

ABSTRACT

Facial features including eye and head movements reveal driver weariness and drunkenness. Characteristics such as facial angle, head movement, and eye blinking have been monitored during the design, construction, and testing phases of a driver impairment monitoring system. This system made use of an IOT module, microcontroller, tilt sensor, drowsiness detection, and an alcohol sensor (MQ3). The alcohol sensor (MQ3), tilt sensor, and open eye sensor or sleepiness detection are the three sensors that are used. There was a microcontroller and an IOT module. There was a buzzer, a display, and a reset switch. The microcontroller will receive signals from three sensors as well as a reset switch. The cover is fastened to the alcohol sensor. The driver would activate if it did not react to the sensor's alarm a light that could cause the engine to shut off. Second, the microcontroller would sound an alarm if the driver closed his eyelids for a predetermined amount of time, detecting it via an infrared open eye sensor. Thirdly, the microcontroller would sound the alarm if the driver felt sleepy and tilted his head for a predetermined amount of time. We propose a smart system based on the Internet of Things. The buzzer alerted the driver and an SMS was sent to the car owners when it identified the driver's closed eyes, head tilt indicated tiredness, or alcohol use for a predetermined amount of time. Ten individuals were used to measure the deviations in face angle, and it was discovered that the average accuracy was quite high.

TABLE OF CONTENTS

<u>CHAPTER NO</u>	<u>TITLE</u>	<u>PAGE NO</u>
	ABSTRACT	iv
	LIST OF TABLES	viii
	LIST OF FIGURES	ix
	LIST OF ABBREVIATIONS	x
1	INTRODUCTION	1
2	LITERATURE SURVEY	9
	2.1 The impact of drivers in attention on near Crash /crash risk.	9
	2.2 Monitoring, managing and motivating driver Safety and well being	10
	2.3 pupil detection in the presence of specular reflection	11
	2.4 Robust real-time pupil tracking in highly off Axis images	12
	2.5 Analysis of driver head and eye movement Correspondence: predicting drive Locations using head rotation data	13
3	EXISTING SYSTEM	14

4	PROPOSED SYSTEM	15
	4.1 Block diagram	16
5	COMPONENTS REQUIRED	17
	5.1 Arduino UNO R3 MC	17
	5.1.1 Specification	18
	5.1.2 Input	21
	5.1.3 Communication	22
	5.1.4 Programming	23
	5.1.5 Automatic Reset	24
	5.1.6 USB Over current protection	24
	5.1.7 Physical Characteristics	25
	5.2 Relay	25
	5.2.1 Basic Design and operation	26
	5.2.2 Applications	27
	5.3 LCD	28
	5.3.1 16x2 LCD	29
	5.3.2 Pin diagram of 16x2 LCD	29
	5.3.3 Pin Specification	29
	5.3.4 Advantages	30
	5.4 Power supplies	33
	5.4.1 Introduction	33
	5.4.2 IC Voltage Regulators	35
	5.4.3 Three Terminal Voltage Regulators	35
	5.4.4 Fixed Positive Voltage Regulator	36

	5.5 Vibration Sensor	37
	5.5.1 NodeMCU	38
	5.5.2 USB to Serial Converter	39
	5.5.3 Gas Sensor	42
	5.6 DC Motor	45
6	SOFTWARE REQUIREMENT	47
	6.1 Embedded C	47
	6.1.1 Introduction	47
	6.1.2 Embedded System Programming	49
	6.1.3 Difference Between C And Embedded C	52
	6.2 PHP	53
	6.2.1 Feature of PHP	53
	6.2.2 Accessing in HTML Page	54
	6.2.3 Accessing in PHP Page	55
	6.2.4 Security	57
	6.2.5 Benefit of PHP	58
	6.3 Back-End Software	61
	6.3.1 MYSQL Introduction	61
	6.4 MATLAB	63
7	RESULT	78
8	FEATURE SCOPE	79
9	CONCLUSION	80
10	REFERENCE	81

LIST OF TABLES

TABLE NO	DESCRIPTION	PAGE NO
5.1	Specification of AT Mega 328P	18
5.2	LCD Pin Description	29
5.3	Positive Voltage Regulator in 7800 Series	36

LIST OF FIGURES

FIG NO	TITLE	PAGE NO
1.1	Wireless Sensor Networks	2
4.1	Block Diagram of Proposed System	16
5.1	Input and Output Data Transfer Microcontroller	22
5.2	16x2 LCD Display	29
5.3	LCD Pin Details	29
5.4	Diagram of Arduino UNO SMD R3	33
5.5	Vibration Sensor	37
5.6	USB to Serial Converter	39
5.7	Gas Sensor	43
5.8	DC Motor	45
7.1	Final Output of the Project	78

CHAPTER 1

INTRODUCTION

Driver distractions are the leading cause of most vehicle crashes and near-crashes. According to a study released by the National Highway Traffic Safety Administration (NHTSA) and the Virginia Tech Transportation Institute (VTTI), 80% of crashes and 65% of near-crashes involve some form of driver distraction. In addition, distractions typically occurred within three seconds before the vehicle crash. Recent reports have shown that from 2011 to 2012, the number of people injured in vehicle crashes related to distracted driving has increased 9%. Distracted driving is defined as any activity that could divert a person's attention away from the primary task of driving. Distractions include texting, using a smartphone, eating and drinking, adjusting a CD player, operating a GPS system or talking to passengers. This is particularly challenging nowadays, where a wide spectrum of technologies has been introduced into the car environment. Consequently, the cognitive load caused by secondary tasks that drivers have to manage has increased over the years, hence increasing distracted driving. NSTHA has reported that texting, browsing, and dialing cause the longest period of drivers taking their Eyes Off the Road (EOR) and increase the risk of crashing by three folds. A recent study shows that these dangerous behaviors are wide-spread among drivers, 54% of motor vehicle drivers in the United States usually have a cell phone in their vehicles or carry cell phones when they drive. Monitoring driver activities forms the basis of a safety system that can potentially reduce the number of crashes by detecting anomalous situations. In authors showed that a successful vision-based distracted driving detection system is built upon reliable EOR estimation. However, building a real time EOR detection system for real driving scenarios is very challenging for several reasons:

- (1) The system must operate during the day and night and under real world illumination conditions;
- (2) changes in drivers' head pose and eye movements.

WIRELESS SENSOR NETWORK

A wireless sensor network (WSN) is a computer network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations. The development of wireless sensor networks was originally motivated by military applications such as battlefield surveillance. However, wireless sensor networks are now used in many civilian application areas, including environment and habitat monitoring, healthcare applications, home automation, and traffic control.

In addition to one or more sensors, each node in a sensor network is typically equipped with a radio transceiver or other wireless communications device, a small microcontroller, and an energy source, usually a battery. The size of a single sensor node can vary from shoebox-sized nodes down to devices the size of grain of dust. The cost of sensor nodes is similarly variable, ranging from hundreds of dollars to a few cents, depending on the size of the sensor network and the complexity required of individual sensor nodes.

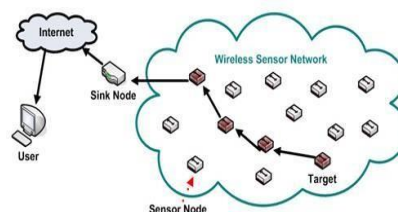


Fig 1.1 Wireless sensor Network

Applications

The applications for WSNs are many and varied. They are used in commercial and industrial applications to monitor data that would be difficult or expensive to monitor using wired sensors. They could be deployed in wilderness areas, where they would remain for many years (monitoring some environmental variable) without the need to recharge/replace their power supplies. They could form a perimeter about a property and monitor the progression of intruders (passing information from one node to the next). There are a many uses for WSNs.

Typical applications of WSNs include monitoring, tracking, and controlling. Some of the specific applications are habitat monitoring, object tracking, nuclear reactor controlling, fire detection, traffic monitoring, etc. In a typical application, a WSN is scattered in a region where it is meant to collect data through its sensor nodes.

- Environmental monitoring
- Habitat monitoring
- Acoustic detection
- Seismic Detection
- Military surveillance
- Inventory tracking
- Medical monitoring
- Smart spaces
- Process Monitoring

Area monitoring

Area monitoring is a typical application of WSNs. In area monitoring, the WSN is deployed over a region where some phenomenon is to be monitored. As an example, a large quantity of sensor nodes could be deployed over a battlefield to detect enemy intrusion instead of using landmines. When the sensors detect the event being monitored (heat, pressure, sound, light, electro- magnetic field, vibration, etc.), the event needs to be reported to one of the base stations, which can take appropriate action (e.g., send a message on the internet or to a satellite). Depending on the exact application, different objective functions will require different data-propagation strategies, depending on things such as need for real-time response, redundancy of the data (which can be tackled via data aggregation techniques), need for security, etc.

Characteristics

Unique characteristics of a WSN are:

- Small-scale sensor nodes
- Limited power they can harvest or store
- Harsh environmental conditions
- Node failures
- Mobility of nodes
- Dynamic network topology
- Communication failures
- Heterogeneity of nodes
- Large scale of deployment
- Unattended operation

Sensor nodes can be imagined as small computers, extremely basic in terms of their interfaces and their components. They usually consist of a processing unit with limited computational power and limited memory, sensors (including specific conditioning circuitry), a communication device (usually radio transceivers or alternatively optical), and a power source usually in the form of a battery. Other possible inclusions are energy harvesting modules, secondary ASICs, and possibly secondary communication devices (e.g. RS232 or USB).

The base stations are one or more distinguished components of the WSN with much more computational, energy and communication resources. They act as a gateway between sensor nodes and the end user.

The ability to accurately detect a vehicles location and its status is the main goal of automobile trajectory monitoring systems. Also the high demand of automobiles has also increased the traffic hazards and the road accidents. This is because of the lack of best emergency facilities available in our country this design is a system which can detect accidents in significantly less time and sends the basic information to first aid center within a few seconds covering geographical coordinates, the time and angle in which a vehicle accident had occurred.

Our project aims to present a technology automatically detecting the accident and a hardware tracking device based on GSM/GPS technology informing at the occurrence of accident with sufficient details like exact location and time at which accident happened. This project will establish a communication between the control station and the unit installed in vehicles. Vehicles will have GPS/GSM enabled tracking modules and will be tracked in real time using cellular networks.

The software embedded in the microcontroller will control the various operations of the device by monitoring waveform from the vibration sensor. In case of accident the device will send an alert message along with location data from GPS module to control station using GSM network. It is a comprehensive and effective solution to the poor rescue response in case of accident. The accident reporting can automatically find a traffic accident, search for the spot and then send the basic information to the rescue agency covering geographical coordinates and the time and circumstances in which a traffic accident took place. At the server end, a control function will extract relevant data and store it in a database, to which accident information from prototypes will be polled in real time. Our system combines advanced hardware design and sophisticated control technology into a compact, reliable package.

Wireless sensor network

Wireless sensor networks (WSN), sometimes called wireless sensor and actuator networks (WSAN), are spatially distributed autonomous sensors

to *monitor* physical or environmental conditions, such as temperature, sound, pressure, etc. and to cooperatively pass their data through the network to a main location. The more modern networks are bi-directional, also enabling *control* of sensor activity. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance; today such networks are used in many industrial and consumer applications, such as industrial process monitoring and control, machine health monitoring, and so on.

The WSN is built of "nodes" – from a few to several hundreds or even thousands, where each node is connected to one (or sometimes several) sensors.

Each such sensor network node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna, a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery or an embedded form of energy harvesting. A sensor node might vary in size from that of a shoebox down to the size of a grain of dust, although functioning "motes" of genuine microscopic dimensions have yet to be created. The cost of sensor nodes is similarly variable, ranging from a few to hundreds of dollars, depending on the complexity of the individual sensor nodes. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and communications bandwidth.

