

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University M'Hamed BOUGARA – Boumerdes



Institute of Electrical and Electronic Engineering
Department of Electronics

Final Year Project Report Presented in Partial Fulfilment of
the Requirements for the Degree of

MASTER

In Electronics

Option: Computer Engineering

Title:

Propeller LED display using Arduino

Presented by:

- **SAADI Adem**
- **CHAOUCH Youcef**

Supervisor:

DR. BELAIDI Hadjira

Registration Number:...../2019

ABSTRACT

As a part of an Electrical Engineering program, it is important to understand the practical application of all those things and concepts that have been taught theoretically, this project report is a summary of practical things learnt by us during the period of our project execution.

This report documents the design, implementation and testing process for a “propeller LED display”: a system in which LEDs move rapidly to create the illusion of a much higher resolution display. A propeller LED display has many advantages over a traditional CRT, LCD or LED display, like power savings, less complexity, easy configuration, attractiveness etc.

The report details the design and building process for the hardware and software of the system, and the integration of these subsystems into a finished product. The hardware incorporates a Propeller part and high-speed brushless motor; the electronics aspect involves designing and manufacturing a printed circuit board consisting of 17 LEDs (11 red, 5 green, 1 blue); software was then written to address and control these LEDs from an Arduino microcontroller. Several programs were written for computing/displaying different images on the LED display.

Overall, the project was a success. Minor issues were identified in testing and subsequently fixed: these are discussed within the report. Potential improvements to the system (namely wireless connectivity and a more balanced, stable structure) are also discussed.

Dedication

This is for you Mom. Thanks for always being there for me. Thanks for your unconditional love. Thanks for everything.

To Dad, who helped and supported me my whole life. You gave me strength when I was hopeless. Thank you

To my great sister Narimane and my little brother Azouaou. Am nothing without you .

To my friends, specially the different ones.

To Boumerdes because wherever you go becomes a part of you somehow.

To the strongest person I know: me.

SAADI Adem

I dedicate this humble work

To my parents who have always been close to me and found them with me
Whenever I needed. It is their unconditional love that motivates me
to set higher targets.

To my sisters and brothers who are my nearest friends before being my siblings.

To all my former friends and teachers of primary school, middle school
and high school, you have always been in my heart.

To all my friends of the institute with whom I shared unforgettable moments.

To everyone I have ever known even for a moment.

CHAOUCH Youcef

Acknowledgment

We would like to acknowledge many people for their role in the completion of this project. We want first to express our gratitude to our supervisors

First of all, we thank Allah, the Most Beneficent, the Most Merciful, for guiding and helping us to finish this humble work.

Throughout our project work, we have been extremely fortunate in having the support and the encouragement of many people. We would like to thank everyone who helped make this thesis possible. We would like to express our gratitude to our supervisor dr. Hadjira BELAIDI for her guidance, knowledge and support during our thesis work. We very much appreciate her insights, as well as the invaluable suggestions she has made to improve our thesis.

A special thanks goes to our teachers at the institute of electrical and electronics engineering, who taught us how to become real engineers and prepared us to face the challenges of life.

Most importantly, we are grateful to our families that supported us every time we were in need, always standing by us, guiding us into searching for answers and striving for a better tomorrow.

Last, but not least, we extend our thanks to our friends and all IGEE personnel.

We certainly hope this project provides as good reading as the pleasure we had to writing it.

Content

Abstract.....	I
Dedication.....	II
Acknowledgement.....	III
Contents	IV
List of tables	VII
List of figures	VIII
List of abbreviations and acronyms	X
Introduction	1
Chapter 1: POV description and generalities	
1.1. Generalities.....	3
1.1.1. Human eye.....	3
1.1.2. Persistence of vision.....	4
1.1.3. Natural occurrences and applications.....	4
a) Thaumatrope.....	4
b) Newton disc.....	5
c) Rubber pencil trick.....	6
d) Propeller LED display.....	6
1.2. Background review and design considerations of Propeller LED display.....	7
1.3. Two dimensional propeller LED display.....	8
1.3.1. Literature review.....	8
1.3.2. Resolution and colors.....	9
1.3.3. Risk assessment.....	10
1.4. Conclusion.....	10

Chapter 2: Propeller LED displa Hardware design

2.1. System overview.....	12
2.2. Propeller LED display Platform description.....	12
2.3. Hardware components.....	13
2.3.1. Arduino.....	13
a) Arduino Nano.....	14
b) Arduino UNO.....	15
2.3.2 Light Emitting Diode (LED).....	16
2.3.3. 2212 Brushless DC Motor.....	17
2.3.4. Electronic Speed Control (ESC).....	18
2.4. Schematic description	19
2.5. Conclusion.....	20

Chapter 3: Software design

3.1. Arduino software.....	22
3.1.1. The sketch.....	23
3.1.2. The uploading process.....	24
3.2. Character's transformation into codes.....	25
3.3. Software algorithm.....	26
3.3.1. Message displaying algorithm.....	26
3.3.2. Message displaying code.....	28
3.4. Algorithm steps for controlling the brushless motor's speed.....	29
3.5. Conclusion.	31

Chapter 4: Functionality testing and results

4.1. Final hardware design.....	33
4.1.1 Propeller circuit.....	33

4.1.2. BLDC motor circuit.....	34
4.2. Tests and troubleshooting.....	34
4.2.1. Testing the Arduino board and connections.....	34
4.2.2. Testing the LED module.....	35
4.2.3. Testing the motor speed and synchronization.....	35
4.3. Applications and results.....	35
4.3.1. Displaying characters' string.....	36
4.3.2. Displaying numbers.....	37
4.3.3 Displaying an analog clock.....	38
4.4. Discussion.....	40
4.4.1. Increase/decrease the delay.....	40
4.4.2. Flipping the motor's phases.....	41
4.4.3. Rotation's speed.....	41
4.5. Conclusion.....	41
Conclusion and perspectives.....	42
References	

List of tables

Table 2.1: Arduino Nano pin layout

Table 2.2: Arduino Nano specifications

Table 2.3: Technical Data of 2212 BLDC

List of Figures

Figure 1.1: Thaumatrope Trick

Figure 1.2: Newton disk

Figure 1.3: Rubber pencil trick

Figure 1.4: 2D oscillating display

Figure 1.5: 3D Propeller display

Figure 1.6: 2D Propeller display

Figure 1.7: 2D Propeller display

Figure 1.8: Seven LEDs Spin

Figure 2.1: Block Diagram

Figure 2.2: Arduino Nano

Figure 2.3: Arduino UNO

Figure 2.4: Function of LED

Figure 2.5: 2212 Brushless DC motor

Figure 2.6: Electronic Speed Controller

Figure 2.7: Propeller Circuit diagram built with Proteus 8.0

Figure 2.8: Arduino BLDC control – Circuit diagram

Figure 3.1: User interaction with computer

Figure 3.2: Example of a sketch form

Figure 3.3: Projection of the character 'B'.

Figure 3.4: Flowchart of the main program

Figure 3.5: Flowchart of the printing function

Figure 3.6: Message displaying code

Figure 3.7: Flowchart of controlling brushless motor's speed

Figure 4.1: Hardware design of Propeller circuit

Figure 4.2: Hardware design of BLDC motor circuit

Figure 4.3: Characters transformation to arrays

Figure 4.4: Propeller display of the string 'IGEE'

Figure 4.4: Numbers transformation to arrays

Figure 4.5: Propeller display of '2019'

Figure 4.6: Propeller display of a clock that point to 12:16:12

Figure 4.7: Propeller display of a clock that point to 12:16:19

Figure 4.8: Displaying 2019 with a small delay

Figure 4.9: 'INELEC' before and after flipping the phases

List of Abbreviations and Acronyms

LED: light emitting diode

CRT: cathode ray tube

LCD: liquid crystal display

RPM: rotation per minute

POV: persistence of vision

DC: direct current

AC: alternative current

Li-Po: lithium polymer

IDE: integrated development environment

USB: universal serial bus

PWM: pulse width modulation

ICSP: in circuit serial programming

BLDC: brushless direct current

ESC: electronic speed controller

MOSFET: C-MOS field effect transistor

PCB: printed circuit board

RGB: red green blue

Introduction

The aim of this project will be to design and implement a “persistence of vision display”, as it will define context of this project. The display will physically consist of a one-dimensional array of LEDs, but these LEDs will be moved rapidly in physical space in order to produce the illusion of a two-dimensional display when viewed. This movement will be in a rotating, rather than oscillating, fashion, and should be implemented using a motor. A microcontroller should be used to receive and/or compute images to be displayed.

Power must be transferred into the rotating body in order to light the LEDs: this means to achieve the goal this factor must to be considered. The microcontroller must be mounted within the rotating portion of the system for transferring the LED data into the rotating body will also need to be considered. The system should be capable of rotating safely at 24 revolutions per second (1440 RPM) in order to achieve a smooth virtual image.

The system display should, however, be capable of running independently of this revolution speed: in other words, the system must have some means of determining its current speed in order that fluctuations in the RPM do not ruin the perceived effect.

The display should be capable of displaying text and low-resolution images such as shapes and cartoon characters or even a clock.

Chapter 1

Persistence of Vision Description and generalities

Propeller is a term associated with circular rotating objects. Conventional methods of displaying images are mainly using LCD display and dot-matrix where a huge number of LED's and power processors are used to create the display. The main idea of this project is to use minimum number of LED's and components to create a virtual display with reduced power consumption.

To achieve this purpose, a set of LED's have been used; hence, the name Propeller LED display. The main advantage of propeller display as compared to the LED matrix board is its lower power consumption. The first propeller clock was created by “Bob Blick” [1], where a single array of LED's was used to produce the display. Propeller clock uses extremely small LED's to display the typescript and symbols on its assembly in an appropriate way.

1.1 Generalities

1.1.1 Human eye

Our eyes offer one of the five specialized means by which our mind is able to form a picture of the world. The eye is a remarkable instrument, having certain characteristics to help us process the light we see in such a way that our mind can create meaning from it.

