**CHAPTER 4**

**SOFTWARE IMPLEMENTATION**

**4.1 Software Implementation**

In this chapter, usage of Arduino software implementation in Elevator is going to be explained. The following steps are used in this thesis. They are:

* Downloading and Installation of Arduino IDE software
* Features Arduino IDE Software
* Arduino Program Analysis

4.1.1. Downloading and Installation of Arduino IDE software

The various types of Arduino IDE can get from the download page on the Arduino official website. The latest Arduino Environment can be found at htpp:// Arduino.cc/en/Main/Software. The user must select the software, which is compatible with the operating system (window, IOS, and Linux). On the website, click on the getting started page to download the latest Arduino IDE software. For window user, they should do double-clicking on windows Installer.

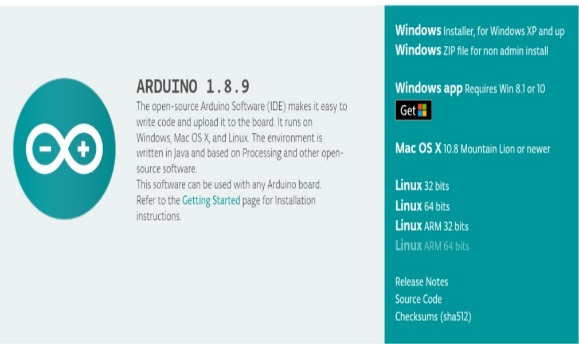
****

Figure 4.1. Arduino IDE Software

Install the Arduino IDE software by running the download file and follow the instructions. Installing the IDE software is like installing any software on a PC, lots of mouse clicking. Read the Arduino License agreement and click the “I Agree” button.

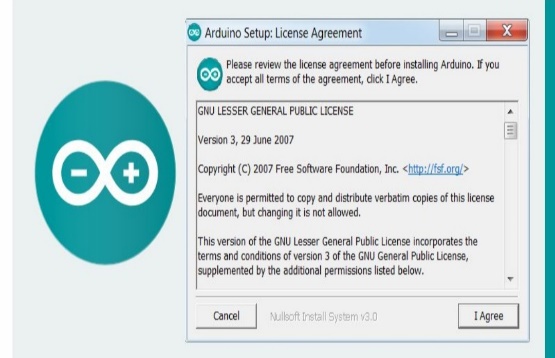


Figure 4.2. License Agreement

Unless the user has a reason to change, keep all the Arduino IDE software components ticked and click the “Next” button.

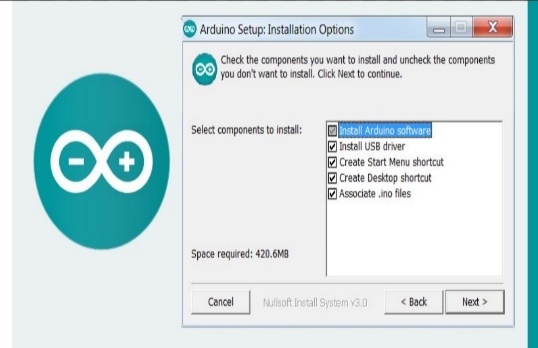
****

Figure 4.3. Installation Options

On the next window, the user can choose to change the folder where the Arduino IDE software is installed, or keep the default location and click the “Install” button. The Arduino software will start to install.

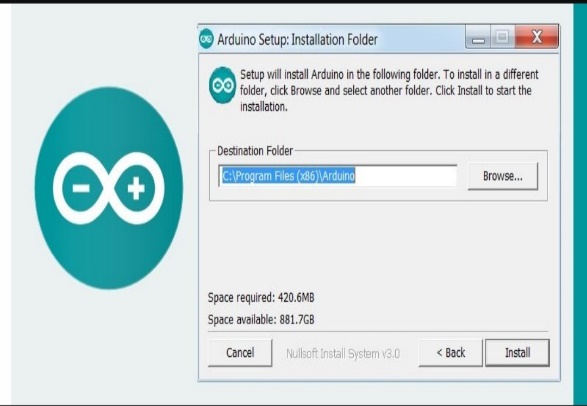
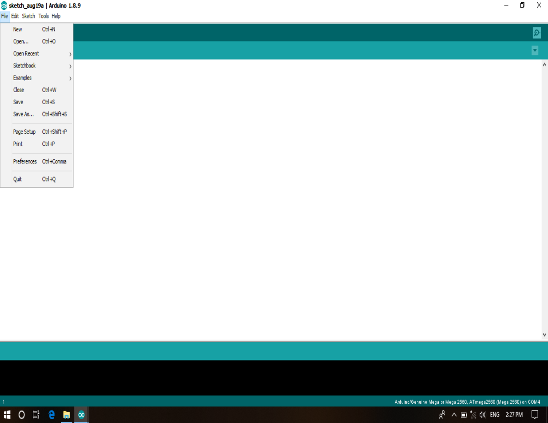