Various government policies have encouraged the combination of hospital accident and emergency departments into centralized units, responsible for large geographic areas. While this provides an improved quality of medical care, travel time to a crash scene is often compromised for those not near these centralized locations. According to Brown, there is a positive association between ambulance delay and the ratio of fatal to serious injuries. This study found an increased mortality rate in counties that had a low population density, further suggesting a link between elevated response time and prognosis. Numerous other studies have also demonstrated the relationship between decreases in response time and corresponding decreases in mortality.

The ability to accurately detect a vehicle's location and its status is the main goal of automobile trajectory monitoring systems & also the high demand of automobiles has also increased the traffic hazards and the road accidents. This is because of the lack of best emergency facilities available in our country. This design is a system which can detect accidents in significantly less time and sends the basic information to first aid center within a few seconds covering geographical coordinates, the time and angle in which a vehicle accident had

occurred. This alert message is sent to the rescue team in a short time, which will help in saving the valuable lives. These systems are implemented using several hybrid techniques that include: wireless communication, geographical positioning and embedded applications. We see that a lot of life spoils in every accident because of typically long response time to access the appropriate care that may be available if informed in-time. Application of our project can significantly shorten the response time of accident. This is a platform for emergency rescue which will operate optimally in order to reduce the golden time of arrival of rescuers in case of road accidents, when every microsecond counts. Our project aims to present a technology automatically detecting the accident and a hardware tracking device based on GSM/GPS technology informing at the occurrence of accident with sufficient details like exact location and time at which accident happened. This project will establish a communication between the control station and the unit installed in vehicles. Vehicles will have GPS/GSM enabled tracking modules and will be tracked in real time using cellular networks. The software embedded in the microcontroller will control the various operations of the device by monitoring waveform from the vibration sensor. In case of accident the device will send an alert message along with location data from GPS module to control station using GSM network. It is a comprehensive and effective solution to the poor rescue response in case of accident. The accident reporting can automatically find a traffic accident, search for the spot and then send the basic information to the rescue agency covering geographical coordinates and the time and circumstances in which a traffic accident took place.

CHAPTER 2

LITERATURE SURVEY

2.1 S. G. Klauer, T. A. Dingus, V. L. Neale, J. D. Sudweeks, and D. J. Ramsey, “The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data,” Tech. Rep., 2006.

Stand-alone framework for measuring air quality in real time that involves specific parameters: PM 2.5, carbon monoxide, carbon dioxide, temperature, moisture and pressure The Internet of Things is now being used widely in all industries and plays a vital role in our air quality network. The Internet of Things that converges with cloud computing. The model proposed consists of environmental sensing units (such as humidity, temperature, heat index, power, etc.) which are able to track the energy consumed in voltage and current parameters of the different household equipment. The regulating mechanism calibrates further to generate aggregated data and eventually gathersthis data on the Internet portal. For this article, we focused mainly on protection precautions for both the driver and the car by utilizing three forms of sensors. The pulse sensor is used to continuously track the driver's pulse rate and avoids IOT incidents. The User Ambulance and police are told of an incident via IOT.

2.2 J. F. Coughlin, B. Reimer, and B. Mehler, “Monitoring, managing, and motivating driver safety and well-being.” IEEE Pervasive Computing, vol. 10, no. 3, 2011.

We deliver a revolutionary program that has quickly completed this mission. Our technology provides an advanced health monitoring device, which utilizes sensors to control the safety of patients and uses the Internet to alert the family in the event of a crisis. The temperature and pulse monitoring of our machine is used to track the health of patients. This document is the protective helmet inside the mine used by the worker. In order to constantly track mine conditions, this helmet is designed with specific environmental indicators such as fire, fumes, air temperature, humidity and air quality. All these sensors are linked to an a microcontroller which is also built into the helmet. The paper alsoexplores the idea of a wireless network to relay data to the central hub utilizing Zigbee technologies from the consumer helmet.

2.3 T. Yoshioka, S. Nakashima, J. Odagiri, H. Tomonori, and T. Fukui, “Pupil detection in the presence of specular reflection,” in Proceedings of the Symposium on Eye Tracking Research and Applications. ACM, 2014, pp. 363–364.

Cloud storage allows fast access to a common pool of distributed computing services, computers, and networks on request. Fog computing can be seen as the cloud computing extension because it offers low latency, low bandwidth and increased data security and privacy. Data violation is an important problem in the healthcare system that can be solved by special data protection acts and special algorithms. In this article, the m-health program, the Web, Fog processing and computer protection problems are discussed thoroughly in IOT. Our project's primary goal is to avoid major car accidents that have a significant impact on people's lives. Unlike a regular breathalyzer, this alcohol monitor is ideal for the measurement of alcohol in your breathing. A hands-free pointer tracker and a remote monitoring and data transfer

2.4 L. ´ Svirsky, A. Bulling, and N. Dodgson, “Robust real-time pupil tracking in highly off-axis images,” in Proceedings of the Symposium on Eye Tracking Research and Applications. ACM, 2012, pp. 173–176.

Eyeball sensors are the most compatible sensors. In fact, we will improve people's wellbeing that have invested a great deal of time in their vehicles by incorporating intelligent sensors that control the internal atmosphere of the vehicle and the driver of the car. A moisturizer that controls the moisture level inside the car and maintains it clean may be used. The pulse oximeter also tests the driver's blood oxygen. The steering wheel of the vehicle may be applied. Headquartered, dispersed and heterogeneous facilities, instruments and knowledge and connectivity systems, the health services and network are also quite diverse and include a wide spectrum of organizations. With the introduction of the Internet of Things (IoT), robots are introduced into the internet as a "item" and linked to other things. This Chapter clearly shows the long-term benefits of people with robotics and IoT in the fields of health, medical emergencies, e-health etc. The implementation and architecture purpose of an intelligent and real-time drainage and control device with the help of the Internet of Things is discussed in this article. A modulus with an interface between the microcontroller and gas sensor, level indicator, and NRF will be mounted for the drainage loops.

2.5 M. Muaz, J. Lee, B. Reimer, B. Mehler, and T. Victor, “Analysis of drivers’ head and eye movement correspondence: Predicting drivers’ glance location using head rotation data,” in Proceedings of the 8th International Driving Symposium on Human Factors in Driver Assessment, Training, and Vehicle Design, Snowbird, UT, 2015, to Appear.

Face and Eye Detection by Machine Learning (ML) and Deep Learning (DL) Algorithms. Jabbar ell proposed Convolutional Neural Network (CNN) technique of the ML algorithm to detect microsleep and drowsiness. In this paper, detection of driver’s facial landmarks can be achieved through a camera that is then passed to this CNN algorithm to properly identify drowsiness. Here, the experimental classification of eye detection is performed through various data sets like without glasses and with glasses in day or night vision. So, it works for effective drowsiness detection with high precision with android modules. The algorithm of Deep CNN was used to detect eye blink and its state recognition as provided by Sanyal and Chakrabarty. Saleh et al developed an algorithm of LSTM and Recurrent Neural Networks (RNN) to classify driver’s behaviors through sensors. Ed-Doughman et al. analyzed the driver’s behaviors through the RNN algorithm. It specially focuses on construction of real-time fatigue detection to prevent roadside accidents. This system formulates a number of drivers’ faces, which works on multilayered 3D CNN models to identify drowsy drivers and provide 92 percentage acceptance rate.

CHAPTER 3

EXISTING SYSTEM

This work is related to four established areas of computer vision: facial feature extraction, head pose estimation, and gaze tracking. The contribution of this paper is in the integration of cutting-edge algorithms and ideas borrowed and modified from each of these fields in order to demonstrate effective eyes- free gaze classification in the wild (a large on-road driving dataset). contribution of the algorithm is an iterative transform of the image to a normalized coordinate system based on the current estimate of the face shape. Also, to avoid the non-convex problem of initially matching a model of the shape to the image data, the assumption is made that the initial estimate of the shape can be found in a linear subspace. Head pose estimation has a long history in computer vision. Murphy-Hutterian and Trivedi describe 74 published and tested systems from the last two decades. Generally, each approach makes one of several assumptions that limit the general applicability of the system in driver state detection Our approach focuses on the head as the proxy for classifying broad regions of eye movement to provide a mechanism for real-time driver state estimation while facilitating a more economical method of assessing driver behavior in experimental setting during design assessment and safety validation.

CHAPTER 4

PROPOSED SYSTEM

The implementation of a preventive program for this matter has become a big challenge. This method calculates the examination of safety criteria and ocular condition. The USB device and the microcontroller are designed by the driver's head. Many times, inattentive drivers don't try to brake or avoid a collision. Therefore, in order to prevent accidents, a system that monitors the driver's health and stops the car right away if the driver exhibits aberrant behavior is built in. The most vulnerable to dozing off while driving are shift workers, business car drivers, and truck drivers. The majority of accidents are caused by intoxicated drivers. In this research, a tilt sensor gas sensor was suggested as well as an eye blink sensor to identify driver tension and dilated pupils. The suggested system senses the driver's tiredness and, if it is found, sounds an alert through a buzzer. If the driver doesn't wake up, the car will slowdown, shift to the left, and come to a halt. which the business owner bears since they are held accountable. Financial loss may result from it. In this talk, we will introduce an application that gives the business owner driving behavior advice as well as an alarm system for adaptive drivers and owners. An Arduino and a gas sensor are interfaced in this setup. The car will automatically slow down and stop if any of these sensors detects an abnormal state in the driver. There is a buzzer installed in the vehicle that warns other cars or the occupants within the vehicle. Simultaneously, the registered cellphone number receives an SMS alert with the driver's location and condition. can be used to track the driver's whereabouts and will assist him by hitting hospitals and his colleague there. Additionally, this information might be forwarded to the server (Cloud) to notify his colleague and warn the driver.

