CHAPTER – 1

INTRODUCTION

1.1 OVERVIEW OF THE ADJUSTABLE ROAD DIVIDER FOR DYNAMIC TRAFFIC MANAGEMENT USING IOT

Traffic congestion is a growing concern in urban and metropolitan areas due to the increasing number of vehicles on the road. Conventional traffic management systems rely on fixed road dividers, which do not adapt to varying traffic patterns, leading to inefficiencies during peak hours. To address this issue, smart and automated solutions are required to dynamically manage road lanes based on real-time traffic conditions. The Adjustable Road Divider for Dynamic Traffic Management using IoT offers an intelligent, automated approach to modifying road layouts based on live traffic density, ultimately reducing congestion and improving road efficiency.

The system is designed using Arduino UNO as the central microcontroller, which processes real-time traffic data collected through Infrared (IR) sensors. These sensors detect the number of vehicles in each lane and send the data to the controller, which then determines whether the road divider needs to be shifted. The dynamic lane reallocation process is displayed on an LCD screen, providing real-time updates for drivers and traffic authorities. This automated adjustment ensures optimal lane distribution, allowing for improved traffic flow and reduced wait times at intersections.

To implement the physical movement of the road divider, a Rack and Pinion System (RPS) is used, driven by a gear motor. This mechanical setup ensures smooth and controlled shifting of the divider, allowing for lane realignment without manual intervention. The gear motor provides the necessary torque to move the divider efficiently, while the RPS ensures precise and stable movement. By integrating this mechanism with IoT-based control, the system enables remote monitoring and automated adjustments, further enhancing traffic management efficiency.

The use of IR sensors for vehicle detection ensures accurate real-time data collection, which is crucial for making quick and effective lane adjustments. These sensors are strategically placed at different sections of the road to continuously monitor traffic density. If a significant imbalance in vehicle distribution is detected, the system automatically triggers the gear motor to shift the road divider, reallocating lanes

accordingly. Additionally, the system can be programmed to operate in pre-set time intervals based on historical traffic data, further optimizing road usage during predictable peak hours.

A major advantage of this system is its ability to improve road efficiency and reduce congestion without the need for additional infrastructure expansion. Traditional solutions, such as widening roads or constructing new flyovers, involve high costs and long implementation times. In contrast, the Adjustable Road Divider using IoT provides a cost-effective, adaptable, and scalable alternative that can be easily deployed on existing road networks. The integration of IoT also allows traffic authorities to monitor and control the system remotely, ensuring seamless operation even in dynamically changing traffic scenarios.

In conclusion, the Adjustable Road Divider for Dynamic Traffic Management using IoT represents a significant advancement in smart traffic control. By utilizing Arduino UNO, LCD displays, IR sensors, RPS, and gear motors, this system enables real-time lane adjustments, reducing congestion and optimizing road usage. With further enhancements such as AI-based traffic prediction and integration with smart city infrastructure, this technology has the potential to revolutionize modern urban transportation systems, making them more efficient, flexible, and sustainable.

CHAPTER -2

LITERATURE SURVEY

2.1 Satya Srikanth Palle , " Implementation of Smart Movable Road Divider and Ambulance Clearance using IOT", IEEE Xplore, May 2019.

The study by Satya Srikanth Palle , " Implementation of Smart Movable Road Divider and Ambulance Clearance using IOT", IEEE Xplore, May 2019.Across the World many Nations are facing Traffic Congestion problems due to the number of automobiles increasing day by day. Though the number of vehicles increased, the Road infrastructure is nearly the same which cannot be able to cope up with the changes like unpredictable travel delays, congestion and accidents. The main problem with Static Road Divider is that the number of lanes on either side of the road is fixed. Since the resources are limited and population as well as the number of cars per family is increasing, there is a significant increase in the number of cars on roads. Controlling of traffic on the road has become a severe problem in today's society.

2.2 Gupta, A., Verma, R., & Singh, P. "Automated Road Dividers for Traffic Congestion Reduction," pp. 120-125, IEEE Xplore, March 2020.

The study by Gupta, A., Verma, R., & Singh, P. titled "Automated Road Dividers for Traffic Congestion Reduction.” published in IEEE Xplore in March 2020.Traditional traffic management systems rely on static lane allocations, which often fail to accommodate varying traffic densities at different times of the day. This study explores motorized road dividers that shift lanes at preset intervals to balance congestion. The system effectively reduces bottlenecks but lacks real-time vehicle sensing. The authors emphasize the need for sensor-based dynamic adjustment rather than time-based movements, as traffic conditions can be unpredictable.

2.2 Sharma, V., & Kumar, R. "Implementation of IR and Ultrasonic Sensors for Smart Traffic Control," pp. 88-92, Springer, October 2021.

The study by Sharma, V., & Kumar, R. titled "Implementation of IR and Ultrasonic Sensors for Smart Traffic Control," Springer, October 2021.Accurate vehicle detection is essential for real-time traffic management. This research evaluates Infrared (IR) sensors and ultrasonic sensors for vehicle detection in smart traffic systems. IR sensors are found to be cost-effective and energy-efficient, providing

reliable vehicle counting under normal conditions. However, the study identifies challenges in dusty, foggy, or rainy environments, where IR sensor accuracy is reduced. To address this, the researchers propose using multiple redundant sensors and data fusion techniques to improve detection accuracy.

2.3 Reddy, K., Banerjee, A., & Joshi, M. "Microcontroller-Based Adaptive Traffic Signal System," pp. 215-220, Elsevier, June 2019.

The study by Reddy, K., Banerjee, A., & Joshi, M. titled "Microcontroller-Based Adaptive Traffic Signal System," pp. 215-220, Elsevier, June 2019.Microcontrollers such as Arduino UNO play a crucial role in automating intelligent traffic management systems. This study develops an adaptive traffic signal system that dynamically adjusts signal timings based on real-time vehicle density. The results indicate that Arduino-controlled systems significantly reduce waiting times at intersections. However, the study does not consider physical lane reallocation, which is critical for dynamic lane management. The authors suggest integrating automated lane adjustment mechanisms alongside adaptive traffic signals to optimize road capacity.

2.4 Patel, S., & Singh, R. "Rack and Pinion-Based Barrier Control for Railway Crossings," pp. 53-58, IEEE Transactions on Automation, September 2022.

The study by Patel, S., & Singh, R. titled "Rack and Pinion-Based Barrier Control for Railway Crossings," pp. 53-58, IEEE Transactions on Automation, September 2022 the use of Rack and Pinion Systems (RPS) with gear motors ensures smooth and precise mechanical movement in traffic control applications. This study focuses on railway crossings, where barriers automatically move based on train schedules. The gear motor-driven pinion smoothly moves the rack, ensuring stability and controlled movement. The study confirms that RPS-based mechanisms are highly reliable and can be adapted for road dividers to facilitate automated lane reallocation. However, the study does not explore its application in dynamic urban traffic environments, which presents a research opportunity for adjustable road dividers.

2.5 Khan, T., Alavi, N., & Rahman, M. "LCD-Based Real-Time Traffic Density Display for Smart Cities," pp. 302-307, IEEE Xplore, December 2020.

The study by Khan, T., Alavi, N., & Rahman, M. titled "LCD-Based Real-Time Traffic Density Display for Smart Cities," pp. 302-307, IEEE Xplore, December 2020 Effective traffic management requires clear visual communication with drivers. This research introduces an LCD-based real-time traffic density display system that informs drivers about lane allocation changes and congestion levels. The results demonstrate that drivers react more efficiently to visual cues, reducing accidents and confusion. The authors recommend integrating IoT-enabled LCD systems with dynamic lane management solutions to enhance public awareness and compliance in automated traffic systems.

2.6 Wong, H., & Lee, C. "IoT-Based Adaptive Traffic Control Using Smart Sensors and Cloud Computing," pp. 115-120, Springer, July 2021.

The study by Wong, H., & Lee, C. titled "IoT-Based Adaptive Traffic Control Using Smart Sensors and Cloud Computing," pp. 115-120, Springer, July 2021.The integration of IoT in traffic management has significantly improved real-time monitoring and control. This paper explores smart sensors and cloud computing for adaptive traffic management systems. The research highlights how IoT-based systems analyze real-time traffic data to optimize lane usage dynamically. The authors suggest that machine learning algorithms can be used to predict traffic congestion trends, allowing for proactive lane adjustments. However, the study lacks an implementation framework for automated mechanical road dividers, which remains an area for further development.

CHAPTER - 3

EXISTING SYSTEM

The current road traffic management system predominantly relies on fixed lane dividers and manual traffic control to regulate vehicle movement. These traditional methods are static in nature, making them inefficient during peak hours or emergency situations where traffic demand fluctuates dynamically. The existing system has several limitations, leading to congestion, longer travel times, and increased fuel consumption.

3.1 Fixed Road Dividers and Traffic Congestion

- In most urban and highway infrastructures, concrete or metal road dividers are permanently installed to separate traffic moving in different directions. However, this system fails to adapt to changing traffic patterns, resulting in:
- Unequal lane usage: Some lanes remain underutilized, while others experience heavy congestion.
- Rigid infrastructure: Changing lane configurations requires costly construction work and road closures.
- Traffic bottlenecks: Fixed dividers do not adjust in real-time to accommodate peak-hour traffic or special conditions (e.g., road repairs, accidents).
- Limitation: The system lacks flexibility, leading to inefficient road space utilization and frequent traffic jams.

3.2 Manual Traffic Control and Its Drawbacks

- Traffic police and operators at control centers manually regulate traffic flow using:
- Traffic signals and signboards.
- Lane barriers for temporary diversions.
- Manually operated movable dividers (in some cities).
- However, manual intervention is inefficient due to:
- Delayed response to real-time traffic conditions.
- High labor costs and increased workload for traffic authorities.
- Human errors leading to mismanagement and accidents.
- Limitation: This system does not provide automated lane adjustments, making it impractical for handling high-traffic scenarios efficiently.

3.3 Smart Traffic Lights and IoT-Based Traffic Systems (Limited Use Cases)

- Some cities have started integrating smart traffic management systems using IoT sensors, cameras, and AI-based algorithms to control signal timings dynamically. However, these systems:
- Focus primarily on traffic signals rather than lane allocation.
- Require significant infrastructure upgrades, making them costly.
- Still rely on fixed road dividers, limiting their efficiency in high-density traffic areas.
- Limitation: While smart traffic lights optimize vehicle flow at intersections, they do not solve the issue of rigid lane allocation on roads and highways.

3.4 Existing Movable Divider Systems (Limited Implementation)

- A few urban areas have experimented with manually or hydraulically adjustable lane dividers, but these systems are not widely implemented due to:
- High implementation costs.
- Complex mechanical structures requiring frequent maintenance.
- Lack of integration with IoT for automation and real-time adjustments.
- Limitation: Current movable divider systems are expensive and not automated, limiting their scalability in large cities.

3.5 Summary of Existing System Limitations

ASPECT	EXISTINGSYSTEM DRAWBACKS
Fixed Road Dividers	Cannot adapt to dynamic traffic conditions.
Manual Traffic Control	Inefficient, labor-intensive, and slow response time.
Smart Traffic Lights	Focus only on signal timing, not lane reallocation.
Movable Dividers	Expensive, not widely implemented, lacks full automation.

Table 3.5 Summary of Existing System Limitations

3.6 Need for an Automated, IoT-Based Adjustable Road Divider System

- To overcome these existing limitations, the proposed Adjustable Road Divider for Dynamic Traffic Management using IoT aims to:
  →Automate lane adjustments using IR sensors and Arduino UNO.
  →Reduce congestion by reallocating lanes in real-time.