Figure 4.4. Installation Folder

This can take sometimes and the user will be presented with various drivers to install including Arduino USB drivers and COM/ LPT port drivers, the user installs them all. When complete the Arduino IDE Software is ready to run.

4.1.2. Features of Arduino IDE

* The project file or the sketch for a project are saved with the file extension.
* Features such as cut/copy/paste are supported in this IDE
* There also is a facility for finding a particular word and replacing it with another by pressing the Ctr +F button on the Keyboard.
* The most basic part or the skeleton of all Arduino code will have two functions.



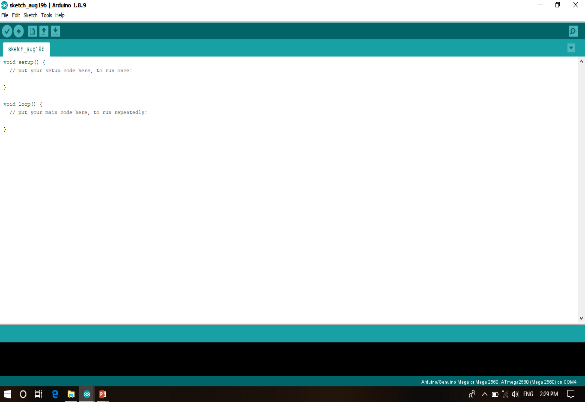


Figure 4.5. Selecting New Sketch

4.1.3. Arduino Program Analysis

The users need to start the Arduino IDE (with Arduino UNO board already connected to the computer) and open the blink sketch. In the sketch shown above, the user can see some lines in grey color in the top of the sketch and some colorful lines in the bottom of the sketch. This is because, any Arduino sketch (or as a matter of fact any program irrespective of the programming language) is a combination of some reference line called as comments and the actual code itself. Comments, as the name indicates, is a part of the program that informs any other going through the program about the way how the program works. They help the source code to be easily understandable by the programmer or user. Generally, comments are ignored by compilers. Hence, do not take up any space while compiling. Comments are again divided in to block comments (also called as multi line comments) and line comments (also called as in line comments). Multiline comments annotate a block or multiple lines in the source code. The grey part in the beginning of the Blink sketch are block comments. They give information like who the developer is, aim of the code, etc.

Multi line comments are usually enclosed between slash (/) – asterisk (\*) and asterisk – slash i.e. “/\*Multiline Comment\*/”. The other type of comments are line comments. They are used to annotate a single line in the code. Single line comments are generally indicated with a double slash in the beginning. The user can see at the end of each line in the code there is a statement with “//”. This is a single line comment. The other part of the sketch is the actual code which is written for Arduino to do a specific task.

In Arduino environment, the sketch is divided in to two parts:

* Setup
* Loop

The setup () is the first function that is executed or called when program starts. It runs only once when the power is turned on or every time the Arduino is reset. Setup is used to initialize pins, variables, libraries etc.

The next function in the Arduino sketch is the loop. As the name indicates, a loop () is a function that runs over and over again i.e. it loops consecutively.

