a natural extension of SCADA (supervisory control and data acquisition), a category of software application program for process control, the gathering of data in real time from remote locations to control equipment and conditions. SCADA systems include hardware and software components. The hardware gathers and feeds data into a computer that has SCADA software installed, where it is then processed and presented it in a timely manner. The evolution of SCADA is such that late-generation SCADA systems developed into first-generation IoT systems.

3.8.2 IoT working

An IoT ecosystem consists of web-enabled smart devices that use embedded processors, sensors and communication hardware to collect, send and act on data they acquire from their environments. IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud

to be analyzed or analyzed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another. The devices do most of the work without human intervention, although people can interact with the devices -- for instance, to set them up, give them instructions or access the data. The connectivity, networking and communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed.

3.8.3 Benefits of IoT

The internet of things offers a number of benefits to organizations, enabling them to:

- Monitor their overall business processes;
- Improve the customer experience;
- Save time and money;
- Enhance employee productivity;
- Integrate and adapt business models;
- Make better business decisions; and
- Generate more revenue.

IoT encourages companies to rethink the ways they approach their businesses, industries and markets and gives them the tools to improve their business strategies.

3.8.4 Consumer and enterprise IoT applications

There are numerous real-world applications of the internet of things, ranging from consumer IoT and enterprise IoT to manufacturing and industrial IoT (IoT). IoT applications span numerous verticals, including automotive, telco, energy and more.

In the consumer segment, for example, smart homes that are equipped with smart thermostats, smart appliances and connected heating, lighting and electronic devices can be controlled remotely via computers, smartphones or other mobile devices.

Wearable devices with sensors and software can collect and analyze user data, sending messages to other technologies about the users with the aim of making users' lives

easier and more comfortable. Wearable devices are also used for public safety -- for example, improving first responders' response times during emergencies by providing optimized routes to a location or by tracking construction workers' or firefighters' vital signs at life-threatening sites.

In healthcare, IoT offers many benefits, including the ability to monitor patients more closely to use the data that's generated and analyze it. Hospitals often use IoT systems to complete tasks such as inventory management, for both pharmaceuticals and medical instruments.

Smart buildings can, for instance, reduce energy costs using sensors that detect how many occupants are in a room. The temperature can adjust automatically -- for example, turning the air conditioner on if sensors detect a conference room is full or turning the heat down if everyone in the office has gone home.

In agriculture, IoT-based smart farming systems can help monitor, for instance, light, temperature, humidity and soil moisture of crop fields using connected sensors. IoT is also instrumental in automating irrigation systems.

In a smart city, IoT sensors and deployments, such as smart streetlights and smart meters, can help alleviate traffic, conserve energy, monitor and address environmental concerns, and improve sanitation.

3.9 NODE MCU

NodeMCU is an open-source firmware and development kit that helps to prototype or build IoT products. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The firmware uses the Lua scripting language. It is based on the eLua project and built on the Espressif Non-OS SDK for ESP8266.

Fig 3.6 ESP8266 NodeMCU

With just a few lines of code can establish a WiFi connection and define input/output pins according to needs exactly like arduino, turning ESP8266 into a web server and a lot more. It is the WiFi equivalent of ethernet module. Now have internet of things (iot) real tool. With its USB-TTL, the nodeMCU Dev board supports directly flashing from USB port. It combines features of WIFI access point and station + microcontroller. These features make the NodeMCU extremely powerful tool for Wifi networking. It can be used as access point and/or station, host a webserver or connect to internet to fetch or upload data.

ESP8266 is highly integrated wireless SOCs, designed for space and power constrained mobile platform designers. It provides unsurpassed ability to embed Wi-Fi capabilities within other systems or to function as a standalone application with low cost and minimal space requirement. The module supports standard IEEE 802.11 b/g/n agreement, complete TCP/IP protocol stack. Users can use the add modules to an existing device networking or building a separate network controller. ESP8266EX offers a complete and self-contained Wi-Fi networking solution, it can be used to host the application or to offload Wi-Fi networking functions from another application processor. When ESP8266EX hosts the application, it boots up directly from an external flash. In has integrated cache to improve the performance of the system in such applications.

Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any micro controller-based design with simple connectivity (SPI/SDIO or I2C/UART interface). ESP8266EX is among the most integrated Wi-Fi chip in the industry. It integrates the antenna switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules. It requires minimal external circuitry and the entire solution, including front-end module are designed to occupy minimal PCB area.

ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor, with on-chip SRAM, besides the Wi-Fi functionalities.

Features

- Finally, programable WiFi module. Arduino-like (software defined) hardware IO.
- Can be programmed with the simple and powerful Lua programming language or Arduino IDE. USB-TTL included, plug & play.
- 10 GPIOs D0-D10, PWM functionality, IIC and SPI communication, 1-Wire and ADC A0 etc. all in one board.
- Wifi networking (can be used as access point and/or station, host a web server), connect to internet to fetch or upload data.
- Event-driven API for network applications, PCB antenna. **Pins**

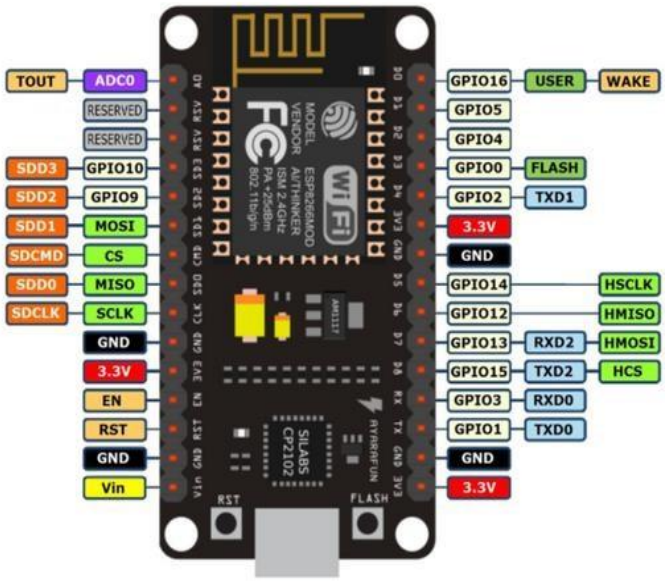

Fig 3.7 Pin Configuration

The above Fig 3.7 represents the Pin Configuration of the NodeMCU. NodeMCU provides access to the GPIO (General Purpose Input/Output) and a pin mapping table is part of the API documentation.

Table No. 3.2 Node MCU Pin Configuration

I/O Index	ESP8266 pin
0[*]	GPIO16
1	GPIO5
2	GPIO4
3	GPIO0
4	GPIO2
5	GPIO14
6	GPIO12
7	GPIO13
8	GPIO15
9	GPIO3
10	GPIO1
11	GPIO9
12	GPIO10

[*] D0 (GPIO16) can only be used for GPIO read/write. It does not support opendrain/interrupt/PWM/I²C or 1-Wire.

CHAPTER 4

TOOLS USED

4.1 SOFTWARE REQUIREMENTS

- Arduino IDE
- Language – Embedded C

4.2 ARDUINO IDE

The Uno can be programmed with the Arduino Software (IDE). Select "Arduino/Genuino Uno" from the Tools > Board menu (according to the microcontroller on board). For details, see the reference and tutorials. The ATmega328 on the Uno comes preprogrammed with a bootloader that allows to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

Bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then reseing the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware can be used. Or the ISP header with an external programmer (overwriting the DFU bootloader) can be used.