3.7 Block Diagram for Existing System

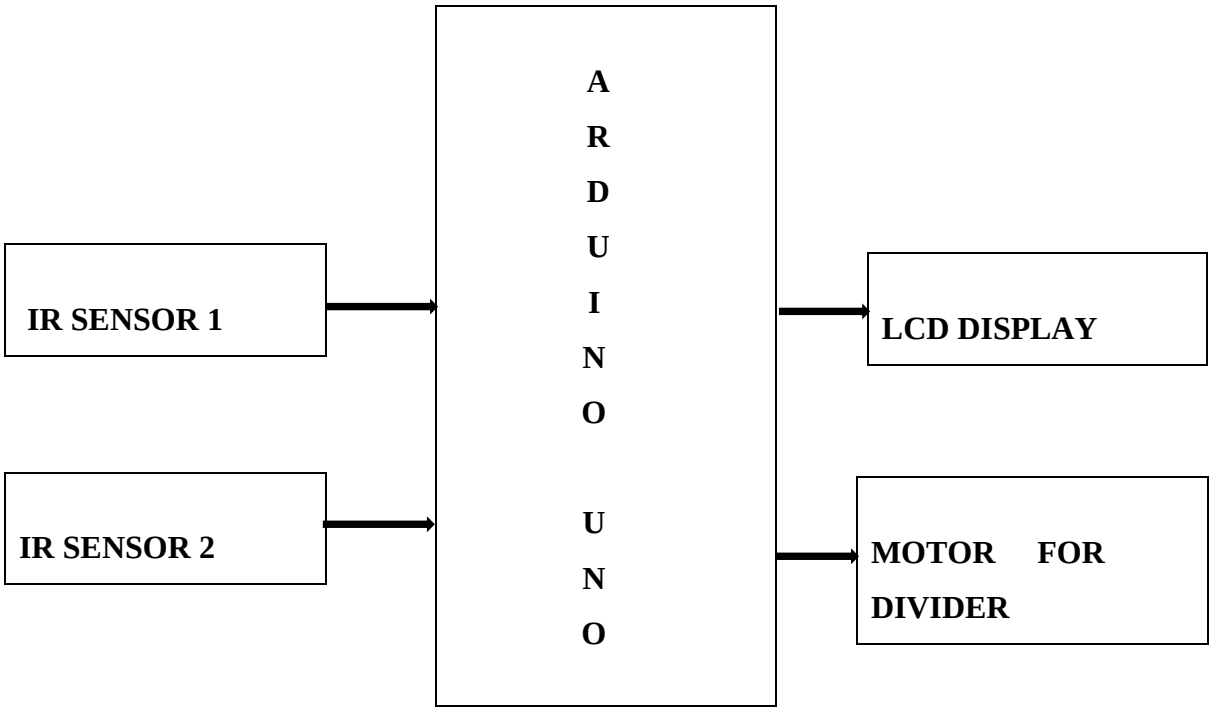


Fig 3.7 Block Diagram for Existing System

CHAPTER - 4

IMPLEMENTED SYSTEM

4.1 INTRODUCTION

The Adjustable Road Divider for Dynamic Traffic Management using IoT project has been designed and implemented with an aim to create a smart, adaptive, and automated system for real-time traffic management. The system integrates various IoT-based sensors, microcontroller automation, and mechanical movement systems to optimize lane allocation and improve overall traffic flow. Below is a detailed explanation of the implemented system.

4.1 SYSTEM ARCHITECTURE OVERVIEW

The implemented system consists of several key components working together to automate traffic management:

4.1.1Sensors for Traffic Detection:

Infrared (IR) Sensors are used to detect the number of vehicles in each lane by measuring vehicle presence.

These sensors send real-time data to the Arduino UNO, which processes the input and determines whether a lane adjustment is required.

The system continuously monitors traffic flow to dynamically allocate lanes based on the real-time traffic density.

4.1.2 Microcontroller (Arduino UNO):

The Arduino UNO acts as the central processing unit in the system.

It receives sensor data (from IR sensors) and makes real-time decisions about lane reallocation.

The Arduino is also responsible for controlling the gear motors that drive the Rack and Pinion System (RPS) to move the road dividers.

4.1.3 Mechanical Movement System (RPS with Gear Motors):

The Rack and Pinion System (RPS) is used to move the adjustable road divider smoothly and accurately.

Gear motors drive the rack and pinion mechanism, allowing the dividers to shift position based on the sensor inputs.

4.1.4 LCD Display for Real-Time Updates:

The LCD screen displays real-time lane status updates to drivers.

It informs them about changes in lane allocation, such as “Lane Shift in Progress” or “Use Left Lane for Faster Traffic.”

This ensures driver awareness and improves compliance with lane changes.

4.1.5 Power Supply:

The system is powered by a low-voltage DC supply, which powers the Arduino UNO, sensors, and gear motors.

The design is energy-efficient, ensuring minimal power consumption while operating continuously.

4.2 System Working and Functionality

Step 1: Traffic Detection

The system starts by using IR sensors placed along the lanes to detect the presence of vehicles.

These sensors continuously track the density of traffic in real-time, providing feedback on the number of vehicles per lane.

Step 2: Data Processing

The data collected by the sensors is sent to the Arduino UNO microcontroller.

The Arduino processes the input and calculates whether there is a need to adjust the lane configuration.

If one side of the road has higher traffic density, the system decides to allocate more lanes to the busier side.

Step 3: Lane Adjustment

Based on the traffic density data, the Arduino UNO sends control signals to the gear motors, which drive the Rack and Pinion System (RPS).The RPS mechanism shifts the road divider to the designated position, reallocating lanes according to the real-time needs of the traffic flow.

Step 4: Displaying Information to Drivers

Simultaneously, the system updates the LCD display to show lane status updates to approaching drivers.

Messages like “Lane Shift in Progress” or “Follow New Lane Allocation” are shown to inform drivers about ongoing lane changes.

Step 5: Continuous Monitoring and Feedback

The system continuously monitors traffic flow and makes adjustments as needed.

If traffic conditions change, such as a decrease in vehicle density, the system can shift the divider back to its original position or reallocate lanes accordingly.

4.3 ADVANTAGES OF THE IMPLEMENTED SYSTEM

- **Real-time Adaptability:** The system adjusts lanes dynamically based on real-time traffic conditions.
- **Automated Control:** The system operates automatically with minimal human intervention, reducing the chances of errors.
- **Improved Traffic Flow:** The lane allocation is optimized in real-time, reducing congestion and improving traffic flow.
- **Cost-Effective:** Using Arduino UNO and simple mechanical systems (RPS and gear motors) keeps the project budget-friendly.
- **Energy Efficient:** The system consumes minimal power, ensuring long-term, cost-effective operation.
- **Scalable and Flexible:** The system can be implemented on various types of roads and scaled up to suit larger urban areas or highways**.**

4.4 BLOCK DIAGRAM OF IMPLEMENTED SYSTEM

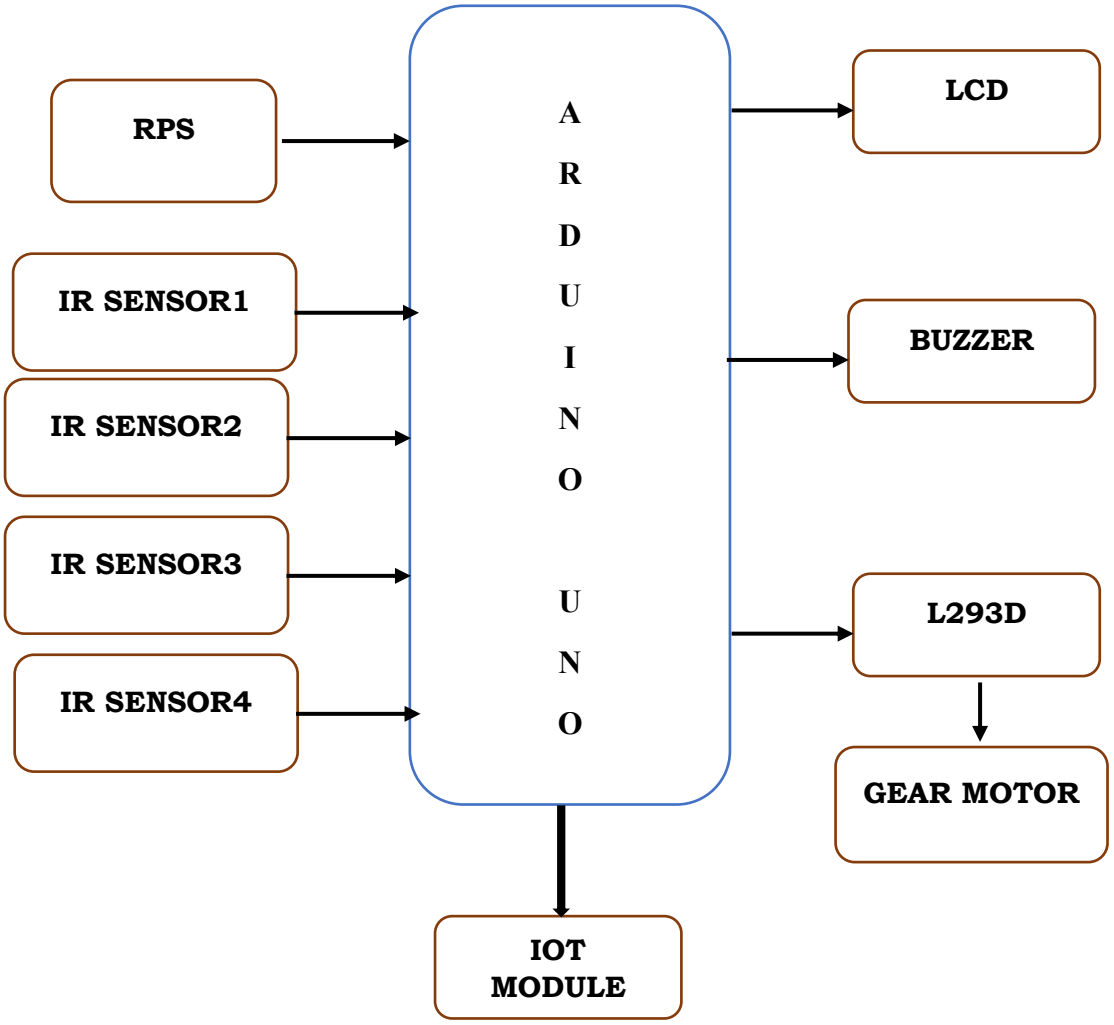


Fig.4.4 Block diagram of Implemented system

4.5 CIRCUIT DIGRAM OF ADJUSTABLE ROAD DIVIDER

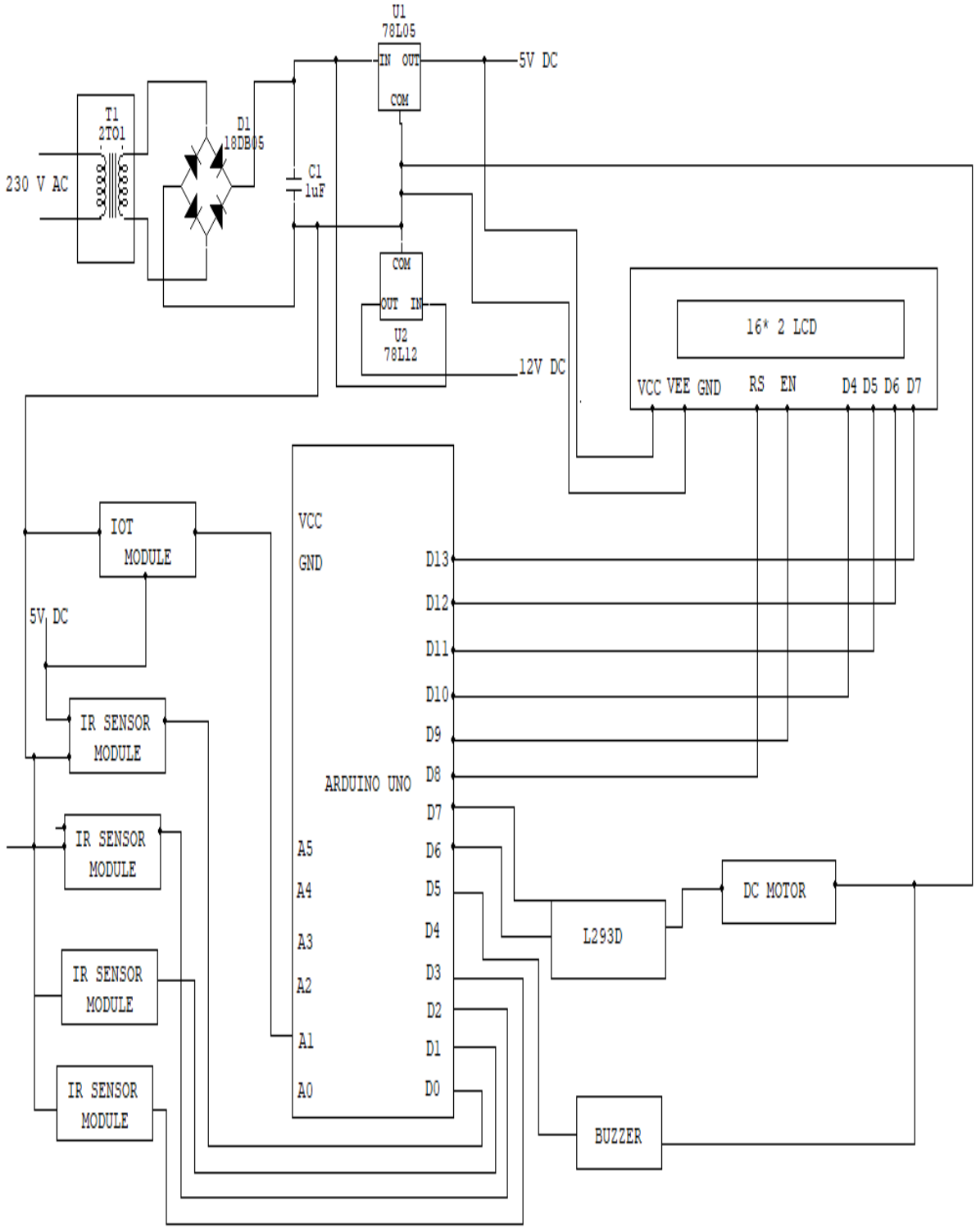


Fig 4.5 Circuit Digram of Adjustable Road Divider

4.5.1 IR Sensors

Multiple IR sensors are installed across the lanes to detect vehicles by emitting infrared light and measuring reflected signals.

