TECHNOLOGICAL UNIVERSITY OF THE PHILIPPINES

Ayala Blvd., Ermita, Manila

COLLEGE OF ENGINEERING

ELECTRONICS ENGINEERING DEPARTMENT

electronics@tup.edu.ph

BEREAL: A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

by

Borjal, Mary Loise O.

Diocera, Michael Conrad L.

Dumandan, Jetlee E.

Gamil, Jessica E.

Pangantihon, Mark B.

Reyes, Jewelle Milles A.

August 2023

APPROVAL SHEET

This project study entitled **"BEREAL: A MODULAR LOW-COST TWO-WAY REFRESHABLE BRAILLE DISPLAY COMMUNICATION AND LEARNING DEVICE FOR BLIND PERSON"**, has been approved and submitted by the following proponents:

Borjal, Mary Loise O	**Gamil**, Jessica E.
Diocera, Michael Conrad L.	**Reyes**, Jewelle Milles A.
Dumandan, Jetlee E.	**Pangantihon**, Mark B.

In partial fulfillment of the requirements for the degree of **Bachelor of Science in Electronics Engineering** is hereby recommended for approval.

ENGR. EDGAR A. GALIDO Project Adviser	**ENGR. JUNE ANTHONY B. ASISTIO** Project Co-Adviser
ENGR. ROMEO L. JORDA JR. Panel Member	**ENGR. LEJAN ALFRED ENRIQUEZ** Panel Member
ENGR. CHERRY G. PASCION Panel Member	**ENGR. JOHN PETER M. RAMOS** Panel Chair

ENGR. GILFRED ALLEN M. MADRIGAL

Accepted and approved in partial fulfillment of the requirements for the degree of **Bachelor of Science in Electronics Engineering**

ENGR. TIMOTHY M. AMADO Head, ECE Department	**ENGR. NILO M. ARAGO** Dean, College of Engineering

ABSTRACT

Braille is a touch-based way used by the blind, deafblind, and visually impaired to read. Therefore, this study aims to develop a low-cost refreshable braille device that blind people can use for education and communication purposes. The said device has a function to convert speech into text, optical character recognition (OCR), text-to-braille. The speech-to-text and text-to-braille functions were designed for the communication of blind, deaf, mute, and deaf-mute people. On the other hand, Optical Character Recognition (OCR) is designed for learning purposes by scanning certain documents which the digital text will be converted to its equivalent braille form and will reflect through the actuator. In addition, the device is composed of seven six-dot braille cells represented by a solenoid actuator which will provide the users with a tactile reading experience. These actuators are controlled by Arduino Uno Microcontroller and Raspberry Pi 4 Model B for processing the data. The implemented Vosk API showed an accuracy rate of 88.896% in testing, and for Tesseract OCR, the text should be positioned three to nine inches away from the camera and have a font size of 25 pt. The effectiveness of the device was evaluated by 11 respondents using Likert Scale, it shown an average of 4.044 which depicts that the respondents were satisfied and agreed that the device was effective. This proposed device offers a practical and cost-effective solution to enhance communication and learning for the blind people community, especially in accessing digital content using technology.

ACKNOWLEDGEMENT

The proponents would like to seize this moment to extend their heartfelt thanks and recognition to all those who have played a part in this undertaking, both on a personal and professional level. Their support, guidance, and encouragement have played a pivotal role in molding the proponent's growth and achievements.

To Engr. Edgar Galido and Engr. June Asistio, for their guidance, knowledge and support throughout the project. Their guidance and expertise played a vital role in completing this project. The dedication and efforts they exerted will never be forgotten.

The proponents would like to express their profound appreciation to every panel member for their willingness to generously impart their wisdom, knowledge, and personal experiences. To, Engr. Romeo L. Jorda Jr., Engr. Lejan Alfred Enriquez, Engr. Cherry G. Pascion, Engr. Gilfred Allen M. Madrigal, and Engr. John Peter M. Ramos for their valuable insights and suggestions that seek for the betterment of the study.

Also, the proponents would like to convey their genuine thanks and deep appreciation to Engr. Nilo M. Arago for his invaluable contribution to this project as his guidance and expertise have been instrumental in the process of the project's success.

To Bagong Silang Elementary School and P. Gomez Elementary School, the proponents would like to thank the school administration, including the principal, vice-principal, students, and administrative staff for their unwavering dedication and commitment to fostering a conducive learning environment.

TABLE OF CONTENTS

Approval Sheet i

Abstract ii

Acknoledgement iii

List of Figures xiv

List of Summary Tables xvi

Chapter 1: THE PROBLEM AND ITS SETTING 1

1.1 Introduction 1

1.2 Background of the Study 3

1.3 Research Gap 4

1.4 Research Objectives 5

1.5 Significance of the Study 5

1.6 Scope and Limitations 6

1.7 Definition of Terms 7

Chapter 2: REVIEW OF RELATED LITERATURE 9

2.1 Braille 9

2.2 Braille System 10

2.2.1 Braille Display 11

2.2.2 Effectiveness of Braille 14

2.3 Algorithm 15

2.3.1 Vosk 15

2.3.2 Tesseract 19

2.4 Braille Technologies 24

2.4.1 Sensors 24

2.4.1.1 Virtual Keyboard 24

2.4.2 Microcontroller 24

2.5 Braille Conversion 26

2.5.1 Text to Braille 26

2.5.2 Speech to Braille 30

2.5.3 Image To Text to Braille 32

Chapter 3: METHODOLOGY 36

3.1 Research Design 36

3.2 Research Process Flow 38

3.3 Designing and construction of seven braille characters that utilized an electromagnetic push-pull solenoid as braille cells in developing a low-cost braille device that incorporates touch, speech, and an image sensor for data inputs. 39

3.3.1 Creation of a low-cost hardware device using push-pull solenoid actuators 39

3.3.1.1 Purchasing the push-pull solenoid actuators for braille cells of the device. 40

3.3.1.2 Determining and purchasing the image, sound, and touch sensors for the device. 40

3.3.1.3 Designing of the Braille Device 40

3.3.1.4 Construction of the device using push-pull solenoid actuators as braille cells and integrating the image, sound, and touch sensors 42

3.4 Designing and developing of a software application with a user-friendly Graphical User Interface (GUI) that integrates the text-to-braille, image-to-braille, and speech-to-braille functions all associated with the push buttons. 42

3.4.1 Construction of Graphical User Interface (GUI) 42

3.5 Developing a program that converts image-to-text using Tesseract OCR and speech-to-text using the Vosk API into a single program 43

