CHAPTER-1

INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

The main aim of this project is to detect the obstacles in the path using an Arduino Uno and also to give this information to the user who is visually impaired individual for enhancing their safety and mobility.

Creating assistive tools for visually impaired individual has become easily applicable. One of the research focuses has been the use of Arduino Uno microcontroller, Ultrasonic sensor, passive many others to sense and measure distances.

The goal is to measure the distance and detect the obstacles with less human assistance as much as possible.

The hardware utilized included the Arduino Uno on a bread board interfaced with LEDs, Buzzer and Ultrasonic sensor.

With the help of Ultrasonic sensor it has become easy to find the distance of the obstacle from sensor.

In this project we will learn how we can interface an Ultrasonic Sensor with a microcontroller like Arduino. We will interface an Arduino with ultrasonic module and blink a LED and beep a Buzzer whenever the obstacle is detected.

The program to run the circuit was developed using Arduino IDE and stored at the memory of the Arduino microcontroller.

Simultaneously the sensor when picks the presence of obstacle within the limits the lights will be glowing, while at the same time producing sound signals and a sound buzzer.

1.2 OBJECTIVE OF THE PROJECT

The objective of the Ultrasonic sensor is to detect the distance between obstacle and the sensor in short time.

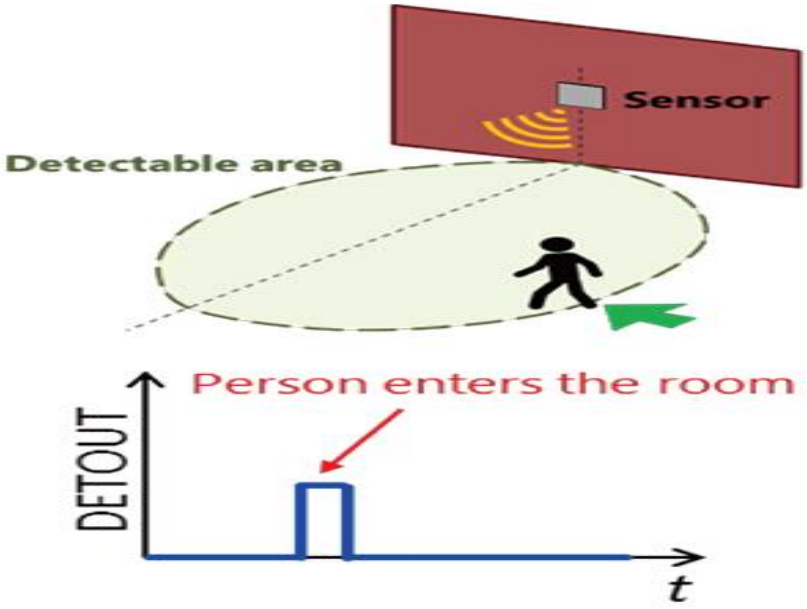

Fig 1.1: detection of a person

By using this sensor we can detects the obstacles within the range of ultrasonic sensor and depending on the distance the obstacle is detected the Arduino using the code sends signals to buzzer to alert the user.

1.3 ORGANIZATION OF THE PROJECT

In the chapter 1, we discussed about the introduction, overview, objective, organization of the project and existing system, proposed system.

In the chapter 2, we discussed about the embedded system.

In the chapter 3, we discussed about the block diagram description, Hardware description.

In the chapter 4, we discussed about the arduino IDE, arduino setup and initialisation, developing the code.

In the chapter 5, we discussed about the working of the project.

In the chapter 6, we discussed about the result of the project.

In the chapter 7, we discussed about the advantages, applications of the project.

In the chapter 8, we discussed about the conclusion, future scope of the project

1.4 EXISTING SYSTEM

- Traditional white canes have been commonly used by visually impaired individuals for mobility.
- However, these canes lack advanced features such as obstacle detection and real-time feedback.
- Existing systems primarily rely on physical feedback but do not provide comprehensive obstacle detection capabilities.

1.5 PROPOSED SYSTEM

- The Smart Blind Stick aims to revolutionize assistive devices for the visually impaired.
- **Key components of the proposed system include:**
  - Arduino Board: Serves as the brain of the device.
  - Ultrasonic Sensor: Detects obstacles by emitting ultrasonic waves and measuring reflections.
  - Buzzer: Provides auditory alerts.
  - LED: Offers visual cues.
- **Real-time response mechanism:**
  - When an obstacle is detected, the Arduino triggers both the buzzer and LED, ensuring immediate feedback.
- **IoT Capabilities:**
  - The Smart Blind Stick can be enhanced with IoT connectivity for remote monitoring and assistance.
  - This holistic solution empowers users to navigate confidently and safely.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION TO EMBEDDED SYSTEM

An embedded system is a computer system—a combination of a computer processor, computer memory, and input/output peripheral devices—that has a dedicated function within a larger mechanical or electrical system.

It is embedded as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has real-time computing constraints

Embedded systems control many devices in common use today. Ninety-eight percent of all microprocessors manufactured are used in embedded systems.

Modern embedded systems are often based on microcontrollers (i.e. microprocessors with integrated memory and peripheral interfaces), but ordinary microprocessors (using external chips for memory and peripheral interface circuits) are also common, especially in more complex systems.

In either case, the processor(s) used may be types ranging from general purpose to those specialized in a certain class of computations or even custom designed for the application at hand. A common standard class of dedicated processors is the digital signal processor (DSP).

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic light controllers, programmable logic

controllers, and large complex systems like hybrid vehicles, medical imaging systems, and avionics.

Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large equipment rack.

Structure of the embedded systems architecture:

The below figure gives an insight to a typical embedded system with the features that makes any product.

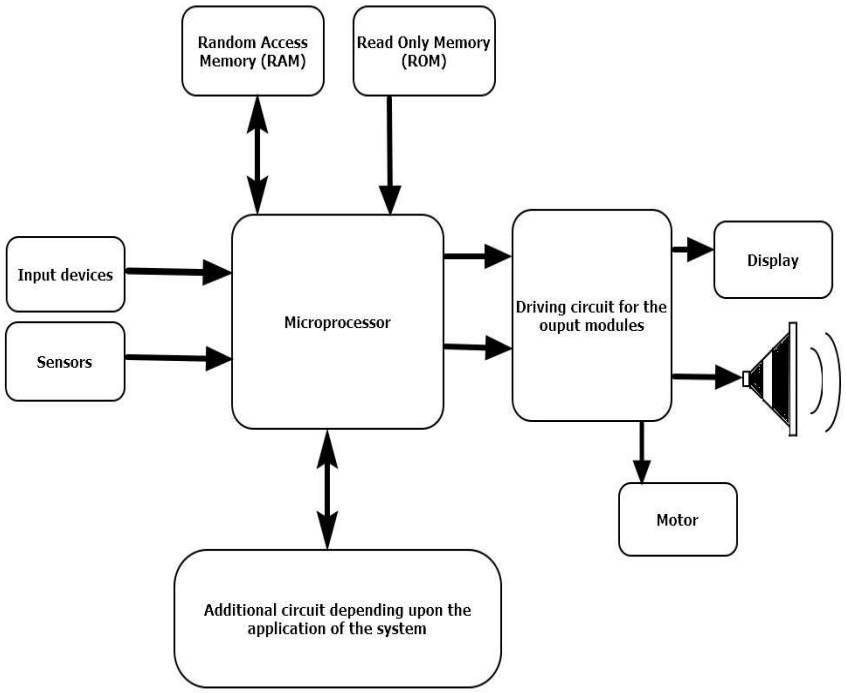

Fig. 2.1 basic architecture of embedded system

It measures the quantities that are physical and converts it to an electrical signal which may be read by an observer or through any electronic tool like an A-D converter. A sensor shops the measured amount to the memory.

A-D Converter:

An Analog-to-digital converter that is used converts the Analog signal sent by using the sensor right into a digital signal.

Processor:

Processors process the records to degree the output and keep it to the memory. **D-A Converter:**

A virtual-to-Analog converter converts the virtual records fed by using the processor to Analog information.

Applications

Embedded systems are commonly found in consumer, industrial, automotive, home appliances, medical, commercial and military applications.

Telecommunications systems employ numerous embedded systems from telephone switches for the network to cell phones at the end user. Computer networking uses dedicated routers and network bridges to route data.

Consumer electronics include MP3 players, television sets, mobile phones, video game consoles, digital cameras, GPS receivers, and printers.

Household appliances, such as microwave ovens, washing machines and dishwashers, include embedded systems to provide flexibility, efficiency and features.