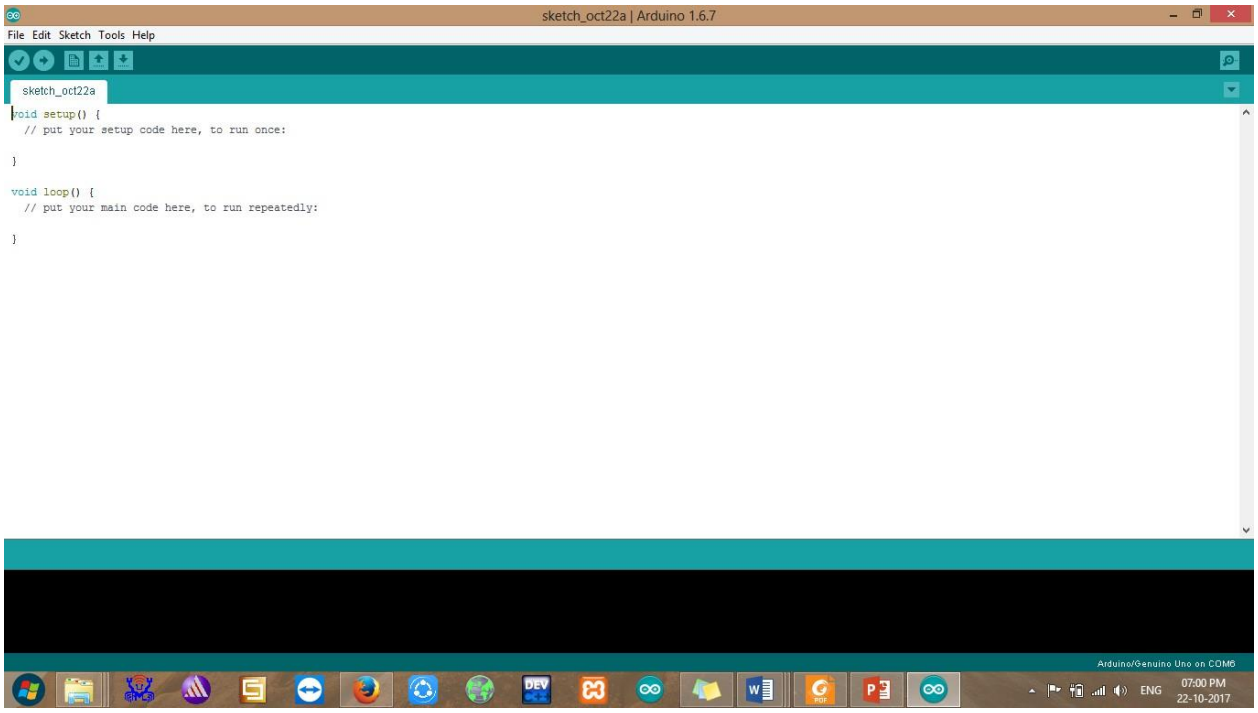

Fig 4.1 Arduino IDE Viewer

4.2.1 Warnings

The Uno has a resettable poly fuse that protects computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

4.2.2 Differences with other boards

The Uno differs from all preceding boards in that it does not use the FTDI USBto-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

4.2.3 Power

The Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector. The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. The power pins are as follows:

- Vin - The input voltage to the Uno board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- 5V - This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage board. We don't advise it.
- 3V3 - A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- GND - Ground pins.
- IOREF - This pin on the Uno board provides the voltage reference with which the microcontroller operates. A properly conFigd shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

4.2.4 Memory

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

4.2.5 Input and Output

See the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical. Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(),digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 2050k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be conFigd to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

There are a couple of other pins on the board:

- AREF - Reference voltage for the analog inputs. Used with analog Reference().
- Reset - Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

4.2.6 Communication

The Uno has a number of facilities for communicating with a computer, another Uno board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed.

However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1. A Software Serial library allows serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

4.2.7 Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader

can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESETEN". may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

4.2.8 Revisions

Revision 3 of the board has the following new features:

- pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V and with the Arduino Due that operates with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

4.2.9 USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

4.3 EMBEDDED C

Embedded C programming builds with a set of functions where every function is a set of statements that are utilized to execute some particular tasks. An Embedded system program allows the hardware to check the inputs & control outputs accordingly. In every embedded system - based projects, Embedded C programming plays a key role to make the microcontroller run & perform the preferred actions. These programs play a prominent role in monitoring and controlling external devices. They also directly operate and use the internal architecture of the microcontroller, such as interrupt handling, timers, serial communication, and other available features. There are different steps involved in designing an embedded c program like the following.

- Comments
- Directives of Processor
- Configuration of Port
- Global variables
- Core Function/Main Function
- Declaration of Variable
- The logic of the Program

4.3.1 Comments

In programming languages, comments are very essential to describe the program’s function. The code of the comments is non-executable but used to provide program documentation. To understand the function of the program, this will make a

simple method to understand the function of the program. In embedded C, comments are available in two types namely single line and mainline comment.

In an embedded C programming language, can place comments in our code which helps the reader to understand the code easily.

4.3.2 Directives of Processor

The lines included within the program code are called preprocessor directives which can be followed through a hash symbol (#). These lines are the preprocessor directives but not programmed statements. The code can be examined through a preprocessor before real code compilation starts & resolves these directives before generating a code through regular statements.