Take the motion picture, the scanning of an image for television, and the sequential reproduction of the flickering visual images they produce. These work in part because of an optical phenomenon that has been called “persistence of vision” and its psychological partner, the *phi* phenomenon—the mental bridge that the mind forms to conceptually complete the gaps between the frames or pictures.

If someone is observing images at a rate of 25 images per second, then they appear to be continuous. The explanation of this phenomena is that whenever light strikes the retina, the brain retains the impression of that light for about a 10th to a 15th of a second (depending on the brightness of the image, retinal field of view, and color) after the source of that light is removed from sight. This is due to a prolonged chemical reaction. As a result, the eye cannot clearly distinguish changes in light that occur faster than this retention period. The changes either go unnoticed or they appear to be one continuous picture to the human observer.

1.1.2 Persistence of vision

Persistence of vision (POV) refers to a phenomenon whereby the human retina retains an after-image of anything it sees for a short time (approximately one twenty-fifth of a second). This phenomenon renders a person unable to distinguish between different images viewed in quick succession.

By taking advantage of this phenomenon, it should be possible to produce an electronic display that appears to have a significantly higher resolution than its hardware would ordinarily dictate, by physically and rapidly moving the display hardware in space and modifying the display output based on its current physical position. If the display hardware is moved fast enough, the eye should perceive the visual output from the hardware at multiple physical positions simultaneously, giving (for example) the perception of a two- or three-dimensional display even though the physical hardware consists of just a one-dimensional “wand” of LEDs. The aim of this project is to design and manufacture such a system.

1.1.3 Natural occurrences and applications of POV

Some natural phenomena and the principles of some optical toys have been attributed to the persistence of vision effect. Patrick d'Arcy recognized [2] the effect in "the luminous ring that we see by turning a torch quickly, the fire wheels in the fireworks, the flattened spindle shape we see in a vibrating cord, the continuous circle we see in a cogwheel that turns with speed". Basically everything that resembles « motion blur » seen in fast moving objects could be regarded as "persistence of vision" and some examples are detailed below:

a) **Thaumatrope**

In April 1825, the first *Thaumatrope* was published by W. Phillips [3]. The fact that the image of one side of the disc seems to blend with the image of the other side when it is looked at while it is twirled very fast is often used as an illustration of persistence of vision. Examples of common thaumatrope pictures include a bare tree on one side of the disk, and its leaves on the other, or a bird on one side and a cage on the other. Many classic thaumatropes also included riddles or short poems, with one line on each side.

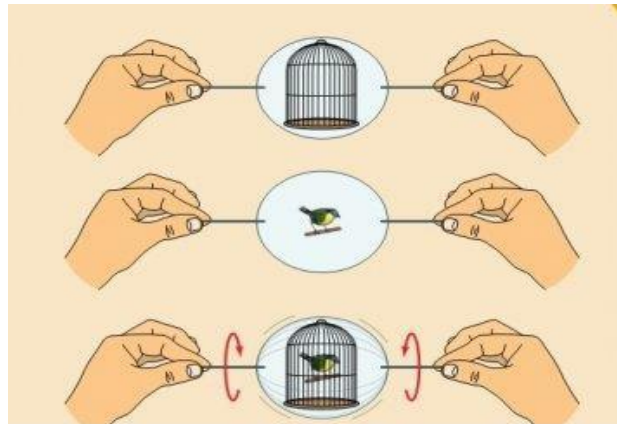


Figure 1.1. Thaumatrope Trick

b) Newton disc:

The Newton disc is a well-known physics experiment with a rotating disc with segments in different colors (usually Newton's primary colors: red, orange, yellow, green, blue, indigo and violet) appearing as white (or off-white or gray) when it spins very fast.

This type of mix of light stimuli is called temporal optical mixing, a version of additive-averaging mixing [4]. The concept that human visual perception cannot distinguish details of high-speed movements is popularly known as persistence of vision.

The disc is named after Isaac Newton; although, he published a circular diagram with segments for the primary colors that he had discovered.

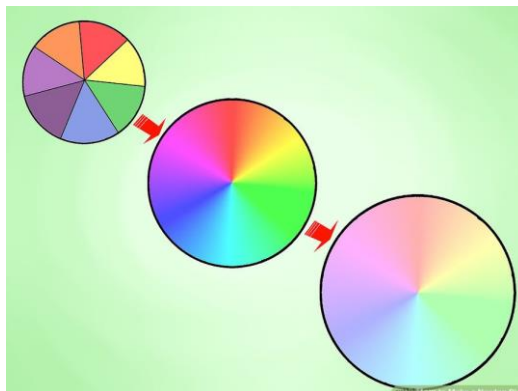


Figure 1.2 Newton disc

c) Rubber pencil trick

A pencil or another rigid straight line can appear as bending and becoming rubbery when it is wiggled fast enough between fingers, or otherwise undergoing rigid motion.

Persistence of vision has been discarded as sole cause of the illusion. It is thought that the eye movements of the observer fail to track the motions of features of the object.

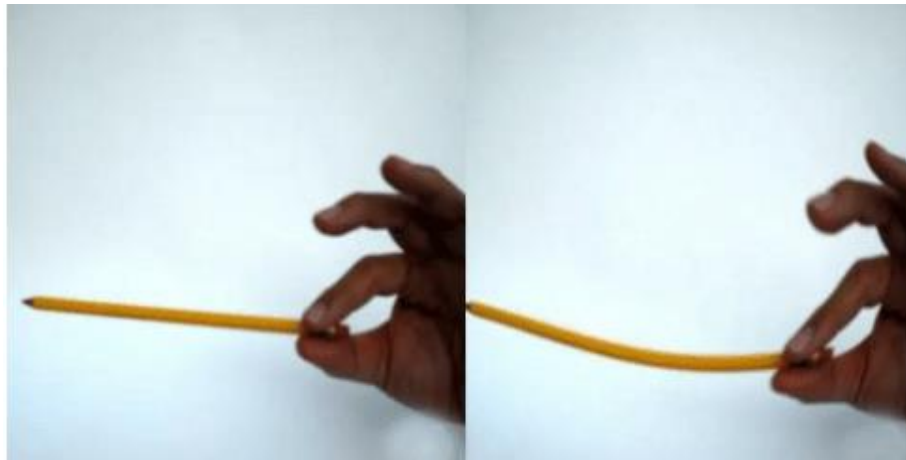


Figure 1.3. Rubber pencil trick

d) Propeller LED display

A propeller LED display is a display created by rotating an array of LEDs rapidly. Due to the fact that human eyes can only render about 10 images per second, the fast spinning LEDs seem like a solid display.

A television is a common example; in which image is re-scanned every 25 times, thereby appear continuous. Further, a glowing objects if rotated in a circle at fast speed, it shows a continuous circle. By modifying this basic idea, 8 LEDs can be rotated in a circle, showing 8 concentric circles. But if these LEDs are switched at precise intervals, a steady display pattern can be shown.

1.2 Background Review and Design Considerations of Propeller LED Display

There are many “Propeller LED display” in existence, both commercial and non-commercial. Figures below show a number of examples. These Propeller LED displays differ vastly in size, shape and capabilities, but all share the ability to physically move a number of LEDs rapidly in physical space.

The configuration of LEDs, together with the axis of rotation, dictates the shape that the display will be, as shown in Figures 1.4 to 1.6.

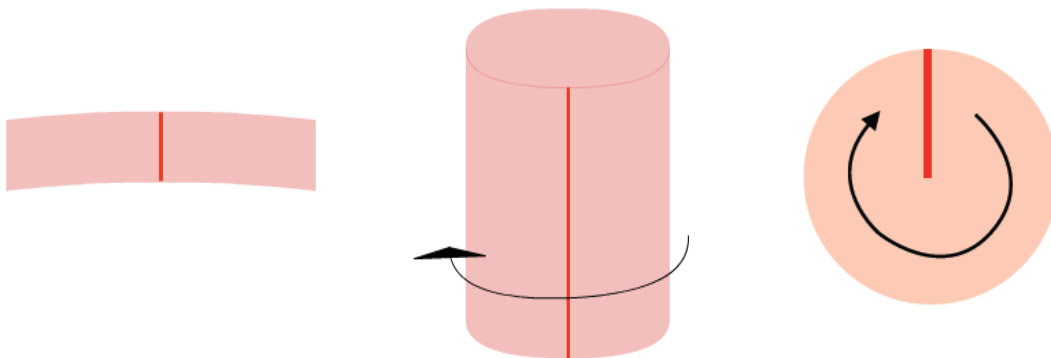


Figure 1.4 - 2D oscillating display

3D Propeller display

Figure 1.6 - 2D propeller display

The Figures show just three examples of possible Propeller LED display shapes. In Figure 1.4, a small number of LEDs are oscillated from side to side in order to produce a two-dimensional arc display. This would commonly be used for displaying text, such as a digital clock.

Figure 1.5 is an example of a three-dimensional Propeller LED display: a one-dimensional array of LEDs are still used but these are rotated about an axis parallel to, rather than perpendicular to, the LEDs in order to produce a cylindrical display (alternatively a sphere could be produced by curving the line of LEDs). A 3 dimensional Propeller LED display is often constructed using a 2D grid of LEDs which is swept or rotated through a volume. Propeller LED display devices can be used in combination with long camera exposures to produce light.

In Figure 1.6, rotation is used rather than oscillation in order to produce the perception of a circular display. This could be used to display text, images, video or any other visual media. This last type of rotation is the one our project is based on and we will display different texts and one video example displaying a Clock.