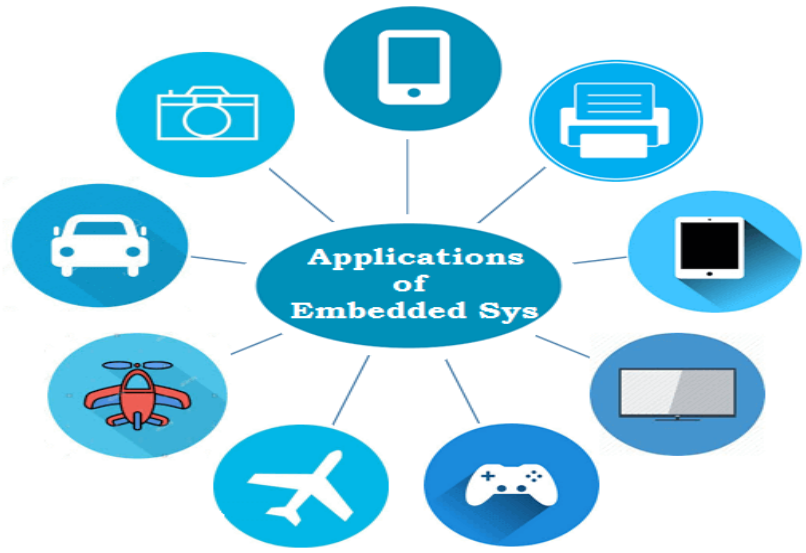

Fig. 2.2 applications of embedded systems

CHAPTER-3

BLOCK DIAGRAM AND DESCRIPTION

3.1 BLOCK DIAGRAM

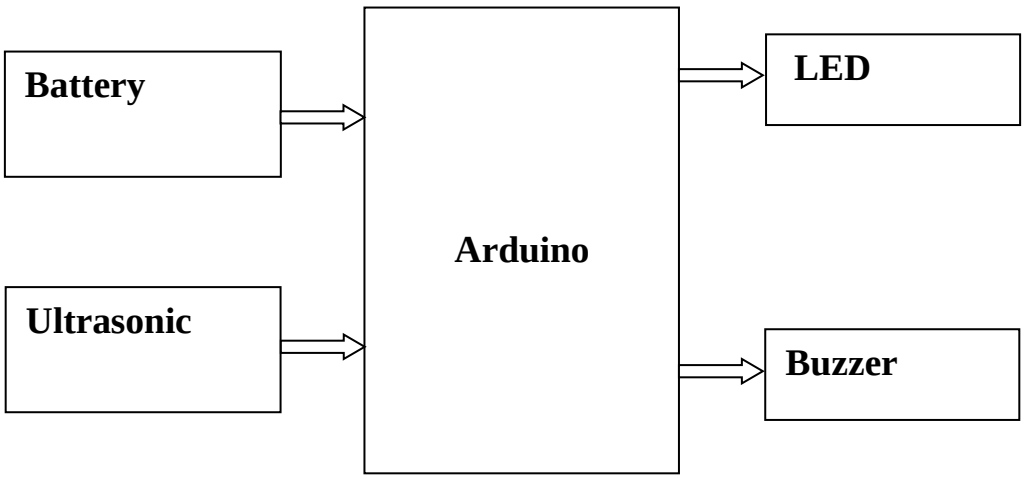

Fig 3.1 Block diagram

This block diagram describes the working flow of our project initially it consists of Arduino Uno, Battery and ultrasonic sensor, buzzer and led. Our project initially starts with the working of Arduino and will be waiting for the data from the sensor.

The Arduino Uno is a microcontroller board based on the ATmega328. It has 20 digital input/output pins (of which 6 can be used as PWM outputs and 6 can be used as analog inputs), a 16 MHz resonator, a USB connection, a power jack, an in-circuit system programming (ICSP) header, and a reset button

Ultrasonic sensors work by emitting sound waves at a frequency too high for humans to hear. They then wait for the sound to be reflected back, calculating distance based on the time required. This is similar to how radar measures the time it takes a radio wave to return after hitting an object.

The vibrating disk in a magnetic buzzer is attracted to the pole by the magnetic field. When an oscillating signal is moved through the coil, it produces a fluctuating magnetic field which vibrates the disk at a frequency equal to that of the drive signal.

A light-emitting diode is a two-lead semiconductor light source. It is a p–n junction diode that emits light when activated. When a suitable voltage is applied to the leads, electrons are able to recombine with electron holes within the device, releasing energy in the form of photon

Ultrasonic sensor works on the principle of ultrasonic sound waves it gives the exact distance from the target to the destination and this will give distance of 2cm to 450cm

Once this process is done it will give the information to Arduino, will give information in the form of buzzer or led.

3.2 HARDWARE DISCRIPTION

3.2.1 INTRODUCTION TO AURDINO UNO

Arduino Uno is a microcontroller board based on 8-bit ATmega328P microcontroller. Along with ATmega328P, it contains other components such as crystal oscillator, serial communication, voltage regulator, etc. to support the microcontroller.

3.2.1 Arduino UNO Board

Arduino Uno has 14 digital input/output pins (out of which 6 can be used as PWM outputs), 6Analog input pins, a USB connection, A Power barrel jack, an ICSP header and a reset button.

Arduino **is** a great tool for developing interactive objects, taking inputs from a variety of switches or sensors and controlling a variety of lights, motors and other outputs**.** Arduin**o** projects can be stand-alone or they can be connected to a computer using USB The Arduino UNO has only 32K bytes of Flash memory and 2K bytes of SRA that is more than 100,000 times LESS physical memory than a low-end PC! And that's not even counting the disk drive! Working in this minimalist environment, you must use your resources wisely.

ARDUINO UNO PIN DIAGRAM

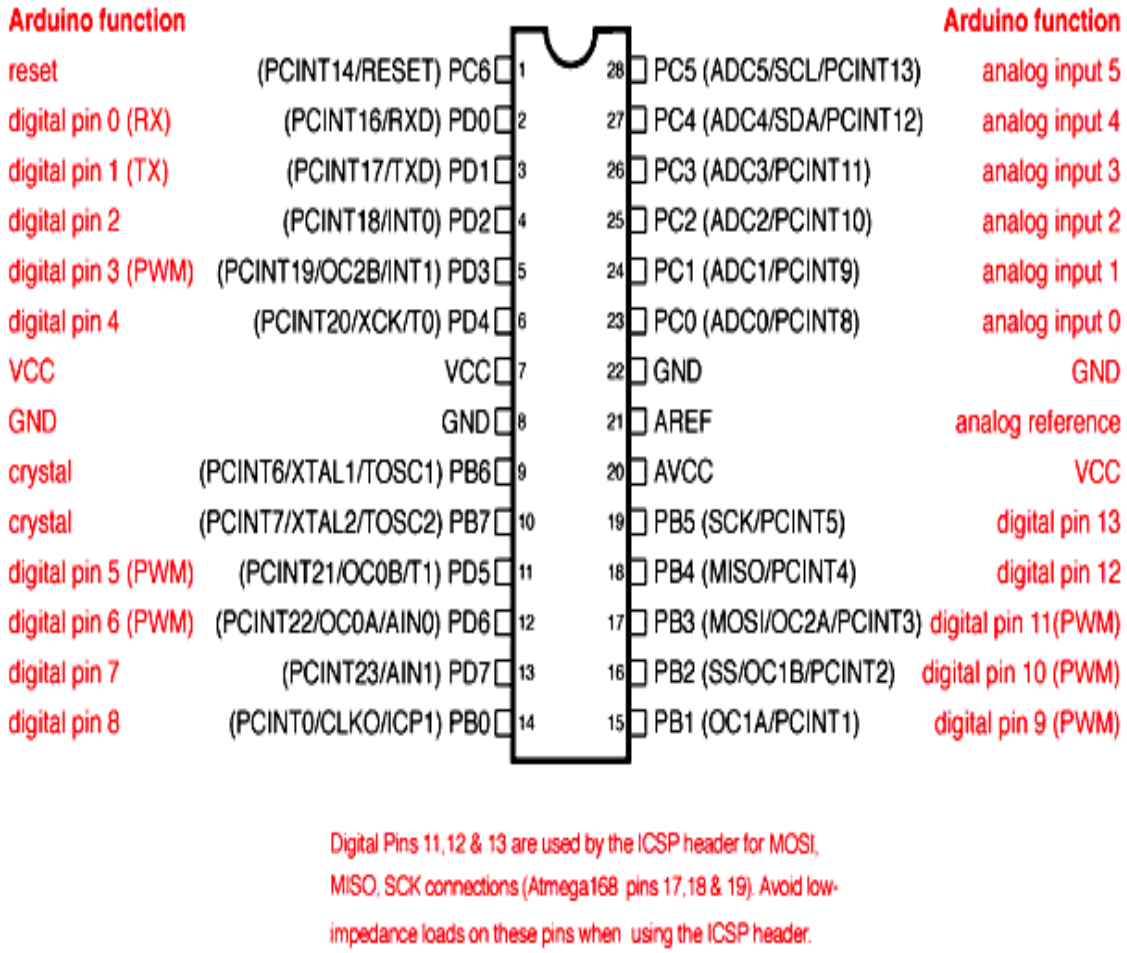

Fig 3.2.2 ARDUINO UNO PIN DIAGRAM

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to 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. Arduino can input and output Analog signals as well as digital signals. An Analog signal is one that can take on any number of values, unlike a digital signal which has only two values: HIGH and LOW.