4.1 BLOCK DIAGRAM

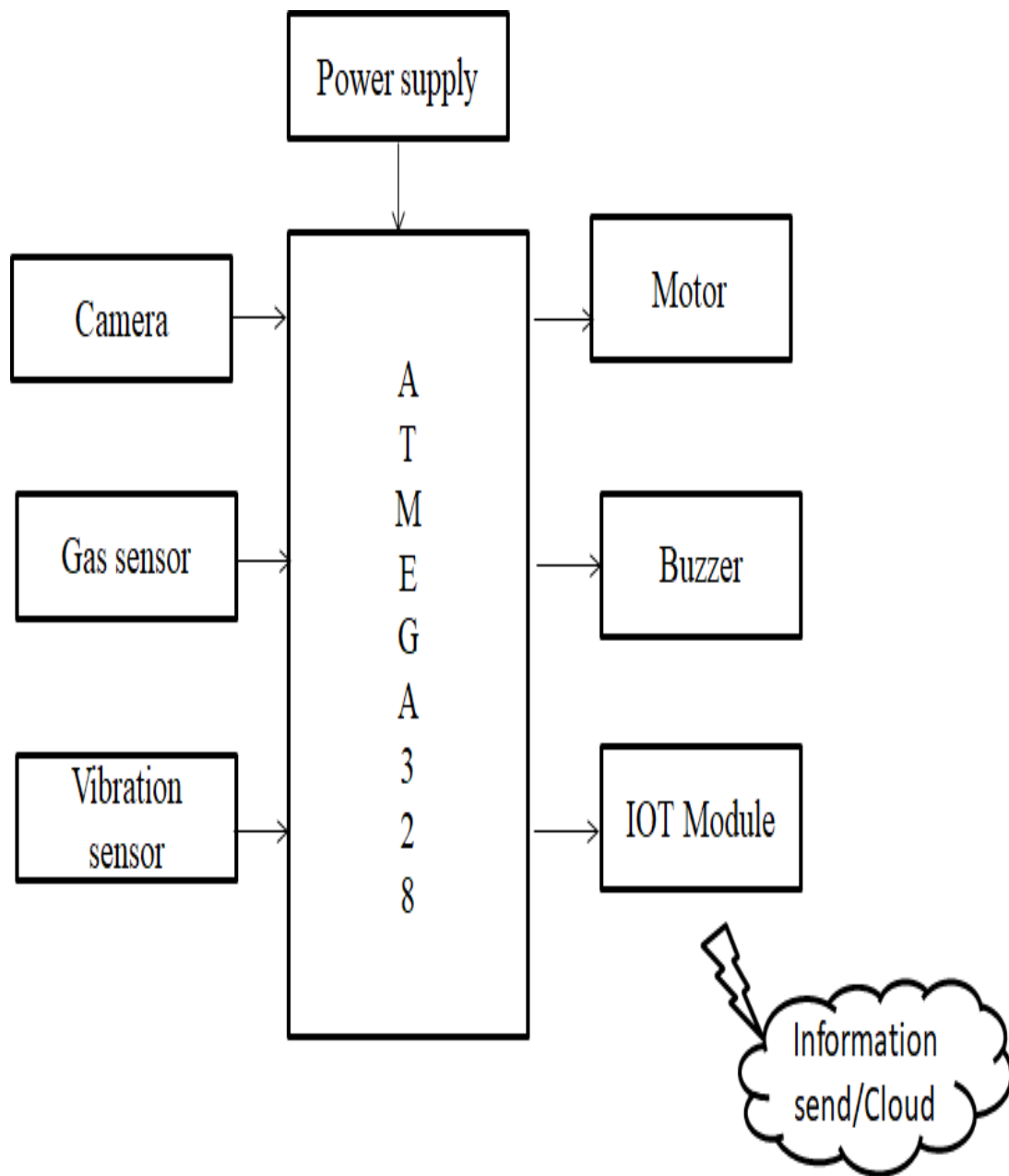


Fig 4.1 Block diagram of proposed solution

CHAPTER-5

HARDWARE REQUIREMENT

5.1 ARDUINO UNO R3 MICROCONTROLLER

The Arduino Uno R3 is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, 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 AC-to-DC adapter or battery to get started.

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

Revision 2 of the Uno board (A000046) has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.

Revision 3 of the board (A000066) has the following new features:

- 1.0 pin out: 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.
- At mega 16U2 replace the 8U2.

5.1.1 SPECIFICATION

- Microcontroller: ATmega328
- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Input Voltage (limits): 6-20V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- DC Current per I/O Pin: 40mA
- DC Current for 3.3V Pin: 50mA
- Flash Memory: 32KB (ATmega328) of which 0.5 KB used by boot loader
- SRAM: 2KB (ATmega328)
- EEPROM: 1KB (ATmega328)
- Clock Speed: 16MHz
- Revision 3 of the board (A000066) has the following new features:
 - ATmega16U2 instead of 8U2 as USB-to-Serial converter
 - 1.0 pin out: added SDA and SCL pins for TWI communication placed 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 and the second one is a not connected pin, that is reserved for future purposes
 - Stronger RESET circuit

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm

Width	53.4 mm
Weight	25 g

Table 5.2 Specification of AT Mega 328P

Power

The Arduino Uno 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 of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be 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 Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

5V- The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.

3V3 - A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND. Ground pins.

5.1.2 Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pin Mode (), digital Write (), and digital Read () functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 m A and has an internal pull-up resistor (disconnected by default) of 20-50 k Ohms. 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 configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attach Interrupt () function for details.

PWM: 3, 5, 6, 9, 10, and 11.

Provide 8-bit PWM output with the analog Write () 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 connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. 3 | Page 3 Arduino Uno 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 it is possible to change the upper end of their range using the AREF pin and the analog Reference() function. Additionally, some pins have specialized functionality:

I2C: 4 (SDA) and 5 (SCL).

Support I2C (TWI) communication using the Wire library.

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.

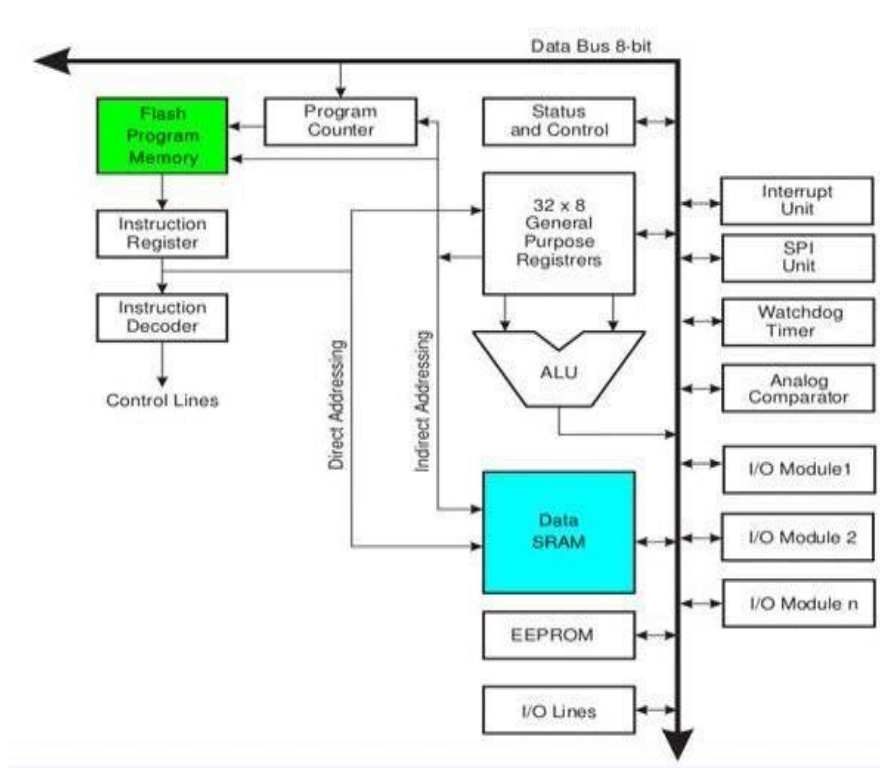


Fig 5.1 Input and Output data Transfer microcontroller

5.1.3 Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital

pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino 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 for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

5.1.4 Programming

The Arduino Uno can be programmed with the Arduino software. Select "Arduino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the reference and tutorials. The ATmega328 on the Arduino Uno comes preboned with a boot loader that allows you 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). You can also bypass the boot loader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details. The ATmega8U2 firmware source code is available. The ATmega8U2 is loaded with a DFU boot loader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU boot loader). See this user-contributed tutorial for more information. Arduino Uno

5.1.5 Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno 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 is connected to the reset line of the ATmega328 via a 100 nano-farad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the boot loader 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 boot loader 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 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 "RESET-EN". You may also be able to disable the auto-reset by connecting a 110ohm resistor from 5V to the reset line; see this forum thread for details.

5.1.6 USB Over current Protection

The Arduino Uno has a resettable polyfused that protects your computer's USB ports from shorts and over current. 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.

5.1.7 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.

5.2 RELAY:

A relay is an electrically operated switch. Many relays use an electromagnet to operate a switching mechanism mechanically, but other operating principles are also used. Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits, repeating the signal coming in from one circuit and re-transmitting it to another. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

A type of relay that can handle the high power required to directly control an electric motor or other loads is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from

overload or faults; in modern electric power systems these functions are performed by digital instruments still called "protective relays".

5.2.1 BASIC DESIGN AND OPERATION:

A simple electromagnetic relay consists of a coil of wire wrapped around a soft iron core, an iron yoke which provides a low reluctance path for magnetic flux, a movable iron armature, and one or more sets of contacts (there are two in the relay pictured). The armature is hinged to the yoke and mechanically linked to one or more sets of moving contacts. It is held in place by a spring so that when the relay is de-energized there is an air gap in the magnetic circuit. In this condition, one of the two sets of contacts in the relay pictured is closed, and the other set is open. Other relays may have more or fewer sets of contacts depending on their function. The relay in the picture also has a wire connecting the armature to the yoke. This ensures continuity of the circuit between the moving contacts on the armature, and the circuit track on the printed circuit board (PCB) via the yoke, which is soldered to the PCB.

When an electric current is passed through the coil it generates a magnetic field that activates the armature, and the consequent movement of the movable contact(s) either makes or breaks (depending upon construction) a connection with a fixed contact. If the set of contacts was closed when the relay was de-energized, then the movement opens the contacts and breaks the connection, and vice versa if the contacts were open. When the current to the coil is switched off, the armature is returned by a force, approximately half as strong as the magnetic force, to its relaxed position. Usually this force is provided by a spring, but gravity is also used commonly in industrial motor starters. Most relays are manufactured to operate quickly. In a low-voltage

application this reduces noise; in a high voltage or current application it reduces arcing.

When the coil is energized with direct current, a diode is often placed across the coil to dissipate the energy from the collapsing magnetic field at deactivation, which would otherwise generate a voltage spike dangerous to semiconductor circuit components. Some automotive relays include a diode inside the relay case. Alternatively, a contact protection network consisting of a capacitor and resistor in series (snubber circuit) may absorb the surge. If the coil is designed to be energized with alternating current (AC), a small copper "shading ring" can be crimped to the end of the solenoid, creating a small out- of-phase current which increases the minimum pull on the armature during the AC cycle.

5.2.2 APPLICATION:

Relays are used for:

- Amplifying a digital signal, switching a large amount of power with a small operating power. Some special cases are:
 - A telegraph relay, repeating a weak signal received at the end of a long wire
 - Controlling a high-voltage circuit with a low-voltage signal, as in some types of modems or audio amplifiers,
 - Controlling a high-current circuit with a low-current signal, as in the starter solenoid of an automobile,
- Detecting and isolating faults on transmission and distribution lines by opening and closing circuit breakers (protection relays)

- Vehicle battery isolation. A 12v relay is often used to isolate any second battery in cars, 4WDs, RVs and boats.
- Switching to a standby power supply.

5.3 LCD:

A liquid crystal display (LCD) is a flat panel display, electronic visual display, or video display that uses the light modulating properties of liquid crystals. Liquid crystals do not emit light directly. LCDs are available to display arbitrary images (as in a general-purpose computer display) or fixed images which can be displayed or hidden, such as preset words, digits, and 7-segment displays as in a digital clock. They use the same basic technology, except that arbitrary images are made up of a large number of small pixels, while other displays have larger elements. LCDs are used in a wide range of applications including computer monitors, televisions, instrument panels, aircraft cockpit displays, and signage. They are common in consumer devices such as video players, gaming devices, clocks, watches, calculators, and telephones, and have replaced cathode ray tube (CRT) displays in most applications. They are available in a wider range of screen sizes than CRT and plasma displays, and since they do not use phosphors, they do not suffer image burn-in. LCDs are, however, susceptible to image persistence.

The LCD screen is more energy efficient and can be disposed of more safely than a CRT. Its low electrical power consumption enables it to be used in battery-powered electronic equipment. It is an electronically modulated optical device made up of any number of segments filled with liquid crystals and arrayed in front of a light source (backlight) or reflector to produce images in color or monochrome. Liquid crystals were first discovered in

1888. By 2008, worldwide sales of televisions with LCD screens exceeded annual sales of CRT units; the CRT became obsolete for most purposes.



Fig 5.2 16x2 LCD Display

5.3.1 16x2 LCD:

A **16x2 LCD** means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data. The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD. Click to learn more about internal structure of a LCD.