4.3.3 Configuration of Port

The microcontroller includes several ports where every port has different pins. These pins can be used for controlling the interfacing devices. The declaration of these pins can be done within a program with the help of keywords. The keywords in the embedded c program are standard as well as predefined like a bit, sbit, SFR which are used to state the bits & single pin within a program.

There are certain words that are reserved for doing specific tasks. These words are known as keywords. They are standard and predefined in the Embedded C. Keywords are always written in lowercase. These keywords must be defined before writing the main program.

4.3.4 Global Variables

When the variable is declared before the key function is known as the global variable. This variable can be allowed on any function within the program. The global variable’s life span mainly depends on the programming until it reaches an end.

4.3.5 Core Function / Main Function

The main function is a central part while executing any program and it begins with the main function simply. Each program utilizes simply one major function since if the program includes above one major function, next the compiler will be confused in begin the execution of the program.

4.3.6 Declaration of Variable

The name like the variable is used for storing the values but this variable should be first declared before utilized within the program. The variable declaration states its name as well as a data type. Here, the data type is nothing but the representation of storage data. In embedded C programming, it uses four fundamental data types like integer, float, character for storing the data within the memory. The data type size, as well as range, can be defined depending on the compiler.

The data type refers to an extensive system for declaring variables of different types like integer, character, float, etc. The embedded C software uses four data types that are used to store data in memory.

The ‘char’ is used to store any single character; ‘int’ is used to store integer value, and ‘float’ is used to store any precision floating-point value. The size and range of different data types on a 32-bit machine are given in the following table. The size and range may vary on machines with different word sizes.

- The char/signed char data type size is 1 byte and its range is from -128 to +128
- The unsigned char data type size is 1 byte and its range is from 0 to 255
- Int/signed int data type size is 2 byte and its range is from -32768 to 32767
- Unsigned int data type size is 2 byte and its range is from 0 to 65535

4.3.7 The logic of the Program

The logic of the program is a plan of the lane that appears in the theory behind & predictable outputs of actions of the program. It explains the statement otherwise theory

regarding why the embedded program will work and shows the recognized effects of actions otherwise resources.

CHAPTER 5

RESULT AND DISCUSSION

The "Autonomous Toilet Cleaning Robot" is equipped with essential features for maintaining restroom cleanliness. It comes with built-in tools such as brushes, water containers, and other necessary items for effective cleaning. Controlled through web applications, it offers convenience and accessibility, with the potential for future upgrades to app-based controls. Its movement and direction can be precisely defined using the web application interface, allowing for efficient navigation within the restroom environment. This innovative robot streamlines the cleaning process, ensuring thorough sanitation and hygiene with minimal human intervention. With its versatile capabilities and user-friendly controls, it represents a significant advancement in restroom maintenance technology.

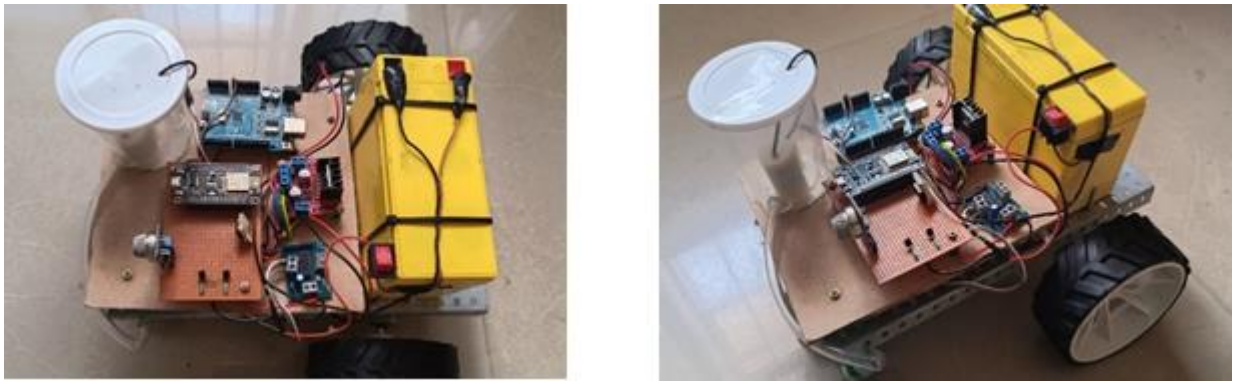

Fig 5.1 Working Model Robot

The above Fig 5.1 (Working model Robot) represents the structure and design of “Autonomous toilet cleaning robot”.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