The function used to output a PWM signal is Analog Write (pin, value). Pin is the pin number used for the PWM output.

ARDUINO UNO PIN SPECFICITIONS

Pin Number	Pin Name	Function
1	PC6	Reset
2	PD0	Digital Pin (RX)
3	PD1	Digital Pin (TX)
4	PD2	Digital Pin
5	PD3	Digital Pin (PWM)
6	PD4	Digital Pin
7	Vcc	Positive Voltage (Power)
8	GND	Ground
9	XTAL 1	Crystal Oscillator
10	XTAL 2	Crystal Oscillator
11	PD5	Digital Pin (PWM)
12	PD6	Digital Pin (PWM)
13	PD7	Digital Pin
14	PB0	Digital Pin
15	PB1	Digital Pin (PWM)
16	PB2	Digital Pin (PWM)
17	PB3	Digital Pin (PWM)
18	PB4	Digital Pin
19	PB5	Digital Pin
20	AVCC	Positive voltage for ADC (power)
21	AREF	Reference Voltage
22	GND	Ground
23	PC0	Analog Input
24	PC1	Analog Input
25	PC2	Analog Input
26	PC3	Analog Input
27	PC4	Analog Input
28	PC5	Analog Input

Table 3.2.1 ARDUINO UNO PIN SPECIFICATIONS

ARDUINO UNO TECHNICAL SPECIFICATION

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

Table 3.2.2 ARDUINO UNO TECHNICAL SPECIFICATIONS

How to use Arduino Board

The 14digital input/output pins can be used as input or output pins by using Pin Mode (), digital Read () and digital Write () functions in Arduino programming.

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

•**Serial Pins 0 (Rx) and 1 (Tax):** Rx and Tax pins are used to receive and transmit TTL serial data. They are connected with the corresponding ATmega328P USB to TTL serial chip.

•**External Interrupt Pins 2 and 3:** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.

•**PWM Pins 3, 5, 6, 9 and 11:** These pins provide an 8-bit PWM output by using Analog Write () function.

•**SPI Pins 10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK):** These pins are used for SPI communication.

•**In-built LED Pin 13:** This pin is connected with a built-in LED, when pin 13 is HIGH – LED is on and when 13pin is LOW, it's off.

Along with 14 Digital pins, there are 6 Analog input pins, each of which provides 10 bits of resolution, i.e. 1024 different values. They measure from 0 to 5 volts but this limit can be increased by using AREF pin with Analog Reference () function.

•Analog pin 4 (SDA) and pin 5 (SCA) also used for TWI communication using Wire library.

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

•**AREF:** Used to provide reference voltage for Analog inputs with Analog Reference () function.

•**Reset Pin:** Making this pin LOW, resets the microcontroller

•**COMMUNICATION**

The Arduino/genuine 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.

Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. A Software Serial library allows serial communication on any of the Uno's digital pins

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to 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.

Functions allow structuring the programs in segments of code to perform individual tasks. The typical case for creating a function is when one needs to perform the same action multiple times in a program. Functions help the programmer stay organize

3.2.2 ULTRASONIC SENSOR

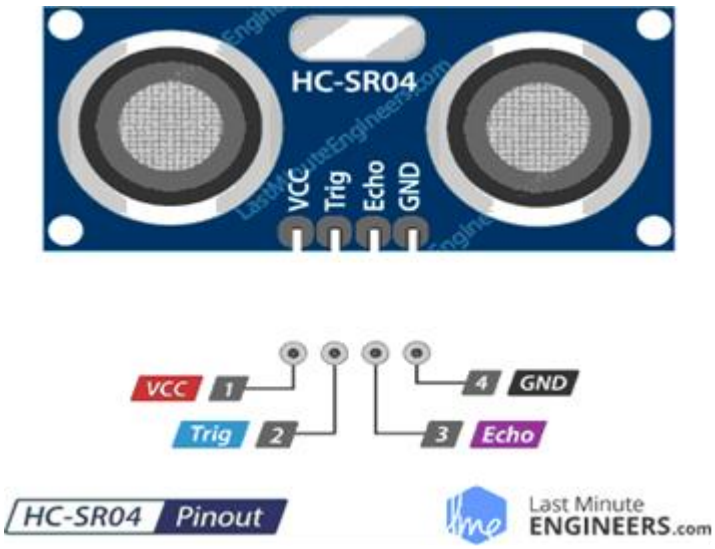

Fig 3.2.3ULTRASONIC SENSOR

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. It comes complete with ultrasonic transmitter and receiver modules.

What is Ultrasound?

Ultrasound is high-pitched sound waves with frequencies higher than the audible limit of human hearing.

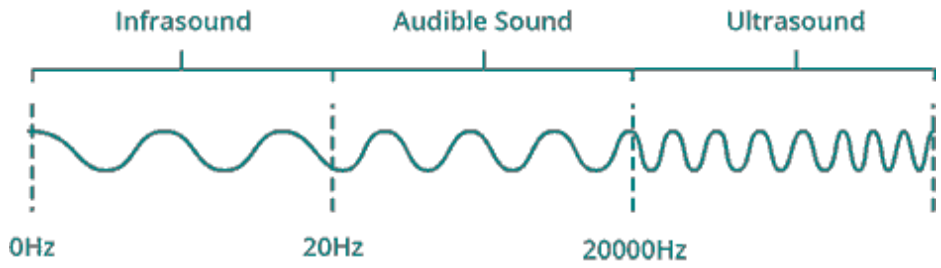

Fig 3.2.4 ULTRA SOUND

Human ears can hear sound waves that vibrate in the range from about 20 times a second (a deep rumbling noise) to about 20,000 times a second (a high-pitched whistling) However, ultrasound has a frequency of over 20,000 Hz and is therefore inaudible to humans

Features

Here's a list of some of the HC-SR04 ultrasonic sensor features and specs:

•Power supply: +5V DC

•Quiescent Current: <2mA

•Working Current: 15mA

•Effectual Angle: <15°

•Ranging Distance: 2cm – 400 cm/1″ – 13ft

•Resolution: 0.3 cm

•Measuring Angle: 30 degree

•Trigger Input Pulse width: 10uS

•Dimension: 45mm x 20mm x 15mm

At its core, the HC-SR04 Ultrasonic distance sensor consists of two ultrasonic. The one acts as a transmitter which converts electrical signal into 40K Hz ultrasonic sound pulses.

The receiver listens for the transmitted pulses. If it receives them it produces an output pulse whose width can be used to determine the distance the pulse travelled. As simple as pie!

The sensor is small, easy to use in any robotics project and offers excellent non-contact range detection between 2 cm to 400 cm (that's about an inch to 13 feet) with an accuracy of 3mm.

Since it operates on 5 volts, it can be hooked directly to an Arduino or any other 5V logic microcontrollers

Ultrasonic sensors work by emitting sound waves at a frequency too high for humans to hear. They then wait for the sound to be reflected back, calculating distance based on the time required. This is similar to how radar measures the time it takes a radio wave to return after hitting an object.

3.2.3 BUZZER

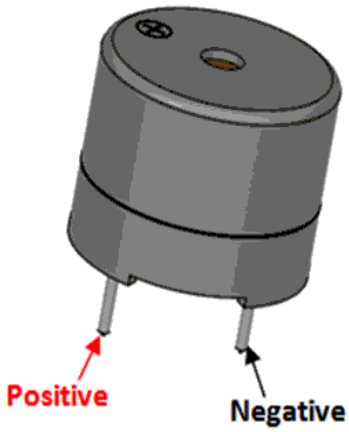

Fig 3.2.5 BUZZER Pin description

Buzzer Pin Configuration

Pin Number	Pin Name	Description
1	Positive	Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC
2	Negative	Identified by short terminal lead. Typically connected to the ground of the circuit

Table 3.2.3 buzzer pin configuration

Buzzer Features and Specifications

•Rated Voltage: 6V DC

•Operating Voltage: 4-8V DC

•Rated current: <30mA

•Sound Type: Continuous Beep

•Resonant Frequency: ~2300 Hz

•Small and neat sealed package

•Breadboard and Per board friendly

This buzzer is an active buzzer, which basically means that it will buzz at a predefined frequency (2300 ±300 Hz) on its own even when you just apply steady DC power. If you are looking for a buzzer can produce varied tones from an oscillating input signal, then take a look at our passive buzzer

Prefer to get active buzzers since they can use them with steady DC power but also be able to produce some variety of tones by applying an oscillating signal. Some consider them to be more versatile than their cousin, the passive buzzer, which is the type that requires an oscillating signal to create any tone.

