A Technical Report on

Social Distancing tool using Ultrasonic sensor and Arduino

Submitted in partial fulfillment of the requirement of the award of the degree of

Bachelor of Technology

in

ELECTRICAL AND ELECTRONICS ENGINEERING

By-*+

BOINA VISHWESH
(19G21A0202)

BENAKAM NAGENDRA
(20G25A0208)

KONEPALLI KRISHNA KOWSHIK
(19G21A0215)

MATLE MADHAN
(19G21A0219)

KALAGANDALA PECHALAMMA
(19G21A0213)

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

AUDISANKARA COLLEGE OF ENGINEERING & TECHNOLOGY

Gudur, Thirupati – 524101, Andhra Pradesh

2022-2023

AUDISANKARA COLLEGE OF ENGINEERING AND TECHNOLOGY

Gudur, Thirupati- 524101, Andhra Pradesh

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

CERTIFICATE

This is to certify that the socially relevant project report entitled **Social Distancing tool using Ultrasonic sensor and Arduino** being submitted by Mr. BOINA VISHWESH Regd.no:19G21A0202, Mr. MATLE MADHAN Regd.no:19G21A0219, Ms. KALAGANDALA PENCHALAMMA Regd.no:19G21A0213, Mr. KONEPALLI KRISHNAKOWSHIK Regd.no:19G21A0215,Mr. BENAKAM NAGENDRA Regd.no:20G25A0208 in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Electrical and Electronics Engineering to Audisankara College Of Engineering And Technology, Gudur.

Project Guide

Mr SK WAHAB., M.Tech

Assistant Professor

Department of Electrical &Electronics Engineering

AUDISANKARA COLLEGE OF ENGG &TECH

GUDUR – THIRUPATI DISTRICT

Head of the Department

J Suresh., M.Tech (PhD).

Associate Professor

Department of Electrical &Electronics Engineering

AUDISANKARA COLLEGE OF ENGG & TECH

GUDUR – THIRUPATI DISTRICT

DECLARATION

BOINA VISHWESH , Regd.no:19G21A0202, MATLE MADHAN, Regd.no:19G21A0219, KALAGANDALA PENCHALAMMA, Regd.no:19G21A0213, KONEPALLI KRISHNA KOWSHIK, Regd.no:19G21A0215, BENAKAM NAGENDRA, Regd.no:20G25A0208, hereby declare that the Mini Project Work entitled **"SOCIAL DISTANCING TOOL USING ULTRASONIC SENSOR AND ARDUINO UNO"** done under the esteemed guidance of Head of the Department of EEE, Asst Prof. SK WAHAB, and is submitted in partial fulfillment of the requirements for the award of the Bachelor degree in **ELECTRICAL & ELECTRONICS ENGINEERING.**

Date:
Place:

BOINA VISHWESH
(119G21A0202)

BENAKAM NAGENDRA
(20G25A0208)

MATLE MADHAN
(19G21A0219)

KALAGANDLA PENCHALAMMA
(19G21A0213)

KONEPALLI KRISHNA KOWSHIK
(19G21A0215)

ACKNOWLEDGEMENT

The satisfaction and elation that accompany the successful completion of any task would be incomplete without the mention of the people who have made it a possibility. It is our great privilege to express our gratitude and respect to all those who have guided us and inspired us during the course of this Mini project work.

First and foremost, we express our sincere gratitude to our honorable chairman **Dr. VANKI PENCHALAIAH, M.A.,M.L.,Ph.D.,** who provided all facilities and necessary encouragement during the course of study.

I am very much thankful to **Dr.A.MOHAN,**Ph.D**.,** Director of Audisankara Group Of Institutions for his valuable encouragement at various stages of my mini project

I extend my gratitude and sincere thanks to our beloved principal **Prof. K Dhanumjaya ,** for motivating and providing necessary infrastructure and permitting us to complete the mini project.

I would like to express the sense of gratitude towards our Head of the Department **Mr. J. SURESH.M.Tech.(PhD).**Our Project Coordinator **Mr. D.DINESH KUMAR.,M.Tech** Assistant Professor and our internal guide **Mr SK WAHAB.,M.Tech.** Assistant Professor, Department of EEE and for their support and encouragement for completion of this mini project.

Finally, I would like to thank **my parents** who support me and I express my sincere thanks to all the teaching and non-teaching staff that guided and helped me to complete the mini project work successfully.

CONTENTS

CHAPTERS PAGE NO

LIST OF FIGURES

LIST OF ABBREVIATIONS

USB	**Universal Serial Bus**
LED	**Light Emitting Diode**
DC	**Direct Current**
Arduino IDE	**Arduino Integrated Development Environment**
Arduino CLI	**Arduino Command-line Interface**
PCB	**Printed Circuit Board**
CPU	**Central Processing Unit**
R	**RESISTOR**
BB	**Bread Board**

1.1 ABSTRACT:

Coronavirus disease (Covid-19) is an infectious disease that is caused by a newly discovered virus, known as the coronavirus that comes under the category of SARS.Due to the increase in the pandemic day by day, the number of people infected by it is in the millions and the deaths are in lakhs. The transmission of the virus is through physical contact from person to person/objects and gets multiplied to pass to the community. The technique which can help to reduce the virus is social distancing. This paper proposes a device that gives actual and real-time information observed by the device. It helps the person to remind of having social distancing. The hardware, run by the Arduino programmed software, is small and can be easily carried and used for an industrial manufactures used or an blind peoples for helping of an like an sticks to measure the distance between the objects.

CHAPTER I: INTRODUCTION

1.2. BLOCK DIAGRAM:

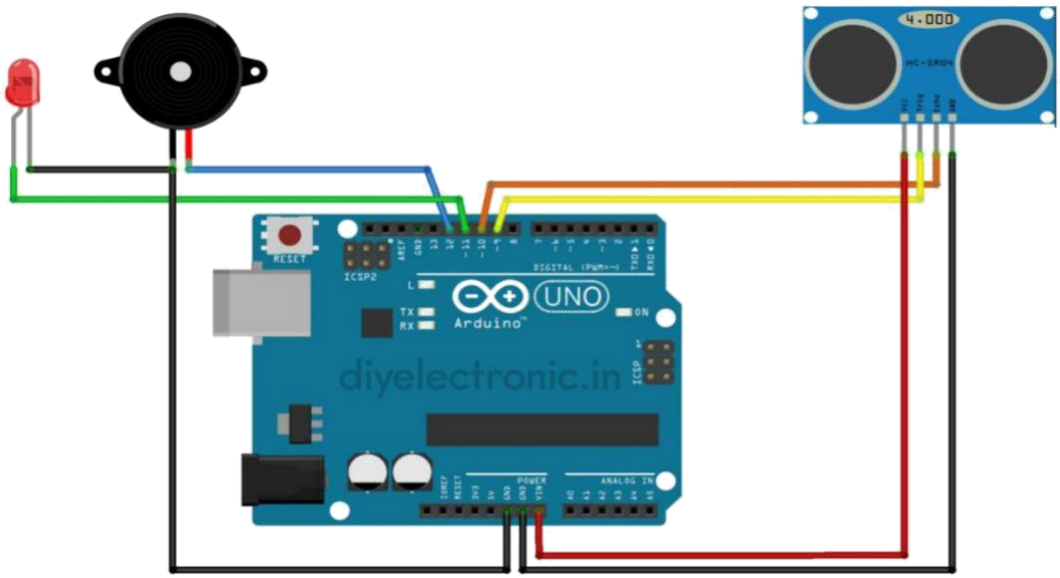

Fig.1.2: Arduino kit connections with all components

1.3. DESCRIPTION:

In this project, we will discuss how to measure distance using ultrasonic sensor HCSR04 and Arduino with circuit diagram and Arduino code. This HC-SR04 sensor and Arduino circuit also can be used as a touchless doorbell or distance measuring tool.

1.4. HARDWARE REQUIREMENTS:

- HC SR04 Ultrasonic sensor
- Arduino Uno/Nano
- DC Buzzer
- Jumper wires
- Bread board
- LED

1.5. SOFTWARE SPECIFICATIONS:

- Arduino compiler, Arduino IDE
- MC Programming Language: Embedded C

CHAPTER- II- WORKING

2.1. EMBEDDED SYSTEMS

2.1.1. DEFINITIONS:

An Embedded system is a combination of hardware and software, it is also named “Firmware”. An embedded system is a special-purpose computer system, which is completely encapsulated by the device it controls. It is a computer-controlled system. An embedded system is a specialized system that is a part of a larger system or machine. As a part of a larger system, it largely determines its functionality. Embedded systems are electronic devices that incorporate microprocessors in their implementations. The main purpose of microprocessors is to simplify the system design and improve flexibility. In embedded systems, the software is often stored in a read-only memory (RAM) chip. Embedded systems provide several major functions including monitoring the analog environment by reading data from sensors and controlling actuators.

The general-purpose definition of embedded systems is that they are devices used to control, monitor, or assist the operation of equipment, machinery, or plant. “Embedded” reflects the fact that they are an integral part of the system. All embedded systems are including computers or microcontrollers. Some of these computers are however very simple systems as compared with a personal computer. The very simple embedded systems are capable of performing only a simple function or set of functions to meet a single predetermined purpose. In more complex systems an application program that enables the embedded system to the user for a particular purpose in a specific application determines the functioning of the embedded system the ability to have a program means that the same embedded system can be used for a variety of different purpose. Controller input comes from a detector or sensor and its output goes to a switch or activator which (for example) may start or stop the operation of a machine

2.1.2. EXAMPLES OF EMBEDDED SYSTEMS:

Embedded systems are found in wide range of application areas. Originally, they were used only for expensive industrial control applications, but as technology brought down the cost of dedicated processors, they began to appear in moderately expensive applications such as automobiles, communication and office equipment’s and television Today's embedded systems are so inexpensive that they are used in almost every electronic product in our life. Embedded systems are often designed for mass production.