3.5.1 Implementation of Tesseract for OCR 43

3.5.2 Implementation of Vosk for Speech Recognition 43

3.5.3 Implementation of Tesseract and Vosk into a Single Program 44

3.5.4 Text to Braille Conversion using Arduino. 44

3.6 Assessment of the device functionality by actual testing it by blind people to deaf, mute, and unimpaired people. 44

3.6.1 Making of the survey in Likert scale form 44

3.6.2 Gathering of the possible respondents for the survey 44

3.6.3 Evaluating and assessing the device's functionality by actual usage 45

3.6.4 Providing the respondents, the survey form 45
3.6.5 Gathering the completed survey form... 45
3.7 Statistical Analysis 45
3.8 Project Workplan... 46
Chapter 4: RESULTS AND DISCUSSION 47
4.1 Project Technical Description 47
4.1.1 Screenshots of the Software... 58
4.2 Project Limitations and Capabilities.. 60
4.3 Project Structural Design 61
4.4 Project Evaluation.. 62
4.5 Deployment Plan.. 73
4.5.1 Overview.. 73
4.5.2 Purpose... 75
4.5.3 Sites/ Seats/ Location 75
Chapter 5: SUMMARY OF FINDINGS, CONCLUSIONS AND RECOMMENDATIONS 76
5.1 Summary Findings.. 76
5.2 Conclusions.. 78
5.3 Recommendations.. 80
REFERENCES.. 81

LIST OF FIGURES

Figure 1. Conceptual Framework of the System 37

Figure 2. *The flow process of the Study* 38

Figure 3. *Rear Side View of the Whole Device with a Dimension of 12x10x5 inches* 41

Figure 4. *Rear Side View for the Enclosure of the Chassis* 41

Figure 5. *Block diagram of the Study* 47

Figure 6. *Flowchart of the system* 48

Figure 7. PCB Layout 49

Figure 8. Schematic diagram of each braille character 49

Figure 9. Schematic Diagram of the Whole Braille of the Whole Braille Character............. 50

Figure 10. *Initial window for input mode selection* 58

Figure 10.1 Schematic of the 1st character 58

Figure 10.2 Schematic of the 2nd character 58

Figure 10.3 Schematic of the 3rd character 58

Figure 10.4 Schematic of the 4th character 58

Figure 10.5 Schematic of the 5th character 58

Figure 10.6 Schematic of the 6th character 58

Figure 10.7 Schematic of the 7th character 58

Figure 11. *GUI for Speech-to-Text Converter* ……………………………………………..58

Figure 12. *GUI of Keyboard for Text to Braille* 59

Figure 13. *GUI for Image-to-Text* 59

Figure 14. *3D Design of the Device: a.) Rear side view b.) Front view* 61

Figure 15. *Actual Prototype: a.) Back view b.) Front view* 61

Figure 16. *Level of Vision Impairment* 62
Figure 17. *Number of respondents who used a screen reader or a braille device before.* 62
Figure 18. *Number of respondents who are familiar with Grade 1 braille alphabet.* 63
Figure 19. *The usefulness and relevance of speech-to-text conversion for communication purposes.* 63
Figure 20. *The usefulness of image-to-braille conversion for learning purposes.* 64
Figure 21. *Number of respondents who agreed that the device meets the needs of blind users.* 64
Figure 22. *Number of respondents satisfied with the braille output quality of the refreshable braille display.* 65
Figure 23. *Number of respondents satisfied with the user-friendliness of the refreshable braille display.* 65
Figure 24. *Accuracy rate of image-to-text conversion* 66
Figure 25. *Accuracy rate of image-to-text conversion* 67
Figure 26. *Accuracy rate of image-to-text conversion* 68
Figure 27. *Image-to-braille conversion* 68
Figure 28. *Speech-to-text conversion* 69
Figure 29. *Text-to-braille conversion* 70
Figure 30. *Word Per Minute* 71

LIST OF SUMMARY TABLES

Table 1. Braille 10

Table 2. Braille System and Braille Display 12

Table 3. Vosk as Speech Recognition 17

Table 4. Tesseract Engine as Optical Character Recognition 21

Table 5. Microcontroller 25

Table 6. Text to Braille 28

Table 7. Speech to Braille 31

Table 8. Image to Braille 33

Table 9. Gantt Chart of the Study 46

Table 10. *Table for Words Per Minute* 72

Table 11. Statistical Measure for Word per Minute 72

Table 12. Comparison Table of the BeReal between the Commercially Available Braile Devices 73

ANNEXES

ANNEX I	BILL OF MATERIALS
ANNEX II	BEREAL CODES
ANNEX III	PROJECT DOCUMETATION
ANNEX IV	EVALUATION FORM
ANNEX V	BEREAL USER’S MANUAL
ANNEX VI	MANUAL ON HOW TO DUPLICATE THE PROTOTYPE
ANNEX VII	PROPONENT’S INFORMATION

Chapter 1:

PROBLEM AND ITS SETTING

1.1 Introduction

Communication involves the transmission and reception of information, either in individual or collective settings, and can occur through direct interactions or by utilizing communication tools and technologies. But among those individuals with a disability or who have senses that can't function well, this skill is difficult for them.

As per the World Health Organization (WHO), roughly 314 million people worldwide are believed to be impacted by visual impairments, encompassing both blindness and low vision. Out of this population, 45 million individuals are blind, with an additional 269 million experiencing low vision [1]. According to the Department of Health (DOH) in the Philippines, as of 2017, the estimated count of individuals with bilateral blindness in the country stands at 332,150, while those with bilateral low vision are estimated to be 2,179,733. The other factors in these cases were cataract, error in refraction (EOR), and glaucoma. [2]

In India, the primary stumbling block is the scarcity of opportunities. Education for blind people is difficult since their parents and families are often unaware of both the necessity and potential of education. The individual may not receive sufficient counseling or be referred to a specific school for blind or visually impaired individuals. [3] In many cases, individuals who are blind are faced with permanent vision loss. Consequently, these individuals must depend on their remaining senses, such as auditory or tactile cues, to navigate their daily routines. Although technology has advanced dramatically, not all people are able to use it. Given the fact that not everyone cannot cope up with the advancement of technology, devices like braille are vital to all visually impaired and blind individuals in terms of learning and communication. But the downfall is that the refreshable braille displays, are extremely expensive, typically the cost ranges from around $2,000 for an 18-character display to $50,000 for a half page of braille. [4] The other problem to be address is how to aid blind people to communicate other people with disabilities, such as the deaf and mute people.

Utilizing the latter, a mechanical system has been specifically devised to cater for the communication between the people with communication difficulties by interfacing braille pads to improve and bridge the gap between them and the progression in technology on a global scale.

Braille is a system specifically created to assist individuals who are visually impaired or have low vision. It has a 6 raised dot system which can be read by touching the fingertips. In this way, it provides literacy for everyone by allowing the blind and low-vision people to communicate and read words using their sense of touch. [5] When reading a Braille device, the fingertips should be moving in a horizontal arrangement, from left to right in the same line.

This research study aims to improve existing Braille integration technologies and provide more communication options for disabled or impaired people by developing and implementing a device with corresponding software that will interpret English speech and/or text and convert it into real-time Braille.

1.2 Background of the Study

According to [6], Software, Voice O Braille, provides a simple learning environment for visually impaired persons. The device could listen and read text and convert it into braille. The program includes an editor that displays the speech that has been recognized. This editor allows users to review any errors that occurred during speech recognition. It features a feature that allows you to save and listen to notes that were converted from voice.