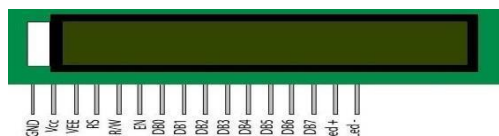


Fig 5.3 LCD pin details

5.3.2 PIN DESCRIPTION:

Pin No	Function	Name
1	Ground (0V)	Ground

2	Supply voltage; 5V (4.7V – 5.3V)	V _{CC}
3	Contrast adjustment; through a variable resistor	V _{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V _{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

Table 5.2 Specification of LCD Description

5.3.3 ADVANTAGES

- Very compact and light.
- Low power consumption. On average, 50-70% less energy is consumed than CRT monitors.
- No geometric distortion.

- The possible ability to have little or no flicker depending on backlight technology.
- Usually no refresh-rate flicker, as the LCD panel itself is usually refreshed at 200 Hz or more, regardless of the source refresh rate.
- Is very thin compared to a CRT monitor, which allows the monitor to be placed farther back from the user, reducing close-focusing related eye-strain.
- Razor sharp image with no bleeding/smearing when used at native resolution.
- Emits less electromagnetic radiation than a CRT monitor.
- Not affected by screen burn-in, though an identical but less severe phenomenon known as image persistence is possible.
- Can be made in almost any size or shape.
- No theoretical resolution limit.

5.4.4 SPECIFICATION:

Important factors to consider when evaluating an LCD:

Resolution versus range

Fundamentally resolution is the granularity (or number of levels) with which a performance feature of the display is divided. Resolution is often confused with range or the total end-to-end output of the display. Each of the major features of a display has both a resolution and a range that are tied to each other but very different. Frequently the range is an inherent limitation of the display while the resolution is a function of the electronics that make the display work.

ARDUINO

Arduino is a computer hardware and software company, project, and user community that designs and manufactures microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL),^[1] permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as do-it-yourself kits.

The project's board designs use a variety of microprocessors and controllers. These systems provide sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ("shields") and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, for loading programs from personal computers. The microcontrollers are mainly programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.

The Arduino project started in 2005 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats, and motion detectors

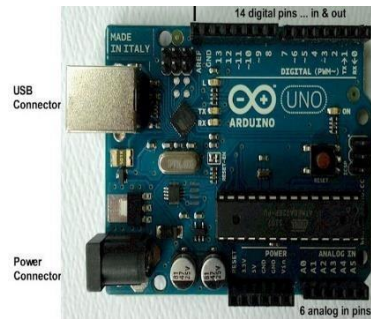


Fig 5.4 Diagram of Arduino Uno SMD R3

Developer	Arduino
Manufacturer	Many
Type	Single-board microcontroller
Operating system	None
CPU	Atmel AVR (8-bit), ARM Cortex-M0+ (32-bit), ARM Cortex-M3 (32-bit), Intel Quark (x86) (32-bit)
Memory	SRAM
Storage	Flash, EEPROM

5.4 POWER SUPPLIES

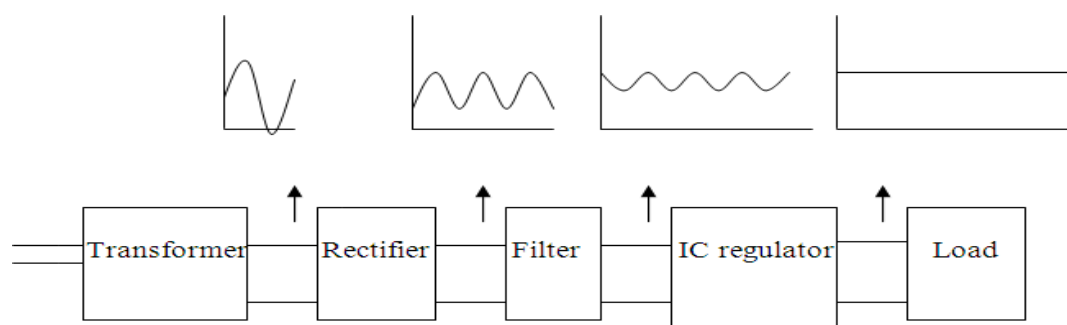
5.4.1 INTRODUCTION

The present chapter introduces the operation of power supply circuits built using filters, rectifiers, and then voltage regulators. Starting with an ac voltage, a

steady dc voltage is obtained by rectifying the ac voltage, then

filtering to a dc level, and finally, regulating to obtain a desired fixed dc voltage. The regulation is usually obtained from an IC voltage regulator unit, which takes a dc voltage and provides a somewhat lower dc voltage, which remains the same even if the input dc voltage varies, or the output load connected to the dc voltage changes.

A block diagram containing the parts of a typical power supply and the voltage at various points in the unit . The ac voltage, typically 120 V rms, is connected to a transformer, which steps that ac voltage down to the level for the desired dc output. A diode rectifier then provides a full-wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation. A regulator circuit can use this dc input to provide a dc voltage that not only has much less ripple voltage but also remains the same dc value even if the input dc voltage varies somewhat, or the load connected to the output dc voltage changes. This voltage regulation is usually obtained using one of a number of popular voltage regulator IC units.



5.4.2 IC VOLTAGE REGULATORS

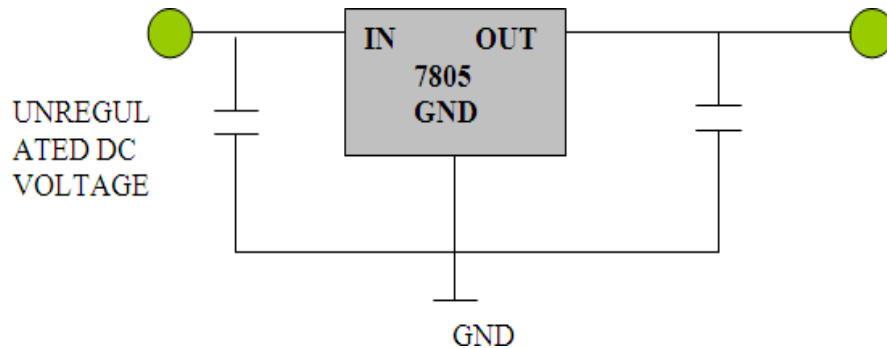
Voltage regulators comprise a class of widely used ICs. Regulator IC units contain the circuitry for reference source, comparator amplifier, control device, and overload protection all in a single IC. Although the internal construction of the IC is somewhat different from that described for discrete voltage regulator circuits, the external operation is much the same. IC units provide regulation of either a fixed positive voltage, a fixed negative voltage, or an adjustably set voltage.

A power supply can be built using a transformer connected to the ac supply line to step the ac voltage to desired amplitude, then rectifying that ac voltage, filtering with a capacitor and RC filter, if desired, and finally regulating the dc voltage using an IC regulator. The regulators can be selected for operation with load currents from hundreds of mill amperes to tens of amperes, corresponding to power ratings from mill watts to tens of watts.

5.4.3 THREE-TERMINAL VOLTAGE REGULATORS

The fixed voltage regulator has an unregulated dc input voltage, V_i , applied to one input terminal, a regulated output dc voltage, V_o , from a second terminal, with the third terminal connected to ground. For a selected regulator, IC device specifications list a voltage range over which the input voltage can vary to maintain a regulated output voltage over a range of load current. The specifications also list the amount of output voltage change resulting from a change in load current (load regulation) or in input voltage (line regulation).

5.4.4 Fixed Positive Voltage Regulators:



The series 78 regulators provide fixed regulated voltages from 5 to 24 V. Figure 19.26 shows how one such IC, a 7812, is connected to provide voltage regulation with output from this unit of +12V dc. An unregulated input voltage V_i is filtered by capacitor C1 and connected to the IC's IN terminal. The IC's OUT terminal provides a regulated + 12V which is filtered by capacitor C2 (mostly for any high-frequency noise). The third IC terminal is connected to ground (GND). While the input voltage may vary over some permissible voltage range, and the output load may vary over some acceptable range, the output voltage remains constant within specified voltage variation limits. These limitations are spelled out in the manufacturer's specification sheets.

Positive Voltage Regulators in 7800 series

IC Part	Output Voltage (V)	Minimum V_i (V)
7805	+5	7.3

7806	+6	8.3
7808	+8	10.5
7810	+10	12.5
7812	+12	14.6
7815	+15	17.7
7818	+18	21.0
7824	+24	27.1

Table 5.3 Positive Voltage Regulator in 7800 Series

5.5 VIBRATION SENSOR

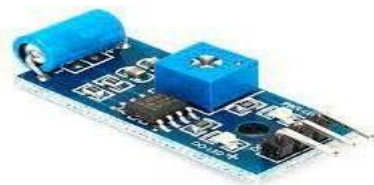


Fig 5.5 Vibration Sensor

The measurement of vibrations should be possible utilizing different sorts of sensors. In spite of the fact that there are no immediate vibration sensors, vibrations can be estimated in a roundabout way, deriving estimates from exemplary mechanical or optical amounts. These sensors contrast in certain highlights. In addition to other things they can be isolated in view of dynamic and latent conduct, there are sensors that action relative and others outright. Other particular highlights are recurrence range, signal elements and the nature of the estimation information. The accompanying sensors displayed here were first organized in a reaching and a non-reaching bunch and inside these in the sub things way, speed and speed increase measurement.

5.5.1 NodeMCU

The NodeMCU (*Node Microcontroller Unit*) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Expressive Systems, contains the crucial elements of a computer: CPU, RAM, networking (Wi-Fi), and even a modern operating system and SDK. That makes it an excellent choice for the Internet of Things (IoT) projects of all kinds.

However, as a chip, the ESP8266 is also hard to access and use. You must solder wires, with the appropriate analog voltage, to its pins for the simplest tasks such as powering it on or sending a keystroke to the “computer” on the chip. You also have to program it in low-level machine instructions that can be interpreted by the chip hardware. This level of integration is not a problem using the ESP8266 as an embedded controller chip in mass-produced electronics. It is a huge burden for hobbyists, hackers, or students who want to experiment with it in their own IoT projects.

But, what about Arduino? The Arduino project created an open-source hardware design and software SDK for their versatile IoT controller. Similar to NodeMCU, the Arduino hardware is a microcontroller board with a USB connector, LED lights, and standard data pins. It also defines standard interfaces to interact with sensors or other boards. But unlike NodeMCU, the Arduino board can have different types of CPU chips (typically an ARM or Intel x86 chip) with memory chips, and a variety of programming environments. There is an Arduino reference design for the ESP8266 chip as well. However, the flexibility of Arduino also means significant variations across different vendors. For example, most Arduino boards do not have WiFi capabilities, and some even have a serial data port instead of a USB port

5.5.2 USB to Serial Converter - CP2102 or CH340G

Incorporated into each NodeMCU is a USB to Serial Converter. The official design is based on the CP2102 chipset and offers the best compatibility. Genuine boards use the CP2102 chipset including the officially licensed Amica NodeMCU modules. The other common USB to Serial Converter used is the CH340G which is common on the lower-priced modules including the LoLin units. Other designs may use drivers including the FTDI chipset, but those designs are

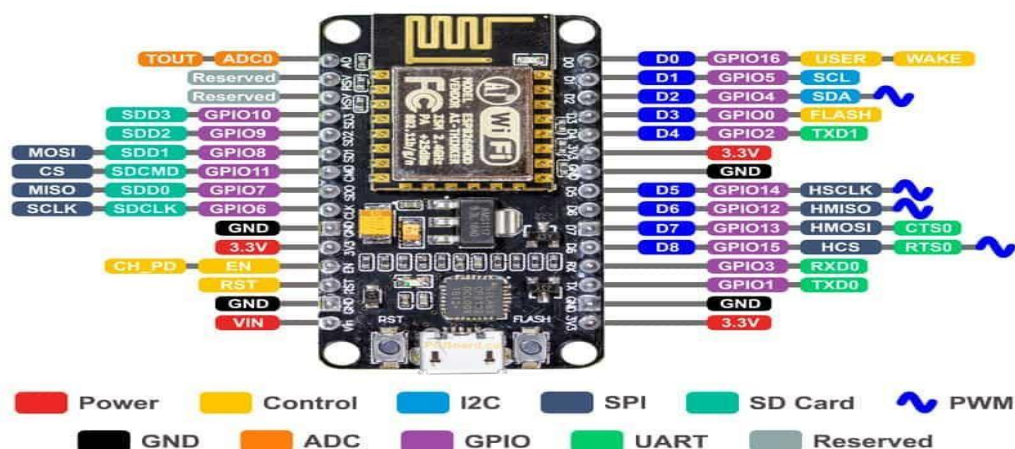


Fig 5.6 USB to Serial Converter

Power Pins There are four power pins. **VIN** pin and three **3.3V** pins.