Some examples of embedded systems:

- Automatic Teller Machines
- Cellular telephone and telephone switches
- Computer network equipment
- Computer printers
- Disk drives
- Engine controllers and antilock brake controllers for automobiles
- Home automation products
- Handheld calculators
- Household appliances
- Medical equipment
- Measurement equipment
- Multifunction wrist watches
- Multifunction printers

2.2. SYSTEM STUDY

2.2.1. Feasibility Study

The feasibility of the project is analysed in this phase and the business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Threes key considerations involved in the feasibility analysis are

ECONOMICAL FEASIBILITY

TECHNICAL FEASIBILITY

SOCIAL FEASIBILITY

2.2.2. Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.2.3. Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.2.4. Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.3. LITERATURE SURVEY

Literature survey is the most important step in Hardware and software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things r satisfied, the ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need a lot of external support. This support can be obtained from senior programmers, from the book, or from websites. Before building the system the above considerations r taken into account for developing the proposed system.

CHAPTER-III: HARDWARE AND MODELS

3.1. HARDWARE DESCRIPTION:

3.1.1. ARDUINO UNO:

The **Arduino UNO** is the best board to get started with electronics and coding. If this is your first experience tinkering with the platform, the UNO is the most robust board you can start playing with. The UNO is the most used and documented board of the whole Arduino family.

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), 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 an AC-to-DC adapter or battery to get started. You can tinker with your Uno without worrying too much about doing something wrong, in the worst-case scenario you can replace the chip for a few dollars and start over again. "Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past, or outdated boards see the Arduino index of boards.

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards can read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

Arduino is an open-source programmable circuit board that can be integrated into a wide variety of maker space projects both simple and complex. This board contains a microcontroller that can be programmed to sense and control objects in the physical world. By responding to sensors and inputs, the Arduino can interact with a large array of outputs such as LEDs, motors, and displays. Because of its flexibility and low cost, Arduino has become a very popular choice for makers and maker spaces looking to create interactive hardware projects.

Arduino was introduced back in 2005 in Italy by Massimo Banzi as a way for non-engineers to have access to a low-cost, simple tool for creating hardware projects. Since the board is open-source, it is released under a Creative Commons license which allows anyone to produce their board. If you search the web, you will find there are hundreds of Arduino compatible clones and variations available but the only official boards have Arduino in their name.

Arduino is a great platform for prototyping projects and inventions but can be confusing when having to choose the right board. If you're brand new to this, you might have always thought that there was just one "Arduino" board and that's it. In reality, there are many variations of the official Arduino boards and then there are hundreds more from competitors who offer clones. But don't worry, we're going to show you which one to start with later on in this tutorial.

Below are a few examples of the different types of Arduino boards out there. The boards with the name Arduino on them are the official boards but there are also a lot of really great clones on the market. One of the best reasons to buy a clone is the fact they are generally less expensive than their official counterpart. Adafruit and Sparkfun for example, sell variations of the Arduino boards which cost less but still have the same quality as the originals. One word of caution, be careful when buying boards from companies you don't know.

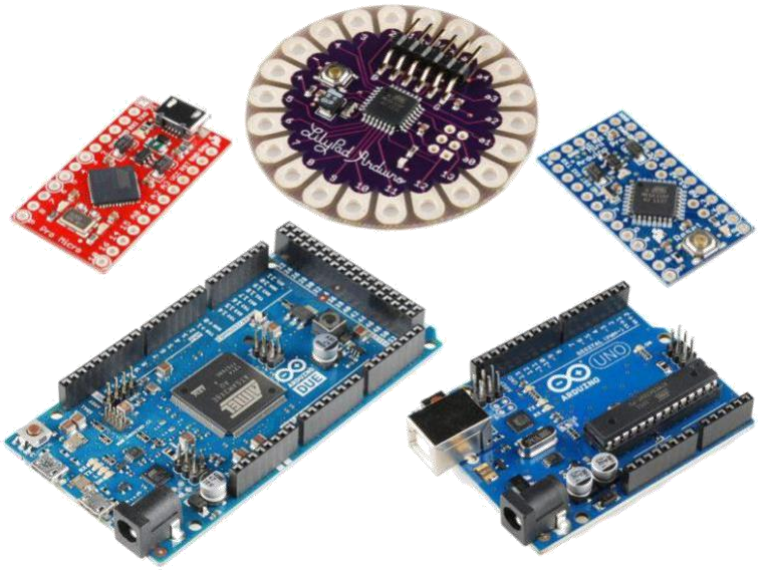

Fig.3.1: Types of Arduino kits are used in present scenario

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino. cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by a USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. It is similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution-Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available.

The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low-cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use them for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step-by-step instructions of a kit, or sharing ideas online with other members of the Arduino community.

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of

microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantages for teachers, students, and interested amateurs over other systems:

- Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50
- Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
- Simple, clear programming environment - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
- Open source and extensible software - The Arduino software is published as an open-source tool, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
- Open source and extensible hardware - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module to understand how it works and save money.

- The word "no" means "one" in Italian and was chosen to mark the initial release of Arduino Software. The Uno board is the first in a series of USB-based Arduino boards; it and version 1.0 of the Arduino IDE were the reference versions of Arduino, which have now evolved to newer releases. The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.
- While the Uno communicates using the original STK500 protocol, it differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead,

it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

3. Arduino UNO R3

Specifications

Microcontroller	Atmega238
Operating Voltage	7-12 V recommended, 6-20 V limits
Digital I/O pins	14 (of which 6 PWM)
Analog input pins	6
DC current per I/O pin	40 mA
DC current for 3.3V pin	50 mA
Flash memory	32 KB
USB to Serial converter	Atmega16U2
UART	1
3V	available