According to [7], the device created recognizes one person's voice, converts it to Braille, and embosses the Braille on a Braille display. The speech input and output signals are both in Tamil. The HM2007 IC is the system's core component, and it can recognize more than 10 words. Electricity-powered solenoid actuators activate when they receive 5V electricity from the PIC and are connected to the Six Solenoids which will be projected as braille dots. According to [8], the device was developed to provide a low-cost device system that helps visually impaired or blind persons in learning braille. Solenoid Valve-Based Braille is applied instead of Braille itself because it is a more expensive product. The Arduino UNO will be used to convert ASCII data to binary. The microcontroller is powered by a USB connector and 12V and 5V solenoids. Using solenoids, ASCII characters are transformed into braille characters by moving pins inside the solenoid housing.

While on [9], the device was developed as a low-cost portable Braille Pad that allows blind or near-sighted persons to learn and read Braille. The device can convert braille to text and convert text to braille. A visually impaired individual wanting to read a file on his/her laptop or computer would provide the program with input during the text-to-braille procedure. The software on Arduino Leonardo converts the text or passage in the file to ASCII binary representation. In the six-combination input system of Braille, each dot sequence is represented in a series of six binary digits which are combinations of 0's and 1's. Users enter text via the keyboard, which then sends a signal to microcontrollers that shift out the information to a series of shift registers. The Arduino Leonardo will transform the Braille letters to text and display them on an LCD screen.

Lastly, based on [10], the text-to-braille Scanner device is created to develop a low- cost braille display. The proposed display runs by sliding two eleven-slot pre-protruded sliders to generate braille characters utilizing variation of the slots. The said device is composed of two series connected 3.7V Li-ion batteries and an HC-05 Bluetooth module which is responsible for getting the text data from the phone and transferring it in ATTiny85 microcontroller. The text scanner is powered by Google's OCR engine and available as a mobile app. The scanner app can transform English text into braille in real time without the need for internet access.

1.3 Research Gap

Several studies concerning braille communication devices and software show that they are capable of handling speech-to-braille and text-to-braille separately. The Braille communication device that consists of the following features is not commercially available. However, developing such a system by integrating the features into a single device would need further study. The utilization of the low-cost braille pad has proven useful in developing a system to handle a single task. Thus, developing a device by integrating the speech-to-text, image-to-text and text-to-braille features, including finding an alternate low-cost braille pad, is proposed. However, a challenge for this system is the accuracy rate of the text-to-braille translation.

1.4 Research Objectives

This research study's main objective is to assist blind people in communicating with deaf, mute, and unimpaired people with emerging modern technology using a portable device that converts text, speech, and image into a braille system.

1. To design and construct seven braille characters that utilized an electromagnetic push-pull solenoid as braille cells in developing a low-cost braille device that incorporates touch, speech, and an image sensor for data inputs.
2. To design and develop a software application with a user-friendly Graphical User Interface (GUI) that integrates the text-to-braille, image-to-braille, and speech-to-braille functions all associated with the push buttons.
3. To develop a program that converts image-to-text using Tesseract OCR and speech-to-text using the Vosk API into a single program.
4. To evaluate and assess the device functionality through actual testing by blind or visually impaired, mute, deaf, professors specialized in teaching braille systems and professionals in creating braille devices.

1.5 Significance of the Study

Communication is an essential human characteristic. To properly maintain relationships, humans must interact with one another. Science and technology have made the modern world so much simpler, but some physically and visually impaired people still have difficulty communicating with others. For people who are blind, deaf, and mute, communication is one of the biggest challenges.

This study aims to create a braille device that is low-cost and has a function that will convert image, speech, and text to a real-time braille to make communication more accessible for people who are blind, deaf, and mute. With the application of the latest technology in the field of health care, which comes under the Harmonized National Research and Development Agenda's (HNRDA): Health: Diagnostic and Industry, Energy and Emerging Technology: Competitive Industries (ICT, Electronics, and Semiconductor) categories.

Also, for the Sustainable Development Goal (SDG), whose objective was to establish a set of global goals to aid in the fight against the world's serious environmental, political, and economic issues. It is also the plan for a better, more sustainable future for everyone. This study will be undertaken under SDG 4 which is all about quality education, SDG 9 which focuses on Industry, Innovation, and Infrastructure, and lastly SDG 10 which focuses on Reduced Inequalities.

Lastly, this study will have a socio-economic impact on family and social support by offering accessible communication devices allowing blind people to easily blend in with the people around them.

1.6 Scope and Limitations

The aim of the research is to improve existing Braille integration technologies and provide accessible communication options for blind people by developing and implementing a device with corresponding software that will interpret English speech/ and or text and convert it into real-time Braille form. The software will be consist of Processing Integrated Development Environment (IDE) software which is responsible for translating the input to braille form. The hardware will be consist of a microphone for speech purposes, a braille cell for the representation of letters, a camera for scanning the text from images/objects and microcontroller for allowing the braille cell to work individually.

All converted braille forms of images, speech, and text will be available only in English, because aside from it being the universal language, the trained data from Tesseract doesn't support Filipino Language. The user of the device must be literate on using braille. In addition, the deaf and mute should be educated in order for them to communicate with blind people. The device will be consist only of seven braille cells considering the budget and size of the hardware.

1.7 Definition of Terms

- ***Braille***

  Braille functions as a tactile method of literacy, supporting individuals who are visually impaired or have limited vision in their reading and writing abilities. It entails perceiving the letters and symbols through the sensitive pads of the fingertips to facilitate understanding and communication.

- ***Braille cells***

  In a braille cell, there are two rows containing three raised dots each, making a total of six dots. These dots are assigned numbers one through six to indicate their specific positions within the cell. By utilizing one or more of these six dots, there are a total of 64 different combinations possible.

- ***Braille device***

  A braille display, alternatively referred to as a braille terminal, is an electromechanical apparatus designed to exhibit characters. It accomplishes this by employing rounded-tipped pins that are raised through openings in a flat surface.

- ***Braille pads***

  Braille pads include an embedded camera that allows you to snap touch photos in real time. The clever Contour Trace application, which runs on Linux, allows you to convert normal visuals into outline tactile images.

- ***EOR (Error of refraction or Refractive)***

  EOR (Error of refraction or Refractive) errors are the type of vision problem which make a person have difficulty in seeing clearly. It occurs when your eyes block light from focusing on your retina. The retina is located at the back of your eye and is a light-sensitive layer of tissue part of an eye.

- ***Glaucoma***

  Glaucoma encompasses a group of eye illnesses that damages the optic nerve located at the back of the eye which results in blindness and vision loss.

- ***Tactile feedback***

  Tactile feedback is involved with replicating pressure patterns on the skin's surface. These pressure patterns are detected by small receptors located throughout the skin and translated into a wide range of sensations by the brain.

Chapter 2

REVIEW OF RELATED LITERATURES AND STUDIES

This section provides facts, information, and concepts as a guide to accumulating knowledge and understanding that is necessary for this study to develop and keep up to date.