1.3 Two dimensional Propeller LED display

A two dimensional propeller LED display is accomplished by means of rapidly moving a single row of LEDs along a linear or circular path. The effect is that the image is perceived as a whole by the viewer as long as the entire path is completed during the visual persistence time of the human eye as shown in the figure 1.7. A further effect is often to give the illusion of the image floating in mid-air. [5]

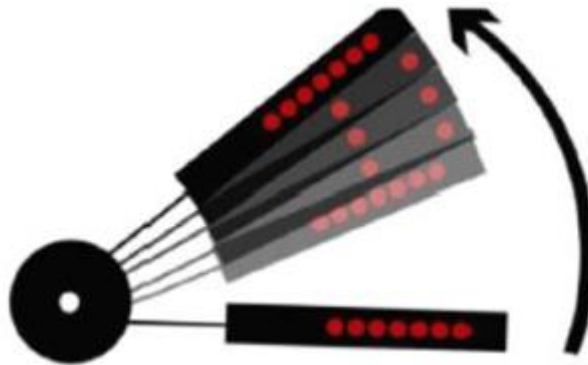


Figure 1.7 - 2D Propeller display

1.3.1 Literature review

Bob Blick invented the first Propeller LED display in 1996 in a form of a propeller clock; he made the clock spinning on a piece of perfboard. The power is provided from the spinning armature of a plain DC motor. In order to run the wires out of the motor, the bearing is removed from one end of the motor, leaving a big hole. There are three terminals inside most small DC motors, and it acts a lot like three-phase alternating current, so it must be rectified back to DC.

A nice side effect of this is that the position of the motor can be detected by taking one of the phases straight into the microprocessor.

Bob Blick used perfboard (Vectorboard) and handwired the circuit together. Use an 18-pin socket for the 16C84 is used because it needs to be programmed before putting it in the circuit. For the 7 current-limit resistors a DIP resistor array is used, because it made it easy to experiment with LED brightness. He settled on 120 ohms. Seven regular

resistors also can be used, because 120 ohms works fine, though it puts the peak current right at the limit for the 16C84. To keep the clock running after turning it off a 47000uf is used, so the time can be set. The LED'S gets power separate from this circuit. The result is shown in Figure 1.8.



Figure 1.8 Seven LEDs Spin.

1.3.2 Resolution and Colors

The resolution of a POV display is partially dictated by the number of LEDs physically present in the system. However, due to the persistence of vision phenomenon caused by physical movement of the LEDs, the perceived (effective) resolution in the direction(s) of motion is much greater than the actual number of the present LEDs.

For example, if a column of 50 LEDs arranged vertically were to be rapidly moved left and right in two dimensional space, and the output of these LEDs were changed 100 times between the left-most and right-most points of movement (*and* the complete sweep from one side to the other were completed before the after-image had disappeared from the retina), the perceived resolution of the display would be 100px by 50px rather than the 1px by 50px resolution that the system hardware would suggest. In other words, the resolution in any directions of physical movement is dictated by the number of times the LED output is modified by software within one twenty-fifth of a second.

1.3.3 Risk Assessment

Any system that incorporates moving parts has risks involved, and it is vital to ascertain what these risks could be in order to mitigate against them. The most obvious risk would be that any part mounted on the rotating body could become disconnected from the system during acceleration or rotation and come into contact with someone or something in the vicinity. To avoid this:

- Care will be taken to ensure that all parts are secured tightly to the system.
- Rotation hardware will be tested thoroughly without any load (such as LEDs), before the load is added and retested.
- The system should be designed so as to not rotate when initially powered on, to prevent accidental rotation. Software and hardware should both be in place to ensure that some kind of intentional input is required before the system is permitted to rotate at all.
- The system should accelerate to full speed gradually: it should take several seconds to reach full speed. Additionally, during initial testing a user input should be required at several points during acceleration before the system is permitted to accelerate further.
- In addition to all of these preventative measures, a transparent but strong protective screen should be in place around the system in every direction during testing as a final layer of protection in the event that any part does become detached from the system.

1.4 Conclusion

The Propeller LED display could be used in the advertising industry, to display company logos, details or animations in a novel way. Alternatively, it could be used as an entertainment product in much the same way as large television displays could be used at various entertainment events. Again, the novelty of the device would be a key factor here, as well as the potentially reduced cost of the lower number of LEDs required (although this would likely be offset by the additional hardware required in producing the physical movement).

Chapter 2

Propeller LED display Hardware Design

A propeller LED display has many advantages over a traditional LCD or LED display, such as power reserves, simplicity, easy configuration, prettiness etc.... To reduce the negative aspect of the old system, we have decided to implement the same display using advanced microprocessor, the Arduino Nano. Arduino Nano board provides us with a low-cost, easy-to use technology to create our project. Hence, this chapter deals with the hardware design of our propeller LED platform.

2.1. System overview

The mechanical part is considered to be a crucial part of the project. Specially, when designing the propeller, several issues arise. The following criteria must be respected when designing our system [6]:

- The size: the smaller a propeller is the better because bigger constructions might have worse aerodynamics which may create more power consumption.
- The weight: if the propeller is too heavy, it may be hard to be balanced while it is moving which may cause major problems (e.g. oscillation).
- LEDs arrangement: it is important that the LEDs are not too far from each other and are in the same row.
- Heavy parts arrangement – the propeller may (but should not) consist of some heavy parts; it is important to make sure that they are located in the right place taking into account the center of the mass – in case of our project, the center of the mass was located right in the middle of the propeller.

2.2. Propeller LED display Platform Description

In this project, there are two main circuits: the first one is the propeller circuit which supposed to rotate and display the desired text or image, it consists of the Arduino microcontroller and LED module, each LED is connected to a 330ohm resistance on one side and to the corresponding Arduino pin on the other side. The circuit is supplied by a 9V Battery.

The second circuit is composed of a Brushless DC motor connected to a LIPO Battery by an electronic speed controller; this part is the responsible of rotating the propeller system. Figure 2.1 summarizes the overall system diagram.

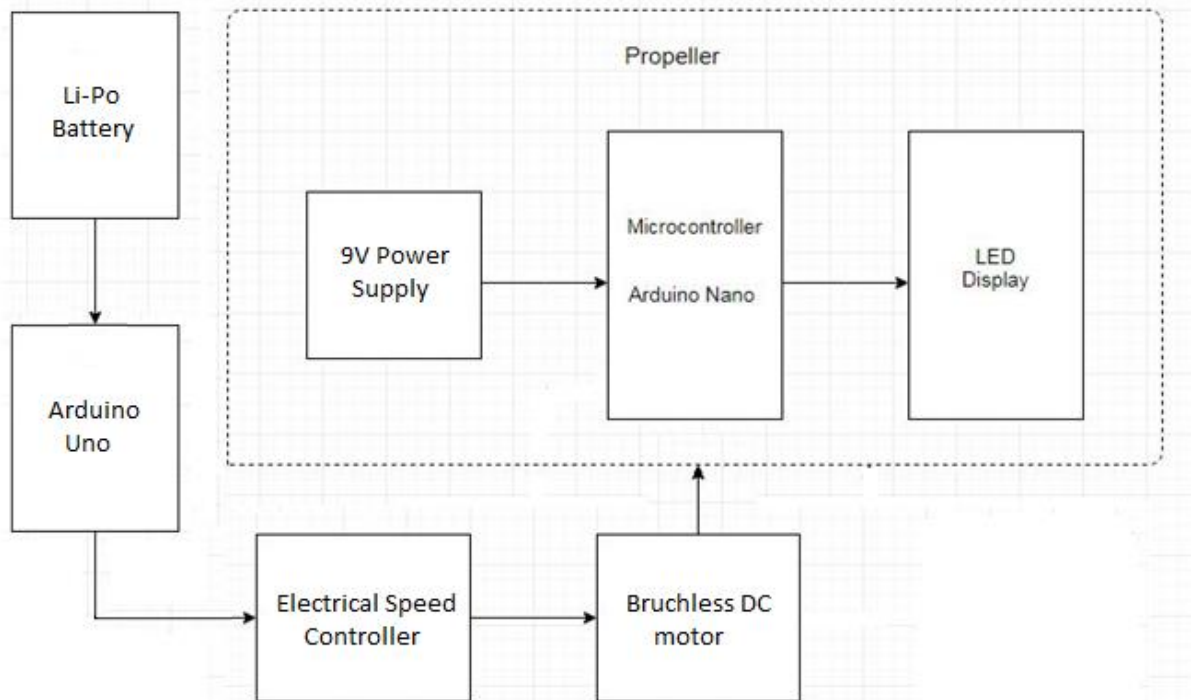


Figure 2.1 Block Diagram

2.3. Hardware Components

Hardware is an important part of any system. If the hardware does not meet the requirements of the system, the system will not work as intended. It is therefore vital for any system to know what is required and why. It is also important to understand what the hardware can and cannot do. In this section each of the main components of the hardware platform are presented. Each subsection features an aspect of the hardware platform and the components used within it.

2.3.1. Arduino

Arduino is an open-source hardware and software. It consists of a circuit-board, microcontroller, and software called Arduino IDE (Integrated Development Environment) which is used to write the code and upload it to the physical board. The code is a dialect of features from the programming languages C and C++.

Arduino boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (*shields*) and other circuits.

The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. These boards are available commercially in preassembled form; in this project we are using two Arduino boards: ArduinoNano for running the propeller part and Arduino UNO for running the motor (we only need one pin for that).

a) Arduino Nano

Arduino Nano is a surface mount breadboard embedded version with integrated USB. It is a smallest, complete, and breadboard friendly. It has everything that Diecimila/Duemilanove has (electrically) with more analog input pins and onboard +5V AREF jumper. Physically, it is missing power jack. The Nano is automatically sense and switch to the higher potential source of power, there is no need for the power select jumper [7]. The Arduino Nano is shown in figure 2.2 and its pin layout is given in table 2.1. Whereas, the Arduino Nano specifications are given in table 2.2.