It is possible, and often done, to still create different tones through an active buzzer when you apply an oscillating signal to the buzzer, but the spectrum of possible different tones is very limited and not as crisp or clean of sound as can be produced with a passive buzzer

Advantage to an active buzzer is that you can still produce a sound from the buzzer connected to a microcontroller, such as an Arduino, by just driving a standard high output on the connected pin. The benefits of this are that you don't need to use processing power, hardware timers, or additional code to produce sound.

Fig 3.2.6 BUZZER

3.2.4 LED

LED stands for Light Emitting Diode. LEDs began as exciting but expensive electronic components in the sixties, used in handheld calculators and other similar devices. Through research and development, LED technology advanced, became more efficient and less expensive, until it reached its current form.

LEDs can now be used for a number of lighting applications and are available across the spectrum of visible, infrared, and ultraviolet light. Affordable 12V LED lights, for example, are often used as a conventional lighting source in homes, offices, and places of business because they are more energy-efficient, last longer, are more physically durable, and are safer than incandescent lighting sources.

Fig 3.2.7 LEDs

However, the same technology that allows LEDs to produce light with less electricity use also means that it has different hardware requirements than incandescent lights. Proper 12V LED light installation includes the use of specific power adapters or drivers to regulate the voltage and heat that reaches the Light Emitting Diode, so that the diodes only receives the right amount of electricity and continues to function properly.

CHAPTER 4

SOFTWARE DESCRIPTION

4.1 Arduino IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code. It connects to the Arduino and Genuine hardware to upload programs and communicate with them.

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension.

The editor has features for cutting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors.

The console displays text output by the Arduino Software (IDE), including complete error messages and other information.

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data.

#include statements will insert one or more at the top of the sketch and compiles the library with your sketch.

Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #include statements from the top of your code.

If you want to program your Arduino Uno while offline you need to install the Arduino Desktop (IDE)

The Uno is programmed using the Arduino Software (IDE), our Integrated Development Environment common to all our boards.

Before you can move on, you must have installed the Arduino Software (IDE) on your PC

The serial monitor displays serial sent from the Arduino or Genuine board over USBor serial connector.

To send data to the board, enter text and click on the "send" button or press enter.

Note that on Windows, Mac or Linux the board will reset (it will rerun your sketch) when you connect with the serial monitor.

Please note that the External terminal program and connect it to the COM port assigned to your Arduino board Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an SERIAL MONITOR

Fig 4.1 Arduino UNO

Arduino was born at the 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

4.2ARDUINO SETUP AND INSTIALLISATION

The Uno is programmed using the Arduino Software (IDE).Connect your Uno board with an USB cable

Step1

Most probably you need to go to the arduino website and download the software

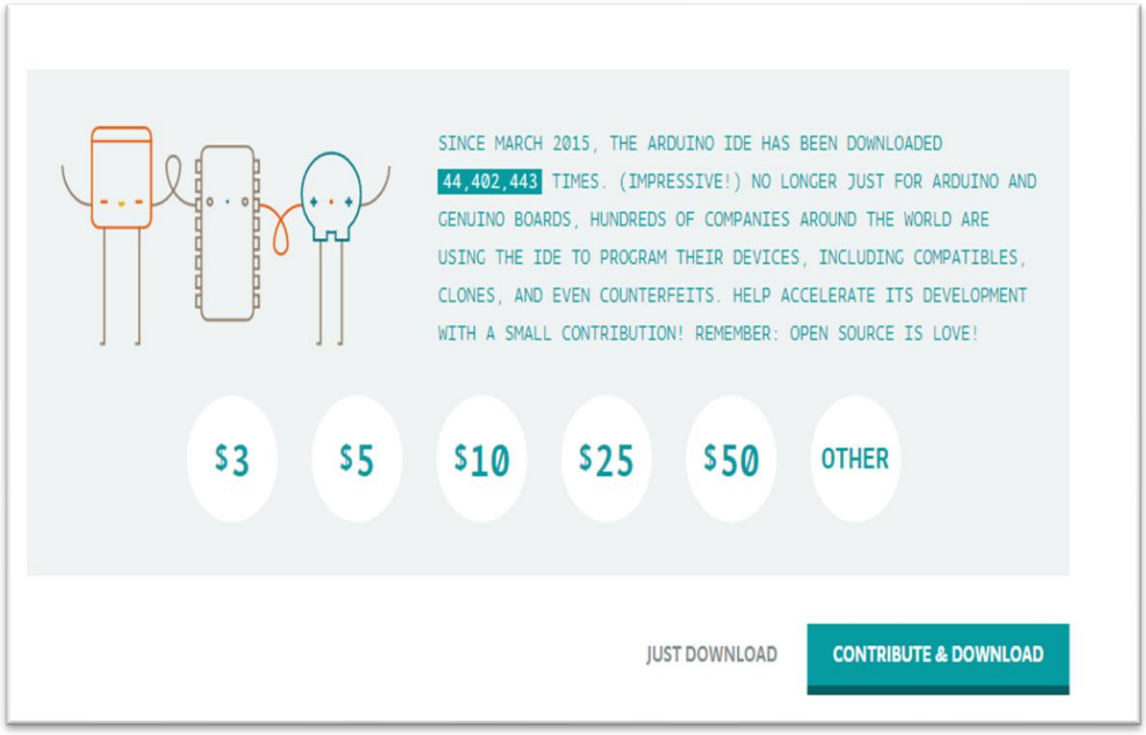

Fig 4.2 downloading Arduino IDE

If you want to program your Arduino Uno while offline you need to install the Arduino Desktop IDE.

The Uno is programmed using the Arduino Software (IDE), our Integrated Development Environment common to all our boards.

Step 2

Initialization of Arduino IDE

After downloading, locate the downloaded file on the computer and extract the folder from the downloaded zipped file. Copy the folder to a suitable place such as your desktop.

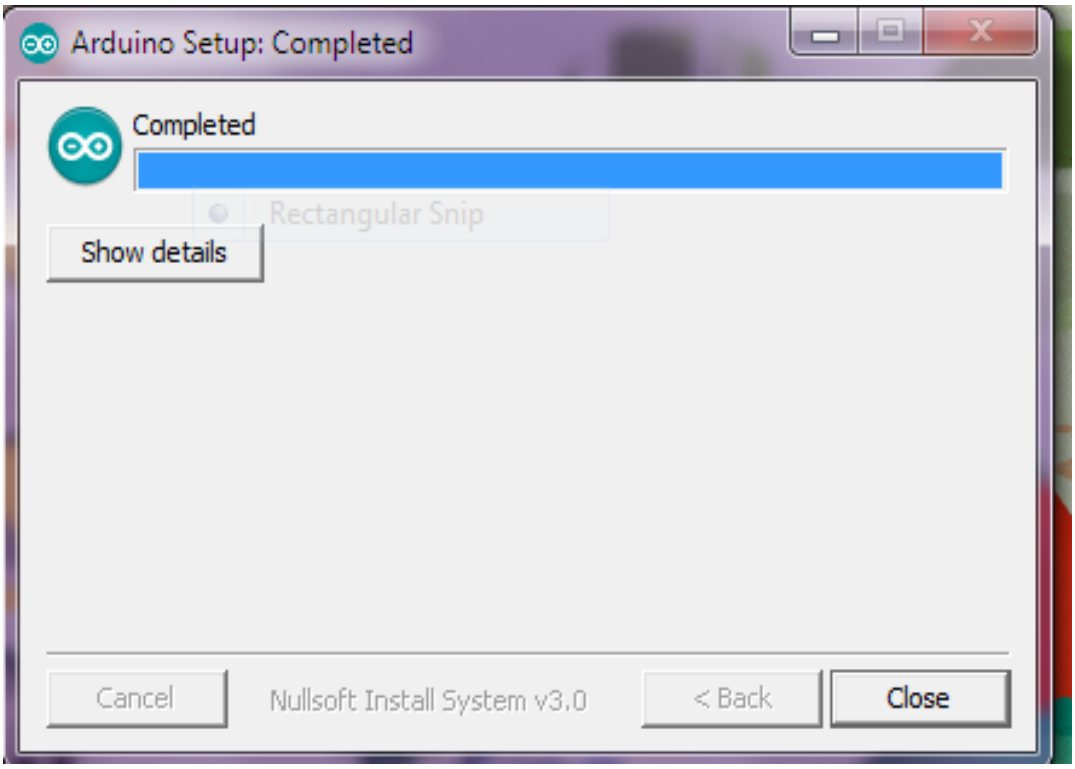

Fig4.3 Arduino setup

Step 3

If you downloaded you need to follow the procedure step by step

Click on the Start Menu, and open up the Control Panel.