4.1.3.1. Pin Mode

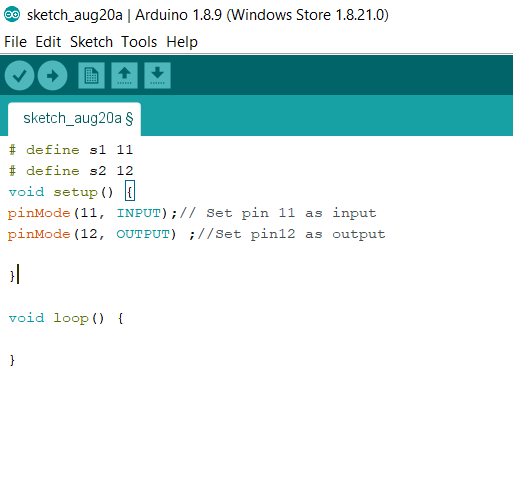


Figure 4.6. PinMode ( ) Function

This command is used to set pins as input or output in the setup loop. Arduino digital pins are set to input by default, hence there is no need to specifically declare them as input using pin mode ( ) .When a pin is configured as input, external data i.e logic HIGH or LOW is given to the Arduino. When a pin is configured as output, Arduino will send either logic LOW or High to that’s pins.

4.1.3.2. DigitalWrite ( )

Sets the pins on or off, by giving them either HIGH or LOW output. The function takes two parameters, the pin number and either HIGH or LOW. The syntax of digitalWrite function is “digitalWrite (pin no, value)”.



Figure 4.7. DigitalWrite ( ) Function

4.1.3.3. DigitalRead ( )

The function is used to read the digital value of a pin. The result is HIGH or LOW. The syntax of digitalRead function is “digitalRead (pin no);”.

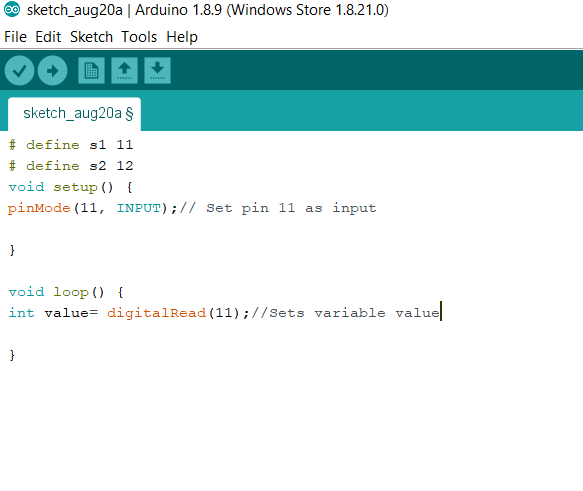


Figure 4.8. DigitalRead ( ) Function

4.1.3.4. AnalogWrite ( )

The function is used to give an analog output. Analog outputs can only be given to pins 3,5,6,9,10 and 11 on the UNO. Vales between 0-255 can be given to the analog pins.

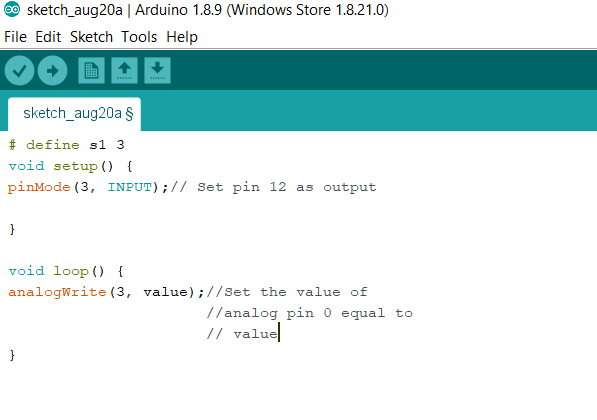


Figure 4.9. AnalogWrite ( ) Function

4.1.3.5. AnalogRead ( )

This function reads the value from a specified analogue pin and returns a value between 0-1023. It only works on the six analogue pins from 0-5.

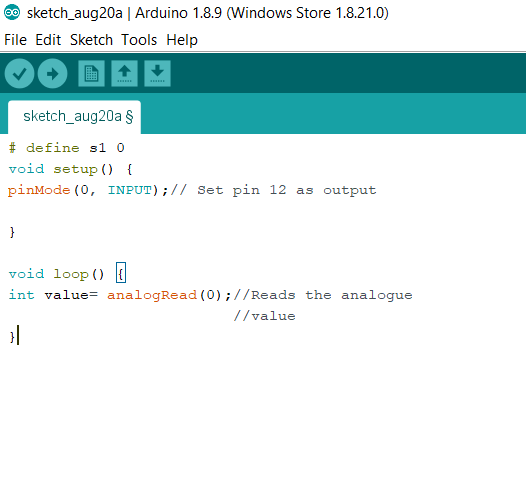


Figure 4.10. AnalogRead ( ) Function

**4.2. Arduino Coding for Stepper Motor and Motor Driver**

Stepper motor is used for up and down movement. Stepper motor is operated by the motor driver.

