**CHAPTER 4. SYSTEM DESCRIPTION**

**4.1 DESIGN**

**BLOCK DIAGRAM OF SMART GRASS CUTTER:**

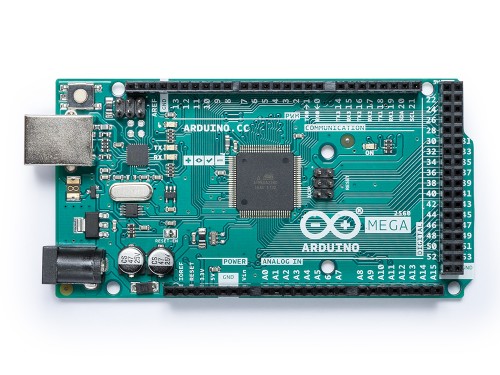
**FLOWCHART OF SMART GRASS CUTTER:**

**4.2 HARDWARE AND SOFTWARE SPECIFICATIONS**

* **HARDWARE:**

1. **Arduino Mega 2560 (1)**

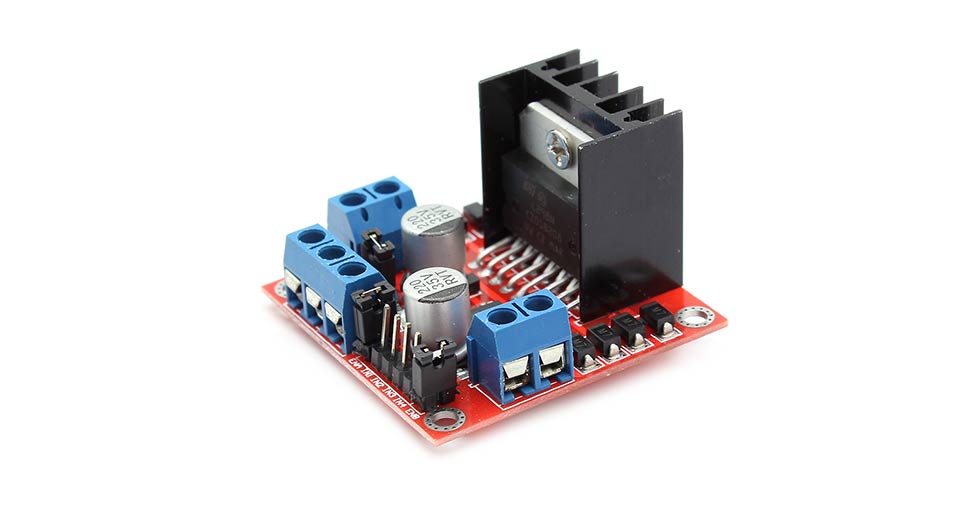
The ArduinoMega2560 is a microcontroller board based on the [Atmega2560](http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf). It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega 2560 board is compatible with most shields designed for the Uno and the former boards Duemilanove or Diecimila.



**Figure 4.1: Arduino Mega**

1. **L298N Motor Driver (1)**

The L298N is an integrated monolithic circuit in a 15- lead Multiwatt and PowerSO20 packages. It is a high voltage , high current dual full-bridge driver de-signed to accept standard TTL logic level sand drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the in-put signals .The emitters of the lower transistors of each bridge are connected together rand the corresponding external terminal can be used for the connection of an external sensing resistor. An additional Supply input is provided so that the logic works at a lower voltage.



**Figure 4.2:L298N Motor Driver**

1. **DC Motors (12V) (3)**

The speed controller works on the fundamental by varying the average voltage sent to the motor. It could do this by simply adjusting the voltage sent to the motor, but this an inefficient method. A better way is to switch the motor supply on and off very quickly. If the switching is fast enough, the motor functioning does not get affected , it only notices the average effect.