The sensors detect the presence of vehicles and communicate the vehicle count to the Arduino UNO.

4.5.2 Arduino UNO

The Arduino UNO is used for data processing, decision-making, and control signal generation.

It receives real-time data from the sensors and actuates the gear motors to move the road divider.

Arduino's simple programming interface and cost-effectiveness make it ideal for this project.

4.5.3 Rack and Pinion System with Gear Motors

The Rack and Pinion System (RPS) is chosen for its smooth and precise movement.

The gear motors control the movement of the divider along the rack, ensuring that the lanes can be dynamically adjusted.

The system is designed for high durability, capable of withstanding continuous use in real-world traffic conditions.

4.5.4 LCD Display

An LCD display is integrated into the system to provide real-time updates to drivers about lane shifts and the status of the road divider.

The display shows clear and simple messages to reduce driver confusion and ensure safe lane transitions.

4.5.5 Power Supply

The system operates on a low-voltage DC power supply for energy efficiency.

Components such as the Arduino, sensors, and gear motors are powered by this supply, ensuring consistent operation.

4.6 WORKING OF IMPLEMENTED STSTEM

The Adjustable Road Divider for Dynamic Traffic Management using IoT is a cutting-edge system that utilizes Internet of Things (IoT) technology to optimize traffic flow by dynamically adjusting road dividers based on real-time traffic conditions. The system is composed of IoT sensors, such as traffic density and environmental sensors, which continuously monitor vehicle count, speed, and road conditions. This data is transmitted to a centralized control system, which processes it using algorithms to determine the best configuration for road dividers. Based on traffic flow, time of day, or unexpected factors like accidents, the system can activate actuators to move or shift the road dividers, thereby expanding or narrowing lanes as needed. This dynamic adjustment improves traffic efficiency by increasing lane capacity in high-traffic areas and reducing congestion in lower-traffic zones. The system not only enhances traffic flow and safety but also reduces travel times and pollution by minimizing bottlenecks and smoothing vehicle movement. Digital signage and app notifications alert drivers to divider changes, ensuring a seamless driving experience. Through continuous data collection and feedback, the system can adapt to ever-changing traffic conditions, making it a scalable and effective solution for modern urban traffic management.

4.7 MODELLING AND PROTOTYPE OF IMPLEMENTED SYSTEM

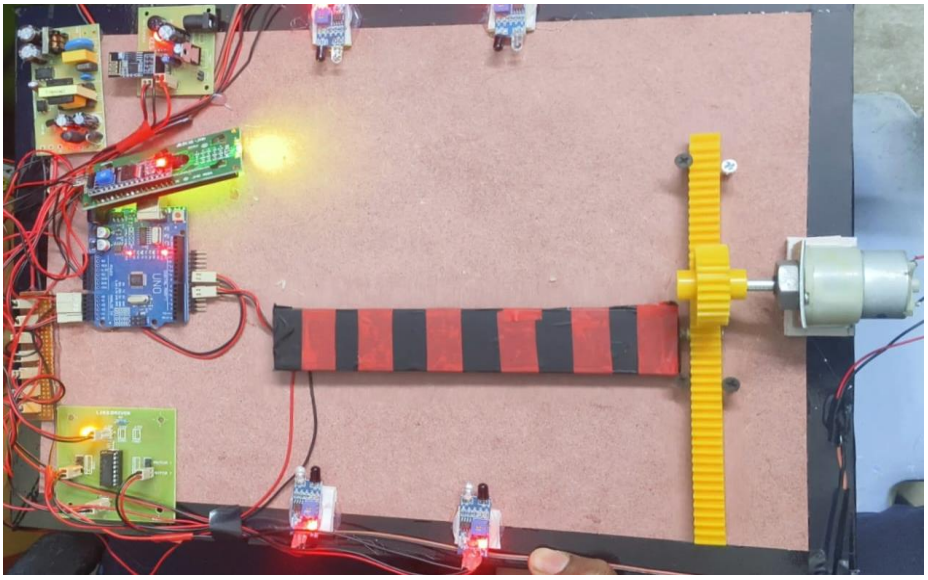


Fig 4.7 Prototype of Implemented System

The Adjustable Road Divider for Dynamic Traffic Management using IoT is an innovative system designed to optimize traffic flow and improve road utilization in real-time. By

leveraging Internet of Things (IoT) technology, the system uses traffic density sensors, environmental sensors, and a centralized control system to continuously monitor traffic conditions. Based on the collected data, the system dynamically adjusts road dividers by shifting or reconfiguring them to increase or decrease lane capacity as required, thereby alleviating congestion and enhancing traffic flow. The system can also adapt to unexpected conditions such as accidents or weather changes, ensuring a responsive and flexible traffic management solution. This dynamic divider adjustment helps reduce travel time, lower emissions, and improve overall road safety. The system's scalability and real-time adaptability make it a viable solution for smart cities aiming to enhance traffic management and reduce congestion, providing significant benefits to both drivers and urban infrastructure**.**

4.8 SOFTWARE APPLICATION

These are the steps you need to follow in order to be up and running:

1. Get an Arduino board
2. Download the Arduino environment
3. Install the USB drivers
4. Connect the board
5. Upload a program

Get an Arduino board:

The Arduino i/o board is a simple circuit featuring the ATmega8 processor from Atmel. The board is composed of a printed circuit board (PCB) and electronic parts.

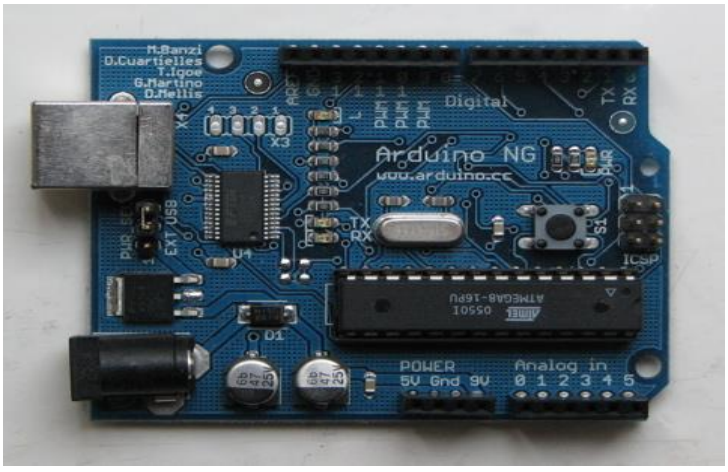


Fig 4.8 Arduino board

There are a few ways to get an Arduino board:

buy a ready made board. See how you can buy a board or just the PCB.

- European distributor
- US distributor

Build your own board. If you want you can build your own PCB just by downloading the CAD files from the Hardware page. Extract the .brd file and send it to a PCB manufacturer. Be aware that manufacturing a single pcb will be very expensive. It's better to get together with other people and make 20 or 30 at a time. Since you get the full CAD files you can make your own customised version of Arduino. if you make modifications or fix bugs please send us your changes!

Purchase parts.

Purchase the parts from any electronics store. The Serial version in particular has been designed to use the most basic parts that can be found anywhere in the world. The USB version on the other hand requires some advanced soldering skills because of the FTDI chip that is an smd part.

Assemble the board.

We put together a step by step guide on how to build an arduino board. Newbies: never soldered before? afraid of trashing thousands of boards before getting one properly soldered? fear not :) learn to master the art of soldering.

Program the boot loader.

In order for the development environment to be able to program the chip, this has to be programmed with a piece of code called *bootloader*. See the bootloader page on how to program it on your chip.

Download the ARDUINO environment

To program the Arduino board you need the Arduino environment.

Download Arduino: From the software page.

Linux note: For help getting the Arduino IDE running on Debian, please see the FAQ ("How can I run the Arduino IDE under Linux?").

Mac OS X note: After downloading the IDE, run the macosx setup.command. It corrects permission on a few files for use with the serial port and will prompt you for your password. You may need to reboot after running this script.

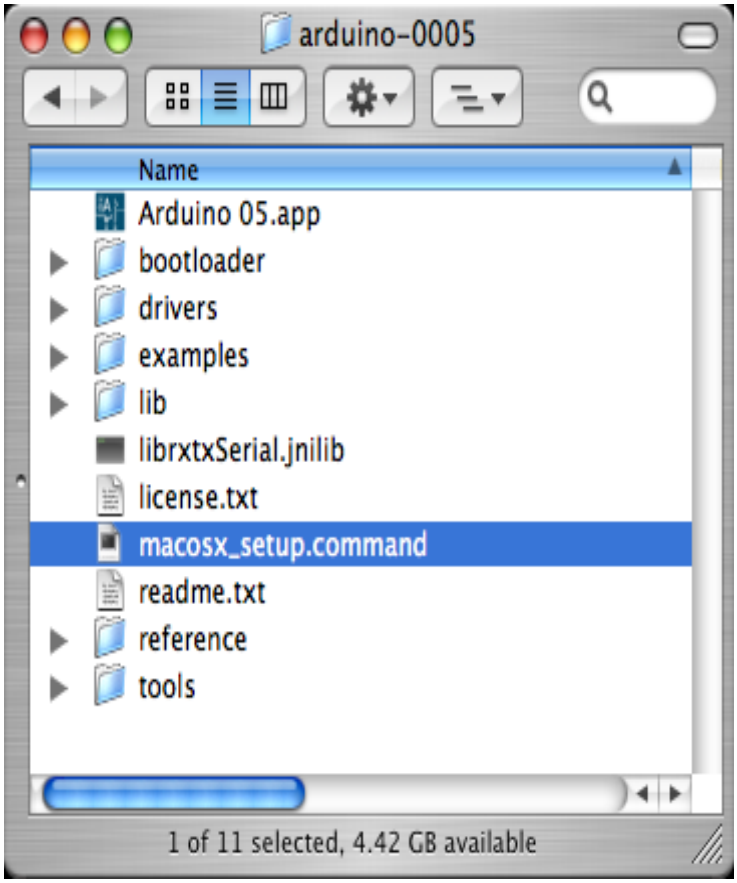


Fig 4.9 Selecting of Macosx Setup.Command

Install the USB drivers

If you are using a USB Arduino, you will need to install the drivers for the FTDI chip on the board.

These can be found in the driver directory of the Arduino distribution.

On Windows, you will need to unzip FTDI USB Drivers.zip. Then, when you plug in the Arduino board, point the Windows *Add Hardware* wizard to the FTDI USB Drivers directory.

On the Mac, mount the FTDIUSBSerialDriver_v2_1_6.dmg (on PPC machines) or the FTDIUSBSerialDriver_v2_2_6_Intel.dmg (on Intel machines) disk image and run the included FTDI USB Serial Driver .pkg.