#include<Stepper.h>

const int stepsPerRevolution = 200;

Stepper myStepper(stepsPerRevolution,8,9,10,11);

void setup()

{ myStepper.setSpeed(65);

Serial.begin(9600);

pinMode(8,OUTPUT);

pinMode(9,OUTPUT);

pinMode(10,OUTPUT);

pinMode(11,OUTPUT);

}

void loop()

{ myStepper.step(885);

}

**4.3. Arduino Coding for Seven Segment Display**

Seven segment display is used to show the location of car in the elevator control system.

#define segA 22

#define segB 24

#define segC 26

#define segD 28

#define segE 30

#define segF 32

#define segG 34

void setup()

{

pinMode(segA,OUTPUT);

pinMode(segB,OUTPUT);

pinMode(segC,OUTPUT);

pinMode(segD,OUTPUT);

pinMode(segE,OUTPUT);

pinMode(segF,OUTPUT);

pinMode(segG,OUTPUT);

}

void loop()

{ digitalWrite(segA,LOW);

digitalWrite(segB,HIGH);

digitalWrite(segC,HIGH);

digitalWrite(segD,LOW);

digitalWrite(segE,LOW);

digitalWrite(segF,LOW);

digitalWrite(segG,LOW);

}

**4.4. Arduino Coding for Inductive Sensors**

Inductive sensors are used to detect the floor level which the elevator cabin exists.

#define ss1 2

#define ss2 3

#define ss3 4

#define LED 5

void setup()

{ Serial.begin(9600);

pinMode(ss1,INPUT);

pinMode(ss2,INPUT);

pinMode(ss3,INPUT);

pinMode(LED,OUTPUT);

}

void loop()

{ if(digitalRead(ss1)||digitalRead(ss2)||digitalRead(ss3)==HIGH)

{ digitalWrite(LED,HIGH);

}

}

**4.5. Arduino Coding for Ultrasonic Sensor**

The ultrasonic sensor is used to sense the object within its range. If the object is out of range, DC motor is operated for door system.

#define trigPin 13

#define echoPin 12

void setup()

{ pinMode(trigPin,OUTPUT);

pinMode(echoPin,INPUT);

Serial.begin(9600);

}

void loop()

{ long duration, distance;

DigitalWrite(trigPin,LOW);

DelayMicroseconds(10);

digitalWrite(trigPin,HIGH);

delayMicroseconds(10);

digitalWrite(trigPin,LOW);

duration=pulseIn(echoPin,HIGH);

distance=(duration/2)/29.1;

Serial.print(distance);

Serial.printIn(“cm”);

}

**4.6. Arduino Coding for Keypad**

The user can use the keypad for choosing the desired floor. The keypad is kept within the elevator.

const int buttonPin1 = 5;

const int buttonPin2 = 6;

const int buttonPin3 = 7;

int buttonState1 = 0;

int buttonState2 = 0;

int buttonState3 = 0;

int lastButtonState1 = 0;

int lastButtonState2 = 0;

int lastButtonState3 = 0;

boolean level1 = true;

boolean level2 = false;

boolean level3 = false;

void setup()

{ pinMode(buttonPin1,INPUT);

pinMode(buttonPin2,INPUT);

pinMode(buttonPin3,INPUT);

}

void loop()

{ buttonState1 = digitalRead(buttonPin1);

buttonState2 = digitalRead(buttonPin2);

buttonState3 = digitalRead(buttonPin3);

if(buttonState1 !=lastButtonState1|| buttonState2 !=lastButtonState2 ||buttonState3 != lastButtonState3)

{ if(level1 == true ){

if(buttonState1 ==HIGH) {

level1 = true;

level2 = false;

level3 = false;

} }