2.1 Braille

The Invention of Braille [11] tackles the person behind braille and how it was invented. On January 4, 1809, Louis Braille was born in Coupvray, France. At the age of three years old, his one eye was accidentally blinded by himself using a sewing awl from the leather shop of his father. While his other eye also became blind because of Sympathetic Ophthalmia. It is an inflammation of both eyes due to the trauma or injury to one eye. He devised a worldwide reading and writing system for blind and visually impaired persons when he was 15 years old, which is now named after him. [12] Over the world, braille has been a known language and has become an essential tool utilized by the blind and visually impaired people. To read a braille, the person's fingertips must pass over the characters that consist of embossed points with an arrangement of one to six. Different alphanumeric characters are represented by the relative placements of these dots.

Table 1. *Summary Table for Braille*

Author	Year	Title	Relevant Findings	Relationship to your study
M. Sen and S. G. Honavar	2022	Louis Braille: Dancing in the Dark	The inventor of the Braille was discussed in this study as well as how Braille started.	Both studies use Braille as the main topic.
Geoff Hudson-Searle	2017	From Louis Braille to leading edge technology that helps people see again	In this study, the usage of braille was discussed and its structure.	Both studies discussed the usage of braille and new creation of prototypes.

2.2 Braille System

The most popular tactile system for communicating between those who are blind and those who are sighted is braille. It is a system used by the blind for reading and writing by touch. It is composed of patterns of dots that form letters of the alphabet, numerals, and punctuation marks that are read by moving the hand or hands down each line from left to right. [13]

A study by [14] presents a novel idea for a portable, low-cost, low-energy, and user-friendly Braille system. The program serves as a Braille writing and reading system that enables people who are blind or visually impaired to improve their Braille writing and reading abilities without the help of a Braille teacher. The intended system accepts input from a Braille keyboard and outputs Braille on a Braille display; the corresponding English characters are displayed on the LCD and, if a laptop is attached, also on the laptop. It also has text-reading capabilities for documents.

2.2.1 Braille Display

By electronically raising and lowering different combinations of pins within braille cells, braille displays make it possible to get information that is presented on a computer screen. A braille display can show up to 80 characters from the screen and is refreshable, which means that it changes as the user moves the pointer around the screen using the Windows and screen reader commands, command keys, or cursor routing keys. [15]

A piece of computer hardware known as a refreshable braille display has a series of fluid braille cells on its surface. Most displays consist of a single line with fourteen to eighty braille cells. Each of these cells has a small pin that may be raised or lowered to mimic a braille dot. Blind people can read braille by tracing their fingers across refreshable braille cells and then moving the display to the following set of characters. [16]

Refreshable braille has several key advantages over other braille technologies. Large amounts of hard copy or paper braille must be generated. The generation of regenerative braille from electronic files. This means that a great deal of data may be stored on a little device.

In the research of [47], an electromagnetic actuator for braille cells has been developed. It consists of a body and a magnet holder. The magnet holder lifts due to the effects of electromagnetic induced current. of the coil when using a coil and a magnet. To enable the braille cell, a coil with 300 turns and a diameter of 0.1 mm is used with a supply voltage of 1.5 volts and a minimum current of 0.08 amperes.

In research [17], when a computer input is received, a prototype of a low-cost Braille display for the blind to read is suggested. In this instance, solenoids that are electromagnetic will be utilized to ease the vertical movement of the Braille display's pins, which is controlled by an Arduino Uno. Blind persons will be able to read braille alphabets, which are unique dot patterns on a braille cell, with this device.

Table 2. *Summary Table for Braille System and Braille Display*

Author	Year	Title	Relevant Findings	Relationship to your study
S. Sultana, A. Rahman, F. H. Chowdhury and H. U. Zaman	2017	Novel Braille pad with dual text-to- Braille and Braille- to-text capabilities with an integrated LCD display	An innovative, inexpensive, portable Braille system is presented in the study that is also user-friendly. A Braille writing and reading system is	Both studies present a low-cost portable Braille system. The two studies use a keyboard for the writing system and an LCD display for the output.

			provided by the system as intended. The proposed system accepts input from a Braille keyboard and outputs Braille on a Braille display.	However, for the present study it also includes a microphone for speech purposes as for the additional input.
M. E. Adnan, N. M. Dastagir, J. Jabin, A. M. Chowdhury and M. R. Islam	2017	A cost-effective electronic braille for visually impaired individuals	It suggests a model of an accessible Braille display that can be read by blind persons when input is provided via a computer.	Both studies propose a low-cost device for the blind people. However, their study focuses on the input given in a computer. The present study includes the input of an image, speech, and text.

M. Nizam, S. Saad, M. Suhaimi et al.	2021	Development of electromagnetic actuator for braille cell	The study shows the ideal amount of turns and diameter of the copper wire. It can produce high electromagnetic force with low current flow.	Both studies present electromagnetic actuators. The two studies present the use of a modular low-cost electronic braille display.

2.2.2 Effectiveness of Braille

Braille reading is a physically intensive tactile activity. Because the pulps of both hands are particularly sensitive in tactile exploration and recognition, reading it heavily relies on the use of both hands' forefingers. The findings demonstrated that handedness has an impact on braille readers' selected reading habits when reading text. Numerous reading styles were observed, and the findings were compared with tactile reading techniques based on dominant hands and fingers. The index of their dominant hand seems to be where readers who chose one-hand braille reading patterns made a lot more mistakes, while readers who chose to read with both hands had a harder time coordinating their indices. Finally, the current study's findings are examined in light of educational practice, applicable theory, and future research. [23]

Despite the widespread adoption of accessible digital technology, Braille continues to play a role in presenting content to blind people. Despite this, many people do not see the value in learning Braille, especially given the time and work required to master it. There are preliminary findings from five interviews with blind people, including Braille teachers and pupils, in which it describes the learning process and Braille usage. [24]

2.3 Algorithm

2.3.1 Vosk

Deep learning can be used to construct several models for speech recognition. A sound prediction procedure takes place with the help of a deep learning network model that has previously been trained for sound detection. Vosk has been and continues to be developed, according to the GitHub repository, since 2019. The API is currently at version 0.3.32. It can be used for speech recognition, smart home device STT, and virtual assistance. In [41] VOSK library and its model were used to extract a segment of speech from a video. The researchers used at least 50 words and phrases for both English and Turkish languages that could refer to any of the categories held. Text extracted from the given video using the Vosk library has been processed using NLP techniques.