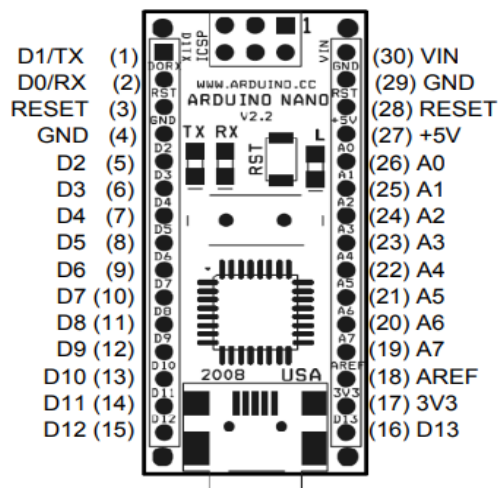


Figure 2.2 Arduino Nano

Table 2.1. Arduino Nano pin layout

Pin No	Name	Type	Description
1-2, 5-16	D0-D13	I/O	Digital input/output port 0 to 13
3, 28	RESET	Input	Reset (active low)
4, 29	GND	PWR	Supply ground
17	3V3	Output	+3.3V output (from FTDI)
18	AREF	Input	ADC reference
19-26	A7-A0	Input	Analog input channel 0 to 7
27	+5V	Output or Input	+5V output (from on-board regulator) or +5V (input from external power supply)
30	Vin	PWR	Supply voltage

Table 2.2. Arduino Nano specifications

Arduino Nano	Specifications
Microcontroller	ATmega328P
Architecture	AVR
Operating Voltage	5 Volts
Flash Memory	32 KB of which 2 KB used by Bootloader
SRAM	2 KB
Clock Speed	16 MHz
Analog I/O Pins	8
EEPROM	1 KB
DC Current per I/O pins	40 milliAmps
Input Voltage	(7-12) Volts

b) Arduino UNO

The Arduino Uno, shown in figure 2.3, is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, 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 [8].

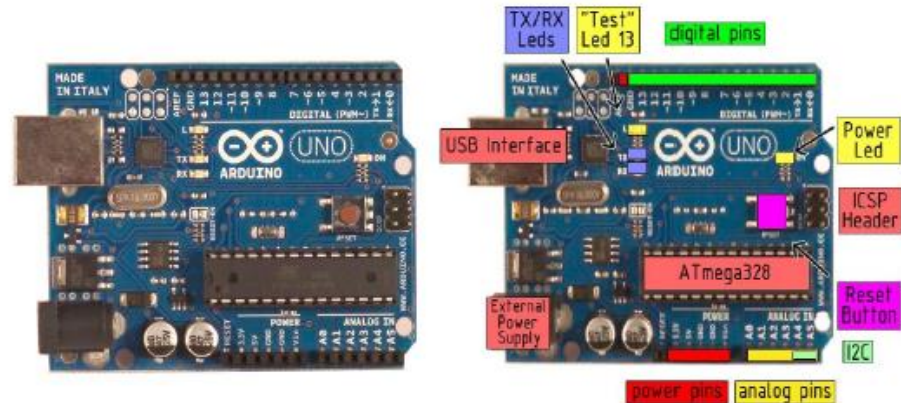


Figure 2.3.Arduino UNO

2.3.2. Light Emitting Diode (LED)

A light-emitting diode is a two-lead semiconductor light source. It is a p-n junction diode, which emits light when activated. When a suitable voltage is applied to the leads, electrons are able to recombine with electron holes within the device, releasing energy in the form of photons. This effect is called electroluminescence, and the color of the light (corresponding to the energy of the photon) is determined by the energy band gap of the semiconductor. LEDs are typically small (less than 1mm²) and integrated optical components can be used to shape the radiation pattern.

A P-N junction can convert absorbed light energy into a proportional electric current. The same process is reversed here (i.e. the P-N junction emits light when electrical energy is applied to it). This phenomenon is generally called electroluminescence, which can be defined as the emission of light from a semiconductor under the influence of an electric field.

The charge carriers recombine in a forward-biased P-N junction as the electrons cross from the N region and recombine with the holes existing in the P-region (see figure 2.4). Free electrons are in the conduction band of energy levels, while holes are in the valence energy band.

Thus the energy level of the holes will be lesser than the energy levels of electrons. Some portion of the energy must be dissipated in order to recombine the electrons and the holes. This energy is emitted in the form of heat and light [9]

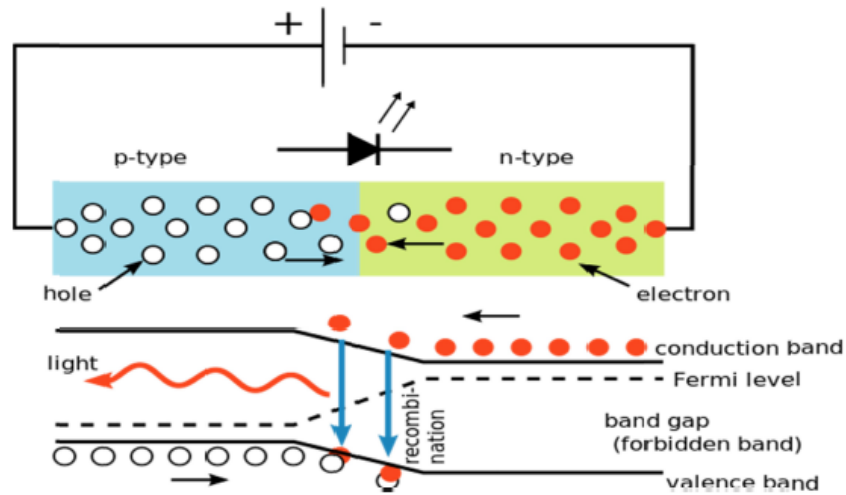


Figure 2.4. Function of LED

2.3.3. 2212 Brushless DC Motor

The 2212 (BLDC) Brushless DC electric motor also known as electronically commutated motor and synchronous DC motor, is synchronous motor powered by DC electricity via an inverter or switching power supply or in our project an electronic speed controller (ESC) which produces an AC electric current to drive each phase of the motor via a closed loop controller, the controller provides pulses of current to the motor windings that control the speed/ torque of the motor (see figure 2.5).



Figure 2.5. 2212 Brushless DC motor

A 2212 Brushless DC motor Technical data are summarized in table 2.3 below.

Table 2.3. Technical Data of 2212 BLDC

No. of Cells:	2 - 3 Li-Poly
Kv:	1000 RPM/V
Max Efficiency:	80%
Max Efficiency Current:	4 - 10A (>75%)
No Load Current:	0.5A @10V
Resistance:	0.090 ohms
Max Current:	13A for 60S
Max Watts:	150W
Weight:	52.7 g / 1.86 oz
Size:	28 mm dia x 28 mm bell length
Shaft Diameter:	3.2 mm
Poles:	14
Model Weight:	300 - 800g / 10.5 - 28.2 oz

2.3.4. Electronic Speed Control (ESC)

An electronic speed control or ESC, shown in figure 2.6, is an electronic circuit that controls and regulates the speed of an electric motor; it may also provide reversing of the motor and dynamic braking. Miniature electronic speed controls are used in electrically powered radio controlled models. ESC's have sturdy construction with 2 separate PCBs for Controller and ESC power MOSFETs. It can be powered with 2-4 lithium Polymer batteries or 5-12 NiMH / NiCd batteries. It has separate voltage regulator for the microcontroller for providing good anti-jamming capability.

An electronic speed control follows a speed reference signal (derived from a throttle lever, joystick, or other manual input) and varies the switching rate of a network of field effect transistors (FETs) [10]. By adjusting the duty cycle or switching frequency of the transistors, the speed of the motor is changed. The rapid switching of the transistors is what causes the motor itself to emit its characteristics high-pitched whine, especially noticeable at lower speeds.

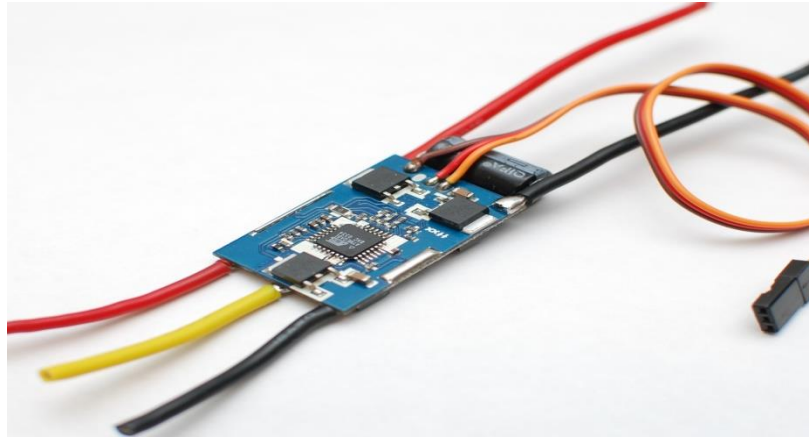


Figure 2.6. Electronic Speed Controller

2.4. Schematic description

As mentioned before, our circuit consists of two main parts: the propeller circuit and the motor circuit. Each part is built independently from the other, but at the end we place the propeller circuit on the BLDC motor in order to have a propeller display. The following figures show the circuits built using Proteus 8.0 software.