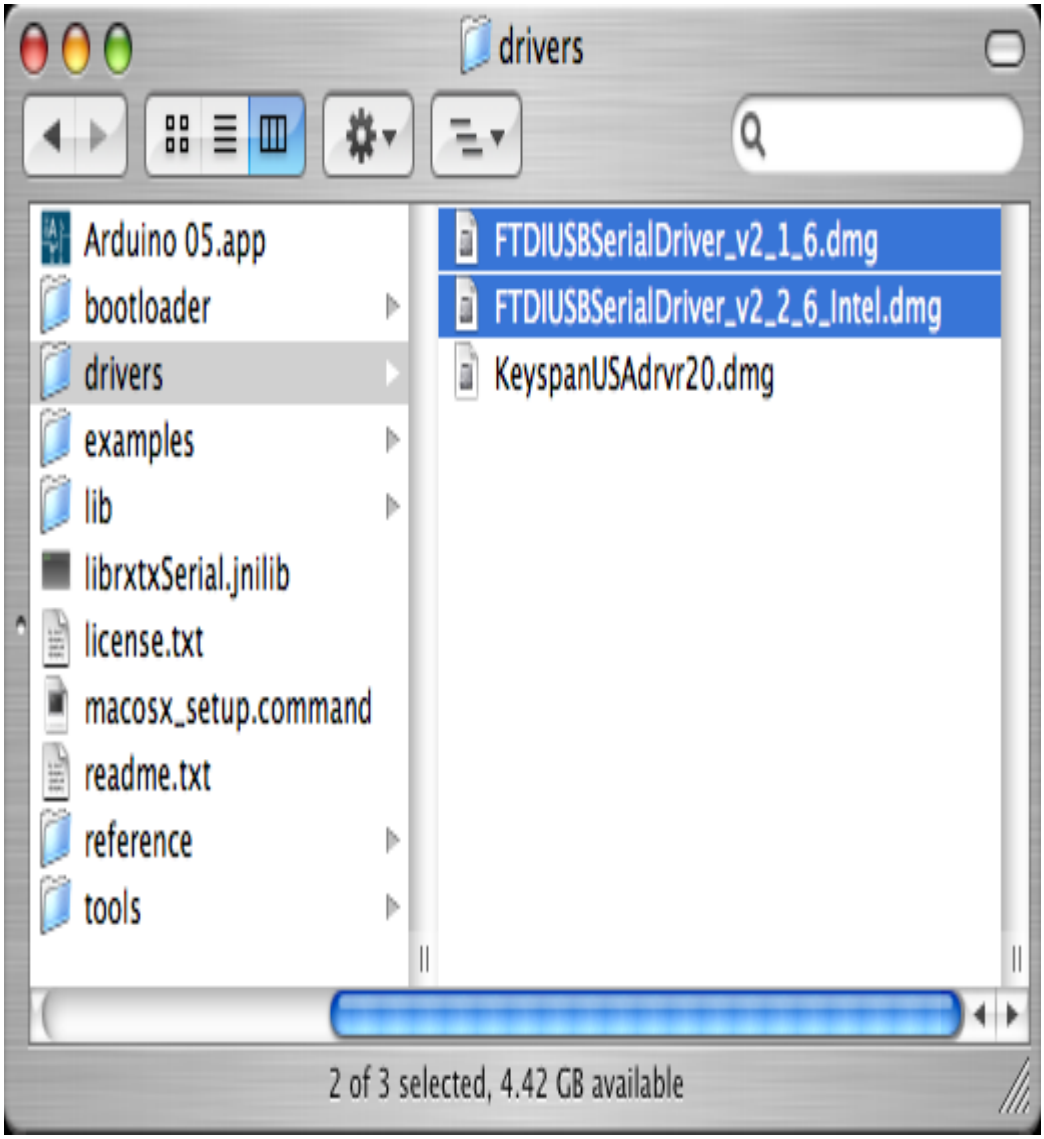


Fig 4.10 The latest version of the drivers can be found on the FTDI_website

Connect the board

If you're using a serial board, power the board with an external power supply (6 to 25 volts DC, with the core of the connector positive). Connect the board to a serial port on your computer.

On the USB boards, the power source is selected by the jumper between the USB and power plugs. To power the board from the USB port (good for controlling low power devices like LEDs), place the jumper on the two pins closest to the USB plug.

To power the board from an external power supply (needed for motors and other high current devices), place the jumper on the two pins closest to the power plug. Either way, connect the board to a USB port on your computer. On Windows, the Add

New Hardware wizard will open; tell it you want to specify the location to search for drivers and point to the folder containing the USB drivers you unzipped in the previous step.

The power LED should go on.

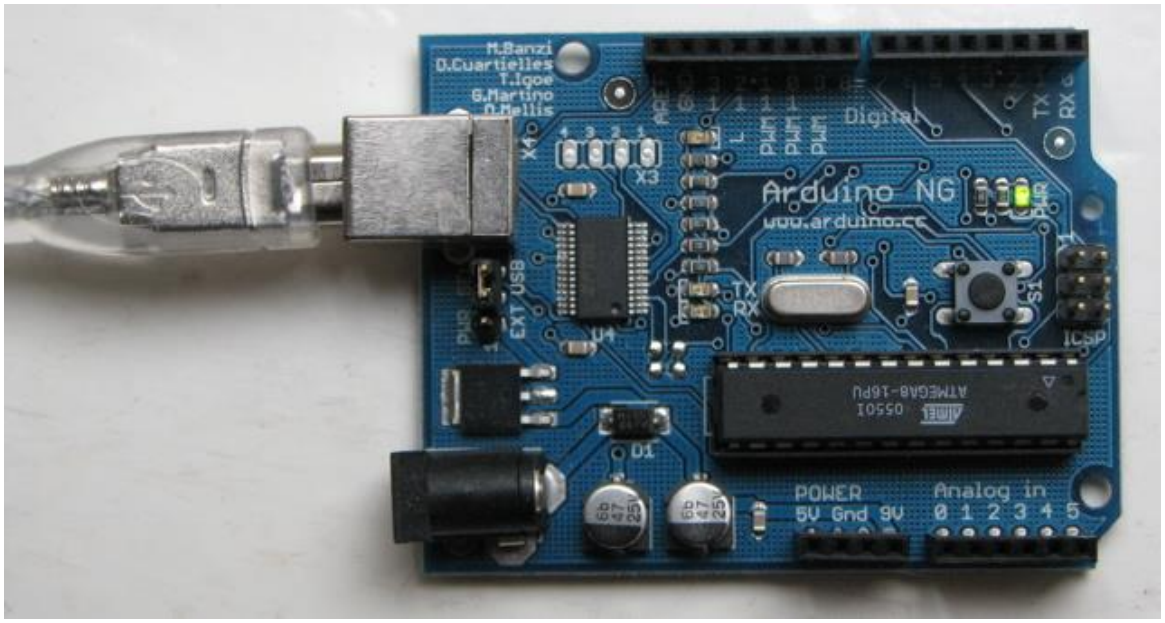


Fig 4.11 Connecting cable to Arduino UNO

Upload a program

Open the LED blink example sketch: File > Sketchbook > Examples > led_blink.

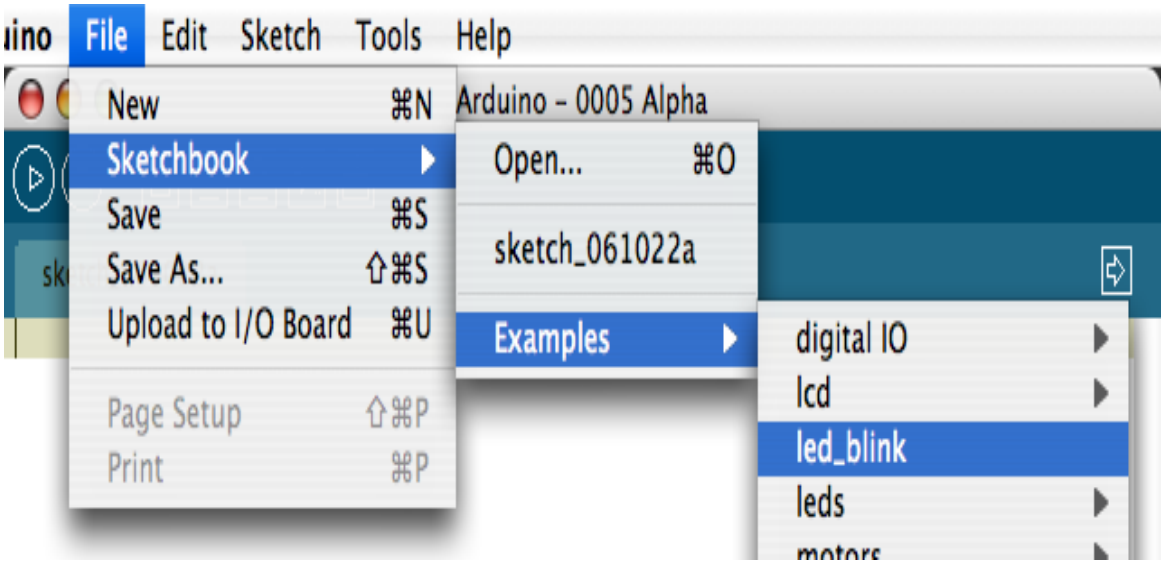


Fig 4.12 Uploading a program to led blink

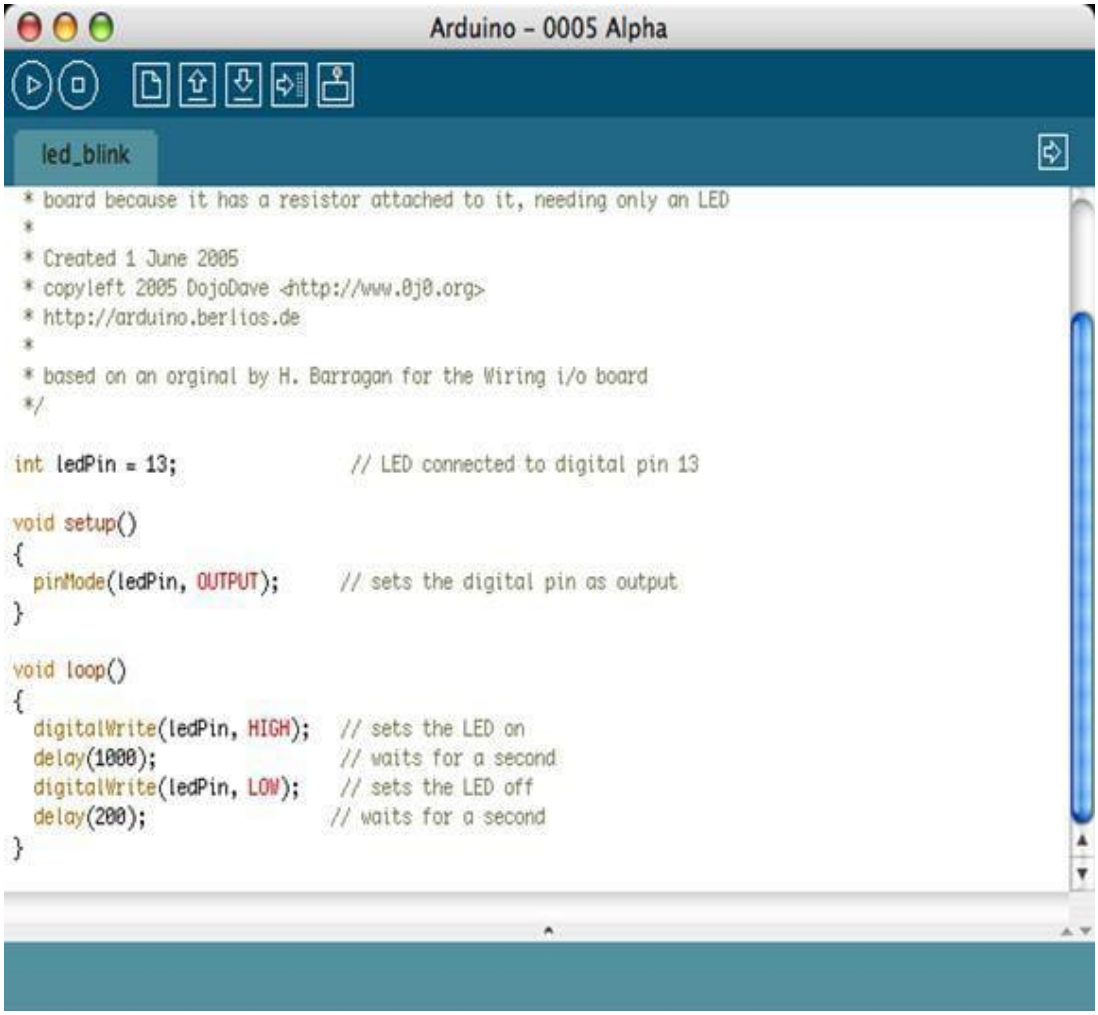
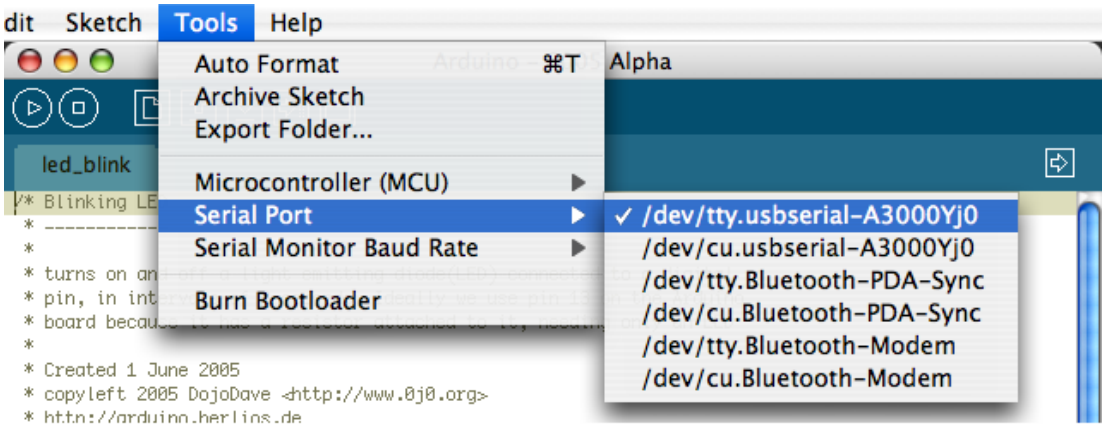


Fig 4.13 Program for led blink

Select the serial device of the Arduino board from the Tools | Serial Port menu. On Windows, this should be COM1 or COM2 for a serial Arduino board, or COM3, COM4, or COM5 for a USB board. On the Mac, this should be something like /dev/cu.usbserial-1B1 for a USB board, or something if using a Keys pan adapter with a serial board (other USB-to-serial adapters use different names).