**Figure 4.3: DC Motor**

1. **Ultrasonic Sensor (HC-SR04) (1)**

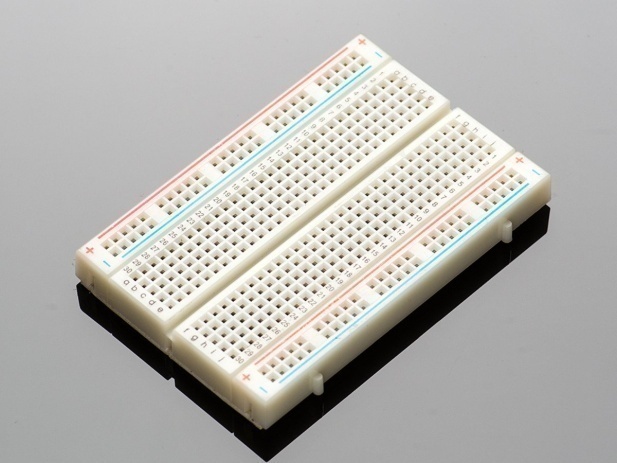
The Ultrasonic distance sensor provides precise, non-contact distance measurements from about 2cm to 3meters. It is very easy to connect to Micro Controllers, propeller chip, or arduino, requiring only one i/o pin. The sensor has amale3-pinheaderused to supply ground, power and signal. The header may be plugged into a directly into Solder less Bread board, or in to a Standard 3- Wire Extension Cable. The sensor detects objects bye matting a short ultra sonic burst and then" listening" for the echo. Under control of a host micro controller, the sensor emits a short 40 KHz burst. This burst travels through the air, hits an object and then bounces back to the sensor. The sensor provides an output pulse to the host that will terminate when the echo is detected hence the width of this pulse corresponds to the distance to the target.



**Figure 4.4: Ultrasonic Sensor**

1. **Breadboard (1)**

A breadboard is a solderless device for temporary prototype with electronics and test circuit designs. Most electronic components in electronic circuits can be interconnected by inserting their leads or terminals into the holes and then making connections through wires where appropriate.



**Figure 4.5: Breadboard**

1. **Wheels (3)**

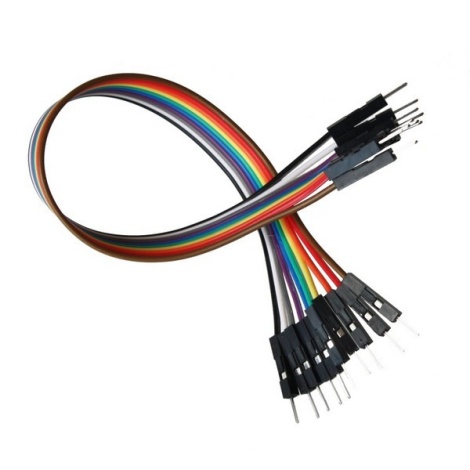
Wheels are used to move forward, backward and left or right in case of any obstacle is detected.



**Figure 4.6: Wheels**

1. **Jumper Wires (20-30)**

Jumper wires are used to connect the sensors, motors, LCD display and all the other components to the Arduino board.



**Figure 4.7: Jumper Wires**

1. **USB Cable (1)**

USB cable is used to give power supply to Arduino board.



**Figure 4.8: USB Cable**

1. **SPST Rocker Switch (1)**

**SPST**rocker switches make or break the connection of a single conductor in a single branch circuit. This switch type typically has two terminals and is referred to as a single-pole switch.



**Figure 4.9: SPST Rocker Switch**

* **SOFTWARE:**

1. **Arduino IDE**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.  
This software can be used with any Arduino board.

1. **C Programming Language**

C is a general-purpose programming language that is extremely popular, simple and flexible.

It is machine-independent, structured programming language.

**Table No 4.1 Hardware, Software Requirements for Smart Grass Cutter.**

|  |  |
| --- | --- |
| **HARDWARE REQUIREMENTS** | **SOFTWARE REQUIREMENTS** |
| Arduino Mega 2560 | Arduino IDE |
| L293N Motor Driver | C Programming Language |
| DC Motors |  |
| SPST Rocker Switch |  |
| Ultrasonic Sensor (HC-SR04) |  |
| Breadboard |  |
| Wheels |  |
| Jumper Wires and USB cable |  |

**4.3 IMPLEMENTATION METHODOLOGY**

STEP 1: Analyzed project requirements and collected various components required to build the project.

STEP 2: Studied the specification of each component along with the pin configuration of the Ultrasonic sensor, L298N motor driver, DC motors and Arduino Mega board.

STEP 3: Tested each component individually, checked the working of sensor and motor driver with motors.  
STEP 4: Ultrasonic sensor interfaced with arduino.