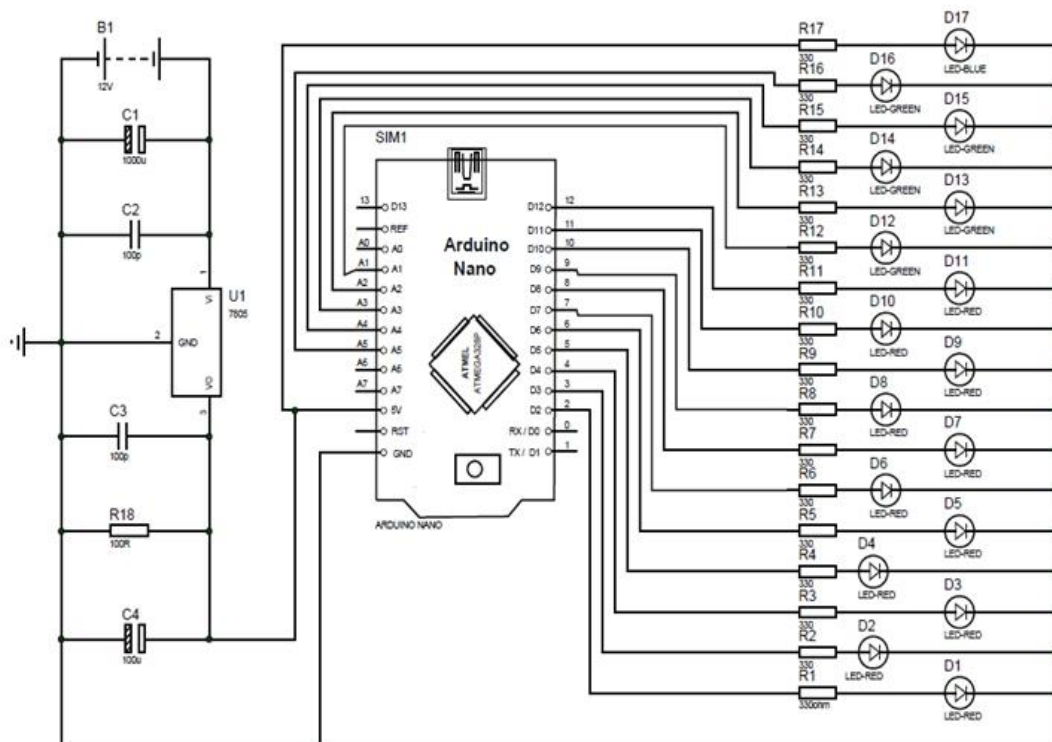


Figure 2.7. Propeller Circuit diagram built with Proteus 8.0

The circuit diagram shown in figure 2.7 represent the propeller part which consists of 3 main components: LED module and Arduini Nano microcontroller and a 9V power supply. The connection details are already shown in the diagram.

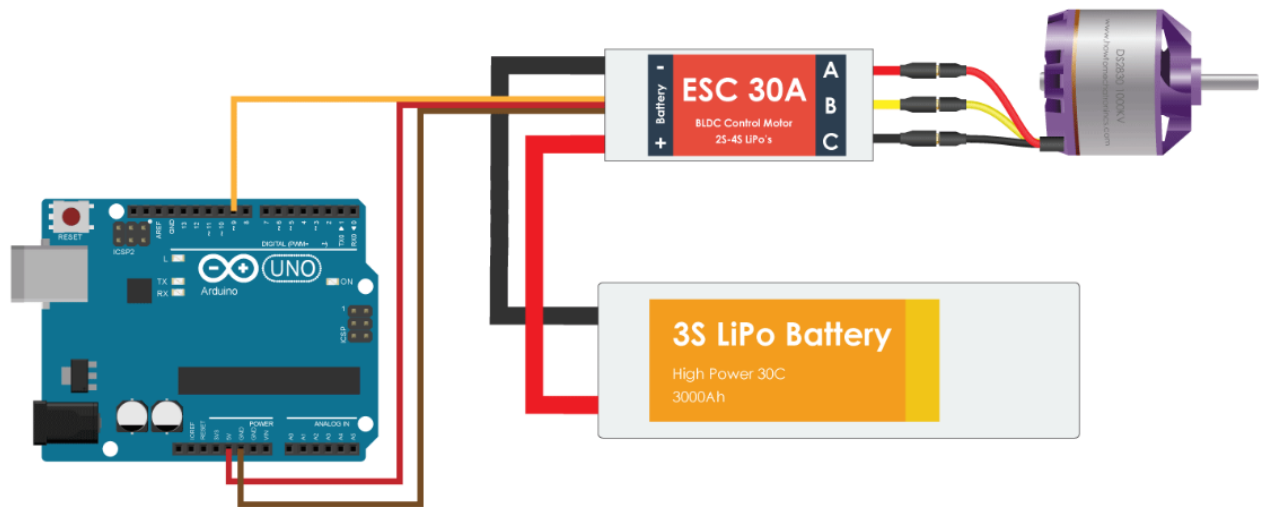


Figure 2.8.Arduino BLDC control – Circuit diagram

Because of the non-existence of the ESC in the Proteus software, we built our circuit as show in figure 2.8, the circuit consists of the BLDC motor controlled by the Arduino UNO through the ESC and alimented by the Li-Po Battery

2.5. Conclusion

The system we have built in this project is composed of two parts consisting of the electronic components stated above (LEDs, Arduino, BLDC motor, etc.), these two parts must work in a synchronized way to achieve the task assigned, means that the motor must rotate with a specific speed and the LEDs must blink in a specific time for this an appropriate and well-defined strategy is developed which will be explained in the next two chapters.

Chapter 3

Software design

Any modern system is built of two main parts the software and the hardware. The software is the set of instructions, data or programs instructing a computer to do specific tasks opposite of hardware which describes the physical component of a computer, software is a generic term used to refer to applications, scripts and programs that runs on a device. This chapter describes the deferent algorithms and programs developed to run our system.

3.1. Arduino software

The most common description of Software is to be the variable part of a computer while the hardware is the unchanging part; where, the user cooperates with software application on a typical computer. The application software layer interfaces with the operating system, which in turn communicates with the hardware. As it is shown in figure 3.1 bellow:

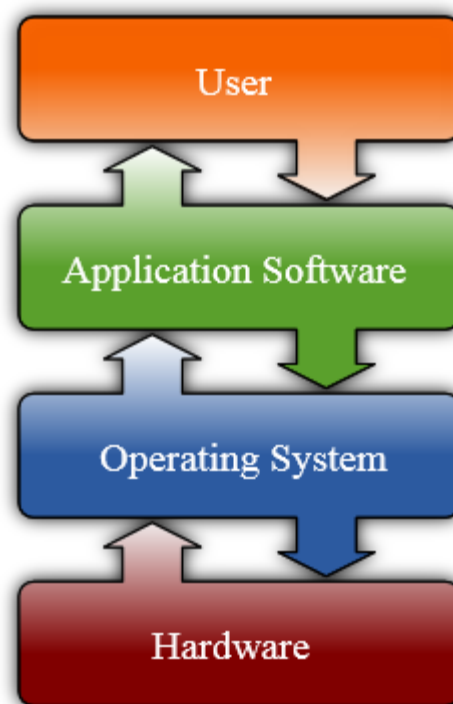


Figure 3.1.User interaction with computer

So, after studying the project and implementing the hardware all our attention turned to building the software, in our case we use Arduino software to write and compile it.

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It originated from the IDE for the languages *Processing* and *Wiring*.

It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple *one-click* mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console; a toolbar with buttons for common functions and a hierarchy of operation menus [11].

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU tool chain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

3.1.1. The Sketch

A *sketch* is a program written with the Arduino IDE. [12] Sketches are saved on the development computer as text files with the file extension **.ino**. A minimal Arduino C/C++ program consists of only two functions

- **Setup ()**: This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- **Loop ()**: After **setup ()** function exits (ends), the **loop ()** function is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

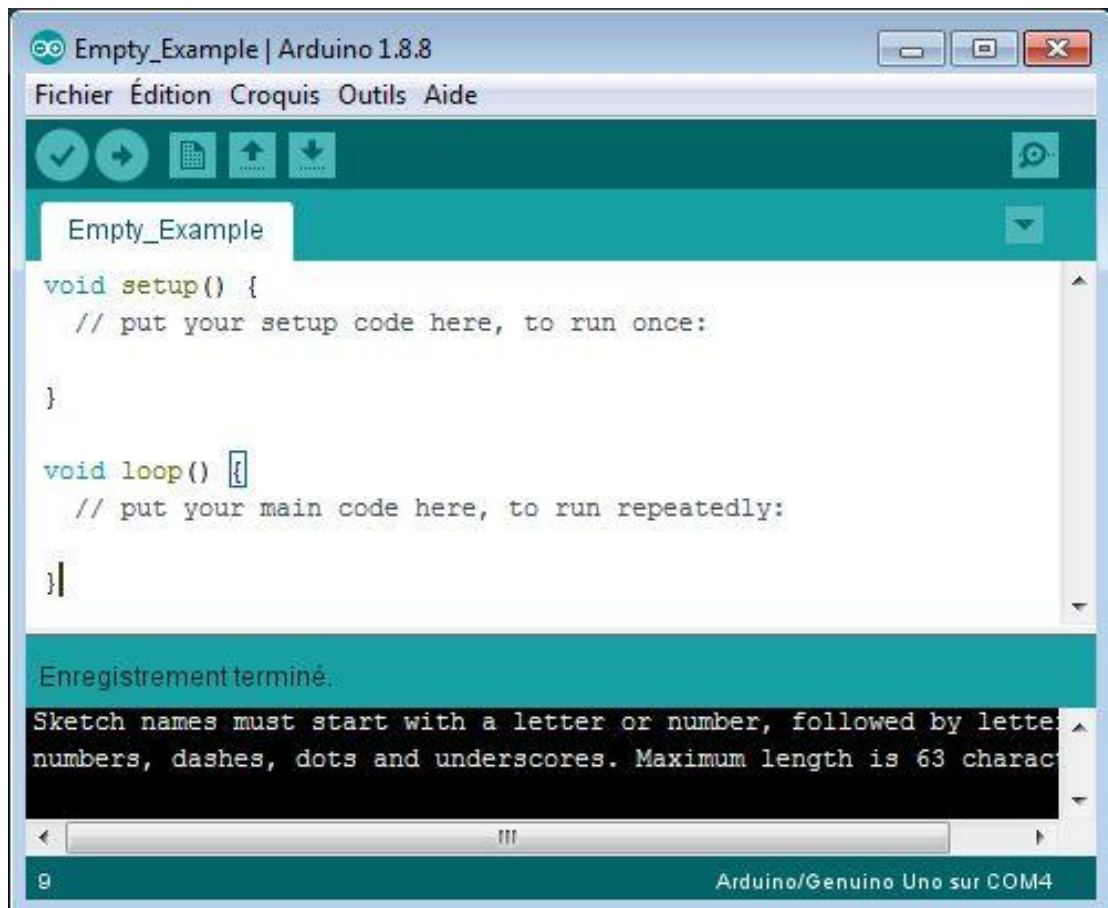


Figure 3.2. Example of a sketch form

3.1.2. The uploading process

A number of procedures have to happen for the Arduino code to get onto the Arduino board. First, the Arduino environment performs some minor pre-processing to turn the code into a C++ program. It then gets passed to a compiler (avr-gcc), which turns the human readable code into machine readable instructions (or object files). Then the code gets combined with (linked against), the standard Arduino libraries that provide basic functions like `digitalWrite()` or `Serial.print()`. The result is a single Intel hex file, which contains the specific bytes that need to be written to the program memory of the chip on the Arduino board. This file is then uploaded to the board: transmitted over the USB or serial connection via the bootloader already on the chip or with external programming hardware.