An autonomous toilet cleaning robot utilizing gas sensor technology and IoT (Internet of Things) represents a significant advancement in restroom hygiene. This innovative system offers several compelling benefits. The gas sensor allows the robot to detect the presence of undesirable odors, triggering cleaning cycles only when necessary. This targeted approach optimizes efficiency and minimizes cleaning solution usage. Additionally, IoT integration enables remote monitoring and control of the robot. Users can track cleaning schedules, receive real-time status updates, and even initiate cleaning cycles remotely. This level of automation ensures a consistently clean and hygienic restroom environment, reducing the workload on cleaning staff and improving public health. Overall, the autonomous toilet cleaning robot with gas sensor and IoT presents a promising solution for promoting cleanliness and enhancing user experience in restrooms.

6.2 FUTURE SCOPE

The future of cleaning robots represents a significant advancement in restroom hygiene technology. As the technology matures, we can expect further improvements, such as: Integration of technologies like UV disinfection for enhanced sanitation, Robots that can automatically refill cleaning solutions and recharge batteries, increasing autonomy and AI-powered robots that can adapt cleaning routines based on real-time sensor data and usage patterns. This innovative technology has the potential to transform public restrooms, leading to a cleaner and healthier future. The robots will become more adaptable to different environments and will be controllable through voice commands or smartphone apps. Expect to see them working not just in homes but also in commercial and industrial settings, freeing up human workers for other tasks.

REFERENCES

[1] R. Bormann, X. Wang, J. Xu and J. Schmidt, "DirtNet: Visual Dirt Detection for Autonomous Cleaning Robots," 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 2020, pp. 1977-1983, doi: 10.1109/ICRA40945.2020.9196559.

[2] R. Parween, M. Vega Heredia, M. M. Rayguru, R. Enjikalayil Abdulkader and M. R. Elara, "Autonomous Self-Reconfigurable Floor Cleaning Robot," in IEEE Access, vol. 8, pp. 114433-114442, 2020, doi: 10.1109/ACCESS.2020.2999202.

[3] Y. Li, D. Zhang, F. Yin and Y. Zhang, "Operation Mode Decision of Indoor Cleaning Robot Based on Causal Reasoning and Attribute Learning," in IEEE Access, vol. 8, pp. 173376-173386, 2020, doi:10.1109/ACCESS.2020.3003343.

[4] J. Moura, W. Mccoll, G. Taykaldiranian, T. Tomiyama and M. S. Erden, "Automation of Train Cab Front Cleaning With a Robot Manipulator," in IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 3058-3065, Oct. 2018, doi: 10.1109/LRA.2018.2849591.

[5] Z. Li, Q. Xu and L. M. Tam, "A Survey on Techniques and Applications of WindowCleaning Robots," in IEEE Access, vol. 9, pp. 111518-111532, 2021, doi: 10.1109/ACCESS.2021.3103757.

[6] M. Seo, S. Yoo, J. Kim, H. S. Kim, and T. Seo, ‘‘Dual ascender robot with position estimation using angle and length sensors,’’ IEEE Sensors J., vol. 20, no. 13, pp. 7422– 7432, Jul. 2020.

[7] M. Choi, M. Seo, H. S. Kim, and T. Seo, ‘‘UKF-based sensor fusion method for position estimation of a 2-DOF rope driven robot,’’ IEEE Access, vol. 9, pp. 12301– 12308, 2021.

[8] T.-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollar, “Focal loss for dense object detection,” in Proceedings of the International Conference on Computer Vision (ICCV), 2017, pp. 2999–3007.

[9] I. Loshchilov and F. Hutter, “Sgdr: Stochastic gradient descent with warm restarts,” in Proceedings of the International Conference on Learning Representations (ICLR), 2016.

[10] H. Zhang, M. Cisse, Y. Dauphin, and D. Lopez-Paz, “mixup: Beyond ·empirical risk minimization,” in Proceedings of the International Conference on Learning Representations (ICLR), 2018.

PUBLICATION DETAILS

CONFERENCE PRESENTATION:

BARATHKUMAR K , KIRUTHIKAA B D, KOWSHIKA M presented a paper entitled **"AN AUTNOMOUS TOILET CLEANING ROBOT"** in National Conference on Multidisciplinary Research and Innovations in Engineering and Technology (**MRIET-2024**) held at **Knowledge Institute of Technology**, Salem.