Layout/connections Arduino UNO R3

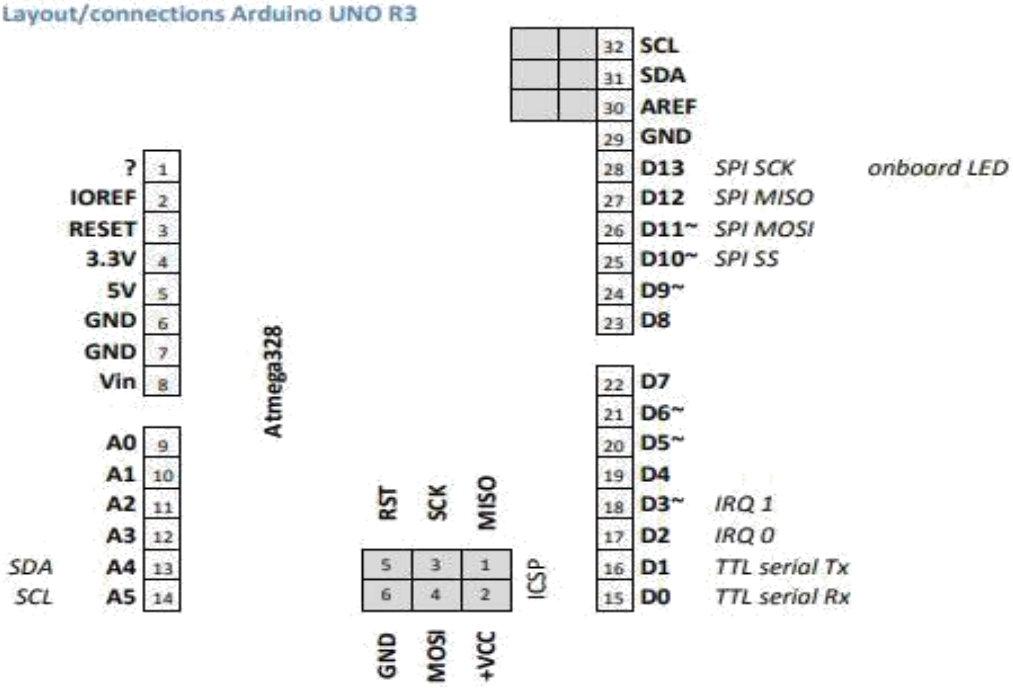

Fig: 3.2: Compatible Arduino kits, Description& pins which are used

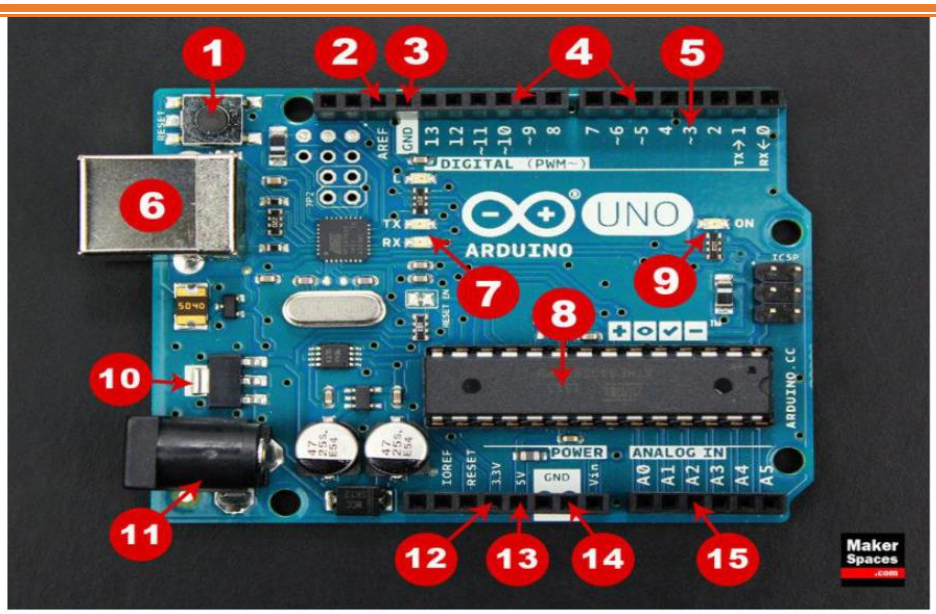

Fig: 3.3: A Typical Arduino UNO Kit with pins

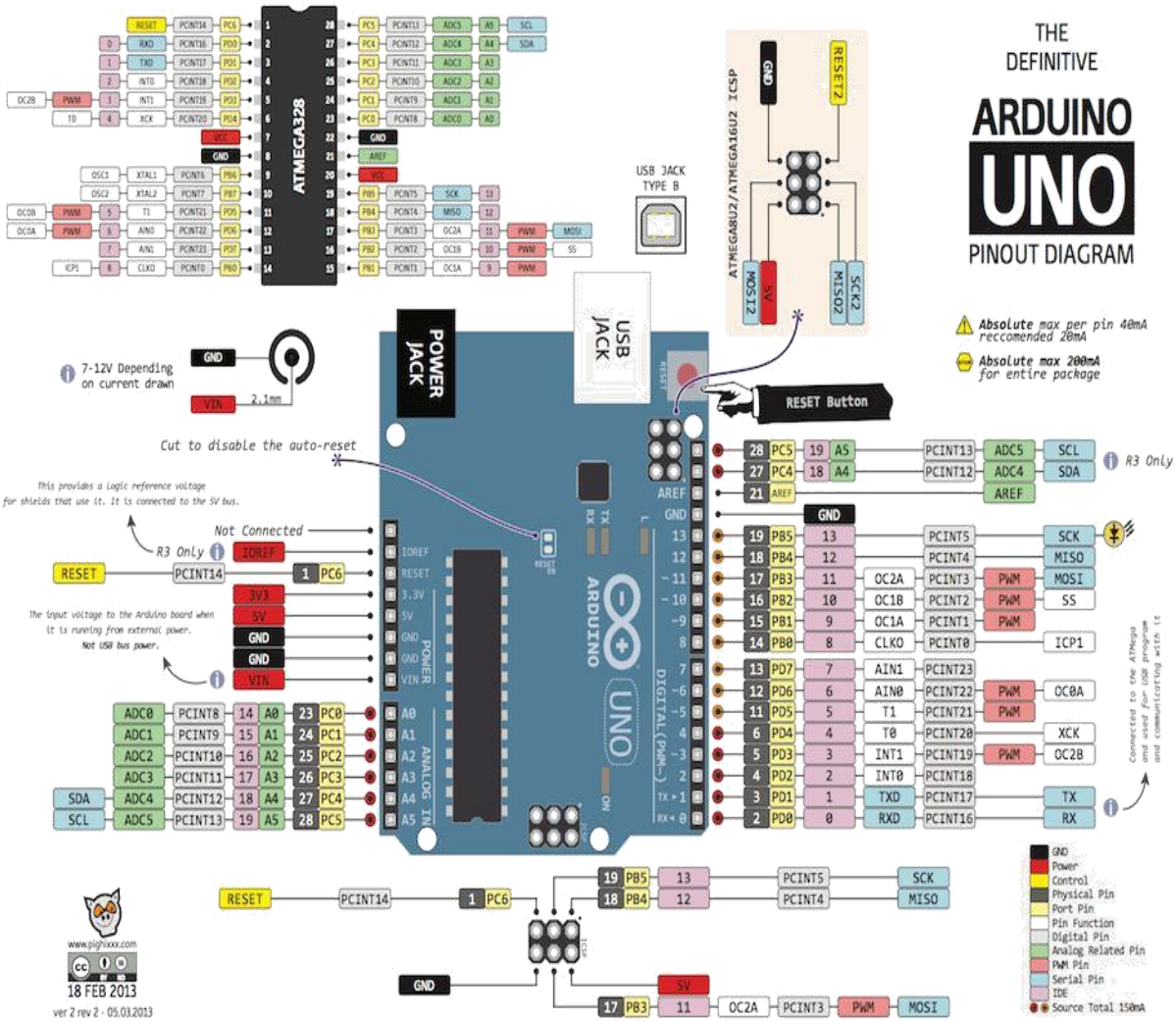

Fig:3.4: Complete pin elaboration in all the components that are used

Board Breakdown

Here are the components that make up an Arduino board and what each of their functions are.

1. **Reset Button** – This will restart any code that is loaded to the Arduino board
2. **AREF** – Stands for "Analog Reference" and is used to set an external reference voltage
3. **Ground Pin** – There are a few ground pins on the Arduino and they all work the same
4. **Digital Input/Output** – Pins 0-13 can be used for digital input or output
5. **PWM** – The pins marked with the (~) symbol can simulate analog output
6. **USB Connection** – Used for powering up your Arduino and uploading sketches
7. **TX/RX** – Transmit and receive data indication LEDs
8. **ATmega Microcontroller** – This is the brains and is where the programs are stored
9. **Power LED Indicator** – This LED lights up anytime the board is plugged in a power source
10. **Voltage Regulator** – This controls the amount of voltage going into the Arduino board
11. **DC Power Barrel Jack** – This is used for powering your Arduino with a power supply
12. **3.3V Pin** – This pin supplies 3.3 volts of power to your projects
13. **5V Pin** – This pin supplies 5 volts of power to your projects
14. **Ground Pins** – There are a few ground pins on the Arduino and they all work the same
15. **Analog Pins** – These pins can read the signal from an analog sensor and convert it to digital

2.1.1. Arduino Power Supply:

The Arduino Uno needs a power source in order for it to operate and can be powered in a variety of ways. You can do what most people do and connect the board directly to your computer via a USB cable. If you want your project to be mobile, consider using a 9V battery pack to give it juice. The last method would be to use a 9V AC power supply.

Fig.3.5: Arduino kit Power Supply Adapter and Cable

3.1.2. Arduino Breadboard

Another very important item when working with Arduino is a solderless breadboard. This device allows you to prototype your Arduino project without having to permanently solder the circuit together. Using a breadboard allows you to create temporary prototypes and experiment with different circuit designs. Inside the holes (tie points) of the plastic housing, are metal clips which are connected to each other by strips of conductive material.

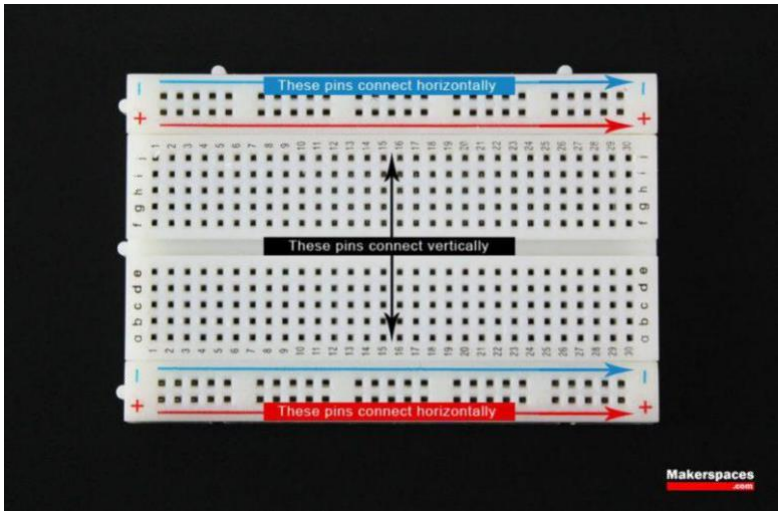

Fig.3.6: A Typical Bread Board

On a side note, the breadboard is not powered on its own and needs power brought to it from the Arduino board using jumper wires. These wires are also used to form the circuit by connecting resistors, switches and other components together.

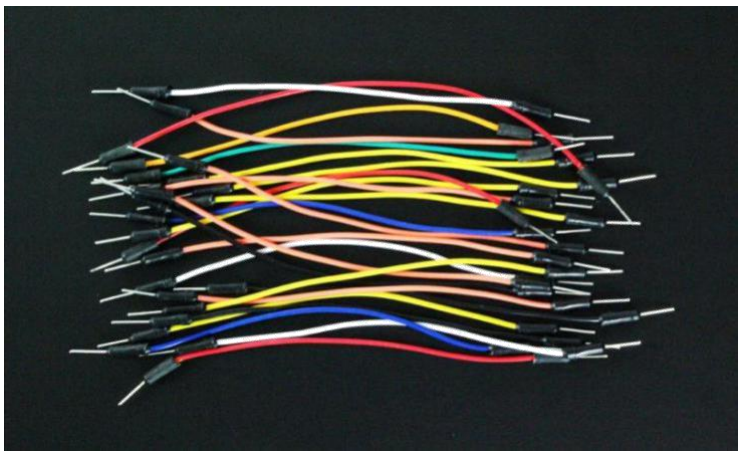

Fig.3.7: Connecting wires/ Patch Chords

Here is a visual of what a completed Arduino circuit looks like when connected to a breadboard.