- **VIN** can be used to directly supply the NodeMCU/ESP8266 and its peripherals. Power delivered on **VIN** is regulated through the onboard regulator on the NodeMCU module – you can also supply 5V regulated to the **VIN** pin
- **3.3V** pins are the output of the onboard voltage regulator and can be used to supply power to external components.

GND are the ground pins of NodeMCU/ESP8266

I2C Pins are used to connect I2C sensors and peripherals. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

GPIO Pins NodeMCU/ESP8266 has 17 GPIO pins which can be assigned to functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel The NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

UART Pins NodeMCU/ESP8266 has 2 UART interfaces (UART0 and UART1) which provide asynchronous communication (RS232 and RS485), and can

communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

SPI Pins NodeMCU/ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:


- 4 timing modes of the SPI format transfer
- Up to 80 MHz and the divided clocks of 80 MHz
- Up to 64-Byte FIFO

SDIO Pins NodeMCU/ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

PWM Pins The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 μ s to 10000 μ s (100 Hz and 1 kHz).

Control Pins are used to control the NodeMCU/ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

- **EN:** The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
- **RST:** RST pin is used to reset the ESP8266 chip.
- **WAKE:** Wake pin is used to wake the chip from deep-sleep.

 Control Pins are used to control the NodeMCU/ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

- **EN:** The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
- **RST:** RST pin is used to reset the ESP8266 chip.
- **WAKE:** Wake pin is used to wake the chip from deep-sleep.

Depending on the Operating System you are using with the NodeMCU, the appropriate driver must be installed. Generally, Windows 10 immediately recognizes the CP2102 chipset while the CH340G may require separate installation.

WCH maintain and update the drivers for the CH340G on a regular basis. Versions of the driver are also available for Windows, Mac, Linux, and Android.

5.6 GAS SENSOR

Each Carbon Dioxide Sensor/ Co2 Sensor monitors any change in the concentration of carbon dioxide. They can be used in a wide variety of applications.



Fig 5.7 Gas sensor

Features

This is a simple to use Gas Sensor Module which can not only sense the presence gases but also their concentration in air. Depending on the sensor used, the module can detect LPG, CNG, CO or Alcohol in the air. It simplifies interface to the odd pin spacing of the sensor and provides interface through 4x 0.1" header pins. It provides both an analog output corresponding to the concentration of the gases in the air and an easy to use digital output indicating the presence of the gas. The onboard potentiometer can be used to set the maximum gas concentration beyond which the digital output gets triggered. An onboard LED indicates the presence of any gas. The digital output can be easily interfaced to microcontrollers and other circuits to detect the presence of the gas. The analog output can be hooked up to an ADC of a microcontroller to detect the concentration of the gas in the air.

Using the Modules

Once the module is powered up, you will have to calibrate the module for the specific environment it will be used in. To calibrate the module, you will have to set the potentiometer by turning its knob by hand or a screw driver. You will have to power the module and rotate the knob of the potentiometer until the output of the module changes from high to low. The potentiometer sets the maximum gas concentration threshold beyond which the digital output of the module gets triggered. When the gas concentration is less than the threshold, the D out pin will have a low (0V) output. When the gas concentration crosses the threshold, the D out pin goes high (5V). The onboard output signal LED indicates the presence of the gas.

The gas sensors being developed by the Chemical Species Gas Sensors team at NASA Glenn were designed with NASA's primary objective of advancing aeronautic and aerospace technology in mind but can be used in a

variety of commercial applications. For example, the hydrogen sensors were originally developed for use on the launch pad of the space shuttle. However, these sensors have been applied to the automotive industry through an interaction with GenCorp Aerojet Corporation. In conjunction with NASA Marshall Space Flight Centre, GenCorp previously developed hardware and software to monitor and control the NASA Glenn/CWRU sensors. The system can be customized to fit the user's needs (e.g. to monitor and display the condition of the tank of a natural gas vehicle). Several of these systems have been purchased for use on the Ford Motor Company assembly line for natural gas vehicles (NGV). It is this complete system that has received the 1995 R&D 100 Award

Makel Engineering, in an STTR is commercializing a next generation hydrogen sensor and oxygen sensor technology for use in space based leak monitoring applications and as well as a variety of commercial applications. Makel Engineering has a second STTR to commercialize the high temperature gas sensors for emission sensing applications. Other applications for the sensors being developed include fuel cell monitoring, combustion process and catalytic reactor monitoring, alarms for high-temperature pressure vessels and piping, polymer production, and volatile organics detection. NASA Glenn, Makel Engineering, Inc., Case Western Reserve University, and NASA Kennedy recently received a Turning Goals into Reality Award for developing and application of Smart Leak Sensor Technology in applications, as varied as the NASA Helices Vehicle, the X43, and the Ford U car.

5.7 DC MOTOR

A **DC motor** is any of a class of rotary electrical motors that converts direct current electrical energy into mechanical energy. The most common types rely

on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current in part of the motor.



Fig 5.8 DC Motor

DC motors were the first form of motor widely used, as they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The universal motor can operate on direct current but is a lightweight brushed motor used for portable power tools and appliances. Larger DC motors are currently used in propulsion of electric vehicles, elevator and hoists, and in drives for steel rolling mills. The advent of power electronics has made replacement of DC motors with AC motors possible in many applications.

A well-known and suitable motor driver is IC L298 which can be used to control two motors. It is a high voltage, high-current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as DC and stepping motors [3]. Two enable inputs of 5 volts are provided to

enable or disable the device independently of the input signal. L298 has 2 amperes per channel current capacity and it can support from 3 volts to 30 volts for outputting. Moreover, L298 works well up to 16 volts without any heat sink.

CHAPTER 6

SYSTEM REQUIREMENTS

Hardware Requirements:

- CPU type : Intel Pentium 4
- Clock speed : 3.0 GHz
- Ram size : 512 MB
- Hard disk capacity : 40 GB
- Monitor type : 15 Inch color monitor
- Keyboard type : internet keyboard

Software Requirements:

- Operating System : Windows OS
- Language : PHP, Embedded, MATLAB

6.1 Embedded C

An embedded system is an application that contains at least one programmable computer (typically in the form of a microcontroller, a microprocessor or digital signal processor chip) and which is used by individuals who are, in the main, unaware that the system is computer-based.

6.1.1 Introduction

Looking around, we find ourselves to be surrounded by various types of embedded systems. Be it a digital camera or a mobile phone or a washing machine, all of them has some kind of processor functioning inside it. Associated with each processor is the embedded software. If hardware forms the

body of an embedded system, embedded processor acts as the brain, and embedded software forms its soul. It is the embedded software which primarily governs the functioning of embedded systems.

During infancy years of microprocessor based systems, programs were developed using assemblers and fused into the EPROMs. There used to be no mechanism to find what the program was doing. LEDs, switches, etc. were used to check correct execution of the program. Some ‘very fortunate’ developers had In-circuit Simulators (ICEs), but they were too costly and were not quite reliable as well.

As time progressed, use of microprocessor-specific assembly-only as the programming language reduced and embedded systems moved onto C as the embedded programming language of choice. C is the most widely used programming language for embedded processors/controllers. Assembly is also used but mainly to implement those portions of the code where very high timing accuracy, code size efficiency, etc. are prime requirements.

Initially C was developed by Kernighan and Ritchie to fit into the space of 8K and to write (portable) operating systems. Originally it was implemented on UNIX operating systems. As it was intended for operating systems development, it can manipulate memory addresses. Also, it allowed programmers to write very compact codes. This has given it the reputation as the language of choice for hackers too.

As assembly language programs are specific to a processor, assembly language didn’t offer portability across systems. To overcome this

disadvantage, several high level languages, including C, came up. Some other languages like PLM, Modula-2, Pascal, etc. also came but couldn't find wide acceptance. Amongst those, C got wide acceptance for not only embedded systems, but also for desktop applications. Even though C might have lost its sheen as mainstream language for general purpose applications, it still is having a strong-hold in embedded programming. Due to the wide acceptance of C in the embedded systems, various kinds of support tools like compilers & cross-compilers, ICE, etc. came up and all this facilitated development of embedded systems using C. Subsequent sections will discuss what is Embedded C, features of C language, similarities and difference between C and embedded C, and features of embedded C programming.

6.1.2 EMBEDDED SYSTEMS PROGRAMMING

Embedded systems programming is different from developing applications on a desktop computers. Key characteristics of an embedded system, when compared to PCs, are as follows. Embedded devices have resource constraints (limited ROM, limited RAM, limited stack space, less processing power). Components used in embedded system and PCs are different; embedded systems typically use smaller, less power consuming components. Embedded systems are more tied to the hardware.

Two salient features of Embedded Programming are code speed and code size. Code speed is governed by the processing power, timing constraints, whereas code size is governed by available program memory and use of programming language. Goal of embedded system programming is to get maximum features in minimum space and minimum time.

Embedded systems are programmed using different type of language

- Machine Code
- Low level language, i.e., assembly
- High level language like C, C++, Java, Ada, etc.
- Application level language like Visual Basic, scripts, Access, etc.

Assembly language maps mnemonic words with the binary machine codes that the processor uses to code the instructions. Assembly language seems to be an obvious choice for programming embedded devices. However, use of assembly language is restricted to developing efficient codes in terms of size and speed. Also, assembly codes lead to higher software development costs and code portability is not there. Developing small codes are not much of a problem, but large programs/projects become increasingly difficult to manage in assembly language. Finding good assembly programmers has also become difficult nowadays. Hence high level languages are preferred for embedded systems programming.

Use of C in embedded systems is driven by following advantages it is small and reasonably simpler to learn, understand, program and debug. C Compilers are available for almost all embedded devices in use today, and there is a large pool of experienced C programmers.

Unlike assembly, C has advantage of processor-independence and is not specific to any particular microprocessor/ microcontroller or any system. This makes it convenient for a user to develop programs that can run on most of the systems. As C combines functionality of assembly language and features of high level languages, C is treated as a ‘middle-level computer language’ or

‘high level assembly language’. It is fairly efficient. It supports access to I/O and provides ease of management of large embedded projects.

Many of these advantages are offered by other languages also, but what sets C apart from others like Pascal, FORTRAN, etc. is the fact that it is a middle level language; it provides direct hardware control without sacrificing benefits of high level languages. Compared to other high level languages, C offers more flexibility because C is relatively small, structured language; it supports low-level bit-wise data manipulation.

Compared to assembly language, C Code written is more reliable and scalable, more portable between different platforms (with some changes). Moreover, programs developed in C are much easier to understand, maintain and debug. Also, as they can be developed more quickly, codes written in C offers better productivity. C is based on the philosophy ‘programmers know what they are doing’; only the intentions are to be stated explicitly. It is easier to write good code in C & convert it to an efficient assembly code (using high quality compilers) rather than writing an efficient code in assembly itself. Benefits of assembly language programming over C are negligible when we compare the ease with which C programs are developed by programmers.