In [40], Dornbirn's inatura museum had designed an installation in the guise of an interactive speech assistant. Visitors were encouraged to ask the display the same questions they would ask a flower. Because of the exhibit's resemblance, open-source offline Speech to Text (STT) engines and speech assistants were investigated. The researchers used Vosk since it is possible to transcribe the audio in real-time. In addition, it can, or has already been incorporated into some Mycroft and Rhasspy are speech aides. In [42], the researchers experimented with the three different speech recognition models which are Vosk, DeepSpeech and google speech recognition on which is the best speech recognition to be used for the robot. After the experiment, the researchers selected Vosk as it has a low word error rate (WER) among others. In [44], the study tested three open-source speech-to-text engines, Mozilla DeepSpeech, CMU Sphinx, and Vosk, using free Common Voice datasets for training. According to the results, Vosk is nearly identical to Google Speech API, with an accuracy rate of 85% for the three languages (English, Spanish, and Portuguese) while CMU Sphinx engine is the fastest with a response time of less than or equal to 1 second. Vosk engine follows with the response time between 1 to 1.5 seconds and DeepSpeech has the slowest response time, averaging more than 2.5 seconds for the English dataset and more than 3 seconds for the Spanish sample.

Table 3. *Summary Table for Vosk as Speech Recognition*

Author	Year	Title	Relevant Findings	Relationship to your study
C. W. An and S. Assistant	2022	Offline Speech To Text Engine For Delimited Context	The small model of Vosk was used to test the algorithms. The main reason for using the small Vosk model for testing was that there was no significant difference between the big and small models	Both studies will use Vosk speech to text (STT) engine as a recognition.
E. Boztepe, B. Karakaya, B. Karasulu et al.	2022	An Approach for Audio-Visual Content Understanding of Video using Multimodal Deep Learning Methodology	The average precision, average recall, average F1 score, average cosine similarity score, average BLEU score, and WER score were calculated using the VOSK speech tests	The study is based on multilingual audiovisual content (English and Turkish) retrieved using a deep

			for English videos given are 0.89, 0.95, 0.92, 0.92, 0.90, and 0.21, respectively. Furthermore, the metrics obtained in For Turkish videos, the VOSK speech tests are 0.96, 0.94, 0.95, 0.95, 0.81 and 0.41.	learning approach. Both studies will use English language however the reference intent is to create a multilingual speech recognition.
F. P. Comas	2022	Language grounding for robotics	The experiment results show that implementing Vosk as speech recognition engine has 8.37%-word error rate (WER) and can be lowered if the	Both studies will implement Vosk as Speech to text engine.

			speaker has English or American Accent.	
T. F. Pereira *et al.*	2021	A web based Voice Interaction framework proposal for enhancing. Information Systems user experience	The results shows that Vosk is almost similar to Google Speech API and it has accuracy rate of 85% for the three languages (English, Spanish and Portuguese)	Both studies will use Vosk for converting English speech into text.

2.3.2 Tesseract

The Tesseract engine is a powerful tool for character recognition in various fonts and sizes. Training the OCR engine procedures, recognition of text, and quantitative assessment of the trained engine; such an extensive training and assessment system, driven by a GUI, enables continuous incremental training to get the optimal results. [43] The goal of the study presented is to show how certain kernels for convolution-based preprocessing might increase the accuracy of the Tesseract 4.0 OCR engine. Although

Tesseract 4.0 displayed excellent performance when compared to a favorable input, its capacity to detect and identify characters in more realistic, unfavorable images is called into question. [38] This study developed a platform for converting printed text documents into equivalent braille code, which activates the visible output of braille cells. The system is made up of two major components: an image reader and a microcontroller-based braille system. The image scanner collects the printed text document and runs a variety of pre-processing algorithms before subjecting the processed image to Tesseract character recognition. [37].

In [39] the researchers used the Tesseract OCR engine in a content-based image retrieval system, as well as an upgraded Levenshtein Algorithm to extract text from a picture and use text-image to identify the whole document containing the text. The retrieved text is checked against the text in the database. The results revealed that the appropriate data was successfully retrieved 100% of the time, even when only a portion of the query photographs were used. Another study [45] demonstrated the usage of tesseract OCR for tracking wine using simply a smart phone with a camera. The application's goal is to be able to scan the information on a wine bottle using a mobile phone. In this study [46] the raspberry pi controller has the tesseract software installed. After the photo has been taken, the software on the Raspberry Pi will process it, and the text that has been processed will be shown in the output window.

Table 4. *Summary Table for Tesseract Engine as Optical Character Recognition*

Author	Year	Title	Relevant Findings	Relationship to your study
R. de Luna	2022	A tesseract-based optical character recognition for a text-to-braille code conversion	The tested photos all use the same Arial font in varied font sizes, including 12, 14, 16, 18, and 20. When tested with Arial font sizes 18 and above, the overall system achieves less than the intended 85% accuracy, while it achieves 0% average accuracy when tested with Arial font sizes 12 and less.	In this study researchers used raspberry pi as a microcontroller-based braille platform. Also, the tesseract engine was utilized in this work as optical character recognition to translate printed text into braille script equivalents.
Sporici, D., Cuşnir, E.,	2020	Improving the Accuracy of Tesseract 4.0 OCR Engine	The approach shows that the F1 value of Tesseract 4.0 can be raised from 0.163 to	Tesseract has been implemented in this study as

& Boiangiu, C. A.		Using Convolution-Based Preprocessing	0.729 using only preprocessing of pictures, for images that weren't part of the set used for training, and with no external knowledge of the dataset.	optical character recognition. In this study it was shown that using image preprocessing can increase the accuracy.
Adjetey, C., & Adu-Manu, K. S.	2021	Content-based Image Retrieval using Tesseract OCR Engine and Levenshtein Algorithm	Further investigation shows that when a copy of a specific image serves as the criteria for searching, and the outcomes obtained are practically one, the comparison ratio is nearly one.	In this study it was shown that the fundamental goal of tesseract OCR is to identify characters in a picture.

S. Cakic, T. Popovic, S. Sandi, S. Krco, and A. Gazivoda	2020	The Use of Tesseract OCR Number Recognition for Food Tracking and Tracing	A 1500 set of small actual photos of wine serial numbers is used to assess the solution. The outcomes demonstrated that an important factor in raising the success rate is picture preprocessing.	Both studies will use python language to implement Tesseract
I. S. Akila, B. Akshaya, S. Deepthi, and P. Sivadharshini	2018	A Text Reader for the Visually Impaired using Raspberry Pi	It is a piece of software that creates speech out of text. The TTS Engine transforms written language into phonetic Illustration.	Both studies will create a device that recognizes the text that is present in the image.

2.4 Braille Technologies

2.4.1 Sensors

2.4.1.1 Virtual Keyboard

A keyboard on a computer is a device of input that inserts characters and operations into a computer system by using buttons or keys. It is the most often used technique of text entering. A keyboard frequently has individual letters, numbers, and special characters, as well as keys for specialized activities. To connect a keyboard to a computer system, a wireless or cable connection is used. [18] The most common keyboard type is an external and physical keyboard wherein the user needs to plug it into the PC to use it. A touch screen is a form of display that allows users to interact with computers through the use of their finger or a pen or pencil. They can be used to browse a GUI instead of a mouse or keyboard (graphical user interface). [19]

2.4.2 Microcontroller

A microcontroller is a low-cost, tiny computer that is designed to execute certain functions in embedded systems such as computers presenting microwave data, receiving remote signals, and so on. The different types of microcontroller boards are Arduino uno, raspberry pi, and beagle bone black. [31]

In the method discussed in this paper [20], researchers used AVR AT mega16 as the microcontroller and with a proper ASCII mapping, it will be downloaded to the PC for presentation on the hyper terminal. The most significant style of communication is that it combines the Braille to computer communication between two gloves. An individual or user wears a glove with switches connected to each finger. Starting with the index finger, the first four fingers of the glove a vibration motor and tactile micro switches will be installed on the thumb.