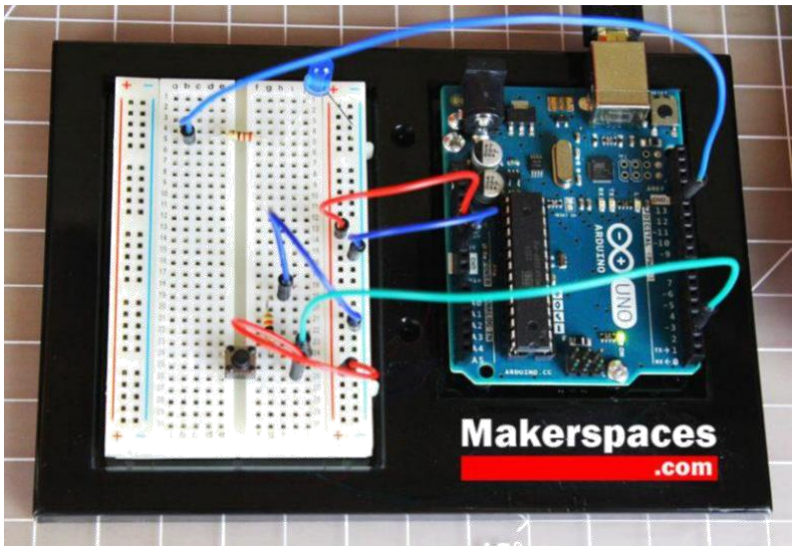

Fig.3.8: Final kit made with all connections

- LED: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it is off.
- VIN: The input voltage to the Arduino/Genuino board when it is 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: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.

- 3V3: A 3.3-volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- GND: Ground pins.
- IOREF: This pin on the Arduino/Genuine board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source, or enable voltage translators on the outputs to work with the 5V or 3.3V.
- Reset: Typically used to add a reset button to shields that block the one on the board.

3.1.3. Special pin functions:

Each of the 14 digital pins and 6 analog pins on the Uno can be used as an input or output, under software control (using pinMode(), digitalWrite(), and digitalRead() functions). They operate at 5 volts. Each pin can provide or receive 20 mA as the recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50K ohm. A maximum of 40mA must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labeled A0 through A5; each provides 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 the range using the AREF pin and the analogReference() function.

In addition, some pins have specialized functions:

- Serial / UART: pins 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: pins 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.
- PWM (pulse-width modulation): pins 3, 5, 6, 9, 10, and 11. Can provide 8-bit PWM output with the analogWrite() function.
- SPI (Serial Peripheral Interface): pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). These pins support SPI communication using the SPI library.
- TWI (two-wire interface) / I^2C: pin SDA (A4) and pin SCL (A5). Support TWI communication using the Wire library.
- AREF (analog reference): Reference voltage for the analog inputs.

3.1.4. Communication:

The Arduino/Genuino Uno has a number of facilities for communicating with a computer,

another Arduino/Genuine board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows serial communication on any of the Uno's digital pins.

3.1.5Automatic (Software) Reset:

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

This setup has other implications. When the Uno is connected to a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.

3.1.6. The Compilation Process:

The Arduino code is actually just plain old c without all the header part (the includes and all). when you press the 'compile' button, the IDE saves the current file as arduino.c in the 'lib/build' directory then it calls a make file contained in the 'lib' directory.

This makefile copies arduino.c as prog.c into 'lib/tmp' adding 'wiringlite.inc' as the beginning of it. this operation makes the arduino/wiring code into a proper c file (called prog.c). After this, it copies all the files in the 'core' directory into 'lib/tmp'. these files are the implementation of the various arduino/wiring commands adding to these files adds commands to the language

The core files are supported by pascal stang's procyon avr-lib that is contained in the 'lib/avrlib' directory

At this point the code contained in lib/tmp is ready to be compiled with the c compiler contained in 'tools'. If the make operation is succesfull then you'll have prog.hex ready to be downloaded into the processor.

NOTE:the next release will see each architecture (avr/pic/8051) to treated as a 'plug-in' to the IDE so that the user can just select from a menu the microcontroller board to use and the IDE will pick the right compilation sequence.

3.2. ULTRASONIC SENSOR:

Ultrasonic sensors (also known as transceivers when they both send and receive, but more generally called transducers) work on a principle similar to radar or sonar which evaluate attributes of a target by interpreting the echoes from radio or sound waves respectively. Ultrasonic sensors generate high frequency sound waves and evaluate the echo which is received back by the sensor. Sensors calculate the time interval between sending the signal and receiving the echo to determine the distance to an object.

This technology can be used for measuring wind speed and direction (anemometer), tank or channel level, and speed through air or water. For measuring speed or direction a device uses multiple detectors and calculates the speed from the relative distances to particulates in the air or water. To measure tank or channel level, the sensor measures the distance to the surface of the fluid. Further applications include: humidifiers, sonar, medical ultra sonography, burglar alarms and non-destructive testing.

Systems typically use a transducer which generates sound waves in the ultrasonic range, above 18,000 hertz, by turning electrical energy into sound, then upon receiving the echo turn the sound waves into electrical energy which can be measured and displayed.

Fig3.8: Ultrasonic Sensor

3.2.1. Ultrasonic Sensor Working:

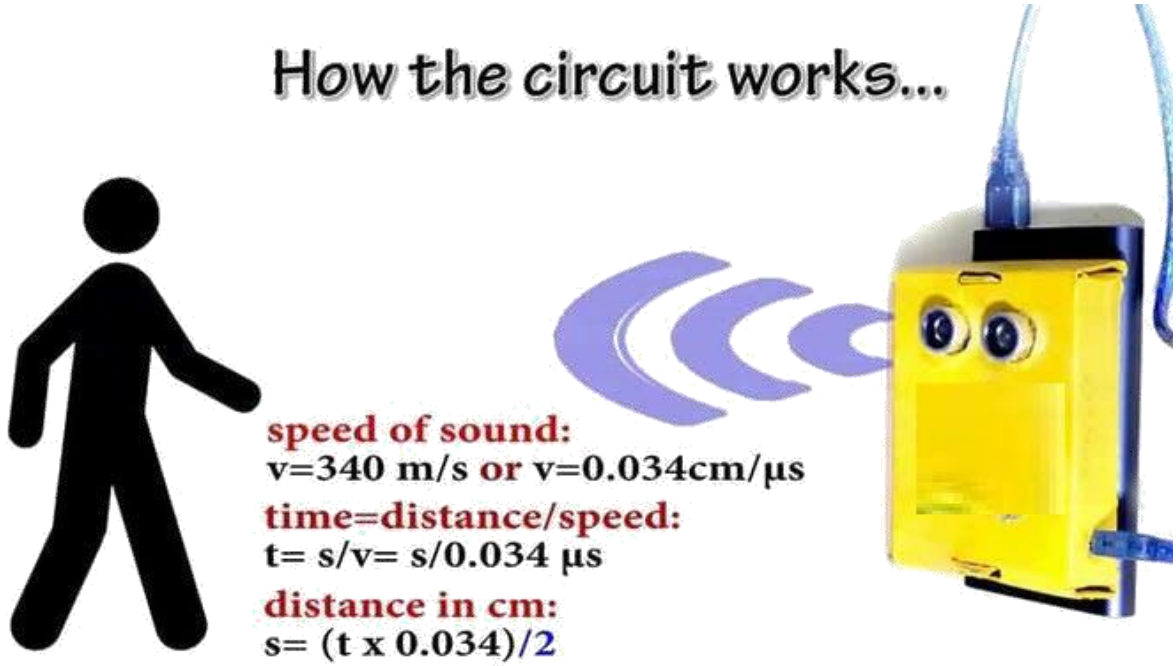

Fig.3.9: Working of Ultrasonic Sensor

The working principle of the HC-SR04 ultrasonic sensor is quite simple. If emit ultrasound at 40000Hz which travels through the air. If there is any object or obstacle on the path it will bounce back to the module. Now considering the travel time and speed of the sound we can calculate the distance.

Fig.3.10: Distance Measurement using Social Distancing Sensor

3.3. BUZZER:

A buzzer or beeper is an audio signaling device, which may mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