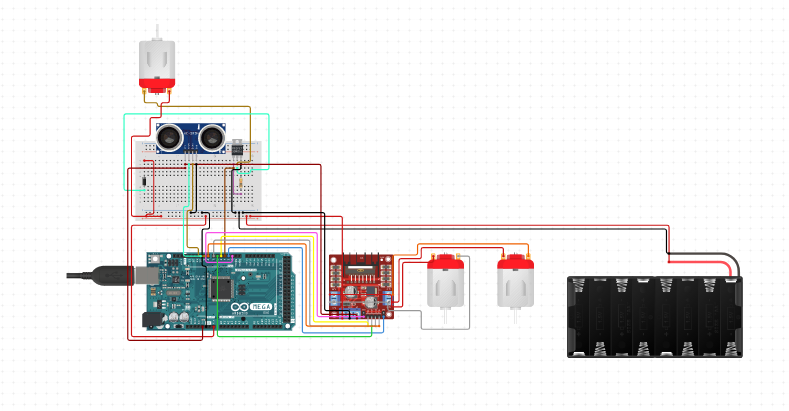
STEP 5: DC motors interfaced with motor driver and the arduino.

STEP 6: Developed the model with plywood and angles (dc motor holder).

STEP7: Connected the sensor and motor driver with arduino board and merged the code for all the components that are used.

STEP 8: Placed all the components on to the plywood model.

**4.4 HARDWARE CIRCUIT DIAGRAM**



**Figure 4.10: Circuit Diagram**

**4.5 CODE**

#include <NewPing.h>

const int LeftMotorForward = 7;

const int LeftMotorBackward = 6;

const int RightMotorForward = 5;

const int RightMotorBackward = 4;

//sensor pins

#define trig\_pin A1 //analog input 1

#define echo\_pin A2 //analog input 2

#define maximum\_distance 200

boolean goesForward = false;

int distance = 100;

NewPing sonar(trig\_pin, echo\_pin, maximum\_distance); //sensor function

void setup(){

pinMode(RightMotorForward, OUTPUT);

pinMode(LeftMotorForward, OUTPUT);

pinMode(LeftMotorBackward, OUTPUT);

pinMode(RightMotorBackward, OUTPUT);

}

void loop(){

int distanceRight = 0;

int distanceLeft = 0;

delay(50);

if (distance <= 20){

Serial.println(distance);

moveStop();

delay(300);

moveBackward();

delay(400);

moveStop();

delay(300);

if (distance >= distanceLeft){

turnRight();

moveStop();

}

else{

turnLeft();

moveStop();

}

}

else{

moveForward();

}

distance = readPing();

}

int readPing(){

delay(70);

int cm = sonar.ping\_cm();

if (cm==0){

cm=250;

}

return cm;

}

void moveStop(){

digitalWrite(RightMotorForward, LOW);

digitalWrite(LeftMotorForward, LOW);

digitalWrite(RightMotorBackward, LOW);

digitalWrite(LeftMotorBackward, LOW);

}

void moveForward(){

if(!goesForward){

goesForward=true;

digitalWrite(LeftMotorForward, HIGH);

digitalWrite(RightMotorForward, HIGH);

digitalWrite(LeftMotorBackward, LOW);

digitalWrite(RightMotorBackward, LOW);

}

}

void moveBackward(){

goesForward=false;

digitalWrite(LeftMotorBackward, HIGH);

digitalWrite(RightMotorBackward, HIGH);

digitalWrite(LeftMotorForward, LOW);

digitalWrite(RightMotorForward, LOW);

}

void turnRight(){

digitalWrite(LeftMotorForward, HIGH);

digitalWrite(RightMotorBackward, HIGH);

digitalWrite(LeftMotorBackward, LOW);

digitalWrite(RightMotorForward, LOW);

delay(500);

digitalWrite(LeftMotorForward, HIGH);

digitalWrite(RightMotorForward, HIGH);

digitalWrite(LeftMotorBackward, LOW);

digitalWrite(RightMotorBackward, LOW);

}

void turnLeft(){

digitalWrite(LeftMotorBackward, HIGH);

digitalWrite(RightMotorForward, HIGH);

digitalWrite(LeftMotorForward, LOW);

digitalWrite(RightMotorBackward, LOW);

delay(500);

digitalWrite(LeftMotorForward, HIGH);