Push the reset button on the board then click the Upload button in the IDE. Wait a few seconds. If successful, the message "Done uploading." will appear in the status bar.

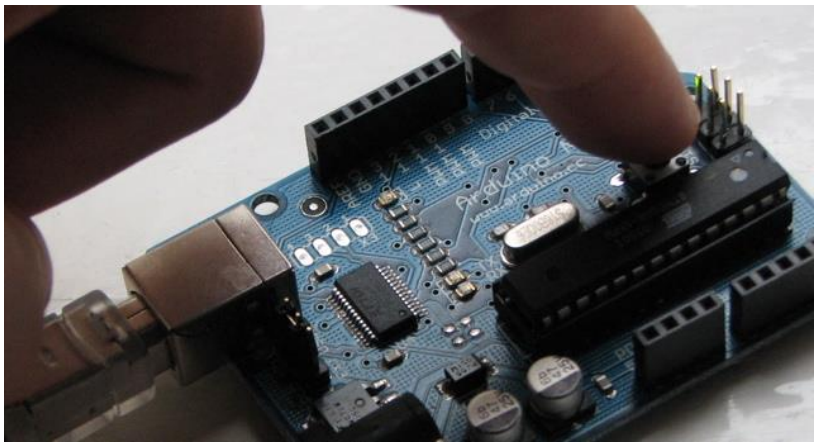


Fig 4.14 After successfully uploaded code in Arduino

If the Arduino board doesn't show up in the Tools | Serial Port menu, or you get an error while uploading, please see the FAQ for troubleshooting suggestions.

A few seconds after the upload finishes, you should see the amber (yellow) LED on the board start to blink.

Example Programs

To open the blink sketch, you will need to go to **File** > **Examples** > **Basics** > **Blink**

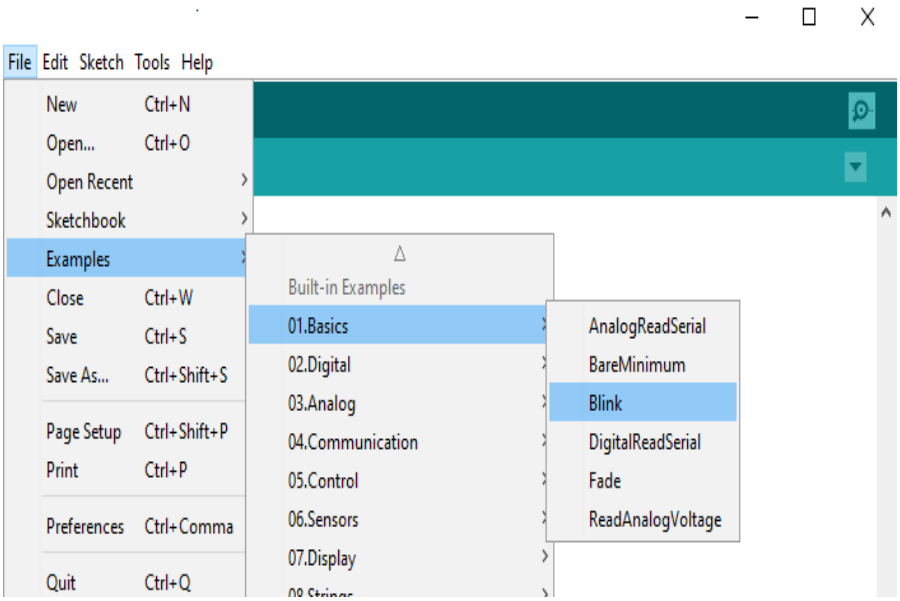


Fig 4.15 To Open the Blink sketch

Fig 4.16 Fully coded Blink Sketch

Next, you need to click on the verify button (check mark) that's located in the top left of the IDE box. This will compile the sketch and look for errors. Once it says "Done Compiling" you are ready to upload it. Click the upload button (forward arrow) to send the program to the Arduino board.

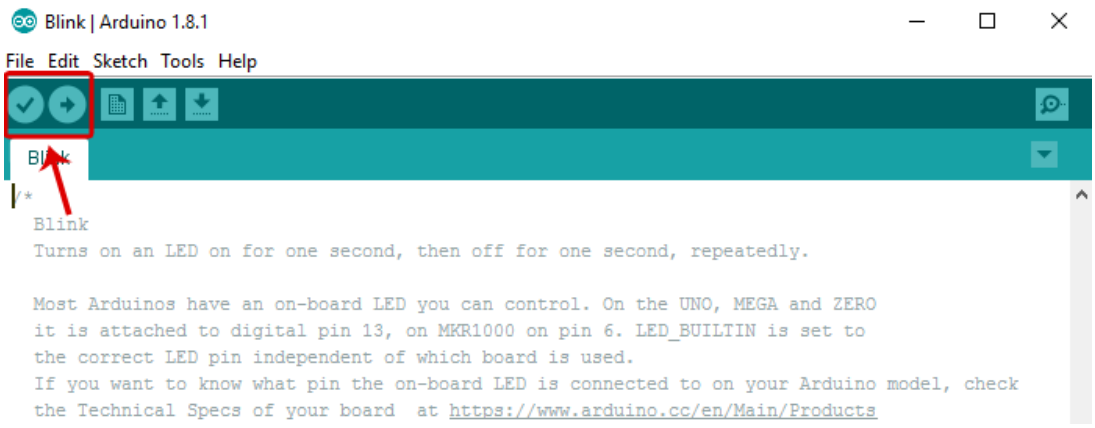


Fig 4.17 Click On Verify Button

The built-in LEDs on the Arduino board will flash rapidly for a few seconds and then the program will execute. If everything went correctly, the LED on the breadboard should turn on for a second and then off for a second and continue in a loop.

Change The Code

Before we go to the next project, lets change some of the code in the "Blink" sketch to make it do something different. Playing around with the sketch will help you start to learn how the code controls the board.

Keep the Arduino board connected and change the delay portion of the code from (1000) to (200). Click the verify button on the top left of the IDE and then click upload. This should make the LED on the breadboard blink faster.

NOTE – Arduino measures time in milliseconds and 1000 milliseconds = 1 second. The original code (1000) turns on the LED for 1 second and then off for 1 second. By adjusting the code from (1000) to (200) it shortens the time between on and off which makes it blink faster.