In this article, the main approach of the researchers is to design an audio-assisted stand-alone microcontroller-based Braille System tutor that seeks to help visually impaired people to improve their literacy. The system has two mode operations, the read mode operation and the write mode operation. Raspberry pi is implemented as the braille tutor for the audio assisted. [32]

Five participants in this study were asked to touch and identify the hijaiyah letters using a specially constructed Arabic braille module. The Arabic braille module was built with small solenoids, the Arduino Uno as a microcontroller, a module for Bluetooth (HC-06), and open-source mobile application software. [33]

Table 5. *Summary Table for Microcontroller*

Author	Year	Title	Relevant Findings	Relationship to your study
M. Garcillanosa,	2017	Audio-assisted standalone microcontroller-	The results show 100% accuracy in both mode	Both studies will use raspberry pi

K.N. Apuyan et al.		based Braille System Tutor for Grade 1 Braille symbols	operations. However, additional testing is possible for a blind somebody who is unfamiliar with braille to double-check the device efficiency.	as a microcontroller.
N. Rahimi, N.Hanif, Z. Janin	2019	Identification of hijaiyah letters using wirelessly controlled Arabic braille module	The total correct answer of the Arabic Braille code of Group A participants obtained an average of 72.9% while Group B participants obtained an average of 65.3%.	Both studies will use grade 1 braille cells to address the problem of blind people.

2.5 Braille Conversion

2.5.1 Text to Braille

There are a lot of ways to convert text into braille language. Using software to swiftly convert text to Braille or vice versa saves longer time than doing it manually. This will help visually challenged people learn faster. [9]. The main approach used by Sruthi Ramachandran et al., (2021) is to have a device which can change any braille character inputs to alphanumeric text format by having two modes of communication, the short distance communication and long-distance communication. In the short term, it employs a Braille keypad as an input, which is supplemented with a push button keyboard and an LCD screen as an output. The GSM module is used in long-distance communication to send messages to a remote receiver with a vibration band to notify the user when a message is sent. [21].

In [10], they developed an app that uses the open-source Optical Character Recognition (OCR) engine from Google. The user will use the camera to scan the image and the system will convert the detected texts into braille form. This is free and the process runs all on the board, so it doesn't need the internet to work. While in [9], instead of using the camera to take the input texts, they used the computer/laptop's keyboard to take the input from the user.

A tactile approach is a method of reading for those who are visually impaired. In [26], the researchers created a system that will convert text to braille which can also create 3D models that can be used as storyboards for short stories or riddles for visually impaired students and staff at the Philippine National School for the Blind (PNSB).

Table 6. *Summary Table for Text to Braille*

Author	Year	Title	Relevant Findings	Relationship to your study
S.Sultana, A. Rahman, F. Chowdhury et al.	2017	A novel Braille pad with dual text-to- Braille and Braille- to-text capabilities with an integrated LCD Display	The device can convert braille to text and convert text to braille. The converted signal will appear in an LCD Display.	Both studies will use LCD displays for displaying the output message.
Sruthi Ramachandran, Niju Rajan, Pallavi K N , Subashree J , Suchithra S, Sonal B	2021	Communication Device for the Visual and Hearing- Impaired Persons to Convert Braille Characters to English Text	Design a conversion between text to braille and braille to text using an LCD display, which can be a tool for having a keyboard as an input.	Both studies use LCD displays for the output. In the LCD display, there can also be a keyboard, which can be used as a communication device between others and

				blind individuals.
S. Hossain, A. Raied , A. Rahman et al.	2018	Text to Braille Scanner with Ultra Low-Cost Refreshable Braille Display	Develop a Text to Braille Scanner that uses a low-cost braille display. The scanner app can transform English text into braille in real time without the need for internet access.	Both studies will use text to braille process and also limited to English text that will convert into braille.
L. Arbes, J.Baybay, J. Turingan et al.	2019	Tagalog text-to-braille translator tactile story board with 3D printing	Created a system that will convert text to braille which can also create 3D models that can be used as storyboards for short stories or riddles.	Both studies will use text to braille process but instead of tactile storyboard, BeReal will use braille cells.

2.5.2 Speech to Braille

To avoid the mechanical human-to-machine interaction like keyboard, mouse, etc. There are Voice Recognition technologies that have been implemented to make it easier for the disabled people, especially blind people, to use a certain appliance or gadget [25]. One type of natural communication is speech recognition that allows the computer or system to have an interaction with humans. In [7], the researchers created a braille device which can recognize more than 10 words through the help of an Acoustic model and HM2007 IC and will eventually convert the detected words into braille form using solenoid drivers. In the project that [22] has developed, it focuses on providing a platform for the especially abled (blind and dumb). It has a Braille keyboard as an input and an LCD as an output. It allows ordinary people to understand the Braille script that the user is typing. At the same time, it offers voice output through the speaker with the help of an SD card-based voice module. While in [6], the researchers created a software which converts the speech into text and at runtime, it shows the text on the screen and turns it to braille. It uses a low-cost dot matrix printer to print the text that is converted into braille form. To recognize the speech, the researchers used Web Kit Speech Recognition since it is an open-source tool kit and with 95% accuracy, this is the most effective recognition tool.

Table 7. *Summary Table for Speech to Braille*

Author	Year	Title	Relevant Findings	Relationship to your study
L. Triyono, T. Yudanto, S. Sukamto et al.	2021	VeRO: Smart home assistant for blind with voice recognition	Developed a technique for turning on and off electronic devices using voice or speech recognition.	Both studies will use a microphone as a sensor to take voice or speech input.
V. Devi	2017	Conversion of Speech to Braille: Interaction device for Visual and Hearing Impaired	Created a braille device which can recognize more than 10 words through the help of an Acoustic model and HM2007 IC for the visual and hearing impaired	Both studies will create a device that can convert speech into braille output and Raspberry Pi as the single-board computer.
J. Pradeepkandhasamy, A. Priya,P. Chellappan	2020	Voice O Braille	Created a software which converts the speech into text and at run time, it shows that as text on the	Both studies will implement voice to text conversion

			screen and translates it to braille.	and convert it into braille form.