Object oriented language, C++ is not apt for developing efficient programs in resource constrained environments like embedded devices. Virtual functions & exception handling of C++ are some specific features that are not efficient in terms of space and speed in embedded systems. Sometimes C++ is used only with very few features, very much as C.

Ada, also an object-oriented language, is different than C++. Originally designed by the U.S. DOD, it didn’t gain popularity despite being accepted as an international standard twice (Ada83 and Ada95). However, Ada language has many features that would simplify embedded software development.

Java is another language used for embedded systems programming. It primarily finds usage in high-end mobile phones as it offers portability across systems and is also useful for browsing applications. Java programs require Java Virtual Machine (JVM), which consume lot of resources. Hence it is not used for smaller embedded devices. Dynamic C and B# are some proprietary languages which are also being used in embedded applications. Efficient embedded C programs must be kept small and efficient; they must be optimized for code speed and code size. Good understanding of processor architecture embedded C programming and debugging tools facilitate this.

6.1.3 Difference between C and embedded C:

Though C and embedded C appear different and are used in different contexts, they have more similarities than the differences. Most of the constructs are same; the difference lies in their applications.

C is used for desktop computers, while embedded C is for microcontroller based applications. Accordingly, C has the luxury to use resources of a desktop PC like memory, OS, etc. While programming on desktop systems, we need not bother about memory. However, embedded C has to use with the limited resources (RAM, ROM, I/Os) on an embedded processor. Thus, program code must fit into the available program memory. If code exceeds the limit, the system is likely to crash.

Compilers for C (ANSI C) typically generate OS dependent executables. Embedded C requires compilers to create files to be downloaded to the microcontrollers/microprocessors where it needs to run. Embedded compilers give access to all resources which is not provided in compilers for desktop computer applications. Embedded systems often have the real-time constraints, which is usually not there with desktop computer applications.

Embedded systems often do not have a console, which is available in case of desktop applications. So, what basically is different while programming with embedded C is the mindset; for embedded applications, we need to optimally use the resources, make the program code efficient, and satisfy real time constraints, if any. All this is done using the basic constructs, syntaxes, and function libraries of 'C'.

6.2 PHP

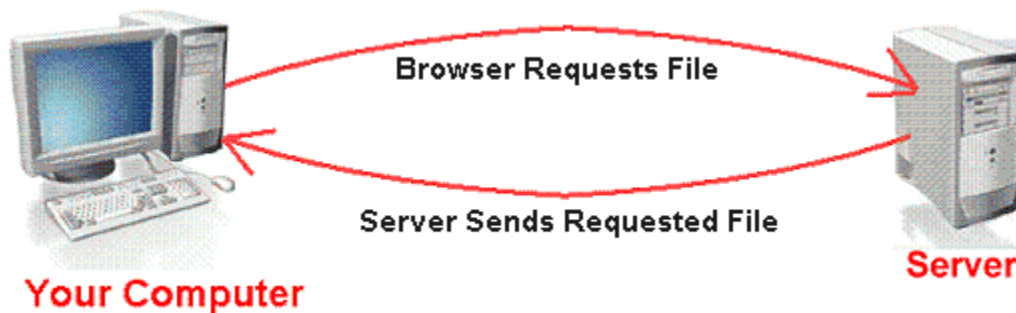
6.2.1 FEATURES OF PHP

PHP: Hypertext Preprocessor (the name is a recursive acronym) is a widely used, general-purpose scripting language that was originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. As a general-purpose programming language, PHP code is processed by an interpreter application in command-line mode performing desired operating system operations and producing program output on its standard output channel. It may also function as a graphical application. PHP is available as a processor for most modern web servers and as standalone interpreter on most operating systems and computing platforms.

PHP was originally created by Rasmus Lerdorf in 1995[1] and has been in continuous development ever since. The main implementation of PHP is now produced by The PHP Group and serves as the de facto standard for PHP as there is no formal specification. PHP is free software released under the PHP License, which is incompatible with the GNU General Public License (GPL) because restrictions exist regarding the use of the term PHP.

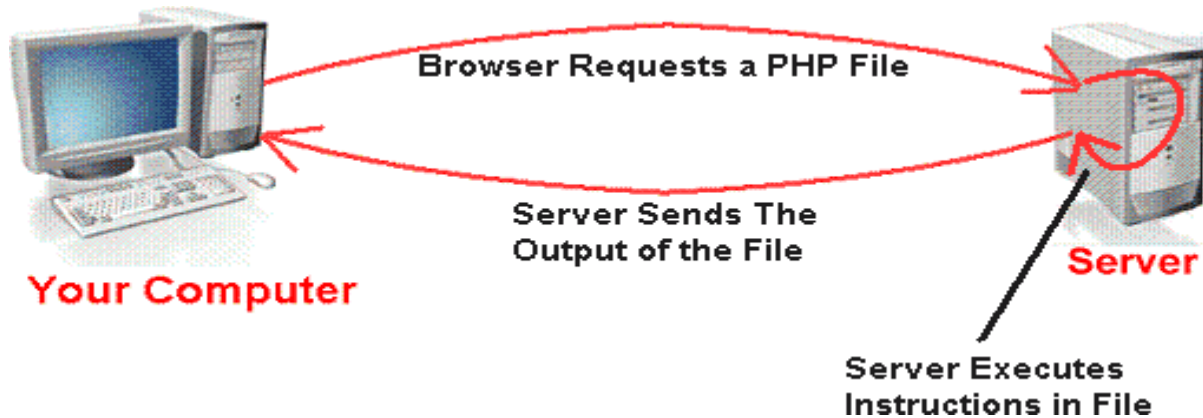
Hypertext refers to files linked together using hyperlinks, such as HTML (Hypertext Markup Language) files. Preprocessing is executing instructions that modify the output. Below is a demonstration of the difference between HTML and PHP files.

6.2.2 Accessing an HTML Page



1. Your browser sends a request to that web page's server (computer) for the file (HTML or image) you wish to view.
2. The web server (computer) sends the file requested back to your computer.
3. Your browser displays the file appropriately.
4. If you request a PHP file (ends with ".php"), the server handles it differently.

6.2.3 Accessing a PHP Page



1. Your browser sends a request to that web page's server for the PHP file you wish to view.
2. The web server calls PHP to interpret and perform the operations called for in the PHP script.
3. The web server sends the output of the PHP program back to your computer.
4. Your browser displays the output appropriately.

PHP originally stood for personal home page. Its development began in 1994 when the Danish/Greenlandic programmer Rasmus Lerdorf initially created a set of Perl scripts he called 'Personal Home Page Tools' to maintain his personal homepage, including tasks such as displaying his résumé and recording how much traffic his page was receiving. He rewrote these scripts as C programming language Common Gateway Interface (CGI) binaries,

extending them to add the ability to work with web forms and to communicate with databases and called this implementation 'Personal Home Page/Forms Interpreter' or PHP/FI. PHP/FI could be used to build simple, dynamic web applications. Lerdorf released PHP/FI as 'Personal Home Page Tools (PHPTools) version 1.0' publicly on June 8, 1995, to accelerate bug location and improve the code. This release already had the basic functionality that PHP has today. This included Perl-like variables, form handling, and the ability to embedHTML. The syntax was similar to Perl but was more limited and simpler, although less consistent. A development team began to form and, after months of work and beta testing, officially released PHP/FI 2 in November 1997.

A new major version has been under development alongside PHP 5 for several years. This version was originally planned to be released as PHP 6 as a result of its significant changes, which included plans for full Unicode support. However, Unicode support took developers much longer to implement than originally thought, and the decision was made in March 2010[13] to move the project to a branch, with features still under development moved to trunk.

Changes in the new code include the removal of register global, magic quotes, and safe mode. The reason for the removals was that register global had given way to security holes, and the use of magic quotes had an unpredictable nature, and was best avoided. Instead, to escape characters, magic quotes may be replaced with the add slashes () function, or more appropriately an escape mechanism specific to the database vendor itself like mysql_real_escape_string () for MySQL. Functions that will be removed in future versions and have been deprecated in PHP 5.3 will produce a warning if used.

PHP currently does not have native support for Unicode or multibyte strings; Unicode support is under development for a future version of PHP and

will allow strings as well as class, method, and function names to contain non-ASCII characters.

PHP interpreters are available on both 32-bit and 64-bit operating systems, but on Microsoft Windows the only official distribution is a 32-bit implementation, requiring Windows 32-bit compatibility mode while using Internet Information Services (IIS) on a 64-bit Windows platform. As of PHP 5.3.0, experimental 64-bit versions are available for MS Windows.

6.2.4 Security

The PHP interpreter only executes PHP code within its delimiters. Anything outside its delimiters is not processed by PHP (although non-PHP text is still subject to control structures described within PHP code). The most common delimiters are `<?php` to open and `?>` to close PHP sections. `<script language="php">` and `</script>` delimiters are also available, as are the shortened forms `<?` or `<?=` (which is used to echo back a string or variable) and `?>` as well as ASP-style short forms `<%` or `<%=` and `%>`. While short delimiters are used, they make script files less portable as support for them can be disabled in the PHP configuration, and so they are discouraged. The purpose of all these delimiters is to separate PHP code from non-PHP code, including HTML.

The first form of delimiters, `<?php` and `?>`, in XHTML and other XML documents, creates correctly formed XML 'processing instructions'. This means that the resulting mixture of PHP code and other markup in the server-side file is itself well-formed XML.

Variables are prefixed with a dollar symbol and a type does not need to be specified in advance. Unlike function and class names, variable names are case sensitive. Both double-quoted ("") and heredoc strings allow the ability to

embed a variable's value into the string. PHP treats newlines as whitespace in the manner of a free-form language (except when inside string quotes), and statements are terminated by a semicolon. PHP has three types of comment syntax: `/* */` marks block and inline comments; `//` as well as `#` are used for one-line comments. The echo statement is one of several facilities PHP provides to output text (e.g. to a web browser).

In terms of keywords and language syntax, PHP is similar to most high level languages that follow the C style syntax. if conditions, for and while loops, and function returns are similar in syntax to languages such as C, C++, Java and Perl.

6.2.5 Benefit of PHP

Because the server does processing, the output of PHP files changes when its input changes. For example, most of the pages on the Horticulture site have only two (2) PHP commands:

1. Include the header file that defines the links on the left, the banner, and the quick links at the top.
2. Include the footer file that displays the mission statement and Horticulture contact information.

Because including the files is performed every time the PHP file is accessed, when the header/footer files change, the new content will be immediately updated. In other words, if you add a new link, every page that includes the header will immediately display the new link.

PHP has become the most popular Web programming language not only because it is free. PHP is a full-fledged programming language (unlike HTML for example, which is more of a presentation means) and many complex

applications can be written in it. Another benefit of applications written in PHP is that they are fast and if written properly, they could be pretty secure. There are also tons of ready PHP scripts and functions, which you can customize to your liking and use in your PHP applications.

Web development is becoming a more prosperous industry lately. Since the Internet and the computer wave in general are becoming quite lucrative, web development is becoming a booming industry in which everyone wants to be a part. However, it is also a very competitive industry since there are many professionals that are quite adept at programming. Therefore, learning proper PHP development strategies is beneficial. If you already have prior knowledge of computer language and coding, PHP should come as a second nature to you. The fact that it can be used in its most primal forms for basic programming as well as incredibly advanced programming only adds to its possible potential for programmers.