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(200);
  digitalWrite(LED_BUILTIN, LOW);
  delay(200);
}
```

Changed to 200

Changed to 200

In order to use a switch, we have to load the file called “Button” which can be found here: **File** > **Examples** > **Digital** > **Button**

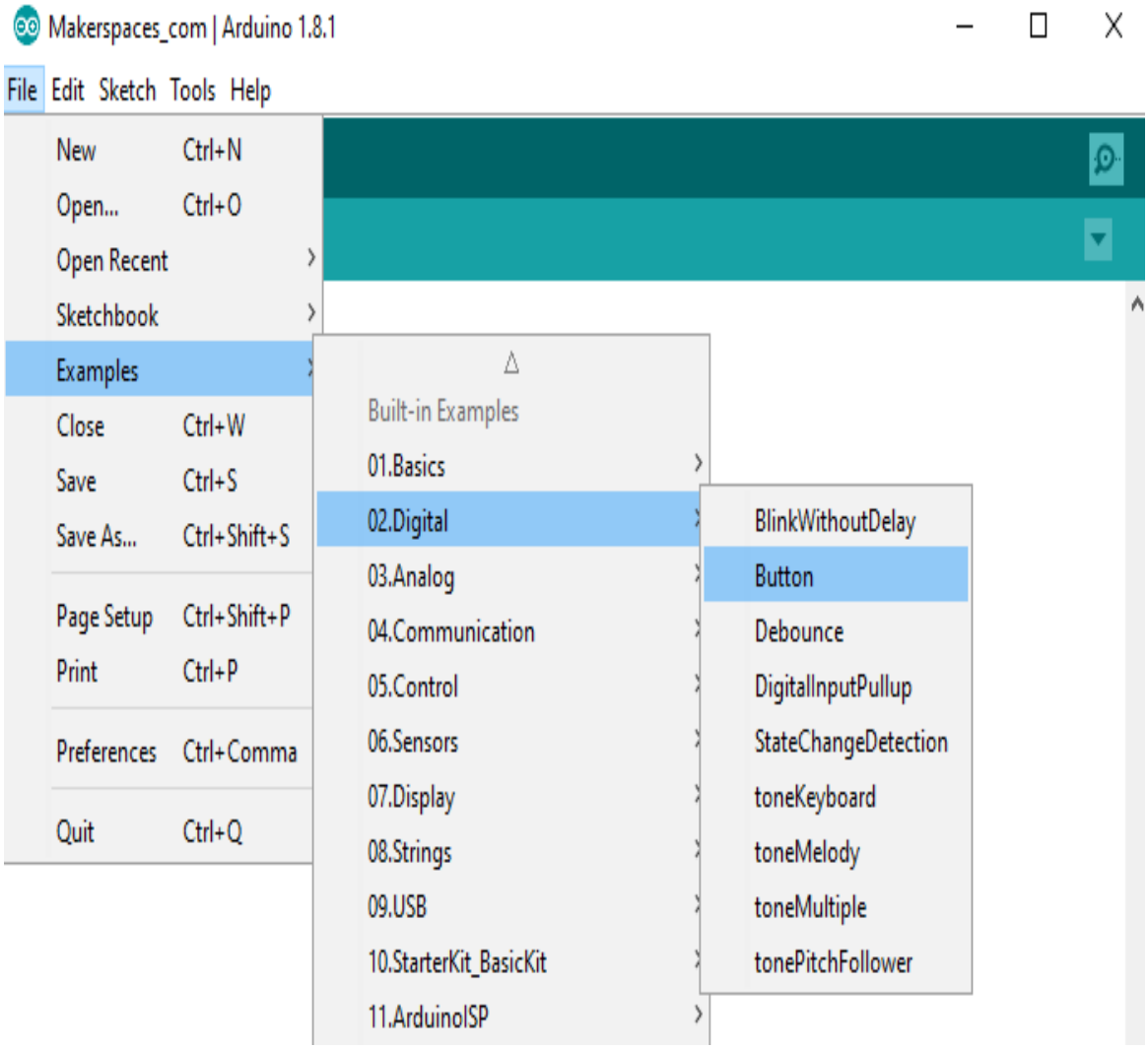


Fig 4.18 In order to use a switch, we have to load the file called “Button

”

Now you should have a fully coded button sketch that looks like the image below.

Fig 4.18 Fully coded output

Next, you need to click on the verify button (check mark) that's located in the top left of the IDE box. Once it says "Done Compiling" you are ready to upload it. Click the upload button (forward arrow) to send the program to the Arduino board.

CHAPTER - 5
COMPONENTS

5.1 ARDUINO

Arduino is an open source microcontroller which can be easily programmed, erased and reprogrammed at any instant of time. Introduced in 2005 the Arduino platform was designed to provide an inexpensive and easy way for hobbyists, students and professionals to create devices that interact with their environment using sensors and actuators. Based on simple microcontroller boards, it is an open source computing platform that is used for constructing and programming electronic devices. It is also capable of acting as a mini computer just like other microcontrollers by taking inputs and controlling the outputs for a variety of electronics devices. It is also capable of receiving and sending information over the internet with the help of various Arduino shields, which are discussed in this paper. Arduino uses a hardware known as the Arduino development board and software for developing the code known as the Arduino IDE (Integrated Development Environment). Built up with the 8-bit Atmel AVR microcontroller's that are manufactured by Atmel or a 32-bit Atmel ARM, these microcontrollers can be programmed easily using the C or C++ language in the Arduino IDE. Unlike the other microcontroller boards in India, the Arduino boards entered the electronic market only a couple of years ago, and were restricted to small scale projects only. People associated with electronics are now gradually coming up and accepting the role of Arduino for their own projects. This development board can also be used to burn (upload) a new code to the board by simply using a USB cable to upload. The Arduino IDE provides a simplified integrated platform which can run on regular personal computers and allows users to write programs for Arduino using C or C++. With so many Arduino boards available in the market, selecting a particular development board needs a variety of survey done with respect to their specifications and capabilities, which can be used for the project execution according to its specified applications.

5.1.1 Need for ARDUINO

Why is there a need to use Arduino in specific? or What makes it different from others? Massimo Banzi, a Co-founder of Arduino mentions some very important reasons for this question.

1) Active User Community:

A group of people using a similar product can hold posted message conversations and share their experiences or solve the problems of the other users in the communities with their own experiences [1]. “If you start charging for everything, everything dies very quickly.” says Banzi, Arduino Cofounder.

2) Growth of Arduino:

Arduino was developed with intent to provide an economical and trouble-free way for hobbyists, students and professionals to build devices that interact with their situation using sensors and actuators. This makes it perfect for newcomers to get started quickly [1].

3) Inexpensive Hardware:

Since Arduino is an open source platform the software is not purchased and only the cost of buying the board or its parts is incurred, thus making it very cheap. The hardware designs are also available online for free from its official website [1].

4) Arduino Board as a Programmer:

To make Arduino board function easy and also making it available everywhere these boards come with a USB cable for power requirements as well as functioning as a programmer [1]. 5) Multi-platform Environment: The Arduino IDE is capable of running on a number of platforms including Microsoft, Linux and Mac OS X making the user community even larger [1].

5.1.2. Type of ARDUINO Boards

Arduino boards are available with many different types of built-in modules in it. Boards such as

Arduino BT come with a built-in Bluetooth module, for wireless communication. These built-in modules can also be available separately which can then be interfaced (mounted) to it. These modules are known as Shield. Some of the most commonly used Shields are:

Arduino Ethernet shield: It that allows an Arduino board to connect to the internet using the Ethernet library and to read and write an SD card using the SD library [2].

Arduino Wireless shield: It allows your Arduino board to communicate wirelessly using Zigbee [2].

Arduino Motor Driver Shield: It allows your Arduino boards to interface with driver of a motor etc. [2].

5.1.3 Elements of ARDUINO Boards

Elements of an Arduino Board can be done into two categories:

• Hardware

• Software

Hardware The Arduino Development Board consists of many components that together makes it work. Here are some of those main component blocks that help in its functioning:

Microcontroller: This is the heart of the development board, which works as a mini computer and can receive as well as send information or command to the peripheral devices connected to it. The microcontroller used differs from board to board; it also has its own various specifications.

External Power Supply: This power supply is used to power the Arduino development board with a regulated voltage ranging from 9 – 12 volts.

USB plug: This plug is a very important port in this board. It is used to upload (burn) a program to the microcontroller using a USB cable. It also has a regulated power of 5V which also powers the Arduino board in cases when the External Power Supply is absent.

Internal Programmer: The developed software code can be uploaded to the microcontroller via USB port, without an external programmer.

Reset button: This button is present on the board and can be used to resets the Arduino microcontroller.

Analog Pins: There are some analog input pins ranging from A0 – A7 (typical). These pins are used for the analog input / output. The no. of analog pins also varies from board to board.

Digital I/O Pins: There are some digital input pins also ranging from 2 to 16 (typical). These pins are used for the digital input / output. The no. of these digital pins also varies from board to board.

Power and GND Pins: There are pins on the development board that provide 3.3, 5 volts and ground through them.

Software

The program code written for Arduino is known as a sketch. The software used for developing such sketches for an Arduino is commonly known as the Arduino IDE. This IDE contains the following parts in it:

Text editor: This is where the simplified code can be written using a simplified version of C++ programming language.

Message area: It displays error and also gives a feedback on saving and exporting the code.

Text: The console displays text output by the Arduino environment including complete error messages and other information

Console Toolbar: This toolbar contains various buttons like Verify, Upload, New, Open, Save and Serial Monitor. On the bottom right hand

5.1.4 Why ARDUINO?

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. 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 it 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 advantage 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 open source tools, 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 own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

5.1.5 ARDUINO Features

High performance, low power AVR® 8-bit microcontroller

- ✓ Advanced RISC architecture
- ✓ 131 powerful instructions – most single clock cycle execution 8 general purpose working registers×● 32
- ✓ Fully static operation
- ✓ Up to 16MIPS throughput at 16MHz
- ✓ On-chip 2-cycle multiplier
- ✓ High endurance non-volatile memory segments
- ✓ 32K bytes of in-system self-programmable flash program memory
- ✓ 1Kbytes EEPROM
- ✓ 2Kbytes internal SRAM
- ✓ Write/erase cycles: 10,000 flash/100,000 EEPROM
- ✓ Optional boot code section with independent lock bits
- ✓ In-system programming by on-chip boot program
- ✓ True read-while-write operation
- ✓ Programming lock for software security
- ✓ Peripheral features
- ✓ Two 8-bit Timer/Counters with separate prescaler and compare mode
- ✓ One 16-bit Timer/Counter with separate prescaler, compare mode, and capture mode
- ✓ Real time counter with separate oscillator
- ✓ Six PWM channels
- ✓ 8-channel 10-bit ADC in TQFP and QFN/MLF package

- ✓ Temperature measurement
- ✓ Programmable serial USART
- ✓ Master/slave SPI serial interface
- ✓ Byte-oriented 2-wire serial interface (Phillips I2 C compatible)
- ✓ Programmable watchdog timer with separate on-chip oscillator
- ✓ On-chip analog comparator
- ✓ Interrupt and wake-up on pin change
- ✓ Special microcontroller features
- ✓ Power-on reset and programmable brown-out detection
- ✓ Internal calibrated oscillator
- ✓ External and internal interrupt sources
- ✓ Six sleep modes: Idle, ADC noise reduction, power-save, power-down, standby, and extended standby

5.2 Pin Description

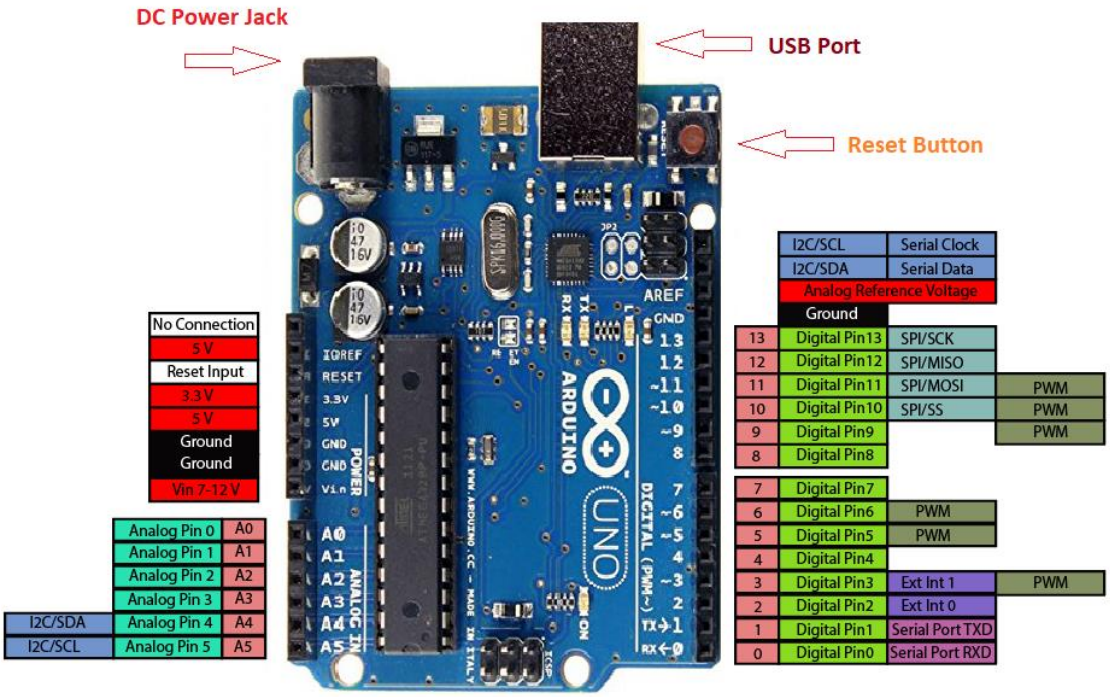


Fig 5.2 Pin Description

Arduino Uno pinout - Power Supply

There are 3 ways to power the Arduino Uno:

- **Barrel Jack** - The Barrel jack, or DC Power Jack can be used to power your Arduino board. The barrel jack is usually connected to a wall adapter. The board can be powered by 5-20 volts but the manufacturer recommends to keep it between 7-12 volts. Above 12 volts, the regulators might overheat, and below 7 volts, might not suffice.
- **VIN Pin** - This pin is used to power the Arduino Uno board using an external power source. The voltage should be within the range mentioned above.
- **USB cable** - when connected to the computer, provides 5 volts at 500mA.

There is a polarity protection diode connecting between the positive of the barrel jack to the VIN pin, rated at 1 Ampere.

The power source you use determines the power you have available for your circuit. For instance, powering the circuit using the USB limits you to 500mA. Take into consideration that this is also used for powering the MCU, its peripherals, the on-board regulators, and the components connected to it. When powering your circuit through the barrel jack or VIN, the maximum capacity available is determined by the 5 and 3.3 volts regulators on-board the Arduino.

- **5v and 3v3**
  They provide regulated 5 and 3.3v to power external components according to manufacturer specifications.
- **GND**
  In the Arduino Uno pinout, you can find 5 GND pins, which are all interconnected.

  The GND pins are used to close the electrical circuit and provide a common logic reference level throughout your circuit. Always make sure that all GNDs (of the Arduino, peripherals and components) are connected to one another and have a common ground.
- **RESET** - resets the Arduino
- **IOREF** - This pin is the input/output reference. It provides the voltage reference with which the microcontroller operates.
-

Arduino Uno Pinout - Analog IN

The Arduino Uno has 6 **analog pins**, which utilize ADC (Analog to Digital converter).

These pins serve as analog inputs but can also function as digital inputs or digital outputs.

Analog to Digital Conversion

ADC stands for Analog to Digital Converter. ADC is an electronic circuit used to convert analog signals into digital signals. This digital representation of analog signals allows the processor (which is a digital device) to measure the analog signal and use it through its operation. Arduino Pins A0-A5 are capable of reading analog voltages. On Arduino the ADC has 10-bit resolution, meaning it can represent analog voltage

by 1,024 digital levels. The ADC converts voltage into bits which the microprocessor can understand.

One common example of an ADC is Voice over IP (VoIP). Every smartphone has a microphone that converts sound waves (voice) into analog voltage. This goes through the device's ADC, gets converted into digital data, which is transmitted to the receiving side over the internet.

Arduino Uno Pinout - Digital Pins

Pins 0-13 of the Arduino Uno serve as digital input/output pins.