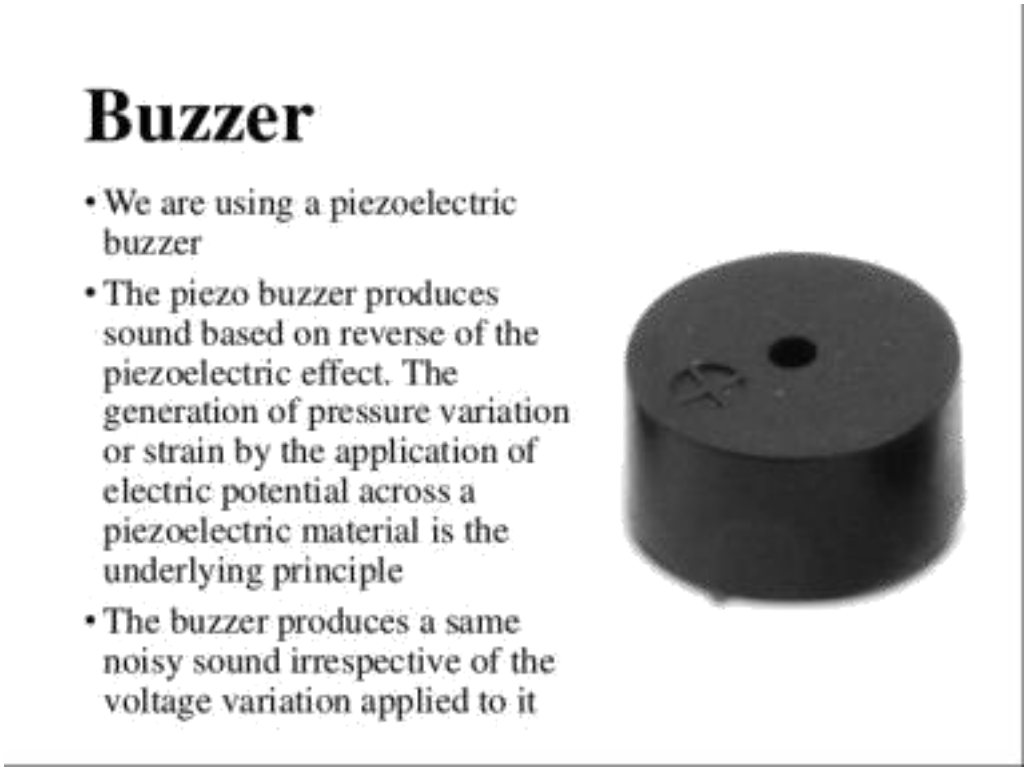

Fig.3.11: Buzzer

3.4. LIGHT EMITTING DIODE (LED):

LEDs are special diodes that emit light when connected in a circuit. They are frequently used as "pilot" lights in electronic appliances to indicate whether the circuit is closed or not. A clear (or often colored) epoxy case enclosed the heart of an LED, the semi-conductor chip.

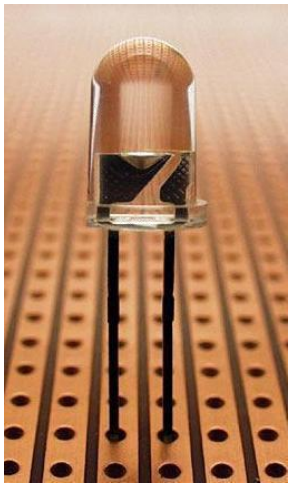

Fig.3.12. LED View

Fig.3.12: LED Symbol

LED's must be connected the correct way round, the diagram may be labeled a or + for anode and k or - for cathode. The negative side of an LED lead is indicated in two ways:

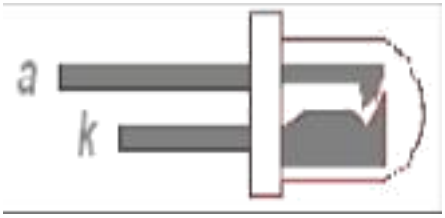

Fig.3.13: LED Internal View

1) By the flat side of the bulb.

2) By the shorter of the two wires extending from the LED.

If you can see inside the LED the cathode is the larger electrode (but this is not an official identification method). The negative lead should be connected to the negative terminal of a battery. LED's operate at relative low voltages between about 1 and 4 volts, and draw currents between about 10 and 40 mille amperes. Voltages and currents substantially about these values can melt a LED chip. The most important part of light emitting diode (LED) is the semi-conductor chip located in the center of the bulb as shown below. The chip has two regions separated by a junction. The p region is dominated by positive electric charges, and the n region is dominated by negative electric charges. The junction acts as a barrier to the flow of electrons between the p and n regions. Only when sufficient voltage is applied to the semi-conductor chip, can the current flow and the electron cross the junction into the p region? In the absence of the large enough electric potential difference (voltage) across the LED leads, the junction presents an electric potential barrier to the flow of electrons.

A light-emitting diode (LED) is a semiconductor light source. LEDs are used as indicator lamps in many devices and are increasingly used for other lighting. Appearing as practical electronic components in 1962, early LEDs emitted low intensity red light, but modern versions are available across the visible, ultraviolet, and infrared wavelengths, with very high brightness.

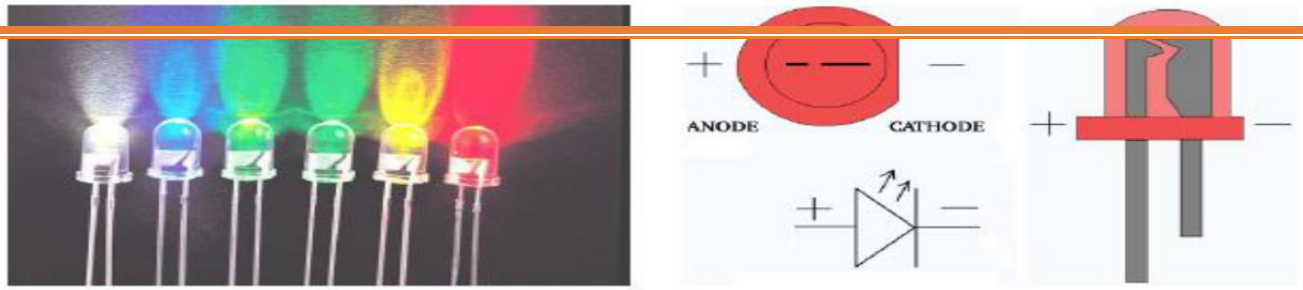

Fig.3.14: Light Emitting Diode

When a light-emitting diode is switched on, electrons are able to recombine with holes within the device, releasing energy in the form of photons. This effect is called electroluminescence and the color of the light (corresponding to the energy of the photon) is determined by the energy band gap of the semiconductor. An LED is often small in area (less than 1 mm2), and integrated optical components may be used to shape its radiation pattern. LEDs present many advantages over incandescent light sources including lower energy consumption, longer lifetime, improved physical robustness, smaller size, and faster switching. However, LEDs powerful enough for room lighting are relatively expensive and require more precise current and heat management than compact fluorescent lamp sources of comparable output.

3.5. BREAD BOARD:

Breadboards are temporary work boards for electronic circuits. The general shape of a breadboard is shown in Fig. 6.3. Compatible with most breadboards, 24-gauge wire is used to connect circuits; solid wire, not stranded. Sometimes, kits may be available with various colors of fixed lengths to specifically fit breadboards. These are a nice convenience.

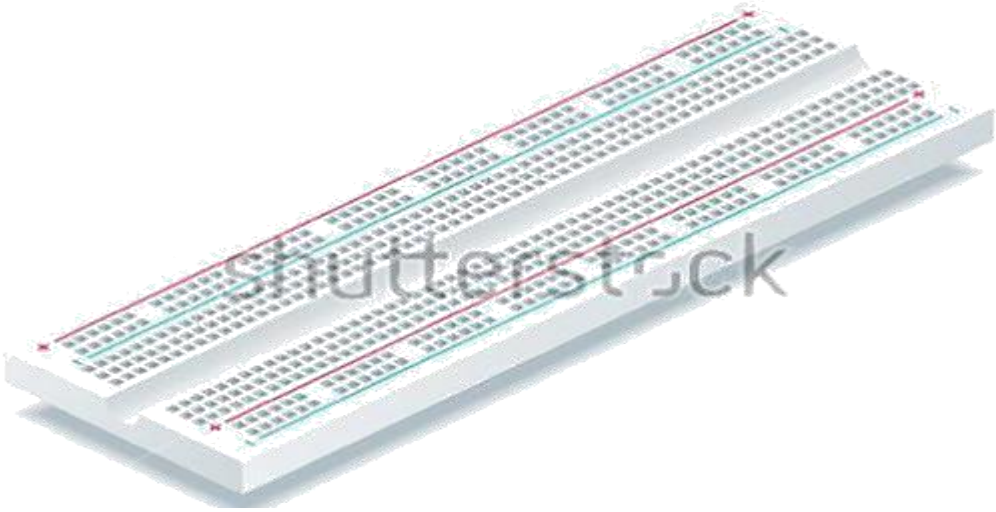

Fig: 3.15: A Typical Bread Board

The horizontal rows are connected throughout the row and may make a complete row with the addition of a simple jumper at the center point. These rows are noted with red and blue or black markings.

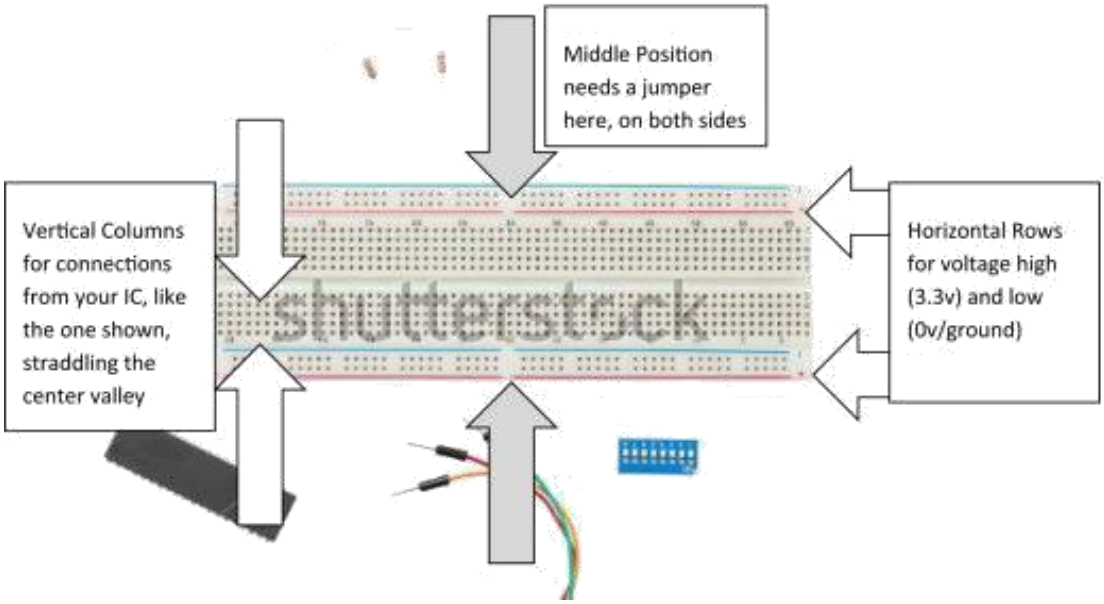

Fig. 3.16: Description of bread board

Breadboard (plugboard) has sets of miniature sockets laid out on a 0.1 inch grid which will accept the manual insertion of component leads and tinned copper wire (TCW) links. It has rows of contacts interconnected in groups placed either side of the center line of the board, where the integrated circuits (ICs) are inserted, giving multiple contacts on each IC pin. At each side of the board, there are long rows of common contacts, which are used for the power supplies. Some types of breadboard are supplied in blocks that link together to accommodate larger circuits, or are mounted on a base with built-in power supplies.