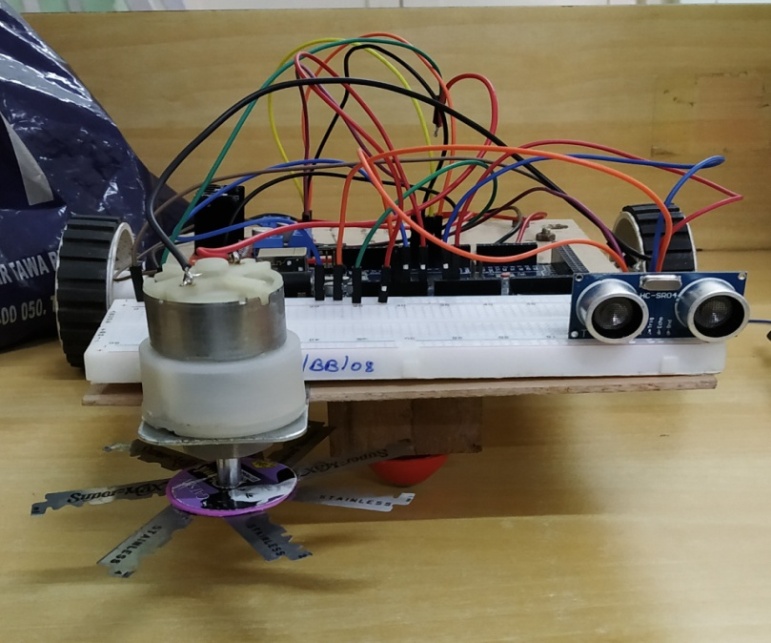
digitalWrite(RightMotorForward, HIGH);

digitalWrite(LeftMotorBackward, LOW);

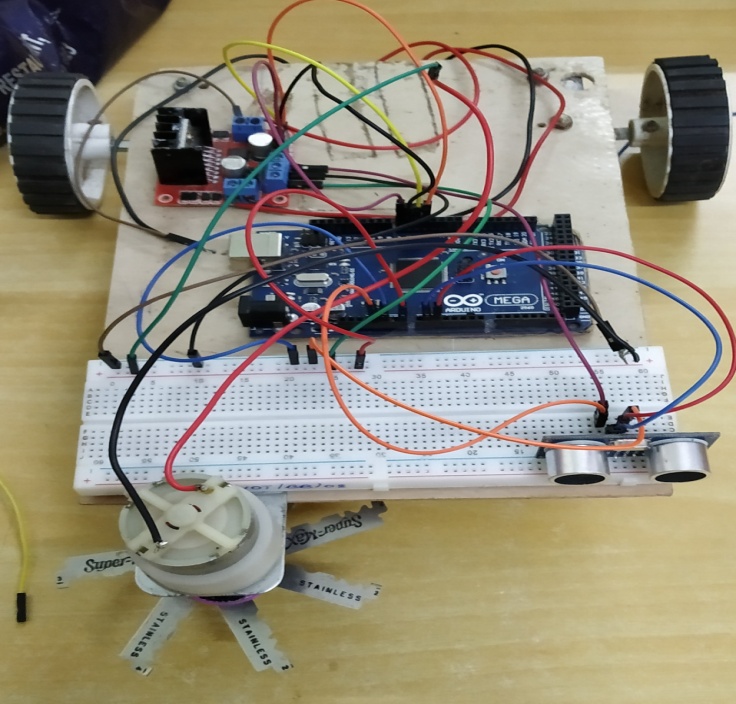
digitalWrite(RightMotorBackward, LOW);

}

**4.6 FINAL PROTOTYPE**

****

**Figure 4.11: Front view of Prototype.**

****

**Figure 4.2: Top view of Prototype.**

**4.7 CONCLUSION AND FUTURE SCOPE OF THE PROJECT**

* **CONCLUSION:**

The previous bot systems were studied and a suitable design was made. The schematic for the same was made on which prototyping will take place. The components have been chosen based on design requirement and based on a few other parameters. The prototype is mounted on a bakelite chassis and the detection is done using ultrasonic sensor and the output was obtained. Since grassing cutting is a mundane task requiring a lot of time, it is believed that human time should not be wasted on such tasks or at least reduced to the bare minimum. The cost effectiveness and the ease provided makes the bot to be a necessity instead of a luxury.

* **FUTURE SCOPE:**

Size can be reduced to make it compact. Efficiency can be amended by incrementing the battery capacity. More sensors can be incorporated for precise results. Bluetooth or Wi-Fi module can be used to operate the robot from anywhere through mobile phone. Solar panels can be used to store power in order to charge the battery. Programming can be enhanced to make the device perform different operations.

**4.8 CONSTRAINTS FOR REAL TIME DEPLOYMENT**

* The battery consumption of the robot is high, hence frequent charging of the battery is required.
* If the height of the grass is too high, the ultrasonic sensor will detect it as an obstacle and won’t cut the grass.
* The prototype is difficult to operate during rainy season.
* High speed cutter motor required to cut the grass smoothly.