2.5.3 Image To Text to Braille

The Camera-Based mechanism helps the visually impaired and blind persons in everyday reading and this method serves as their eye [30]. Optical Character Recognition (OCR) is a technique used to translate a text in a scanned image of printed, handwritten, or typed documents into machine-encoded text which the output can be manipulated by a computer [28]. In [29], the researchers tested the use of Optical Braille character recognition (OBCR) process using the artificial neural networks (ANN) in converting the identified letters in scanned images in Cyrillic alphabet into Braille representation system to fill the gaps between the visually impaired, blind, and ability-sighted people in communicating. There were 33 photos for the training experiment and 8 photos for the testing per test word experiment. The results for training image and testing image experiment were 97.1% and 95% accuracy respectively. In [27], the researchers created an Automatic Object Detection for the visually impaired using Smart Glass which is connected to the smartphone via Bluetooth. In this way, the user can just capture an image using the controller (smart phone) and the image detected will be converted into braille data which can now recognize by the user the form of the object with the tactile picture produced through the braille pad.

In [36], the researchers created a system where the OCR engine is utilized by capturing an image which will undergo through image processing using Tesseract and converting the detected text into braille form which is represented through braille cells. The system has an overall reliability of 95.68% by testing 180 in different fonts.

Table 8. *Summary Table for Image to Braille*

Author	Year	Title	Relevant Findings	Relationship to your study
S. Kumar, S. Mathew, N. Anumula et al.	2020	Portable Camera- Based Assistive Device for Real- Time text Recognition on Various Products and Speech Using Android for Blind People	Created a camera-based system in which the image's text is retrieved and fed into a speaker, which outputs audio.	Both studies will use a sensor which is a camera to assist the blind person from getting an input and convert it into text into braille.
K. Smelyakov,	2018	Braille Character Recognition Based	Used an Optical Braille character recognition	Both studies will use an Image to text

A. Chupryna, D. Yeremenko et al.		on Neural Networks	(OBCR) system using artificial neural networks (ANN) in converting the identified letters in scanned images in Cyrillic alphabet into Braille representation system.	process but BeReal will use another algorithm to convert scanned image into braille form.
D. Lee & J. Cho	2020	Automatic Object Detection Algorithm-Based Braille Image Generation System for the Recognition of Real-Life Obstacles for Visually Impaired	Created an Automatic Object Detection for the visually impaired which the user can just capture an image using the controller(smartphone) and the image detected will be converted into braille data	Both studies will use the Image to Braille process by scanning or capturing an image.

		People		
J. Dela Cruz, J. Ebreo, R. Inovejas et al.	2017	Development of a text to braille interpreter for printed documents through optical image processing	Created a system that will capture an image and detect the text via Tesseract OCR and convert it into braille form.	Both studies will use image to text process but instead of using OCR, BeReal will use Image Classification using Tenserflow.

Chapter 3

METHODOLOGY

3.1 Research Design

Descriptive research design was implemented in this study. Descriptive statistics is the presentation, collection, description, analysis, and interpretation of data using graphical and numerical techniques. Its purpose is to summarize the data collection from a set of values which can be described by any data set, either a population or a sample, and set the conclusion on it [34]. This allowed the researchers to gather data from the surveys answered by the user target or respondents and make a conclusion out of it. This can be used for future research.

Developmental research design was also implemented in this study. Developmental research design, which is especially significant in the discipline of educational technology, can be described as the systematic study of planning, producing, and assessing processes, products, and programs that must meet efficiency and consistent requirements. It involves a product development process in which the final product is being evaluated and analyzed [35]. This enabled the researchers to develop and produce a Braille Device System for the Blind.

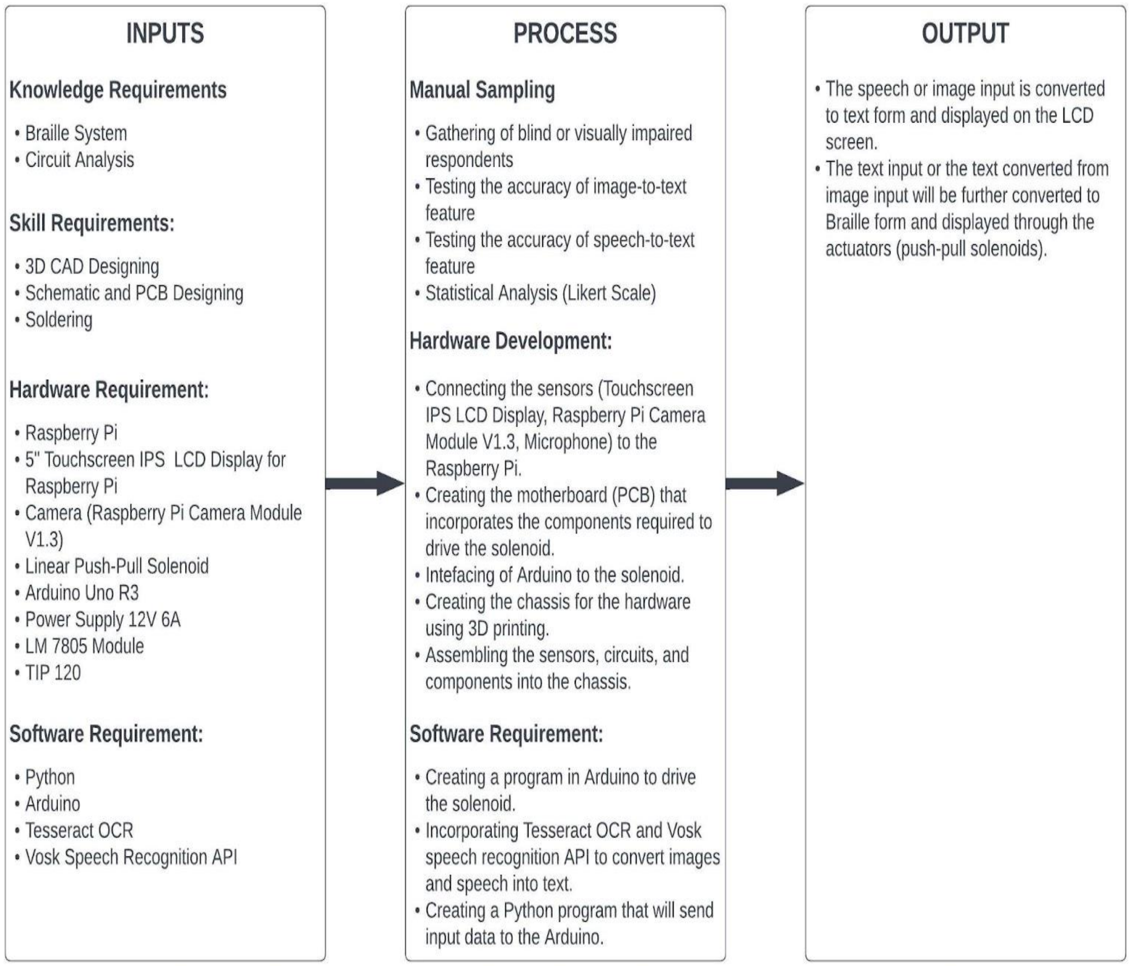


Figure 1. *Conceptual Framework of the System*

Figure 1 illustrates the Input-Process-Output of the device. The input focused on the gathering of materials such as the sensors and microcontroller, including the software requirements. The data accumulated was delivered to the microcontroller for the execution of a suitable machine learning algorithm based on the input. Subsequently, the processed data converted into text form was shown in the LCD display for confirmation, and the text form converted into braille form was shown in the actuators.