While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager

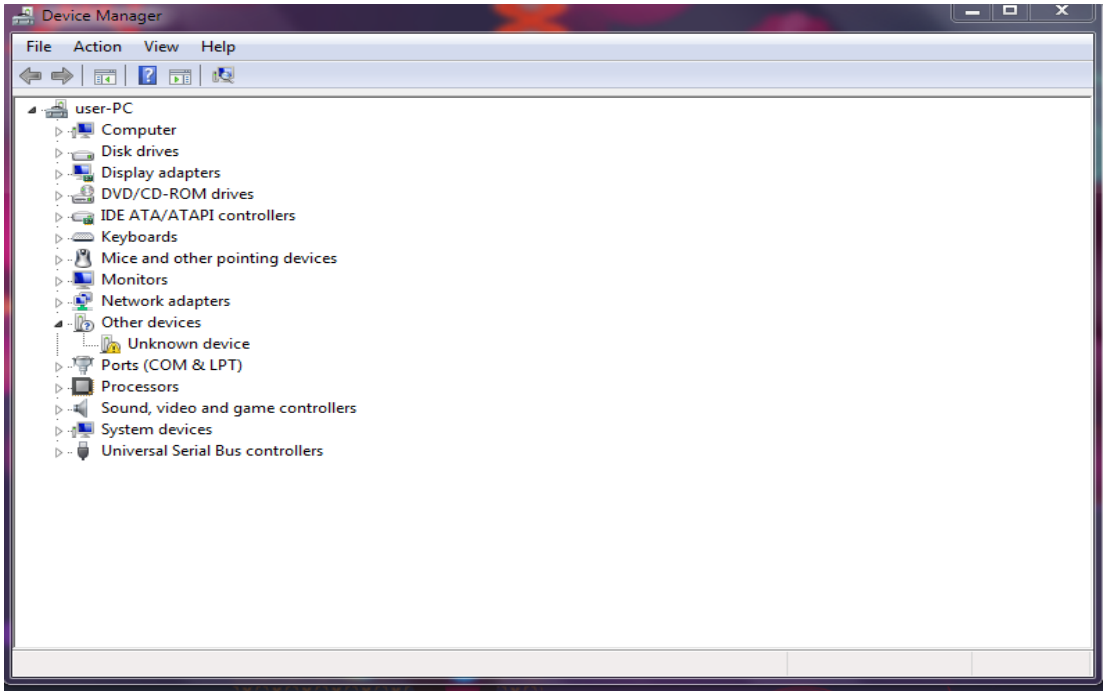

Fig 4.4 device manager

Step 4

Connecting Arduino UNO

Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COM xx)". If there is no COM & LPT section, look under "Other Devices" for "Unknown Device".

You will be able to see Arduino Uno after connecting the Arduino Uno only

Right click on the "Arduino UNO (COMxx)" port and choose the "Update Driver Software" option.

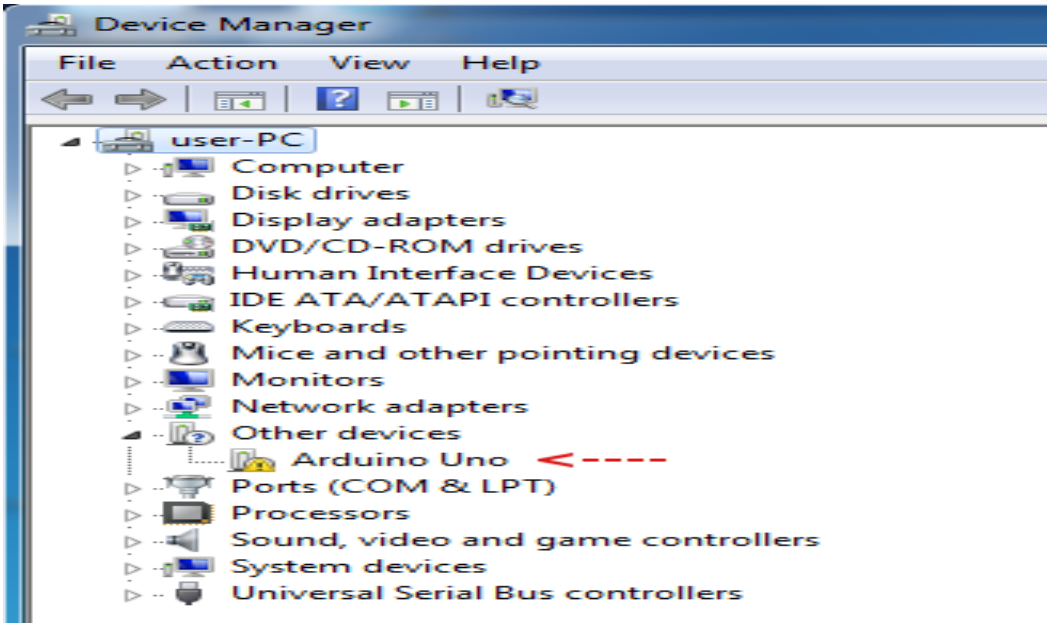

Fig 4.5device manager>Arduino Uno

Step 5

- Installing the device driver

In the Device Manager Window, right-click the Arduino board and then click **Update Driver Software...** on the pop-up menu:

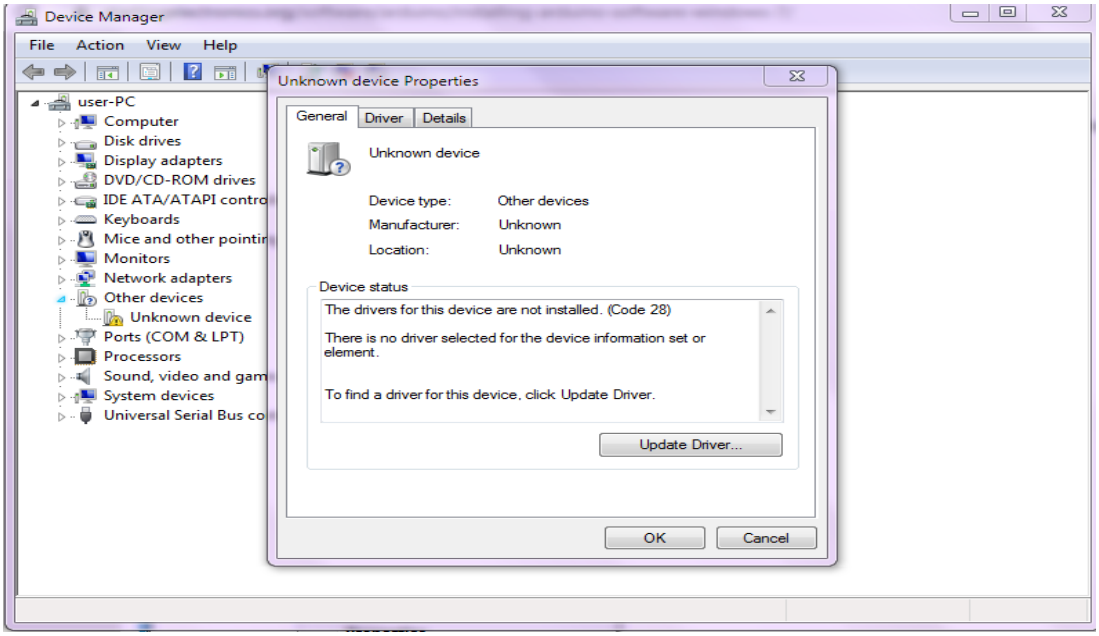

Fig 4.6 driver software status (finding driver for device)

Step 6

Next, choose the "Browse my computer for Driver software" option.