3.2. Character's transformation into codes

To display a message, it is more useful to transform each character into a code. That code is taken from the projection of a character on a two dimensional table containing five rows with eight columns which is the number of LEDs used to display the message, figure 3.3 illustrates the process.

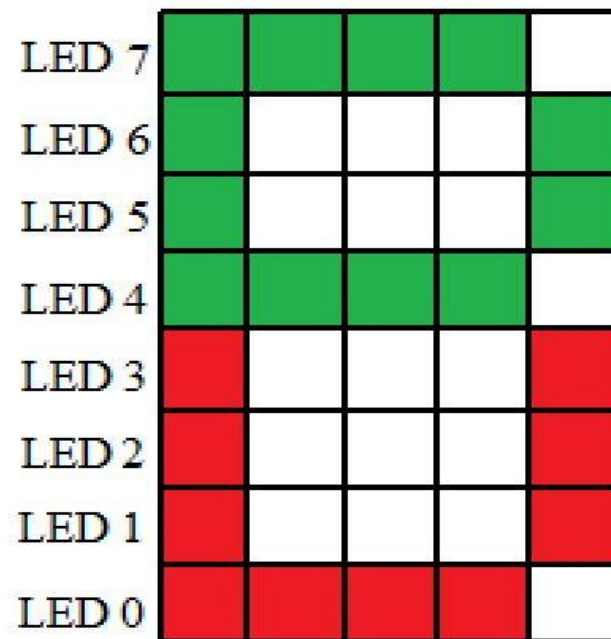


Figure 3.3. Projection of the character 'B'.

Starting from top to bottom and from left to right of the table shown in the figure 3.3, the colored squares considered as binary '1' while the uncolored ones as binary '0'. Instead of using two dimensional array to store the code, the use of one dimensional array is less complex and efficient for programming the Arduino board.

For this example, the array containing the character 'B' will be as follows:

```
intB[] = {1, 1, 1, 1, 1, 1, 1, 1,
          1, 0, 0, 1, 0, 0, 0, 1,
          1, 0, 0, 1, 0, 0, 0, 1,
          1, 0, 0, 1, 0, 0, 0, 1,
          0, 1, 1, 0, 1, 1, 1, 0};
```

3.3. Software algorithm

An algorithm is a fancy to-do list for a computer. Algorithms take in zero or more inputs and give back one or more outputs. A recipe informs us what we need to do step by step; so, it can be considered as a good illustration of an algorithm. It takes inputs (ingredients) and produces an output (the completed dish).

To write a computer program, we have to instruct the computer, step by step, exactly what we want it to do. When we are telling the computer what to do, we also get to choose how it's going to do it. That's where computer algorithms come in. The algorithm is the basic technique used to get the job done.

3.3.1. Message displaying algorithm

The design of the circuit is based on two microcontrollers (Arduino Nano and Arduino Uno) one to drive the LEDs and the other to control the speed of the brushless motor, each Arduino kit is independent from the other, no connection needed between the two because of the rotating part which is containing the LEDs, so each microcontroller is loaded with its appropriate sketch. The algorithm steps for each part are explained below.

- **Step 1: declaring variables & constants**

In this section of code, we declare the variables (space, dot) and we assign each character with an equivalent code as 1D array

- **Step 2 :the setups & the displaying functions :**

We divide this section into two main parts:

- **Part 1:**

- Setup the Arduino input & output ports
- Define the space: which is the space time between letters in (ms)
- Define the dot: which is the waiting time of displaying the next row of a single character.

- **Part 2:** The printing function:

This function accept one dimension array of integers as an argument, since every character is coded as 1D array, we pass to this

function one character at a time, so we print the first row of character then wait for some time that

We defined before as dot time (ms) to avoid collision; then we print the second row and we wait for another dot time until the end of a character's rows. After that, we leave a space time that we defined before as space.

- **Step 3 : printing a message repeatedly**

In this part not much work left to do all we have to do is calling the printing function by the characters of the message that we want to display.

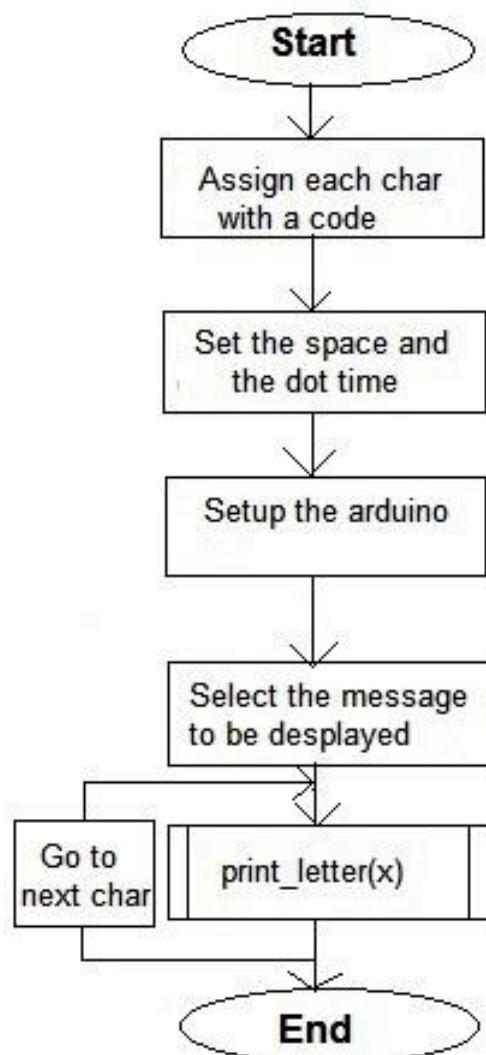


Figure 3.4. Flowchart of the main program

Once the printing function called from the loop function of the sketch its execution goes as follow:

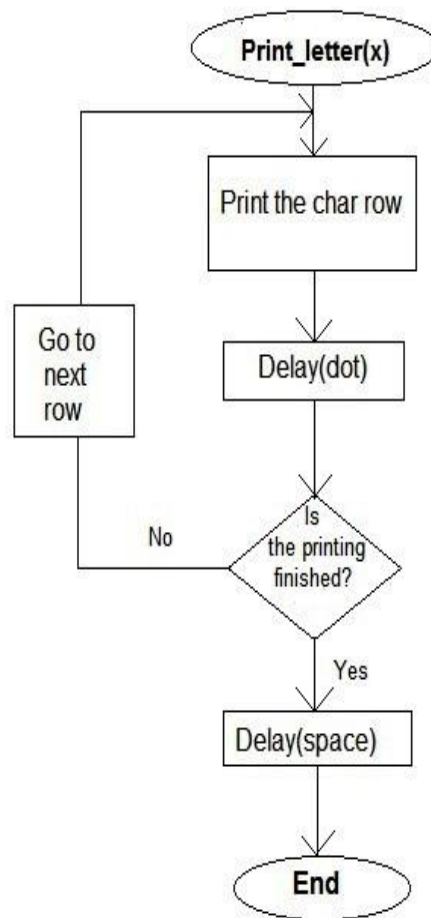



Figure 3.5. Flowchart of the printing function

3.3.2. Message displaying code

After gathering all the data needed to build our software, the figure 3.6 describes briefly the code for displaying any message we want.



```

PseudoCode $

//we write for each character its corresponding array

int _[] = {0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0};
int A[] = {1,1,1,1,1,1,1,1, 1,0,0,1,0,0,0,0, 1,0,0,1,0,0,0,0, 1,0,0,1,0,0,0,0, 1,1,1,1,1,1,1,1};
int B[] = {1,1,1,1,1,1,1,1, 1,0,0,1,0,0,0,1, 1,0,0,1,0,0,0,1, 1,0,0,1,0,0,0,1, 0,1,1,0,1,1,1,0};
.....
.....

int Z[] = {1,0,0,0,0,1,1,1, 1,0,0,0,1,0,0,1, 1,0,0,1,0,0,0,1, 1,0,1,0,0,0,0,1, 1,1,0,0,0,0,0,1};
int Space;
int dot;

void setup()
{ Serial.begin(9600);
// setting the ports of the leds to OUTPUT

pinMode(i0, OUTPUT);
pinMode(i1, OUTPUT);
.....
.....

pinMode(in, OUTPUT);

// defining the space between the letters (ms)
letterSpace = ...;
// defining the time dots appear (ms)
dotTime = ...;
}

void printLetter(int letter[])
{
.....
}

void loop()//write here (in this example we want to display the message 'Hello')
{
printLetter (H);
printLetter (E);
printLetter (L);
printLetter (L);
printLetter (O);
}

```

Figure 3.6. Message displaying code

3.4. Algorithm steps for controlling the brushless motor's speed

In order to control the motor's speed we must send the appropriate signal from the Arduino Uno to the electronic speed control (ESC), the process is performed as follow:

- **Step 1: The declaration**

Include the servo library to control the ESC and creating an instance of the servo class named esc.

- **Step 2: The setups**

- Specify the esc signal pin.
- Initialize the signal.

- **Step 3: The infinite loop**

The motor must accelerate slowly until the desired speed is reached, using the variable “val” as the signal to esc, if the variable “val” generates a signal that led to a speed less than the desired we increment “val” with the scale of 10 ’ until the motor is stabilized at desired speed.