Pin 13 of the Arduino Uno is connected to the built-in LED.

In the Arduino Uno - pins 3,5,6,9,10,11 have PWM capability.

It's important to note that:

Each pin can provide/sink up to 40 mA max. But the recommended current is 20 mA.

The absolute max current provided (or sank) from all pins together is 200mA

What does digital mean?

Digital is a way of representing voltage in 1 bit: either 0 or 1. Digital pins on the Arduino are pins designed to be configured as inputs or outputs according to the needs of the user. Digital pins are either on or off. When ON they are in a HIGH voltage state of 5V and when OFF they are in a LOW voltage state of 0V.

On the Arduino, When the digital pins are configured as **output**, they are set to 0 or 5 volts.

When the digital pins are configured as **input,** the voltage is supplied from an external device. This voltage can vary between 0-5 volts which is converted into digital representation (0 or 1). To determine this, there are 2 thresholds:

- Below 0.8v - considered as 0.
- Above 2v - considered as 1.

When connecting a component to a digital pin, make sure that the logic levels match. If the voltage is in between the thresholds, the returning value will be undefined.

What is PWM?

In general, Pulse Width Modulation (PWM) is a modulation technique used to encode a message into a pulsing signal. A PWM is comprised of two key components: **frequency** and **duty cycle**. The PWM frequency dictates how long it takes to complete a single cycle (period) and how quickly the signal fluctuates from high to low. The duty cycle determines how long a signal stays high out of the total period. Duty cycle is represented in percentage.

In Arduino, the PWM enabled pins produce a constant frequency of ~ 500Hz, while the duty cycle changes according to the parameters set by the user. See the following illustration:

PWM signals are used for speed control of DC motors, dimming LEDs and more.

Communication Protocols

Serial (TTL) - Digital pins 0 and 1 are the serial pins of the Arduino Uno.

They are used by the onboard USB module.

What is Serial Communication?

Serial communication is used to exchange data between the Arduino board and another serial device such as computers, displays, sensors and more. Each Arduino board has at least one serial port. Serial communication occurs on digital pins 0 (RX) and 1 (TX) as well as via USB. Arduino supports serial communication through digital pins with the SoftwareSerial Library as well. This allows the user to connect multiple serial-enabled devices and leave the main serial port available for the USB.

Software serial and hardware serial - Most microcontrollers have hardware designed to communicate with other serial devices. Software serial ports use a pin-change interrupt system to communicate. There is a built-in library for Software Serial communication. Software serial is used by the processor to simulate extra serial ports. The only drawback with software serial is that it requires more processing and cannot support the same high speeds as hardware serial.

SPI - SS/SCK/MISO/MOSI pins are the dedicated pins for SPI communication. They can be found on digital pins 10-13 of the Arduino Uno and on the ICSP headers.

What is SPI?

Serial Peripheral Interface (SPI) is a serial data protocol used by microcontrollers to communicate with one or more external devices in a bus like connection. The SPI can also be used to connect 2 microcontrollers. On the SPI bus, there is always one device that is denoted as a Master device and all the rest as Slaves. In most cases, the microcontroller is the Master device. The SS (Slave Select) pin determines which device the Master is currently communicating with.

SPI enabled devices always have the following pins:

- MISO (Master In Slave Out) - A line for sending data to the Master device
- MOSI (Master Out Slave In) - The Master line for sending data to peripheral devices
- SCK (Serial Clock) - A clock signal generated by the Master device to synchronize data transmission.

I2C - SCL/SDA pins are the dedicated pins for I2C communication. On the Arduino Uno they are found on Analog pins A4 and A5.

What is I2C?

I2C is a communication protocol commonly referred to as the "I2C bus". The I2C protocol was designed to enable communication between components on a single circuit board. With I2C there are 2 wires referred to as SCL and SDA.

- SCL is the clock line which is designed to synchronize data transfers.
- SDA is the line used to transmit data.

Each device on the I2C bus has a unique address, up to 255 devices can be connected on the same bus.

Aref - Reference voltage for the analog inputs.

Interrupt - INT0 and INT1. Arduino Uno has two external interrupt pins.

External Interrupt - An external interrupt is a system interrupt that occurs when outside interference is present. Interference can come from the user or other hardware devices in the network. Common uses for these interrupts in Arduino are reading the frequency a square wave generated by encoders or waking up the processor upon an external event.

Arduino has two forms of interrupt:

- External
- Pin Change

There are two external interrupt pins on the ATmega168/328 called INT0 and INT1. both INT0 and INT1 are mapped to pins 2 and 3. In contrast, Pin Change interrupts can be activated on any of the pins.

Arduino Uno Pinout - ICSP Header

ICSP stands for In-Circuit Serial Programming. The name originated from In-System Programming headers (ISP). Manufacturers like Atmel who work with Arduino have developed their own in-circuit serial programming headers. These pins enable the user to program the Arduino boards' firmware. There are six ICSP pins available on the Arduino board that can be hooked to a programmer device via a programming cable..

5.3 LCD

Fig 5.3 2x16 LCD Display

To display interactive messages we're using LCD Module. We take a look at an smart LCD show of lines, sixteen characters in keeping with line this is interfaced to the controllers. The protocol (handshaking) for the display is as proven. Whereas D0 to D7th bit is the Data strains, RS, RW and EN pins are the control pins and last pins are +5V, -5V and GND to offer supply. Where RS is the Register Select, RW is the Read Write and EN is the Enable pin.

The most commonly used ALPHANUMERIC displays are 1x16 (Single Line & 16 characters), 2x16 (Double Line & 16 character per line) & 4x20 (four lines & Twenty characters per line). The LCD requires 3 control lines (RS, R/W & EN) & 8 (or 4) data lines. The number on data lines depends on the mode of operation. If operated in 8-bit mode then 8 data lines + 3 control lines i.e. total 11 lines are required.

And if operated in 4-bit mode then 4 data lines + 3 control lines i.e. 7 lines are required. How do we decide which mode to use? It's simple if you have sufficient data lines you can go for 8 bit mode & if there is a time constrain i.e. display should be faster then we have to use 8-bit mode because basically 4-bit mode takes twice as more time as compared to 8-bit mode.

The display consists of internal byte-wide registers, one for instructions (RS=zero) and the second for characters to be displayed (RS=1). It additionally includes a user-programmed RAM vicinity (the man or woman RAM) that may be programmed to generate any favored character that can be fashioned the usage of a dot matrix. To distinguish among these two facts regions, the hex command byte 80 could be used to indicate that the display RAM deal with 00h can be selected.Port1 is used to grant the command or statistics type, and ports 3.2 to3.4 provide register pick and study/write ranges.

The show takes various quantities of time to perform the features as listed. LCD bit 7 is monitored for common sense high (busy) to ensure the display is overwritten.

Liquid Crystal Display also called as LCD may be very useful in offering user interface in addition to for debugging purpose. The most not unusual sort of LCD controller is HITACHI 44780 which provides a easy interface between the controller & an LCD. These LCD's are quite simple to interface with the controller as well as are price effective.

5.4 BUZZER

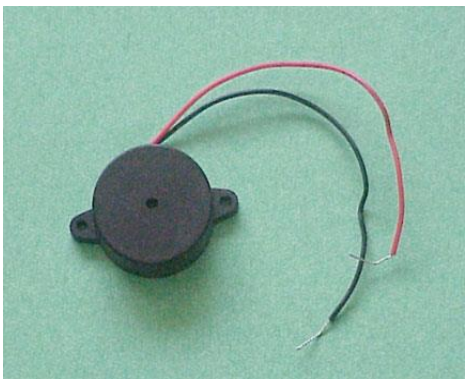


Fig 5.4 Buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or electronic. Typical uses of buzzers and beepers include alarms, timers and confirmation of user input such as a mouse click or keystroke.

FEATURES

• The PB series are high-performance buzzers with a unimorph piezoelectric ceramic element and an integral self-excitation oscillator circuit.

• They exhibit extremely low power consumption in comparison to electromagnetic units.

• They are constructed without switching contacts to ensure long life and no electrical noise.

• Compact, yet produces high acoustic output with minimal voltage.

Mechanical

A joy buzzer is an example of a purely mechanical buzzer.

Electromechanical

Early devices were based on an electromechanical system identical to an electric bell without the metal gong. Similarly, a relay may be connected to interrupt its own actuating current, causing the contacts to buzz. Often these units were anchored to a wall or ceiling to use it as a sounding board. The word "buzzer" comes from the rasping noise that electromechanical buzzers made.

Electronic

A piezoelectric element may be driven by an oscillating electronic circuit or other audio signal source. Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep. Electronic buzzers find many applications in modern days.

Uses

- Annunciator panels
- Electronic metronomes
- Game shows
- Microwave ovens and other household appliances
- Sporting events such as basketball games

5.5 IR Sensor Module

Fig 2.5 IR Sensor Module

The sensor module adaptable to ambient light, having a pair of infrared emitting and receiving tubes, transmitting tubes emit infrared certain frequency, when the direction of an obstacle is detected (reflection surface), the infrared reflected is received by the reception tube, After a comparator circuit processing, the green light is on, but the signal output interface output digital signal (a low-level signal), you can adjust the detection distance knob potentiometer, the effective distance range of 2 ~ 30cm, the working voltage of 3.3V- 5V. Detection range of the sensor can be obtained by adjusting potentiometer, with little interference, easy to assemble, easy to use features, can be widely used in robot obstacle avoidance, avoidance car, line count, and black and white line tracking and many other occasions.

Specification

1. When the module detects an obstacle in front of the signal, the green indicator lights on the board level, while the OUT port sustained low signal output, the module detects the distance 2 ~ 30cm, detection angle 35 °, the distance can detect potential is adjusted clockwise adjustment potentiometer, detects the distance increases; counter clockwise adjustment potentiometer, reducing detection distance.

2. The sensor active infrared reflection detection, target reflectivity and therefore the shape is critical detection distance. Where the minimum detection distance black, white, maximum; small objects away from a small area, a large area from the Grand.

3. The sensor module output port OUT port can be directly connected to the microcontroller IO can also be directly drive a 5V relay; Connection: VCC-VCC;

4. GND-GND; OUT-IO

5. With the screw holes 3mm, easy fixed installation;

6. Board size: 3.2CM * 1.4CM

7. Each module has been shipped threshold comparator voltage adjusted by potentiometer good, non-special case, do not adjustable potentiometer.

5.6 L293D MOTOR DRIVER MODULE

L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. **L293D** is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motor with a single L293D IC. Dual H-bridge Motor Driver integrated circuit (IC).

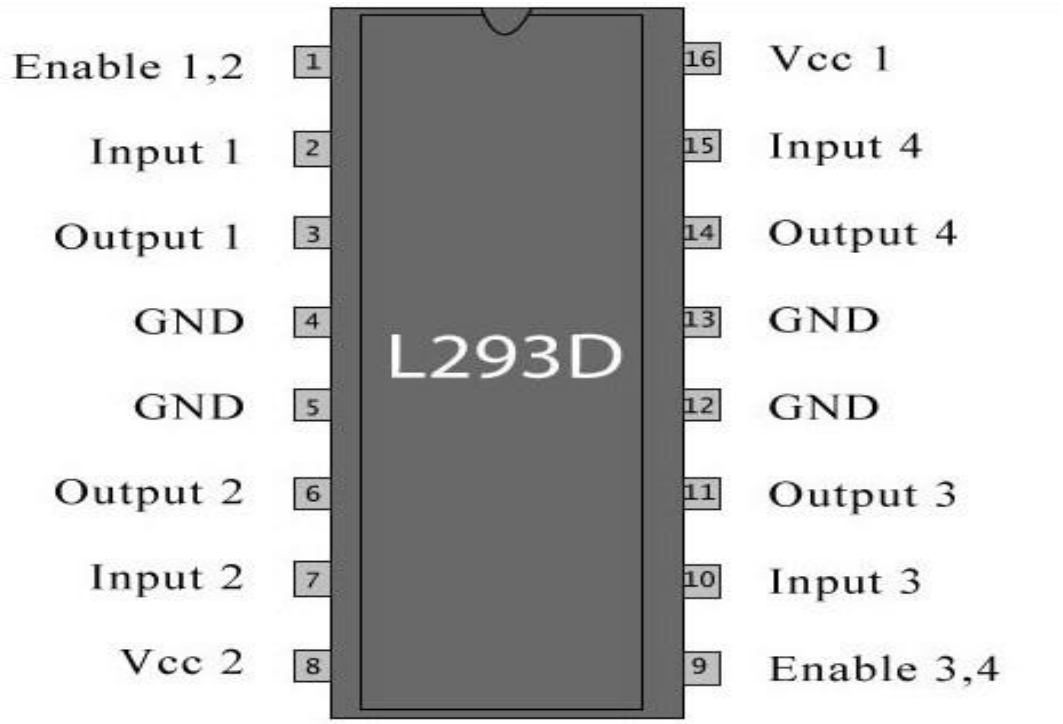


Fig 5.6 L293D Motor Driver Module

Detailed Description:

It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. As you know voltage need to change its direction for being able to rotate the motor in clockwise or anticlockwise direction, Hence H-bridge IC are ideal for driving a DC motor.

In a single L293D chip there are two h-Bridge circuit inside the IC which can rotate two dc motor independently. Due its size it is very much used in robotic application for controlling DC motors. Given below is the pin diagram of a L293D motor controller.