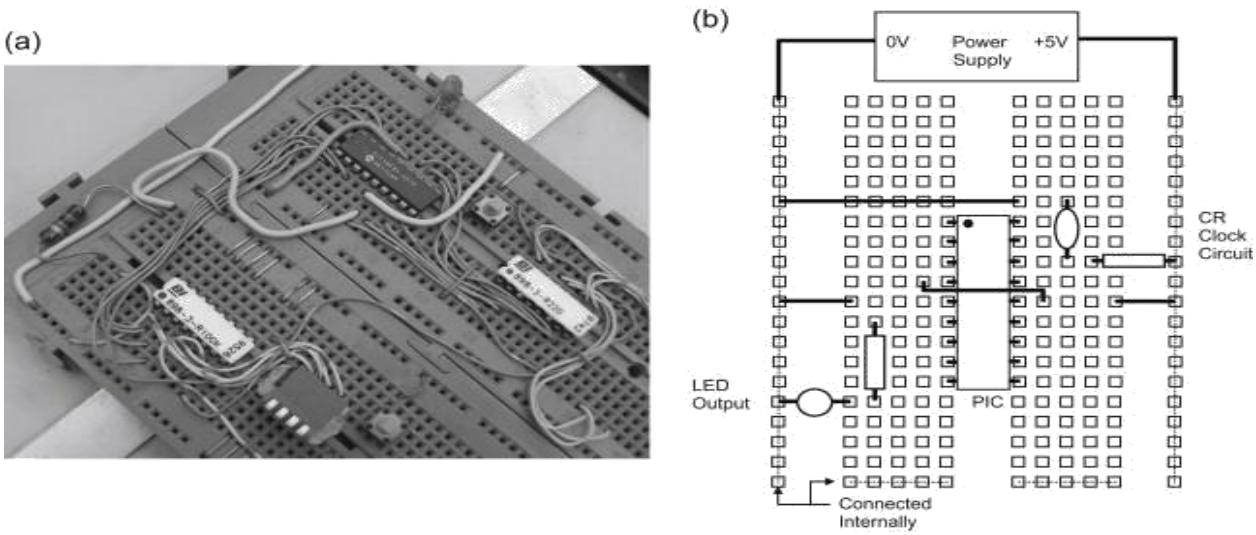

Fig: 3.17: A Typical bread board connections

3.5.1 Breadboarding Techniques:

The breadboard is both the designer's playground and proving ground. It is there that Reality resides, and paper (or computer) designs meet their ruler. More than anything else, breadboarding is an iterative procedure, an odd amalgam of experience guiding an innocent, ignorant, explorative spirit. A key is to be willing to try things out, sometimes for not very good reasons. Invent problems and solutions, guess carefully and wildly, throw rocks and see what comes loose. Invent and design experiments, and follow them wherever they lead. Reticence to try things is probably the number one cause of breadboards that "don't work". Implementing the above approach to life begins with the physical construction methods used to build the breadboard.

A high speed breadboard must start with a ground plane. Additionally, bypassing, component layout and connections should be consistent with high speed operations. Because of these considerations there is a common misconception that breadboarding high speed circuits is time consuming and difficult. This is simply not true. For high speed circuits of moderate complexity a complete and electrically correct breadboard can be assembled in 10 minutes if all necessary components are on hand. The key to rapid breadboarding is to identify critical circuit nodes and design the layout to suit them. This permits most of the breadboard's construction to be fairly sloppy, saving time and effort. Additionally, use all degrees of freedom in making connections and mounting components. Don't be bashful about bending IC pins to suit desired low capacitance connections, or air wiring components to achieve rapid or electrically optimum layout. Save time by using components, such as bypass capacitors, as mechanical supports for other components, such as amplifiers. It is true that eventual printed circuit construction is required, but when initially breadboarding forget about PC and production constraints. Later, when the circuit works, and is well understood, PC adaptations can be taken care of.

3.6. Jumper Wires:

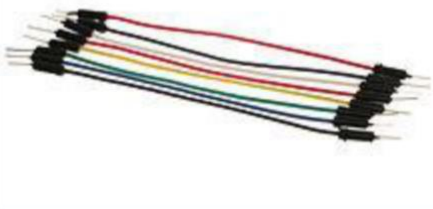

Fig: 3.18: Jumper or connecting cables

A jump wire (also known as jumper, jumper wire, jumper cable, DuPont wire, or DuPont cable – named for one manufacturer of them) is an electrical wire, or group of them in a

cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering. Types:

- Male to Male
- Female to Male
- Female to Female

A jump wire (also known as jumper, jumper wire, jumper cable, DuPont wire or cable) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test.

The difference between each is in the end point of the wire. Male ends have a pin protruding and can plug into things, while female ends do not and are used to plug things into. SHOWA jumper wire (NSL: New Showa Lead) is a lead-free tin-plated annealed copper wire. Tin plating is tin: 99.2%, copper: 0.8%. In general, it is said that hot plating is difficult to control the plating thickness compared with electroplating, but we control the plating thickness by the original processing method.

A jumper wire is a short insulated wire with bare (stripped of insulation) ends. You use jumper wires, such as the one shown in the following figure, to connect two points in a breadboard circuit. Gauged devices allow you to strip insulation without worrying about cutting the wire underneath the insulation. When you set a jumper, you place a plug on the prongs that completes a contact. In effect, the jumper acts as a switch by closing (or opening) an electrical circuit. Jumpers can be added or removed to change the function or performance of a PC component. A group of jumpers is sometimes called a jumper block.

3.7. POWER SUPPLY:

input power supply for the project 5v,1Amp

Fig:3.19: Power Supply adapter

3.8. PCB:

A printed circuit board (PCB) mechanically supports and electrically connects electrical or electronic components using conductive tracks, pads and other features etched from one or more sheet layers of copper laminated onto and/or between sheet layers of a non-conductive substrate. Components are generally soldered onto the PCB to both electrically connect and mechanically fasten them to it.

Printed circuit boards are used in all but the simplest electronic products. They are also used in some electrical products, such as passive switch boxes.

Alternatives to PCBs include wire wrap and point-to-point construction, both once popular but now rarely used. PCBs require additional design effort to lay out the circuit, but manufacturing and assembly can be automated. Electronic computer-aided design software is available to do much of the work of layout. Mass-producing circuits with PCBs is cheaper and faster than with other wiring methods, as components are mounted and wired in one operation. Large numbers of PCBs can be fabricated at the same time, and the layout only has to be done once. PCBs can also be made manually in small quantities, with reduced benefits.

PCBs can be single-sided (one copper layer), double-sided (two copper layers on both sides of one substrate layer), or multi-layer (outer and inner layers of copper, alternating with layers of substrate). Multi-layer PCBs allow for much higher component density, because circuit traces on the inner layers would otherwise take up surface space between components. The rise in popularity of multilayer PCBs with more than two, and especially with more than four, copper planes was concurrent with the adoption of surface mount technology. However, multilayer PCBs make repair, analysis, and field modification of circuits much more difficult and usually impractical.

A basic PCB consists of a flat sheet of insulating material and a layer of copper foil, laminated to the substrate. Chemical etching divides the copper into separate conducting lines called tracks or circuit traces, pads for connections, vias to pass connections between layers of copper, and features such as solid conductive areas for electromagnetic shielding or other purposes. The tracks function as wires fixed in place, and are insulated from each other by air and the board substrate material. The surface of a PCB may have a coating that protects the copper from corrosion and reduces the chances of solder shorts between traces or

undesired electrical contact with stray bare wires. For its function in helping to prevent solder shorts, the coating is called solder resist or solder mask.

A printed circuit board can have multiple copper layers. A two-layer board has copper on both sides; multi layer boards sandwich additional copper layers between layers of insulating material. Conductors on different layers are connected with vias, which are copper-plated holes that function as electrical tunnels through the insulating substrate. Through-hole component leads sometimes also effectively function as vias. After two-layer PCBs, the next step up is usually four-layer. Often two layers are dedicated as power supply and ground planes, and the other two are used for signal wiring between components.

"Through hole" components are mounted by their wire leads passing through the board and soldered to traces on the other side. "Surface mount" components are attached by their leads to copper traces on the same side of the board. A board may use both methods for mounting components. PCBs with only through-hole mounted components are now uncommon. Surface mounting is used for transistors, diodes, IC chips, resistors and capacitors. Through-hole mounting may be used for some large components such as electrolytic capacitors and connectors.