Before building a website you need to know which language you are going to use in a professional looking website. PHP is one of the best and easy to use programming languages as it can be run on any operating system. PHP is a free language so that is the huge advantage of this language. For handling database connections, formatting dates, editing strings, handling emails and all PHP can be very useful. It can be easily extended for some specific functions that you would like to add in your website. Reliability of this language is extraordinary as PHP already runs on millions of servers around the world, which means that it's powerful enough for even the most demanding situations. It provides web developers much more liberation in creating websites with some of the outstanding features and they can use regular elements frequently. PHP can be very much successful for creating Dynamic Websites. PHP programmers with the use of open source codes benefit from the flexibility of editing, modifying and updating the source code when there is a mandatory.

PHP is based on C++ programming language and the syntax used in PHP is fairly similar to C, C++. There is huge community of developers who still believes that C/C++ is still the best programming language. For every website to get reasonable progress it can be use Content Management System such as Joomla, Word Press etc. here PHP and MySQL are very helpful in successful CMS running. There are so many IT companies which provide best quality

PHP web development work from India. The reason for Outsourcing PHP development to India is that it is very cost effective with better quality. In the professional field of Web and software development services we have achieved great amount of victory with skilled and experienced PHP programmers.

PHP has been supported by almost every hosting company. PHP can be the best choice to run an application on Linux based hosting platforms. PHP is a server side scripting language originally designed to build dynamic websites. Modern web 2.0 applications are largely characterized by mashups and desktop style user interfaces. php is an excellent choice for interacting with other websites and providing rich user experience. Simple php commands like curl or fopen allow you to grab data from other websites with relative ease. Php works well with javascript so you can provide your end users with modern, responsive interfaces that are way beyond the old static interfaces of days past.

6.3 BACK END SOFTWARE

6.3.1 MySQL Introduction

The MySQL® database has become the world's most popular open source database because of its consistent fast performance, high reliability and ease of use. It's used on every continent -- Yes, even Antarctica! -- by individual Web developers as well as many of the world's largest and fastest-growing organizations to save time and money powering their high-volume Web sites, business-critical systems and packaged software -- including industry leaders such as Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, and Zappos.com.

Not only is MySQL the world's most popular open source database, it's also become the database of choice for a new generation of applications built on the LAMP stack (Linux, Apache, MySQL, PHP / Perl / Python.) MySQL runs on more than 20 platforms including Linux, Windows, Mac OS, Solaris, HP- UX, IBM AIX, giving you the kind of flexibility that puts you in control.

Whether you're new to database technology or an experienced developer or DBA, MySQL offers a comprehensive range of certified software, support, training and consulting to make you successful.

MySQL can be built and installed manually from source code, but this can be tedious so it is more commonly installed from a binary package unless special customizations are required. On most Linux distributions the package management system can download and install MySQL with minimal effort, though further configuration is often required to adjust security and optimization settings.

Though MySQL began as a low-end alternative to more powerful proprietary databases, it has gradually evolved to support higher-scale needs as well. It is still most commonly used in small to medium scale single-server

deployments, either as a component in a LAMP based web application or as a standalone database server. Much of MySQL's appeal originates in its relative simplicity and ease of use, which is enabled by an ecosystem of open source tools such as phpMyAdmin. In the medium range, MySQL can be scaled by deploying it on more powerful hardware, such as a multi-processor server with gigabytes of memory.

There are however limits to how far performance can scale on a single server, so on larger scales, multi-server MySQL deployments are required to provide improved performance and reliability. A typical high-end configuration can include a powerful master database which handles data write operations and is replicated to multiple slaves that handle all read operations.[18] The master server synchronizes continually with its slaves so in the event of failure a slave can be promoted to become the new master, minimizing downtime. Further improvements in performance can be achieved by caching the results from database queries in memory using memcached, or breaking down a database into smaller chunks called shards which can be spread across a number of distributed server clusters.

6.4 MATLAB

6.4.1 INTRODUCTION

MATLAB, which stands for Matrix Laboratory, is a software package developed by Math Works, Inc. to facilitate numerical computations as well as some symbolic manipulation. The collection of programs (primarily in FORTRAN) that eventually became MATLAB were Developed in the late 1970s by Cleve Moler, who used them in a numerical analysis course, he was teaching at the University of New Mexico. Jack Little and Steve Bangert later reprogrammed these routines in C, and added M-files, toolboxes, and more powerful graphics (original versions created plots by printing asterisks on the screen). Moler, Little, and Bangert founded Math Works in California in 1984.

What is MATLAB?

- MATLAB (“Matrix Laboratory”) is a tool for numerical computation and visualization. The basic data element is a matrix, so if you need a program that manipulates array-based data it is generally fast to write and run in MATLAB (unless you have very large arrays or lots of computations, in which case you’re better off using C or Fortran).
- High level language for technical computing
- Stands for **Matrix Laboratory**
- Everything is a matrix - easy to do linear algebra

Using MATLAB

The best way to learn to use MATLAB is to sit down and try to use it. In this handout are a few examples of basic MATLAB operations, but after you’ve gone through this tutorial you will probably want to learn more. Check out the “Other Resources” listed at the end of this handout. The Beginning When you start MATLAB, the command prompt “>>” appears. You will tell MATLAB

What to do by typing commands at the prompt.

Creating matrices

The basic data element in MATLAB is a matrix. A scalar in MATLAB is a 1x1 matrix, and a vector is a 1xn (or nx1) matrix.

Advanced operations

There's a lot more that you can do with MATLAB than is listed in this handout. Check out the MATLAB help or one of the "Other Resources" if you want to learn more about the following more advanced tools:

- Numerical integration (quad)
- Discrete Fourier transform (fft, ifft)
- Statistics (mean, median, std, var)
- Curve fitting (cftool)
- Signal processing (sptool)
- Numerical integration of systems of ODEs (ode45)

M-files and functions

If you are doing a computation of any significant length in MATLAB, you will probably want to make an m-file. Anything that you would type at the command prompt you can put in the m-file (for example, "script.m") and then run it all at once (by typing the name of the m-file, e.g. "script", at the command prompt). You can even add comments to your m-file, by putting a "%" at the beginning of a comment line.

File I/O

MATLAB allows you to save matrices and read them in later. The simplest way to do this is using the commands “save” and “load”. Typing in “save A” saves matrix A to a file called Asmat. If you want to read in matrix A later, just type “load A”. You can also use the load command to read in ASCII files, as long as they are formatted correctly. Formatted correctly means that the number of columns in each line is the same and the columns are delimited with a space. Suppose you have a file called “datafile.dat” that contains the following lines:

```
12.5 6 9
```

```
1 3.5 125
```

```
2 4 0
```

You can put multiple individual plots in the same figure window using the “subplot”

Command. Type “help subplot” for more information. Once you’re done with your plot,

you’ll probably want to label the axes:

```
>> xlabel('x')
```

```
>> ylabel('y')
```

You can also give it a title:

```
>> title('My plot')
```

Now, let’s do a 3-D example. First, generate some sample data:

```
>> z = peaks;
```


The “peaks” command generates a sample function. The default is a 49x49 matrix, but

you can specify a different size. “Peaks” can be very useful if you want to test your

plotting script or just play around with making plots.

We can make a 3-D shaded surface plot using the “surf” command:

```
>> surf(z)
```

OUTLINE:

- Introduction and where to get MATLAB
- Data structure: matrices, vectors and operations
- Basic line plots
- File I/O.

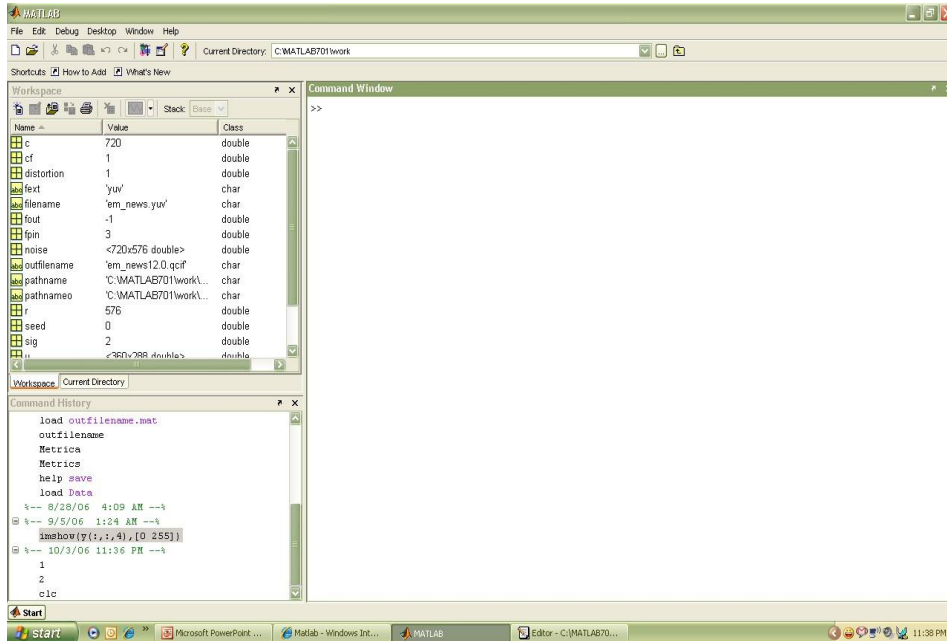
The MATLAB System:

- Development Environment
- Mathematical Function Library
- MATLAB language
- Application Programming Language.

Creating Matrices:

- zeros(m, n): matrix with all zeros
- ones(m, n): matrix with all ones.
- eye(m, n): the identity matrix
- rand(m, n): uniformly distributed random
- randn(m, n): normally distributed random

- `magic(m)`: square matrix whose elements have the same sum, along the row, column and diagonal.
- `Pascal(m)` : Pascal matrix.



Basic Mathematical Operations

Addition: `>> C = A + B`

Subtraction: `>> D = A - B`

Multiplication: `>> E = A * B` (Matrix multiplication)

`>> E = A .* B` (Element wise multiplication)

Division: Left Division and Right Division

`>> F = A ./ B` (Element wise division)

`>> F = A / B` ($A * \text{inverse of } B$)

>> $F = A ./ B$ (Element wise division)

>> $F = A \setminus B$ (inverse of $A * B$)

MATLAB consists of:

The MATLAB language

- A high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features.

The MATLAB working environment

- The set of tools and facilities that you work with as the MATLAB user or programmer, including tools for developing, managing, debugging, and profiling

Handle Graphics

- The MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics.

The MATLAB function library.

- A vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix Eigen values, Bessel functions, and fast Fourier transforms as well as special image processing related functions

The MATLAB Application Program Interface (API)

- A library that allows you to write C and Fortran programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

Starting and Quitting MATLAB

To start MATLAB click on the MATLAB icon or type in MATLAB, followed by pressing the enter or return key at the system prompt. The screen will produce the MATLAB **prompt** `>>` (or `EDU >>`), which indicates that MATLAB is waiting for a command to be entered.

- In order to quit MATLAB, type **quit** or **exit** after the prompt, followed by pressing the enter or return key.

Display Windows

MATLAB has three display windows. They are

1. A Command Window which is used to enter commands and data to display plots and graphs.
2. A Graphics Window which is used to display plots and graphs.
3. An Edit Window which is used to create and modify M-files. M-files are files that contain a
 - Program or script of MATLAB commands.

Entering Commands

Every command has to be followed by a carriage return <cr> (enter key) in order that the command can be executed. MATLAB commands are case sensitive and lower case letters are used throughout.

To execute an M-file (such as Project_1.m), simply enter the name of the file without its extension (as in Project_1).

MATLAB Expo

In order to see some of the MATLAB capabilities, enter the demo command. This will initiate the MATLAB EXPO. MATLAB EXPO is a graphical demonstration environment that shows some of the different types of operations which can be conducted with MATLAB.

Abort

In order to abort a command in MATLAB, hold down the control key and press c to generate a local abort with MATLAB.