There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high. For driving the motor with left H-bridge you need to enable pin 1 to high. And for right H-Bridge you need to make the pin 9 to high. If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working. It's like a switch.

Instructions/Manuals/Technical details:

VCC is the voltage that it needs for its own internal operation 5v; L293D will not use this voltage for driving the motor. For driving the motors, it has a separate provision to provide motor supply VSS (V supply). L293d will use this to drive the motor. It means if you want to operate a motor at 9V then you need to provide a Supply of 9V across VSS Motor supply.

The maximum voltage for VSS motor supply is 36V. It can supply a max current of 600mA per channel. Since it can drive motors Up to 36v hence you can drive pretty big motors with this l293d.

VCC pin 16 is the voltage for its own internal Operation. The maximum voltage ranges from 5v and up to 36v.

TIP: Don't Exceed the Vmax Voltage of 36 volts or it will cause damage.

Let's consider a Motor connected on left side output pins (pin 3,6). For rotating the motor in clockwise direction, the input pins has to be provided with Logic 1 and Logic 0.

- Pin 2 = Logic 1 and Pin 7 = Logic 0 | Clockwise Direction
- Pin 2 = Logic 0 and Pin 7 = Logic 1 | Anticlockwise Direction
- Pin 2 = Logic 0 and Pin 7 = Logic 0 | Idle [No rotation]
- Pin 2 = Logic 1 and Pin 7 = Logic 1 | Idle [No rotation]

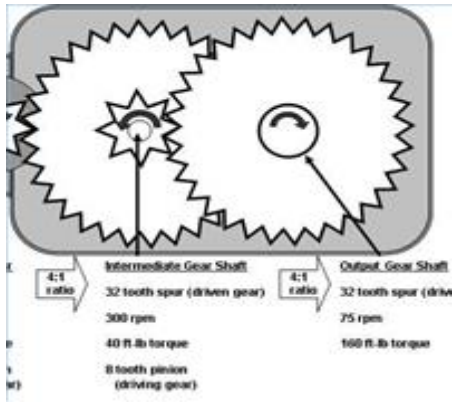
In a very similar way the motor can also operate across input pin 15,10 for motor on the right hand side.

Key Features:

- Design based on highly proven IC L293D driver
- Direct input from 5V microcontroller for L293D driver supply.
- Output terminal for both motors.
- Powered from external 12V or from wire header

5.7 GEAR MOTOR

Gear motors are complete motive force systems consisting of an electric motor and a reduction gear train integrated into one easy-to-mount and -configure package. This greatly reduces the complexity and cost of designing and constructing power tools, machines and appliances calling for high torque at relatively low shaft speed or RPM. Gear motors allow the use of economical low-horsepower motors to provide great motive force at low speed such as in lifts, winches, medical tables, jacks and robotics. They can be large enough to lift a building or small enough to drive a tiny clock.



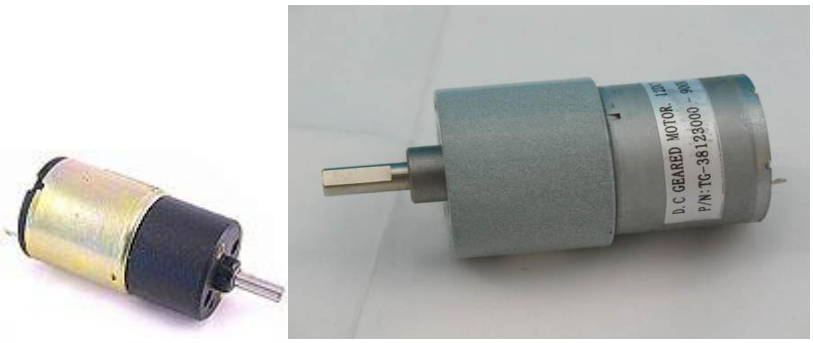


Fig 5.7 12V High Torque DC GEAR MOTOR

Operation Principle :

Most synchronous AC electric motors have output ranges of from 1,200 to 3,600 revolutions per minute. They also have both normal speed and stall-speed torque specifications. The reduction gear trains used in gear

motors are designed to reduce the output speed while increasing the torque. The increase in torque is inversely proportional to the reduction in speed. Reduction gearing allows small electric motors to move large driven loads, although more slowly than larger electric motors. Reduction gears consist of a small gear driving a larger gear. There may be several sets of these reduction gear sets in a reduction gear box.

Speed Reduction:

- Sometimes the goal of using a gear motor is to reduce the rotating shaft speed of a motor in the device being driven, such as in a small electric clock where the tiny synchronous motor may be spinning at 1,200 rpm but is reduced to one rpm to drive the second hand, and further reduced in the clock mechanism to drive the minute and hour hands. Here the amount of driving force is irrelevant as long as it is sufficient to overcome the frictional effects of the clock mechanism.

Torque Multiplication:

- Another goal achievable with a gear motor is to use a small motor to generate a very large force albeit at a low speed. These applications include the lifting mechanisms on hospital beds, power recliners, and heavy machine lifts where the great force at low speed is the goal.

Motor Varieties:

- Most industrial gear motors are AC-powered, fixed-speed devices, although there are fixed-gear-ratio, variable-speed motors that provide a greater degree of control. DC gear motors are used primarily in automotive applications such as power winches on trucks, windshield wiper motors and power seat or power window motors.

Many Applications:

What power can openers, garage door openers, stair lifts, rotisserie motors, timer cycle knobs on washing machines, power drills, cake mixers and electromechanical clocks have in common is that they all use various integrations of gear motors to derive a large force from a relatively small electric motor at a manageable speed. In industry, gear motor applications in jacks, cranes, lifts, clamping, robotics, conveyance and mixing are too numerous to count

CHAPTER - 6

RESULTS AND DISCUSSION

6.1 RESULTS & DISCUSSION

1. The experiment on movable road divider was conducted with many IR sensors which are deployed on the either side of the divider to detect the presence of density of vehicles, it senses the traffic density. Now depending on density of vehicles as determined by observing the IR sensors the divider is moved on either side to give more space for the traffic to flow smoothly in the dense area.

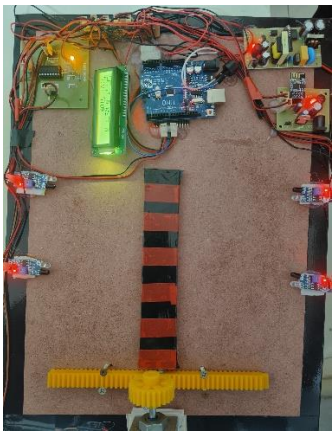


Fig 6.1 Divider in Constant Position

2. The below Fig represents the prototype of Movable Road Divider, where the traffic is high in lane I and the divider is 1moved towards lane 2 after intimating the drivers about the movement of the divider through the buzzer.

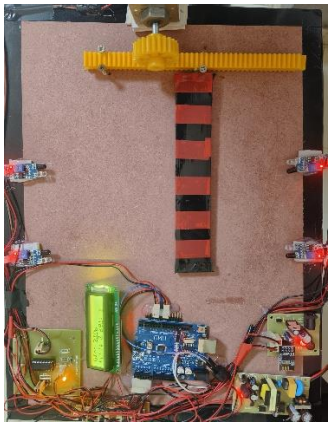


Fig 6.2 Divider is Moving Towards Right side

3. The below Fig represents the prototype of Movable Road Divider, where the traffic is high in lane 2 and the divider is moved towards lane I after intimating the drivers about the movement of the divider through the buzzer and the divider is moved back once the traffic is normal.

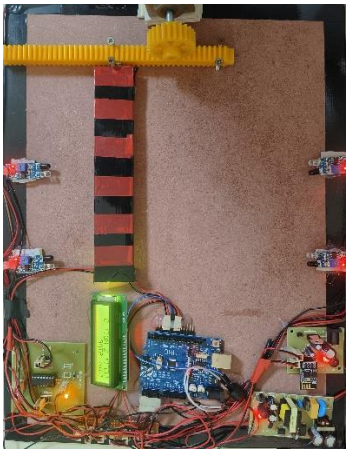


Fig 6.3 Divider is Moving Towards left side

4. The density of the traffic is displayed for both the lanes. If the density of the traffic is less, the LCD display will be "Less Traffic" corresponding to the lanes. If the density of the traffic is medium, then "Medium Traffic" will be displayed on the LCD display. If the traffic is high, then "High Traffic" is displayed on the LCD, a buzzer makes the sound, indicating that the divider is about to move and then the divider is moved.

Fig 6.4 LCD Display the Traffic Density

CHAPTER - 7

CONCLUSION & FUTURE SCOPE

7.1. CONCLUSION

Traffic congestion is a significant challenge in urban areas, leading to delays, fuel wastage, pollution, and road accidents. Traditional traffic management systems, including fixed dividers and static lane configurations, fail to dynamically adapt to changing traffic conditions, resulting in inefficient road space utilization. The Automatic Movable Smart Road Divider system provides an innovative sensor-based solution that enables real-time lane allocation based on traffic density.

By utilizing Arduino, IR sensors, an L293D motor driver, and gear motors, the system monitors lane traffic and shifts the divider accordingly. This ensures that road space is effectively utilized, reducing congestion without requiring major infrastructure changes. The LCD display provides real-time feedback on lane adjustments, enhancing transparency and monitoring

7.2. FUTURE SCOPE

The future scope of the project "Adjustable Road Divider for Dynamic Traffic Management using IoT" presents numerous exciting possibilities. Integrating AI and machine learning can enhance the system's ability to predict traffic patterns and dynamically adjust dividers for optimized flow. Advances in sensor technology could improve the accuracy of traffic density detection while ensuring adaptability to environmental conditions. Scalability is another promising avenue, allowing the system to be expanded for large urban areas with intricate traffic networks. Incorporating solar-powered mechanisms could make the project more energy-efficient and sustainable.

Vehicle-to-infrastructure communication opens up possibilities for real-time coordination between vehicles and road dividers, while features for prioritizing emergency vehicles can make traffic management more efficient during critical situations. Aligning the system with smart city initiatives could lead to more cohesive urban management. Efforts to reduce costs by utilizing affordable materials and technologies would make the system more accessible .User-friendly interfaces for traffic authorities could further enhance operational efficiency, while global adaptability could tailor the system to suit varying traffic rules and conditions worldwide. Moreover, providing commuters with real-time traffic updates through mobile applications or digital boards can improve travel planning.

REFERENCES

[1] Smith, J., & Brown, A. (2018). Sensor-based traffic control systems: A review. International Journal of Smart Infrastructure, 12(4), 230-245.

[2] Gupta, R., & Sharma, K. (2020). IoT-based real-time traffic management using ultrasonic sensors. *IEEE* Transactions on Transportation Systems, 7(2), 110-125.

[3] Kumar, P., & Patel, M. (2021). Computer vision in traffic monitoring: Challenges and advancements. Journal of Artificial Intelligence in Transportation, 9(1), 55-72.

[4] Wang, L., & Zhang, T. (2019). Automated movable road dividers for dynamic lane allocation. Smart Traffic Engineering, 15(3), 90-105.

[5] Sharma, R., & Reddy, S. (2022). Arduino-based smart road divider for urban traffic optimization. International Journal of Embedded Systems, 10(5), 340-355.

[6] Chen, Y., & Lee, J. (2023). LIDAR-based lane monitoring for intelligent traffic management. Smart Transportation Research, 6(2), 120-135.

[7] Singh, A., & Verma, S. (2021). Adaptive traffic control using Arduino and IR sensors. International Conference on Embedded Technologies, 78-89.

[8] Jadhav, P., & Mehta, N. (2022). Implementation of dynamic lane switching using microcontrollers. Journal of Intelligent Systems and Control, 5(3), 210-225.

[9] Department of Transport and Infrastructure. (2021). Smart traffic solutions for urban mobility. Government White Paper.

[10] World Economic Forum. (2022.Future of urban mobility and traffic automation. Annual Mobility Report.

[11] S. Jyothirmayee, G. Vamshi Krishna, 1. Nanditha, B. Shashank Yadav, "Controlling of Traffic using Movable Road Dividers", IJAERD, Volume 5, Issue 04, April 2018.

[12] Er. Faruk Bin Poyen, Amit Kumar Bhakta, B. Durga Manohar, Imran Ali, Arghya Santra, Awanish Pratap Rao "Density Based Traffic Control", Vol 2, Issue 8, Aug 2016.

[13] Rajeshwari Sundar, Santhosh S Hebbar, Varaprasad Golla, "Implementing intelligent Traffic Control System for Congestion Control Ambulance Clearance and Stolen Vehicle Detection", IEEE Sensors journal,vol. 15, no. 2, February 2015.

[14] Shabbir Bhusari, "Traffic control system using Raspberry-pi", Global Journal of Advanced Engineering Technologies, vol. 4, no. 4, pp. 413-415, MARCH 2015

[15] K. Vidhya, A. Bazila Banu, "Density Based Traffic Signal System", Volume 3, Special Issue 3, March 2014.

[16] Priyanka Khanke, Prof. P. S. Kulkarni, "A Technique on Road Track Analysis using Image Processing", Vol. 3 Issue 2, February 2014.

[17] Soufiene Djahel, "Reducing Emergency Services Response Time in Smart Cities: An Advanced Adaptive and Fuzzy Approach", IEEE, pp. 978-986, 2015

[18] S. Lokesh, "An Adaptive Traffic Control System Using Raspberry PI", International journal of engineering sciences & research Technology, pp. 831-835, June 2014.

[19] Shivashankar "Improvement of Speed in Data Collection Rate in Tree based Wireless Sensor Network", 2016 IEEE conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT 2016), ISBN: 978-1-5090-0774-5, May 20th and 21st, 2016.pp-720-723.

[20] Shivashankar "Design and development ofnew apparatus in VANETs for safety and accident avoidance. ", 2016 IEEE conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT-2016), ISBN: 978-1-5090-0774-5, May 20th and 21st, 2016.pp 1695-1698.

[21] Shivashankar "Development of Agile Frequency Synthesizer.", 2016 IEEE conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT-2016), ISBN: 978-1-5090-0774-5, May 20th and 21st, 2016.pp- 1943-1945.

[22] Shivashankar "An Efficient Mobile Sink Path Selection Approach for WSNs", 2016 IEEE conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT-2016), ISBN: 978 1-5090-0774-5, May 20th and 21st, 2016.pp- 151-155.