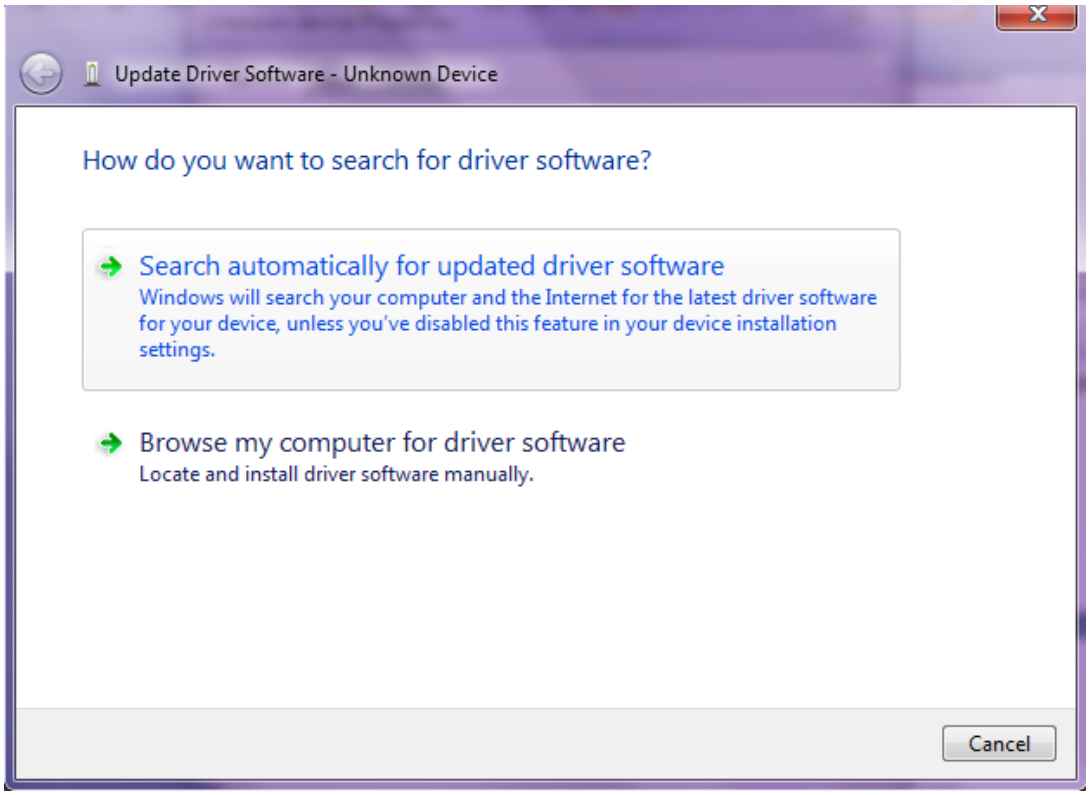

Fig 4.7 Locating and installing driver software manually

Step 7

Finally, navigate to and select the driver file named "arduino.inf",

It is located in the "Desktop" folder of the Arduino Software download not the "FTDI USB Drivers" sub-directory. If you are using an old version of the IDE older one, choose the Uno driver file named "ArduinoUNO.inf"

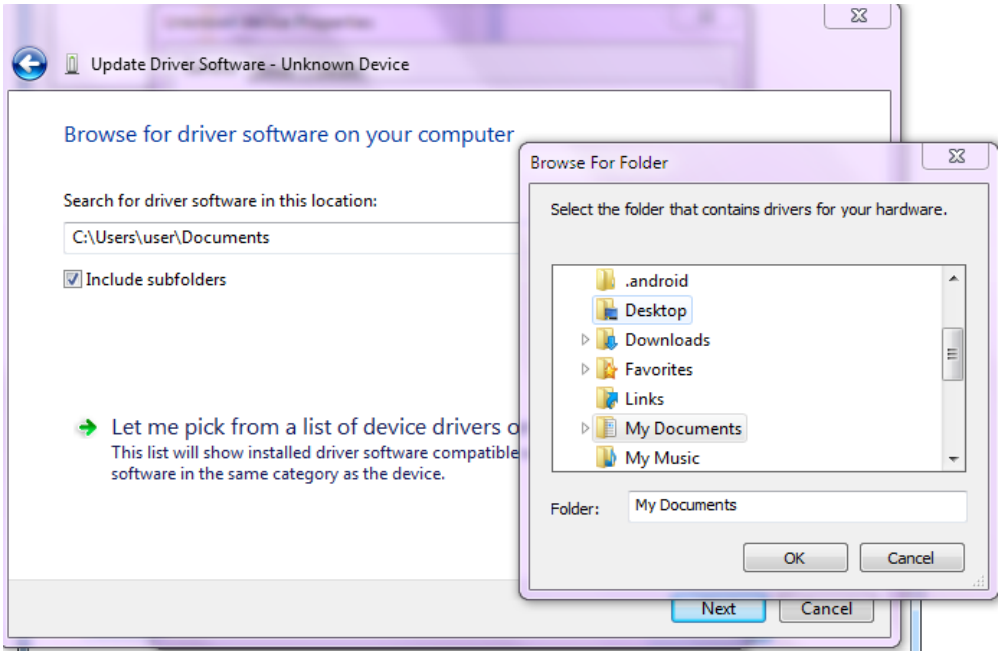

Fig 4.8 Navigating folder

Step 8

After selecting the driver folder, click the **Next** button

Fig 4.9 Browse for driver software

Step 9

If, windows was unable to detect any software in your pc/device then installing driver software is necessary

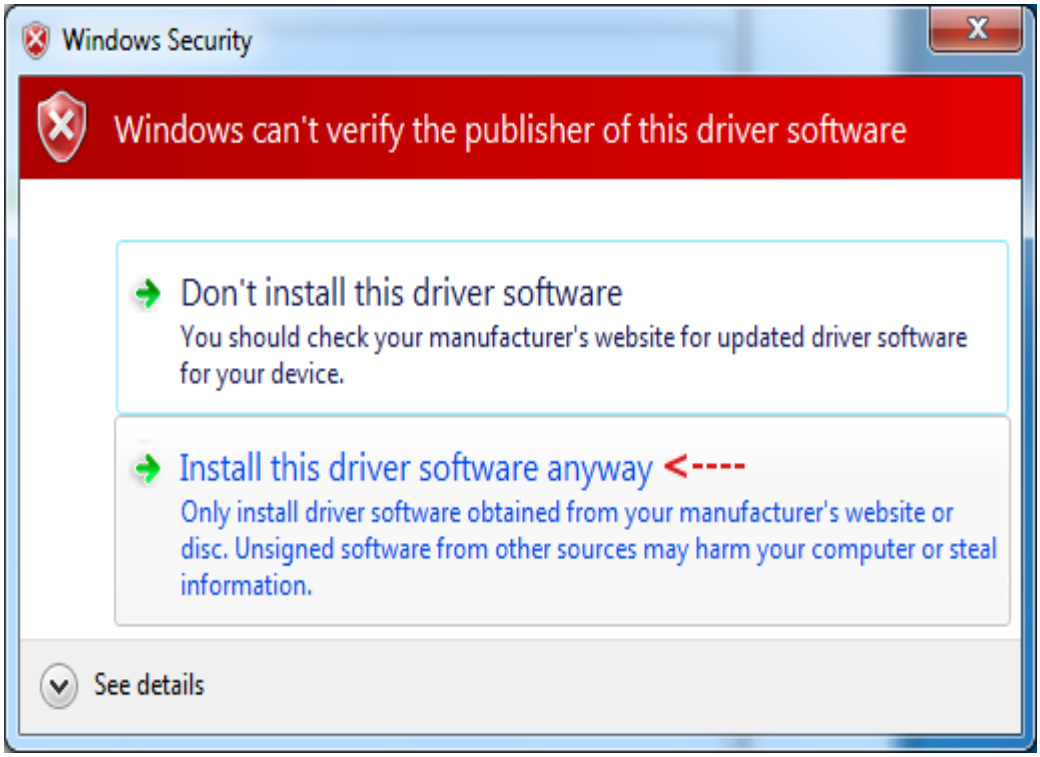

Fig 4.10 Driver software

After some time, the driver installation will finish and you will see the following dialog box. Take note of the port that the Arduino was configured as. In this case it was **COM3**.

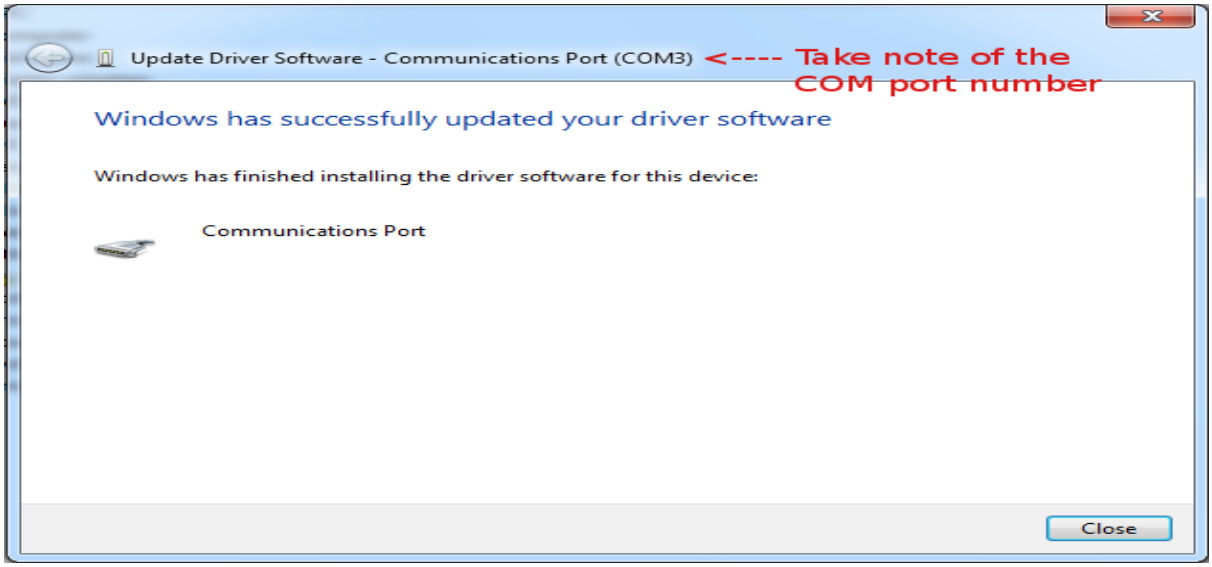

Fig 4.11Communication port

Step 10

Setting up the Arduino Software

The setup will only need to be done once, unless you change the board type or port that the Arduino is connected to.