The flowchart below describes the previous steps:

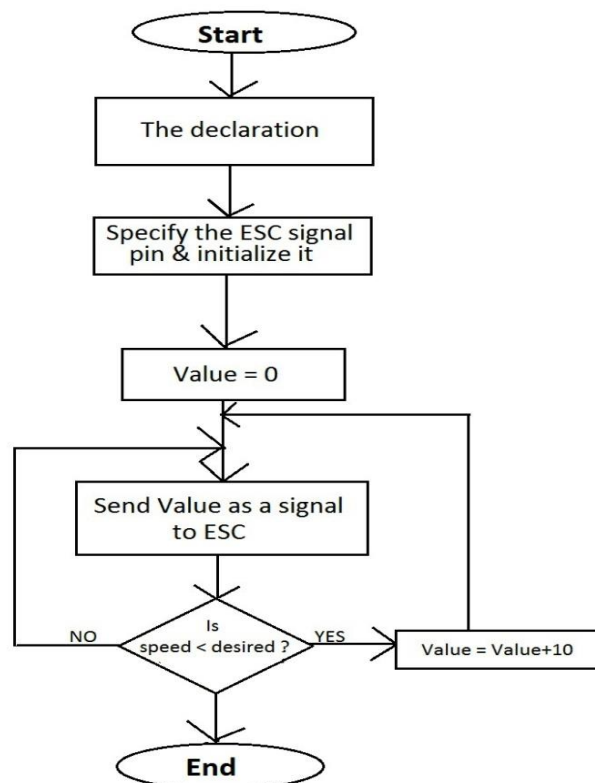


Figure 3.7. Flowchart of controlling brushless motor's speed

3.5. Conclusion

In this chapter we have discussed all the steps needed to write a complete program for propeller LED display using Arduino IDE software where we configured the LEDs to blink in a specific pattern to create the message or the image we want to display on one hand. On the other hand we also used the Arduino IDE through the Electronic Speed Control (ESC) to drive the brushless DC motor at any desired speed. Now, all that left is to synchronize between the two above to get the best desired display.

Chapter 4

Functionality testing and results

The focus of this project is to use only one row of LEDs connected to a microcontroller on a small board which will be rotated by a motor, to realize a two-dimensional screen in order to display different images. This chapter describes the final hardware design and the testing that was performed to check the functionality of each component; and then the different applications done to ensure the success of our project.

4.1. Final hardware design

The final hardware design of the project consists, as mentioned before, of two main systems: LEDs' circuit and motor's block, each one of them must be composed very carefully and very precisely in order to reach the project's goals. It is therefore convenient to divide the system into two subsections summarized as follow:

4.1.1. Propeller circuit

The propeller circuit is the rotating part of the built system. It consists of 17 LED (11 red, 5 green and 1 blue) driven by an Arduino Nano microcontroller, and a 9V power supply; all this components are connected on a 15x3 cm² PCB board as mentioned in figure 4.1. This board must be very balanced to avoid oscillations and loss of speed.

To have a clear image, the distance between LEDs must be as small as possible and close to the center of rotation.

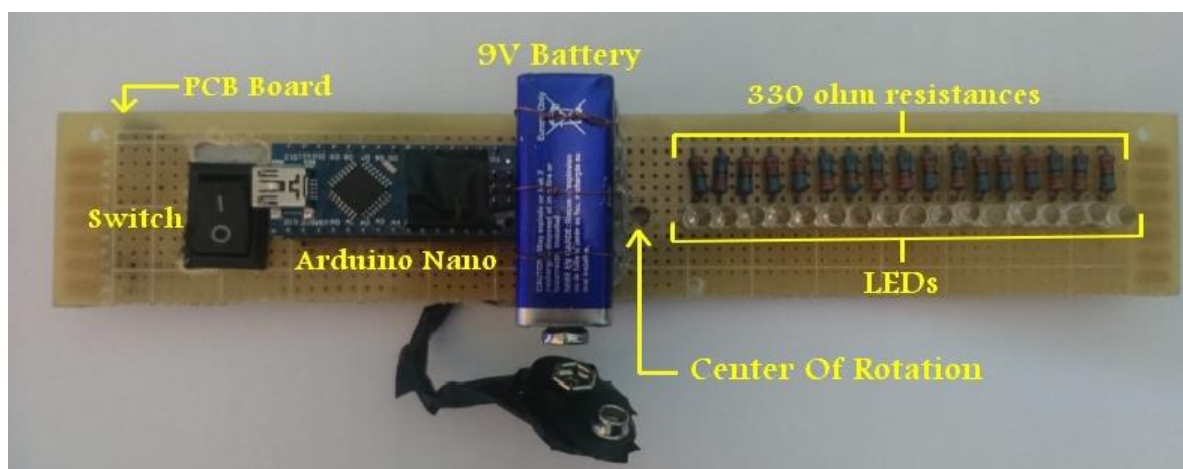


Figure 4.1. Hardware design of Propeller circuit

4.1.2. BLDC motor circuit

In order to reach different speeds levels, a brushless motor is used, controlled by an Arduino UNO through an electronic speed controller (ESC), powered up by a Li-Po battery. The figure 4.2 below illustrates the arrangement of the components.

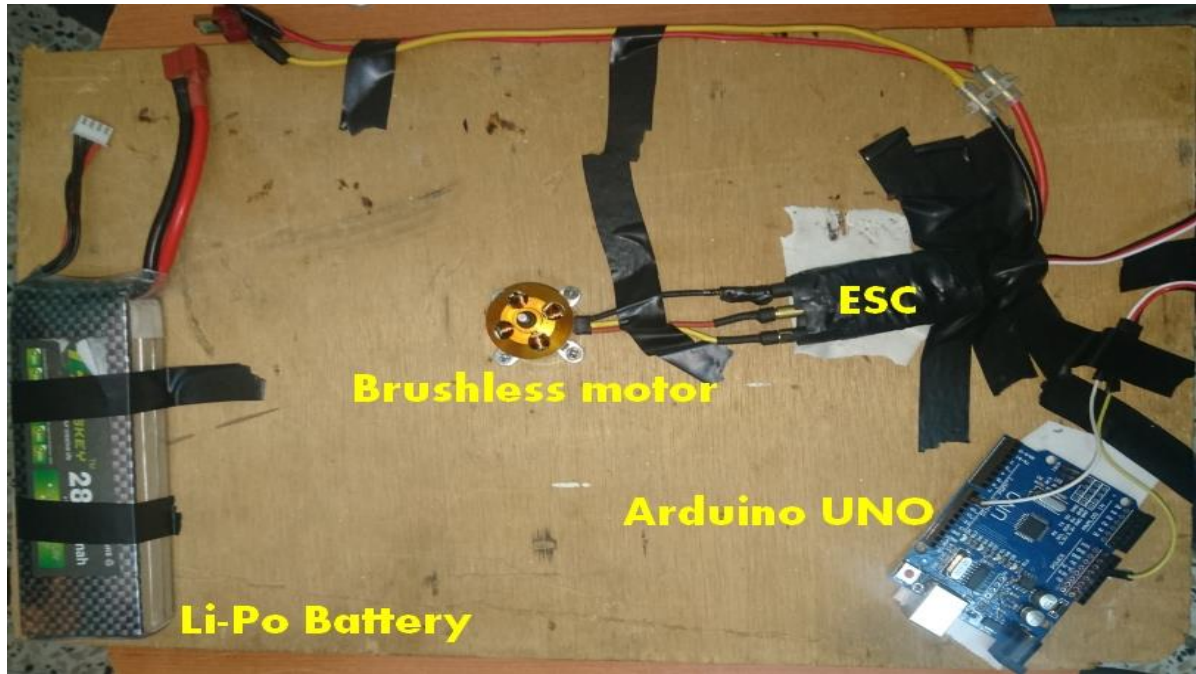


Figure 4.2. Hardware design of BLDC motor circuit

4.2. Tests and troubleshooting

To avoid any damage that can happen to the project components or any danger that can face the user due to the high speed rotation, some tests must be done to all the parts of our system as follows:

4.2.1. Testing the Arduino Board and Connections

When first time we switch on the board, we will see that an LED labeled L connected to the pin no. 13 of the board will light up to indicate that the board is working. The led in the LED column connected to this pin 13 will also glow. Thus we will get only one LED to glow, the one connected to pin 13. The board can then be connected to a pc and the desired program can be loaded into it to obtain desired glowing LED effect. After the desired program is loaded, disconnect the USB from the board.

4.2.2. Testing the LED Module

Before we run the display on full scale, we need to check that all the LEDs are connected to the Arduino board and that each one responds. The most common problem that occurs in connection of LEDs is when we connect the LEDs to wrong pins in the controller. This may cause a totally different pattern to be displayed and may give an impression that the display is out of sync. Care must be taken to connect the LEDs in a sequential manner to the pins on the controller. Miss sequenced pins may cause the display to appear inverted or in the form of a mirror image of what is originally to be displayed.

4.2.3. Testing the Motor Speed and Synchronization

Now we are ready to rotate the display. But before we start the motor, make sure that no wires are tangles to any of the parts that will rotate when we turn on the motor. This can cause extensive hardware damage. Now we connect the Li-Po battery to the ESC terminals, this last one is already connected to the Arduino UNO.

After that we upload the code that controls the motor's speed to the Arduino UNO and initially we keep the signal at lowest value and check that no part is hanging out that may cause the assembly to dug or to get tangled during operation. Finally when everything is checked out, slowly start increasing the speed by sending the appropriate signal to the ESC until a satisfying resolution is obtained. If still the display doesn't work properly, try to remove excess weight from the mounting assembly which may cause the motor to operate at low RPMs.

4.3. Applications and results

The main goal of our project is to display on a rotating board a desired pattern that can be a string of characters, numbers, irregular shapes or even a clock with use of only 17 LEDs or less, instead of more than 500 LEDs like in LCD panels or even in Televisions.

So after testing all the components of our projects, we can now start applying some applications and try to make the display as well as possible.