The pattern to be etched into each copper layer of a PCB is called the "artwork". The etching is usually done using photoresist which is coated onto the PCB, then exposed to light projected in the pattern of the artwork. The resist material protects the copper from dissolution into the etching solution. The etched board is then cleaned. A PCB design can be mass-reproduced in a way similar to the way photographs can be mass-duplicated from film negatives using a photographic printer.

In multi-layer boards, the layers of material are laminated together in an alternating sandwich: copper, substrate, copper, substrate, copper, etc.; each plane of copper is etched, and any internal vias (that will not extend to both outer surfaces of the finished multilayer board) are plated-through, before the layers are laminated together. Only the outer layers need be coated; the inner copper layers are protected by the adjacent substrate layers.

FR-4 glass epoxy is the most common insulating substrate. Another substrate material is cotton paper impregnated with phenolic resin, often tan or brown.When a PCB has no components installed, it is less ambiguously called a printed wiring board (PWB) or etched wiring board. However, the term "printed wiring board" has fallen into disuse. A PCB populated with electronic components is called a printed circuit

usage, the term "printed circuit board" most commonly means "printed circuit assembly" (with components). The IPC preferred term for assembled boards is circuit card assembly (CCA),[4] and for assembled backplanes it is backplane assemblies. "Card" is another widely used informal term for a "printed circuit assembly". For example, expansion card.

A PCB may be "silkscreen" printed with a legend identifying the components, test points, or identifying text. Originally, an actual silkscreen printing process was used for this purpose, but today other, finer quality printing methods are usually used instead. Normally the screen printing is not significant to the function of the PCBA.

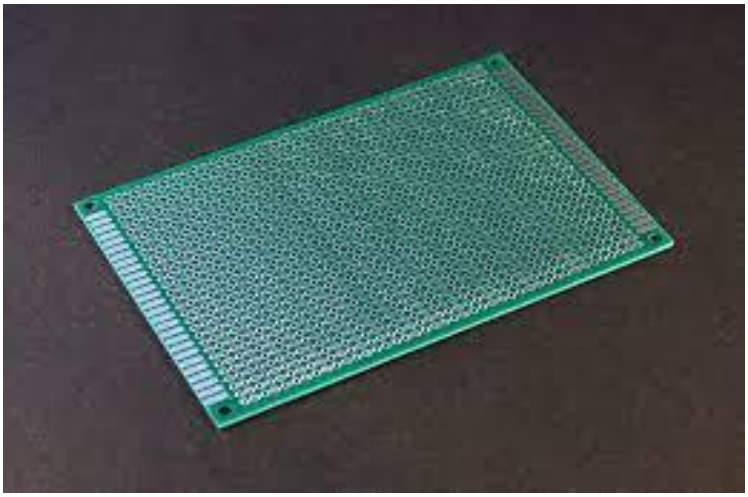

Fig.3.20: Model PCB used

A minimal PCB for a single component, used for prototyping, is called a breakout board. The purpose of a breakout board is to "break out" the leads of a component on separate terminals so that manual connections to them can be made easily. Breakout boards are especially used for surface-mount components or any components with fine lead pitch.

Advanced PCBs may contain components embedded in the substrate.

3.9. Characteristics:

The first PCBs used through-hole technology, mounting electronic components by leads inserted through holes on one side of the board and soldered onto copper traces on the other side. Boards may be single-sided, with an unpleated component side, or more compact double-sided boards, with components soldered on both sides. Horizontal installation of through-hole parts with two axial leads (such as resistors, capacitors, and diodes) is done by bending the leads 90 degrees in the same direction, inserting the part in

the board (often bending leads located on the back of the board in opposite directions to improve the part's mechanical strength), soldering the leads, and trimming off the ends. Leads may be soldered either manually or by a wave soldering machine.

Through-hole manufacture adds to board cost by requiring many holes to be drilled accurately, and it limits the available routing area for signal traces on layers immediately below the top layer on multi-layer boards, since the holes must pass through all layers to the opposite side. Once surface-mounting came into use, small-sized SMD components were used where possible, with through-hole mounting only of components unsuitably large for surface-mounting due to power requirements or mechanical limitations, or subject to mechanical stress which might damage the PCB (e.g. by lifting the copper off the board surface).

CHAPTER- IV: IMPLEMENTATION

IMPLEMENTATION:

4.1. HARDWARE INSTALLATION:

The USB standard requires a 1.5 kΩ pullup resistor on D+, but this board is known to have a wrong value (R10 on the board). It ships with either a 10 kΩ resistor or a 4.7 kΩ resistor, but it should be replaced with a 1.5 kΩ resistor, or put an appropriate resistor value (e.g 1.8 kΩ) in between PA12 and 3.3V. It is also true that some PCs are tolerant of incorrect value so, before you change the resistance, you can try if it works in your case.

4.2. TEST PLAN:

Check the power supply connections

· Insert smart card in to the smart card reader · Check LEDs

· Check Microcontroller minimum requirements

· Welcome message will be printed on LCD

· Consumed units and available units are displayed on LCD

· A message will be displayed if there is no smart card

4.2.1. Window – Files.

Now use from the menu Project – Select Device for Target and select a CPU for your project. The Select Device dialog box shows the µVision2 device database. Just select the microcontroller you use. We are using for our examples the Philips 80C51RD+ CPU. This selection sets necessary tool options for the 80C51RD+ device and simplifies in this way the tool

Configuration

4.2.2. Building Projects and Creating a HEX Files

Typical, the tool settings under Options – Target are all you need to start a new application. You may translate all source files and line the application with a click on the Build Target toolbar icon. When you build an application with syntax errors, µVision2 will display errors and warning messages in the Output Window – Build page. A double click on a message line opens the source file on the correct location in a µVision2 editor window. Once you have

successfully generated your application you can start debugging. After you have tested your application, it is required to create an Intel HEX file to download the software into an EPROM programmer or simulator. µVision2 creates HEX files with each build process when Create HEX file under Options for Target – Output is enabled. You may start your PROM

programming utility after the make process when you specify the program under the option Run User Program #1.

4.2.3. CPU Simulation:

µVision2 simulates up to 16 Mbytes of memory from which areas can be mapped for read, write, or code execution access. The µVision2 simulator traps and reports illegal memory accesses. In addition to memory mapping, the simulator also provides support for the integrated peripherals of the various 8051 derivatives. The on-chip peripherals of the CPU you have selected are configured from the Device

4.2.4. Database selection:

You have made when you create your project target. Refer to page 58 for more information about selecting a device. You may select and display the on-chip peripheral components using the Debug menu. You can also change the aspects of each peripheral using the controls in the dialog boxes.

4.2.5. Start Debugging:

You start the debug mode of µVision2 with the Debug – Start/Stop Debug Session command. Depending on the Options for Target – Debug configuration, µVision2 will load the application program and run the startup code µVision2 saves the editor screen layout and restores the screen layout of the last debug session. If the program execution stops, µVision2 opens an editor window with the source text or shows CPU instructions in the disassembly window. The next executable statement is marked with a yellow arrow. During debugging, most editor features are still available. For example, you can use the find command or correct program errors. Program source text of your application is shown in the same windows. The µVision2 debug mode differs from the edit mode in the following aspects: The "Debug Menu and Debug Commands" described on page 28 are available. The additional debug windows are discussed in the following. The project structure or tool parameters cannot be modified. All build commands are disabled.

4.2.6. Disassembly Window:

The Disassembly window shows your target program as mixed source and assembly program or just assembly code. A trace history of previously executed instructions may be displayed with Debug – View Trace Records. To enable the trace history, set Debug – Enable/Disable Trace Recording. If you select the Disassembly Window as the active window all program step commands work on CPU instruction level rather than program source lines. You can select a text line and set or modify code breakpoints using toolbar buttons or the context menu commands.

4.3. SENSORS:

With some simple code, the Arduino can control and interact with a wide variety of **sensors** - things that can measure light, temperature, degree of flex, pressure, proximity, acceleration, carbon monoxide, radioactivity, humidity, barometric pressure, you name it, you can sense it!

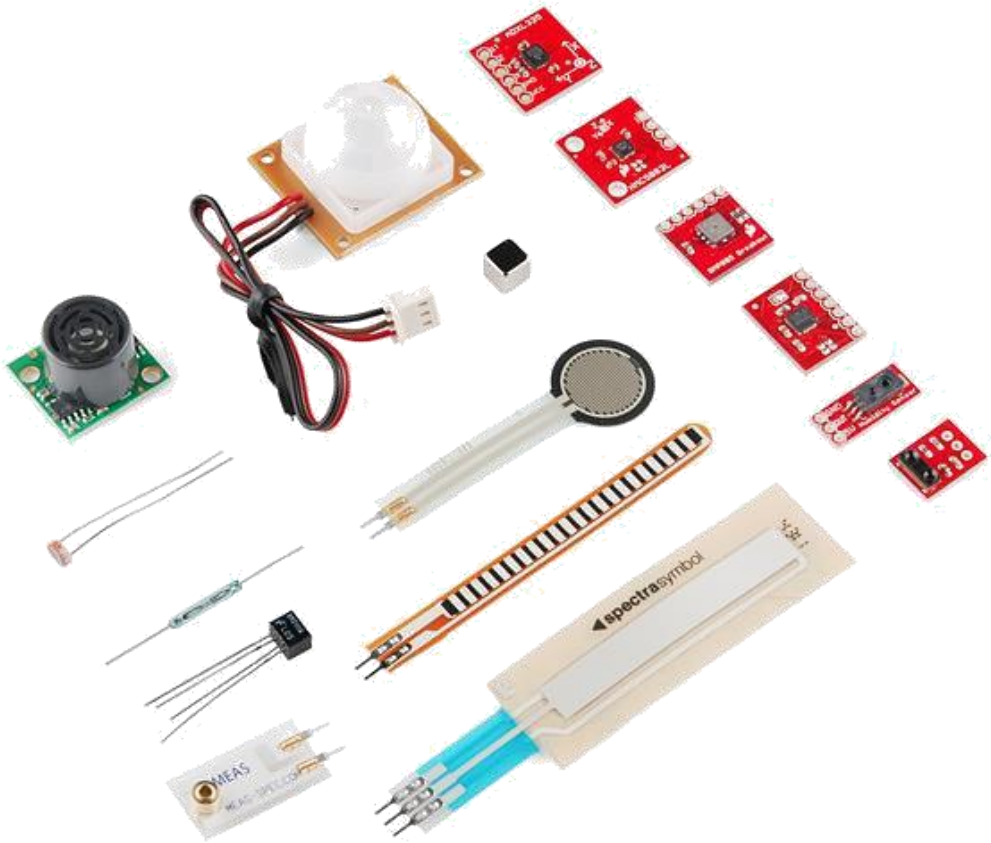

Fig:4.1: Just a few of the sensors that are easily compatible with Arduino

Arduino is an open source development platform. This document describes how to use the platform on Gentoo. Note that in addition to official and clone Arduino products based on Atmel AVR microprocessors, the environment can also support other Atmel AVR microprocessors