Navigate to the folder that you downloaded and start the Arduino software IDE by double-clicking the Arduino application:

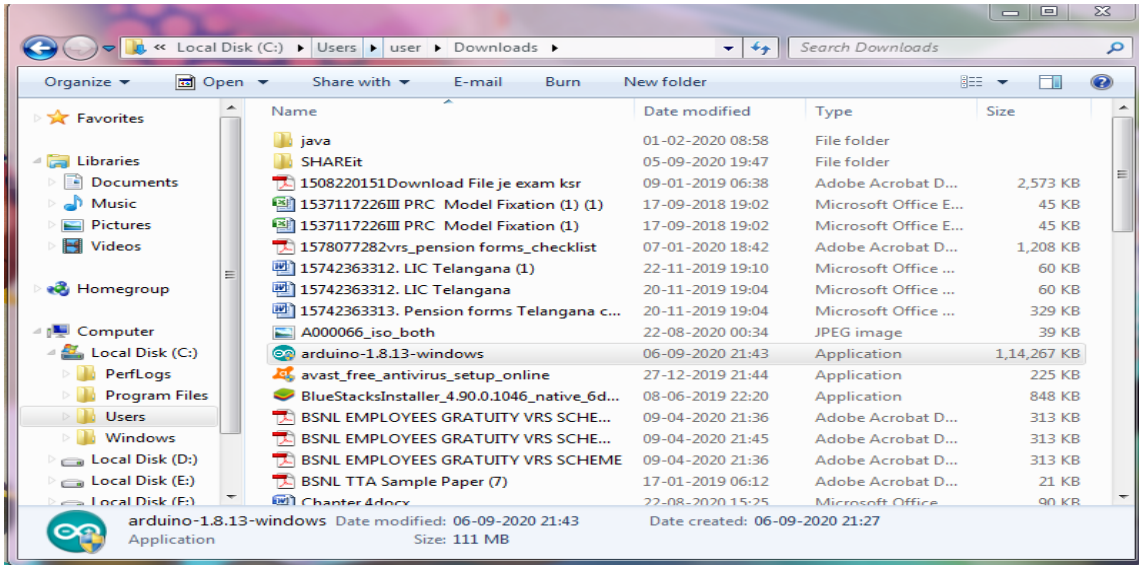

Fig 4.12 starting the Arduino software

Step 11

Checking the correct Arduino board is selected. Change if necessary

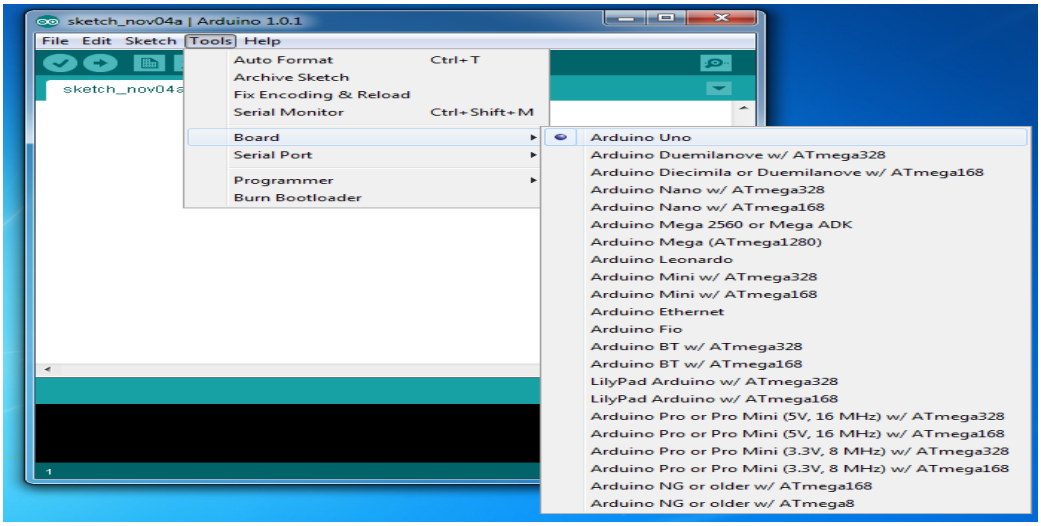

Fig 4.13 checking the Arduino board type

Step 12

Checking that the correct serial port is selected

Now check that the correct serial port is selected and change if necessary. This is the serial port that you took note of after installing the Arduino driver.

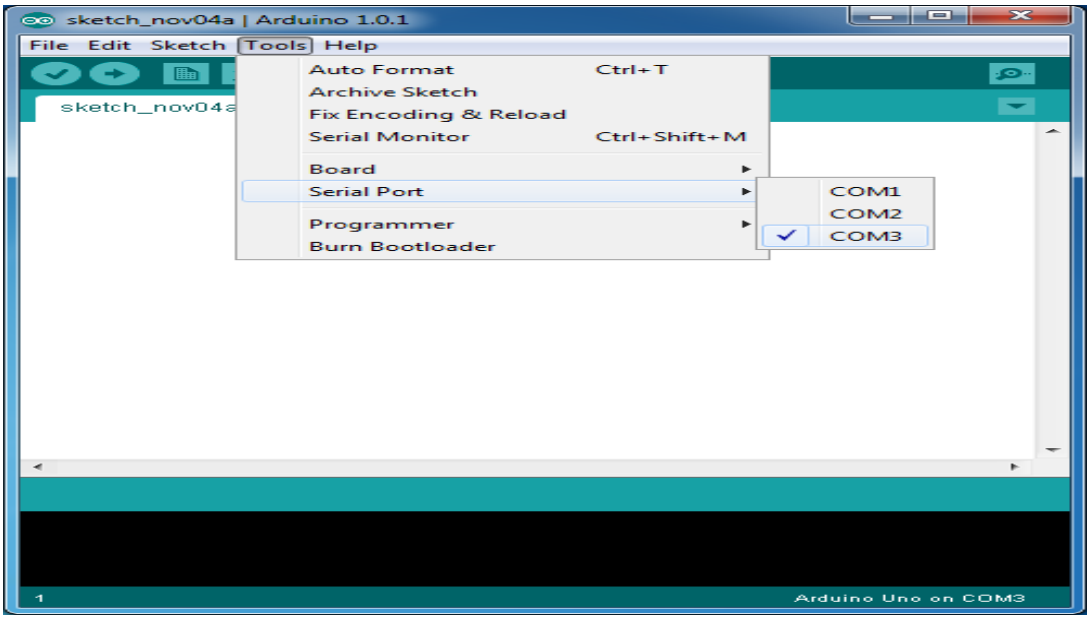

Fig 4.14 checking the serial port

Step 13

Checking the installation

Open the Blink sketch in the Arduino IDE:

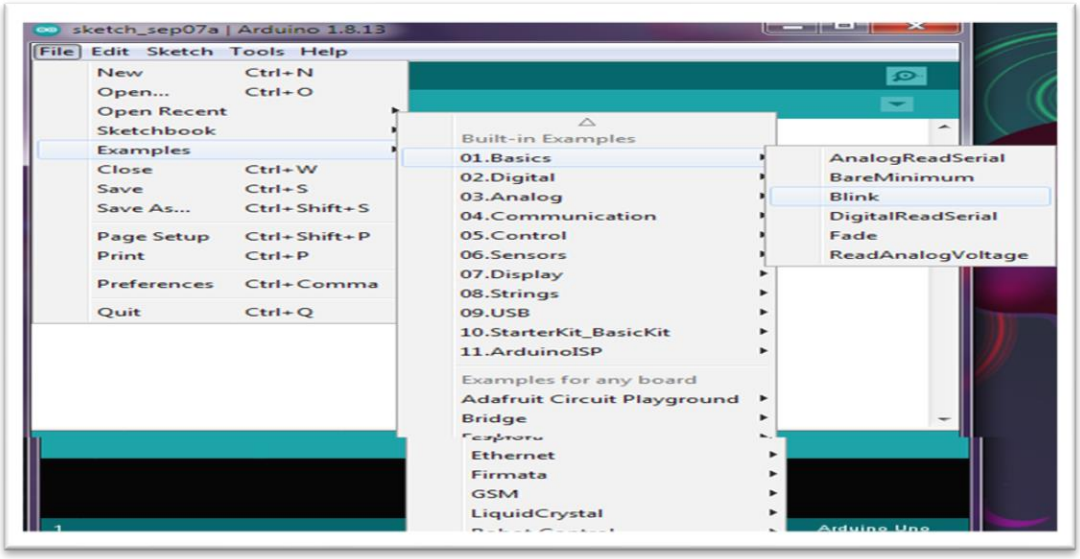

Fig 4.15 testing the installation-open blink

Finally, Click the Arduino Upload button to load the sketch to the Arduino. This sketch will flash the on-board LED on the Arduino. If the sketch runs then you know that you have successfully installed the Arduino software and drive.