4.3.1. Displaying characters' string

The first thing that will be displayed in our project is a string of characters or a message that consists of only alphabet letters and no string is better than the name of the institut we study in which is 'IGEE', so if we want to use a row of 8 LEDs (4 green and 4 red) we need to project each character on a two-dimensional table in order to get the array corresponding to each letter as shown in figure 4.3 below.

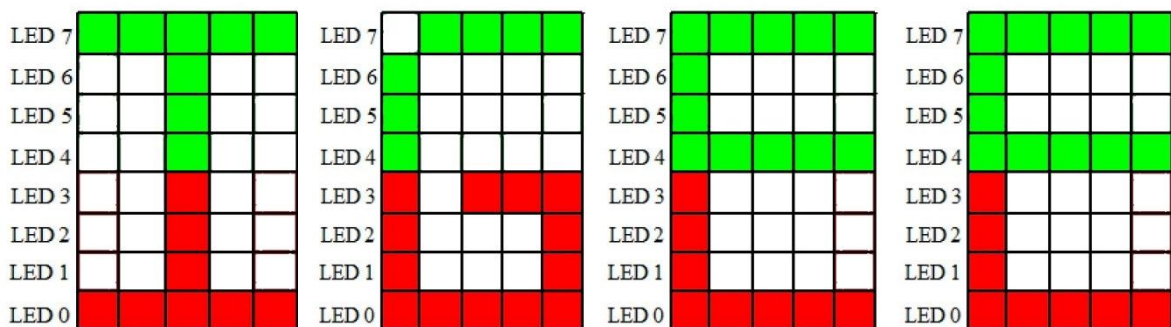


Figure 4.3. Characters transformation to arrays

So for example the array corresponding to the letter 'G' is:

```
int G[] = {0,1,1,1,1,1,1,1, 1,0,0,0,0,0,0,1, 1,0,0,0,1,0,0,1, 1,0,0,0,1,0,0,1, 1,0,0,0,1,1,1,0};
```

After getting an array for each character, we write our program and upload it to the Arduino Nano and then we use the Arduino UNO to set the BLDC motor's speed to a value that allow us to read clearly the message displayed (the speed is around **1430 RPM**). The result is shown in the following pictures (figure 4.4).



Figure 4.4. Propeller display of the string 'IGEE'

4.3.2. Displaying numbers

In order to display numbers, we are applying the same procedure done to characters with use of the same number of LEDs. The figure 4.5 below describes how we got the array corresponding to each number.

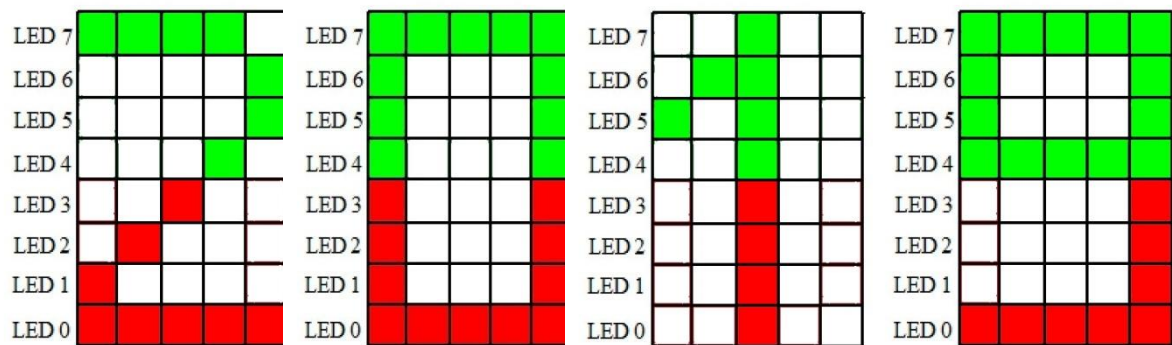


Figure 4.4. Numbers transformation to arrays

So for example the array corresponding to the number '9' is:

```
Int NUMBER9[]={1,1,1,1,0,0,0,1, 1,0,0,1,0,0,0,1, 1,0,0,1,0,0,0,1, 1,0,0,1,0,0,0,1,
1,1,1,1,1,1,1,1};
```


After getting the data needed to display a number which is '2019', we do the same procedure done before to upload the program to the Arduino Nano and set the BLDC motor's speed which is the same as the speed needed to display characters. The results are shown in figure 4.5.



Figure 4.5. Propeller display of '2019'.

4.3.3. Displaying an Analog clock

The system built in this project is not limited on displaying characters only, many applications can be performed also, displaying an analog clock is one of them. In order to do so, the Arduino Nano is uploaded with a sketch to be able to drive the LEDs (11 red, 5 green, 1 blue) to create the clock, also the motor speed must be synchronized in a manner that allows the clock hands to indicate the real time.

Building the analog clock software

To display the analog clock The LEDs can be programmed as follow:

- The blue LED (LED17): always on to draw the clock's circle.
- The green LEDs (LED16...LED12) : are used to display the clock scales (hours and minutes and quarter scale), an integer variable is used to keep track of the clock position called 'p', the value of p keep increasing by '1' inside the

infinite loop, if it exceed '60' it goes back to '0', each time p increased the LED16 is on to create the minutes scale.

if $p=\{0,5,10,15,20,25,30,35,40,45,50,55\}$, the LEDs (16,15,14) are on to display hours scale, if $p=\{0,15,30,45\}$ the LEDs(16,15,14,13,12) are on to create the quarter scale.

- The red LEDs (11.....1): are used to display the clock hands, after each second elapsed 's' is incremented by '1', if $s=60$ then $s=0$ and $m=m+1$, if $m=60$ then $m=0$ and $h=h+1$ (s , m and h are variable for second minute and hour respectively), if $p = h*5$ then the LEDs (7.....1) are on to draw the hours hand, if $p=m$ then the LEDs (9.....1) are on to draw the minutes hand also if $p=s$ then the LEDs (11.....1) are on to draw the seconds hand , to draw the inner circle on the clock LED1 is always on .

The figures 4.6 and figure 4.7 down show the obtained results.



Figure 4.6 – Propeller display of a clock that point to 12:16:12



Figure 4.7. Propeller display of a clock that point to 12:16:19

Unfortunately, the clock is not stable; the motor's speed must accurately be synchronized with delays used in the software.

4.4. Discussion

4.4.1. Increase/ decrease the delay

The number of characters displayed in the system is related with the delay used in the software. A large delay allows a few characters to be displayed, while using a small delay lead to display more characters, the figure 4.8 below shows the result obtained.



Figure 4.8. Displaying 2019 with a small delay

4.4.2. Flipping the motor's phases

It is known that flipping the motor's phases make the motor rotate to the opposite direction. In our system, the rotation to opposite direction make the displayed image reversed like a mirror image, the word "INELEC" was ran on the system on both ways, the figure 4.9 below illustrates this process.



'INELEC' after flipping the phases. (a) 'INELEC' before flipping the phases. (b)

Figure 4.9. Flipping the motor's phases.

4.4.3. Rotation's speed

To apply the persistence of vision, the eye must see more than 25 images per second, so the minimum speed that the motor must rotate with is around 1400 RPM.

In our project we stabilized the motor's speed at minimum that can make the propeller part rotate (considering the weight of the component), which is certainly much higher than 1400 RPM.

4.5. Conclusion

In our project, we have presented a system to display a desired image on rotating board, using 17 LEDs only for less power consumption, taking in consideration many factors that can affect the system. The rotation's speed is one of these factors, the system require a high speed to get a good resolution (more than 1400 RPM), also the LEDs must be perfectly programmed and synchronized in order to create the desired image on the board.

Conclusion and perspectives

This project concentrated on the design and implementation of a two dimensional display based on the persistence of vision phenomenon, the built system consists of single dimensional array of LEDs well arranged on a rotating board, many aspects in term of coasts, power requirement, hardware requirement, ease of use and maintenance were considered.

The display is extremely attractive and gives the impression of being a transparent display using a motor of higher RPM and torque , the task will be perfectly performed by a BLDC motor , the result obtained from displaying a string of characters or numbers are satisfying, other applications can be done by the system , an analog clock is one of them , it was displayed on the system but sadly the real time wasn't indicated on it

For future work we would like to develop the system to have the ability to display colored images using RGB LEDs instead of a single color LEDs , also to control the system and change the displayed pattern wirelessly via a PC or a phone , and upgrade the system to display a three dimensional images.

REFERENCES:

- [1] Ancy David, KJ Josphin², Sreejith S, Thomasukutty Zacharia, “*Propeller LED Display*” UG Scholars, Department of Electrical and Electronics Engineering, AmalJyothi College of Engineering, April 2016
- [2] d’Arcy, Patrick *Sur la durée de la sensation de la vue* in *Histoire de l’Académie Royale des Sciences - Année M. DCCXV*. p. 439-451 (1768)
- [3] Herbert, Stephen. "Wheel of Life - The Taumatrope"
- [4] Briggs, David (2012-08-12). "Additive mixing, additive-averaging". *The Dimensions of Colour*. Retrieved 2018-08-11.
- [5] Robinson P. Paul et al., “Persistence of Vision Control Using Arduino”, I.J. Intelligent Systems and Applications, 01, 102-111, 2014.
- [6] Przemysław Bartosik, Michał Zasłona, “Propeller Clock”, Intermediate Project Final report, Department of Cybernetics and Robotics Wrocław University of Technology, February 15, 2016.
- [7] Arduino Due Released, Arduino.cc
- [8] Reference Image : arduino.cc/en/uploads/Main
- [9] K. Anusha, L. Baala Gajakreedan, S. Nishanth Kumar and S. Vignesh Rajarathinam, Persistence of Vision using Arduino. *International Journal of Electronics and Communication Engineering and Technology*, 8(4), 2017, pp. 7–12.
- [10] <http://www.stefanv.com/electronics/escprimer.html>
- [11] Arduino Due Released, Arduino.cc
- [12] *Programming Arduino Getting Started with Sketches*. McGraw-Hill. Nov 8, 2011. ISBN 978-0071784221.