4.4. PREPARE THE KERNEL FOR USB CONNECTION

The arduino boards will be connected via USB to the computer. With this connection it is possible to write binaries to the atmega microprocessor and get debug messages from the board during run mode. Different boards have different USB interface chips. In case youprogramming any more, but you may still need it for debugging.

4.5. HARDWARE INSTALLATION:

The USB standard requires a 1.5 kΩ pullup resistor on D+, but this board is known to have a wrong value (R10 on the board). It ships with either a 10 kΩ resistor or a 4.7 kΩ resistor, but it should be replaced with a 1.5 kΩ resistor, or put an appropriate resistor value (e.g 1.8 kΩ) in between PA12 and 3.3V. It is also true that some PCs are tolerant of incorrect value so, before you change the resistance, you can try if it works in your case.

4.6. SOFTWARE INSTALLATION:

A bootloader needs to be flashed using USB to Serial or ST-Link (SWD). See Flashing the bootloader.

Note that after first flashing the bootloader you may have to place the board into "perpetual bootloader" mode before you can upload a sketch; place a resistor between pin PC14 and 3.3V, and then reset the board. You should now be able to flash a blank sketch, remove the resistor, and restart the board, after which uploading new sketches should work as expected. If you find that the IDE successfully resets your board, but dfu-util complains about no DFU-devices being present you may have to edit the maple-upload script in tools-folder. Find the line where it calls upload-reset, and increase the value given to it.

4.7. COMMUNICATING WITH ARDUINO THROUGH PC

Another major problem related to the Arduino board was the communication with it from PC. Since, we require RS-232 to TTL conversion for the communication, so we tried some methods:

1. Firstly we used the MAX-232 IC to communicate with the Arduino as with the 8051 but due to large voltage drop and mismatch in the speed, it failed to communicate.
2. Next, we tried to use a dedicated AVR as USB to Serial converter as in the original arduino board, the difference being DIP AVR used by us instead of the SMD Mega16U2 controller. But, unfortunately we were unable to communicate through it.

   At last we had no other choice but to complete the project in time by using the FTDI FT-232R chip for USB to Serial conversion.

``

CHAPTER- V: TESTING, SOURCE CODES & ADVANTAGES

TESTING OF PROJECT

With the knowledge of operation of the system was tested step by step to the transistor output and the load was connected across the collector terminal of the transistor.

5.1 ASSEMBLING

The whole system was packed in a plastic casing and provision was made for the IR to sense light from the outside.

5.2. SOFTWARE INSTALLATION:

A bootloader needs to be flashed using USB to Serial or ST-Link (SWD). See Flashing the bootloader.

Note that after first flashing the bootloader you may have to place the board into "perpetual bootloader" mode before you can upload a sketch; place a resistor between pin PC14 and 3.3V, and then reset the board. You should now be able to flash a blank sketch, remove the resistor, and restart the board, after which uploading new sketches should work as expected. If you find that the IDE successfully resets your board, but dfu-util complains about no DFU-devices being present you may have to edit the maple-upload script in tools-folder. Find the line where it calls upload-reset, and increase the value given to it.

Software program: (Main Source Code)

```
const int trigPin = 9;
const int echoPin = 10;
long duration;
int distanceCm, distanceInch;
void setup()
{
Serial.begin(9600);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(11,OUTPUT);
pinMode(12,OUTPUT);
```

```
}
void loop () {
digital Write (trig Pin, LOW);
delay Microseconds (2);
digitalWrite (trig Pin, HIGH);
delay Microseconds (10);
digital Write (trig Pin, LOW);
duration = pulseIn (echo Pin, HIGH);
distance Cm= duration*0.034/2;
distance Inch = duration*0.0133/2;
Serial. Printin  ("Distance: ");
Serial. println (distance Cm);
If (distance Cm < 100)
{
  Digital Write (11, HIGH);
  digital Write (12, HIGH);
}
else
{
  Digital Write (11,  LOW);
  Digital Write (12, LOW);
}
}
```

5.3. TESTING:

5.3.1. System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the
Software system meets its requirements and user expectations and does not fail in an

unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

5.4. TYPES OF TESTS:

5.4.1. Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

5.4.2. Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

5.4.3. Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective

value of current tests is determined.

5.4.4..System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

5.4.5. White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

5.4.6. Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

5.4.7. Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

5.5. TEST OBJECTIVES

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

5.6. FEATURES TO BE TESTED

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct

page. Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface

defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

5.7. TEST RESULTS: All the test cases mentioned above passed successfully. No defects encountered.

5.7.1. Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

CHAPTER- VI: RESULTS & CONCLUSION

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points

BIBLIOGRAPHY

References:

[1]. Habibolah Arasteh rad & Aeshia Badi (2020). A Study on control of novel corona-virus (2019-nCov) disease process by using PID controller. MedRxiv, pp 1-19. doi: https://doi.org/10.1101/2020.04.19.20071654.

[2]. Diah Handayani, Dwi Rendra Hadi, Fathiyah Isbaniah, Erlina Burhan & Heidy Agustin (2020). Penyakit Virus Corona 2019. Jurnal Respirologi Indonesia. 40(2), pp 119-129.

[3]. Giselle Ann Alcoran Alvarez, Marc Brian Garcia & Dave Unabia Alvarez (2020). Automated Social Distancing Gate with Non-Contact Body Temperature Monitoring using Arduino Uno. International Research Journal of Engineering and Technology. 07(07), pp 4351-4356.

[4]. David Cababaro Bueno (2020). Physical distancing: A rapid global analysis of public health strategies to minimize COVID-19 outbreaks. Institutional Multidisiplinary Research and Development Journal. 3, pp 31-53. doi: https://doi.org/10.13140/RG.2.2.30429.15840/1

[5]. Con T. Nguyen, Yuris Mulya Saputra, Nguyen Van Huynh, Ngoc-Tan Nguyen, & Trat Viet Khoa (2020). Enabling and Emerging Technologies for Social Distancing: A Comprehensive Survey. ArXiv, pp 1–42.

[6]. Giuseppe Gaeta (2020). Social distancing versus early detection and contacts tracing in epidemic management. ArXiv, pp 1-12.

[7]. Luigi Cirrincione, Fulvio Plescia, Caterina Ledda, Venerando Rapisarda, Daniela Martorana, Raluca Emilia Moldovan, Kelly Theodoridou & Emanuele Cannizzaro (2020). COVID-19 Pandemic: Prevention and protection measures to be adopted at the

workplace. Sustainability (Switzerland), 12(9), pp 1–18. Doi: https://doi.org/10.3390/SU12093603.

[8]. C Raina Macintyre, & Quanyi Wang (2020). Comment Physical distancing, face masks, and eye protection for prevention of COVID-19. The Lancet, 6736(20), pp 19–20. doi: https://doi.org/10.1016/S0140-6736(20)31183-1.

[9]. Reza Aminnejad & Alikhani, R. (2020). Physical distancing or social distancing: that is the question. Canadian Journal of Anesthesia/Journal Canadien d'anesthésie, 11 May. doi: https://doi.org/10.1007/s12630-020-01697-2.

[10] Nicholas R Jones, Zeshan U Qureshi, Robert J Temple, Jessica P J Larwood & Trisha Greenhalgh (2020). Two metres or one: what is the evidence for physical distancing in Covid-19?. The BMJ, pp 1–6. doi: https://doi.org/10.1136/bmj.m3223.

[11] Hitesh Mohapatra & Amiya Kumar Rath (2020). Social Distancing Alarming Through Proximity Sensors for COVID-19. Easy Chair, 18 June.

[12]. Yuda Irawan, Muhardi, Rian Ordila & Roni Diandra (2021). Automatic Floor Cleaning Robot Using Arduino and Ultrasonic Sensor. Journal of Robotics and Control (JRC), 2(4), pp 4–7. doi: https://doi.org/10.18196/jrc.2485. Jenli Susilo, Anita Febriani, Uci Rahmalisa & Yuda Irawan (2021). Car Parking Distance Controller Using Ultrasonic Sensors Based on Arduino Uno. Journal of Robotics and Control (JRC), 2(5), pp 353–356. doi: https://doi.org/10.18196/jrc.25106.