4.3 DEVELOPING THE CODE

It consists of an ultrasonic transmitter and an ultrasonic receiver. First ultrasonic transmitter sends an ultrasonic wave. This signal will collide with the object and reflect the signal. The receiver will receive the reflected signal. The distance calculated by the time taken to receive the reflected and the speed of sound in air. The speed of sound in air at room temperature is 340 Meter/Second or 0.034 centimeter/microsecond. The equation for calculating time

Using,

Time=Distance/speed of sound

If the object is 5 CM away from the sensor, you will get the time as per the equation is,

5/0.034 = 147.055 Microseconds

But you will get the value from the Echo pin is 294.11. This is because of the sound wave needs to travel forward and bounce backward. So we need to divide that value by two for getting the actual value (time). Here we want to calculate the distance from the time. So re-arrange the equation we will get,
Distance=Time x speed of sound

Time to start:

Open the ARDUINO IDE.

// defines pins numbers

const int trigPin = 9;

const int echoPin = 10;

const int buzzer = 11;

const int ledPin = 13;

```
// defines variables
long duration;
int distance;
int safetyDistance;



void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
pinMode(buzzer, OUTPUT);
pinMode(ledPin, OUTPUT);
Serial.begin(9600); // Starts the serial communication
}



void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
```

```
digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);

// Calculating the distance
distance= duration*0.034/2;

safetyDistance = distance;
if (safetyDistance <= 5){
  digitalWrite(buzzer, HIGH);
  digitalWrite(ledPin, HIGH);
}
else{
  digitalWrite(buzzer, LOW);
  digitalWrite(ledPin, LOW);
}

// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
}
```

CHAPTER-5

WORKING OF THE PROJECT

5.1 DESCRIPTION OF THE PROJECT

The Smart Blind Stick scans the path in front of it using the ultrasonic sensor.

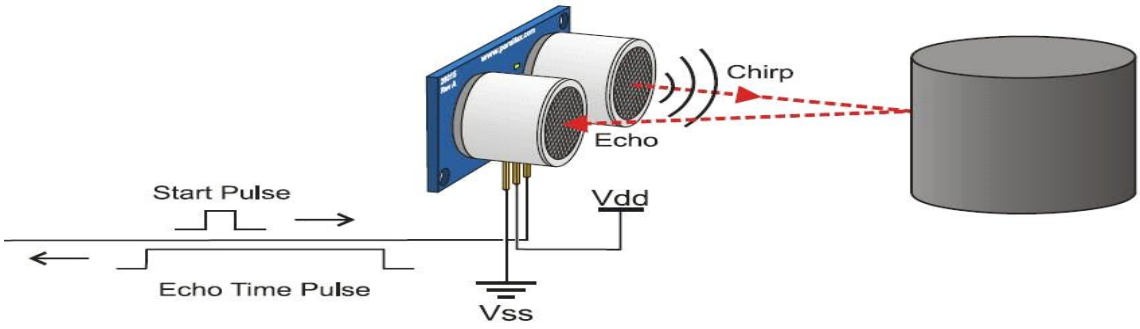

Fig 5.1 ultrasonic sensor functioning.

- When the sensor detects an object (such as a vehicle or obstacle), the following actions occur:
    - The buzzer starts beeping to alert the user.
    - Simultaneously, an LED turns on, providing visual feedback.
- The blind person can hear the buzzer and adjust their path accordingly, avoiding potential collisions or injuries.
- Real-time distance values (in centimeters) can also be monitored via the Arduino serial monitor.

CHAPTER-6

RESULT

The Smart Blind stick when tested it was able to successfully detect all the obstacles and within the range of 50cm the buzzer was alarming and light was also blinking. hence we are successful in our project.

CHAPTER-7

7.1 APPLICATIONS

- **Assistive Mobility**: The primary application is to assist visually impaired individuals in navigating their surroundings safely.
- **Outdoor Navigation**: Whether on sidewalks, streets, or public spaces, the smart stick helps users avoid obstacles.
- **Indoor Navigation**: It can also be used indoors, such as in buildings, malls, or public transportation hubs.

7.2 ADVANTAGES

- **Obstacle Detection**: The ultrasonic sensor detects obstacles in real time, providing immediate feedback.
- **Auditory and Visual Alerts**: The combination of a buzzer (auditory) and LED (visual) ensures effective communication.
- **User-Friendly**: The stick is lightweight, portable, and easy to use.
- **Cost-Effective**: Arduino-based solutions are affordable and accessible.
- **IoT Integration (Optional)**: Adding IoT capabilities allows remote monitoring and assistance.

CHAPTER-8

CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

This prototype of sticks for blind people that uses sensor technology to aid alertness and movement in the blind. The blind can sense objects up to 70 centimetres away and receive feedback in the form of sound and vibration. At this point, it's worth noting that the study's primary objective, which was to create and launch a smart walking stick for the blind, was fully achieved. The Smart Stick is a building block for the next generation of assistive technology that can help the visually impaired navigate both indoor and outdoor environments safely. It is both effective and powerful. Within a three-meter range, it performs well in detecting in the user's direction. With a noticeable short response time, this system provides a low-cost, dependable, lightweight, low-power, and robust navigation solution. Despite being hard-wired with sensors and other parts, the system is lightweight. Ultimately , the continued development and implementation of such tools can bring happiness into the lives visually impaired individuals.

8.2 FUTURE SCOPE

Further enhancements can be made:

- **Advanced Feedback Mechanisms**: Explore using additional sensors (e.g., vibration motors, voice feedback).
- **Navigation Assistance**: Integrate GPS or other location-based services.
- **Connectivity**: Consider adding Bluetooth or Wi-Fi for smartphone integration.
- **Machine Learning**: Train models to recognize specific obstacles or environments.

REFERENCES

Patel, J., & Singh, A. (2019). Smart stick for blind using Arduino and ultrasonic sensors. *International Journal of Engineering and Advanced Technology, 8(5), 2311-2314.*

Kumar, P., & Sharma, R. (2020). Development of an Arduino-based blind navigation system. *Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, 8(3), 123-127.*

Nair, A., & Mathew, L. (2021). Smart walking stick for visually impaired using Arduino and IoT. *International Journal of Computer Applications, 183(37), 7-12.*

Dhar, A., & Rahman, S. (2020). Design and implementation of a smart blind stick using Arduino and sensors. *International Journal of Advanced Research in Computer Science and Software Engineering, 10(8), 250-255.*

Pawar, R., & Kulkarni, S. (2019). Smart cane for visually impaired using Arduino and ultrasonic sensors. *International Journal of Innovative Technology and Exploring Engineering, 9(1), 523-526.*

Gupta, V., & Verma, R. (2021). IoT-based smart blind stick using Arduino and GPS module. *International Journal of Scientific and Research Publications, 11(2), 45-50.*

Suresh, K., & Kumar, M. (2020). Arduino-based smart stick for visually impaired persons. *International Journal of Research in Engineering and Technology, 9(4), 112-116.*

Rathore, A., & Choudhary, R. (2021). Smart blind stick for obstacle detection and navigation. *Journal of Electrical Engineering and Automation, 13(5), 334-338.*

Reddy, K., & Ramakrishna, S. (2019). Design of a smart stick for visually impaired using Arduino. *International Journal of Engineering Research & Technology, 8(7), 196-199.*

Mishra, A., & Sen, P. (2020). Implementation of a smart blind stick using Arduino and voice recognition module. *International Journal of Innovative Research in Science, Engineering and Technology, 9(8), 4500-4505.*