The Semicolon (;)

If a semicolon (;) is typed at the end of a command, the output of the command is not displayed.

Typing %

When per cent symbol (%) is typed in the beginning of a line, the line is designated as a comment. When the enter key is pressed, the line is not executed.

The clc Command

Typing clc command and pressing enter cleans the command window. Once the clc command is executed, a clear window is displayed.

Creating and Saving a Script File

Any text editor can be used to create script files. In MATLAB, script files are created and edited in the Editor/ Debugger Window. This window can be opened from the Command Window. From the Command Window, select File, New and then M-file. Once the window is open, the commands of the script file are typed line by line. The commands can also be typed in any text editor or word processor program and then copied and pasted in the Editor/Debugger Window. The second type of M-files is the function file. Function file enables the user to extend the basic library functions by adding ones own computational procedures. Function M-files are expected to return one or more results. Script files and function files may include reference to other MATLAB toolbox routines.

MATLAB function file begins with a header statement of the form:

- Function (name of result or results) = name (argument list)

Running a Script File

A script file can be executed either by typing its name in the Command Window and then pressing the Enter key, directly from the Editor Window by clicking on the Run icon. The file is assumed to be in the current directory, or in the search path.

Input to a Script File

There are three ways of assigning a value to a variable in a script file.

1. The variable is defined and assigned value in the script file.
2. The variable is defined and assigned value in the Command Window.
3. The variable is defined in the script file, but a specified value is entered in the Command Window

CODING

```
#include <LiquidCrystal.h>
#include <EEPROM.h>
const int rs = 13, en = 12, d4 = 11, d5 = 10, d6 = 9,
d7 = 8;
const int b1 = A0, b2 = 6, b3 = 5, bz=4;
int a=0,b=0;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup()
{
  Serial.begin(9600);
  pinMode(b1, INPUT);

  pinMode(b2, OUTPUT);
  pinMode(b3, OUTPUT);
  pinMode(bz, OUTPUT);
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print("    WELCOME    ");
  lcd.setCursor(0, 1);
  lcd.print("DROWSINESS D SYS    ");
  delay(500);
  digitalWrite(bz, LOW);
  digitalWrite(b3, LOW);
  digitalWrite(b2, LOW);
}
void loop()
{

  lcd.setCursor(0, 0);
  lcd.print("    WELCOME    ");
  lcd.setCursor(0, 1);
  lcd.print("DROWSINESS D SYS    ");
  delay(500);
  lcd.setCursor(0, 0);
  lcd.print("        WELCOME    ");
  lcd.setCursor(0, 1);
```



```

lcd.print("VEHICLE MONITOR          ");
delay(500);

```

```

if(Serial.available())
{
String key = Serial.readString();
writeString(11, key);
delay(10);
String recivedData;
recivedData = read_String(11);

```

```

if(recivedData[0]=='A')
{digitalWrite(b2,HIGH);
digitalWrite(b3,HIGH);
lcd.setCursor(0,0);
lcd.print("*****");
digitalWrite(bz,HIGH);
lcd.setCursor(0,1);
lcd.print("Drowsiness Detect ");
delay(5000);
digitalWrite(bz,LOW);
//}
}
//if(recivedData[0]=='b')
//{
//b=b+1;
//lcd.setCursor(0,0);
//lcd.print("Successfull voted ");
//digitalWrite(bz,HIGH);
//lcd.setCursor(0,1);
//lcd.print(" Voted to DMK ");
//delay(5000);
//digitalWrite(bz,LOW);
//}
//}
//if(digitalRead(b3)==HIGH)
//{

```

```

//if(a<b)
//{
//lcd.setCursor(0,0);
//lcd.print("    DMK Won    ");
//lcd.setCursor(0,1);
//lcd.print("ADMK:");
//lcd.print(a);
//lcd.print(" DMK:");
//lcd.print(b);
//Serial.print("    DMK Won    ");
//
//Serial.print("ADMK:");
//Serial.print(a);
//Serial.print(" DMK:");
Serial.println(digitalRead(7));
//delay(10000);
//}
//else if(b<a)
//{
//lcd.setCursor(0,0);
//lcd.print("    ADMK Won    ");
//lcd.setCursor(0,1);
//lcd.print("ADMK:");
//lcd.print(a);
//lcd.print(" DMK:");
//lcd.print(b);
//
//
//
//Serial.print("    ADMK Won    ");
//
//Serial.print("ADMK:");
//Serial.print(a);
//Serial.print(" DMK:");
//Serial.println(b);
//
//delay(10000);
//}

```

```

//else if(a==b)
//{
//lcd.setCursor(0,0);
//lcd.print("          Equal          ");
//lcd.setCursor(0,1);
//lcd.print("ADMK:");
//lcd.print(a);
//lcd.print(" DMK:");
//lcd.print(b);
//delay(10000);
//
//Serial.print("    Equal    ");
//
//Serial.print("ADMK:");
//Serial.print(a);
//Serial.print(" DMK:");
//Serial.println(b);
//
//
//
//}

```

```

int dd=digitalRead(7);

```

```

if(dd == 0)
{digitalWrite(b2,LOW);
digitalWrite(b3,HIGH);
lcd.setCursor(0,0);
lcd.print("*****");
digitalWrite(bz,HIGH);
lcd.setCursor(0,1);
lcd.print("GAS Detect          ");
delay(5000);
digitalWrite(bz,LOW);
//}

```

```

}
}

```

```

void writeString(char add,String data)
{
    int _size = data.length();
    int i;
    for(i=0;i<_size+1;i++)
    {
        EEPROM.write(add+i,data[i]);
    }
    EEPROM.write(add+_size,'\0');    //Add termination
null character for String Data
}

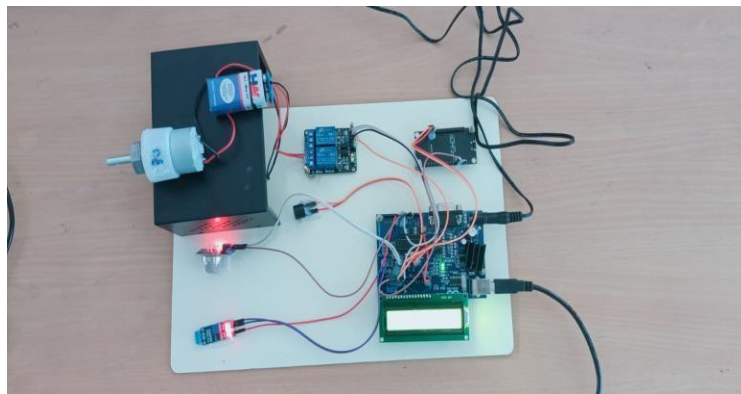
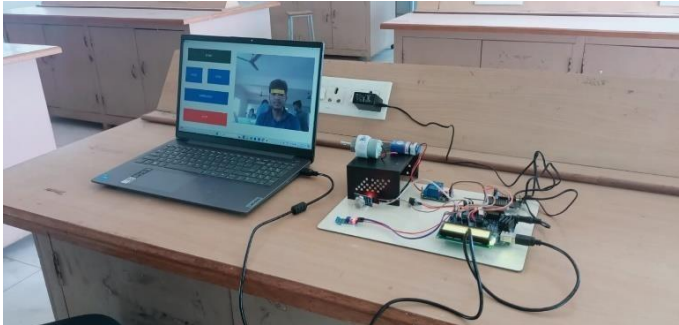
```

```

String read_String(char add)
{
    int i;
    char data[100]; //Max 100 Bytes
    int len=0;
    unsigned char k;
    k=EEPROM.read(add);
    while(k != '\0' && len<=10)    //Read until null
character
    {
        k=EEPROM.read(add+len);
        data[len]=k;
        len++;
    }
    data[len]='\0';
    return String(data);
}

```

SCREEN SHOT



DROWSINESS IOT			
Home			
DELETE ALL			
Sno	Message	Date	ACTION
1	*Accident Detect	07-05-2024	DELETE
2	*Accident Detect	07-05-2024	DELETE
3	*Accident Detect	07-05-2024	DELETE
4	*Accident Detect	07-05-2024	DELETE
5	*Accident Detect	01-05-2024	DELETE

Fig 7.1 Final output of the project

CHAPTER-8

FUTURE SCOPE

Enhancing an Anti-sleep Alarm for Drivers with a DIP(Driver's Information Panel monitoring system could involve integrating real-time data from the vehicle's sensors to detect signs of drowsiness more accurately.

This could include analyzing steering patterns, Lane departure, eye movements and even heart rate through wearable devices to provide timely alerts and prevent accidents.

Additionally, incorporating machine learning algorithms could improve the system's ability to recognize individual patterns and adjust alert thresholds accordingly.

CHAPTER-9

CONCLUSION

This paper shows that spatial configuration of facial landmarks provides sufficient discriminating information to accurately classify driver gaze into six gaze regions. The proposed framework has the capacity to distinguish the constant condition of the driver in day and night conditions with the assistance of a camera. The discovery of the Face and Eyes applied dependent on the balance. We have built up a non-meddling model of a PC vision-based framework for ongoing checking of the driver's sleepiness. Third, the problem of two region gaze classification (“driving-related” versus center stack) that is especially relevant to driver safety results in higher accuracy than the more general six-region classification problem. Fourth, the classification accuracy varies significantly between subjects and within subjects. Our future work will explore and exploit this inter-person and intra-person variation as it relates to the relationship between eye and head movement.

CHAPTER-10

REFERENCES

- [1] S. G. Klauer, T. A. Dingus, V. L. Neale, J. D. Sudweeks, and D. J. Ramsey, “The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data,” Tech. Rep., 2006.

- [2] J. F. Coughlin, B. Reimer, and B. Mehler, “Monitoring, managing, and motivating driver safety and well-being.” *IEEE Pervasive Computing*, vol. 10, no. 3, 2011.

- [3] T. Yoshioka, S. Nakashima, J. Odagiri, H. Tomimori, and T. Fukui, “Pupil detection in the presence of specular reflection,” in *Proceedings of the Symposium on Eye Tracking Research and Applications*. ACM, 2014, pp. 363–364.

- [4] L. ´ Swirski, A. Bulling, and N. Dodgson, “Robust real-time pupil tracking in highly off-axis images,” in *Proceedings of the Symposium on Eye Tracking Research and Applications*. ACM, 2012, pp. 173–176.

- [5] M. Muoz, J. Lee, B. Reimer, B. Mehler, and T. Victor, “Analysis of drivers’ head and eye movement correspondence: Predicting drivers’ glance location using head rotation data,” in *Proceedings of the 8th International Driving Symposium on Human Factors in Driver Assessment, Training, and Vehicle Design*, Snowbird, UT, 2015, to Appear.

- [6] W. J. Talamonti, W. Huang, L. Tijerina, and D. Kochhar, “Eye glance and head turn correspondence during secondary task performance in simulator

driving,” in Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 57, no. 1. SAGE Publications, 2013, pp. 1968–1972.

[7] V. Kazemi and J. Sullivan, “One millisecond face alignment with an ensemble of regression trees,” in Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on. IEEE, 2014, pp. 1867–1874.

[8] E. Murphy-Chutorian and M. M. Trivedi, “Head pose estimation in computer vision: A survey,” Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 31, no. 4, pp. 607–626, 2009.

[9] A. Al-Rahayfeh and M. Faezipour, “Eye tracking and head movement detection: A state-of-art survey,” Translational Engineering in Health and Medicine, IEEE Journal of, vol. 1, pp. 2 100 212– 2 100 212, 2013.

[10] B. Mehler, D. Kidd, B. Reimer, I. Reagan, J. Dobres, and A. McCartt, “Multi-modal assessment of on-road demand of voice and manual phone calling and voice navigation entry across two embedded vehicle systems,” Ergonomics, 2015.