3.2 Research Process Flow

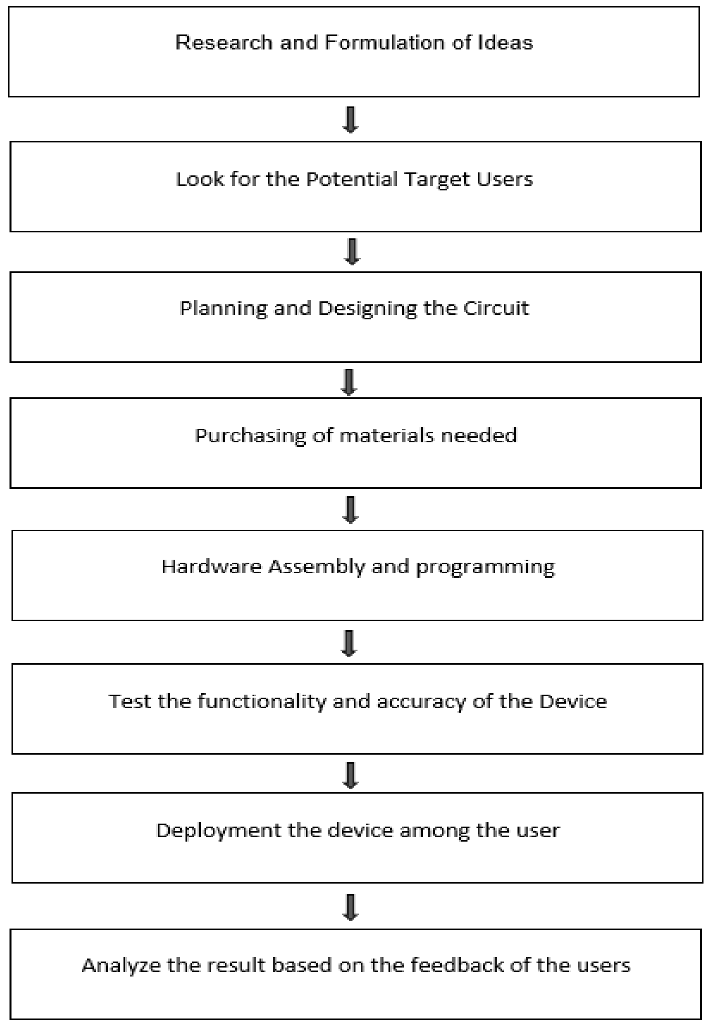


Figure 2. *The flow process of the Study*

Figure 2 depicts the flow of the study. The first step was to research and formulate the ideas. This step focused on formulating a problem statement, which could then be answered by establishing the study's objective. The next step was to look for the potential target users. In this step, the researchers decided to conduct a survey in P. Gomez Elementary School and Bagong Silang Elementary School. The third step was planning and designing the circuit. This included canvassing and planning for the construction of the prototype and circuit needed. Then, the materials needed for the building of the prototype were purchased. The next process was building the hardware and programming the software for the conversion of the three conversions: image to braille, text to braille, and speech to braille separately. After that, the functionality and accuracy of the device were tested. Then, the device was deployed among the users. In this way, the researchers could analyze and assess the compatibility of the device with the potential users. Lastly, this step was to analyze the results based on the feedback of the users. If there were errors and a lack of results after the testing, the researchers could develop the device to be more suitable for the users. After completing the results gathered, the objective of the study was completed.

3.3 Designing and construction of seven braille characters that utilized an electromagnetic push-pull solenoid as braille cells in developing a low-cost braille device that incorporates touch, speech, and an image sensor for data inputs.

3.3.1 Creation of a low-cost hardware device using push-pull solenoid actuators.

The Push-Pull Solenoid Actuators is a design that is mainly used for braille system devices. It is an electronic Braille character with a magnetic retention mechanism that has very few parts and is simple to replicate, alter, or scale up for manufacturing. Because of its low-cost production, this was used as a braille character in the braille device and could be customized according to the user preferences.

3.3.1.1 Purchasing the push-pull solenoid actuators for braille cells of the device.

The ready-to-use push-pull solenoid actuators were purchased by the researchers from online stores. A total of 42 solenoids were acquired by the researchers to create a seven (7) character display, which was used for the device itself.

3.3.1.2 Determining and purchasing the image, sound, and touch sensors for the device.

The camera was determined by its quality and its used voltage. The proposed camera for the device was the Raspberry Pi Camera Module V.13. The microphone was also chosen based on its quality and suitability for the device. The LED display or the touch sensor selected was the 5" Touchscreen IPS LCD Display for Raspberry Pi. Its Resistive touch control and support for any Raspberry Pi revision made it suitable for the braille device.

3.3.1.3 Designing of the Braille Device

The researchers' proposed design was a small rectangular gadget with an LCD screen, braille characters, buttons, and a microphone on top. The camera was located in the front of the device, while the remaining components were contained within. Its measure was approximately 12 inches long, 10 inches wide, and 5 inches tall. The size and portability of the device would allow the user to transport it anywhere, and its components and functionality were appropriately sized.

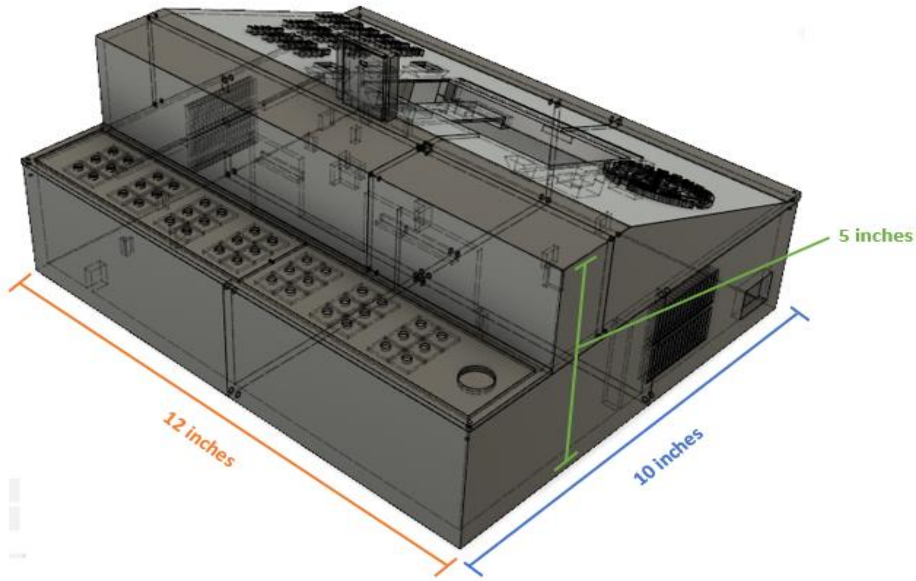


Figure 3. *Rear Side View of the Whole Device with a Dimension of 12x10x5 inches*

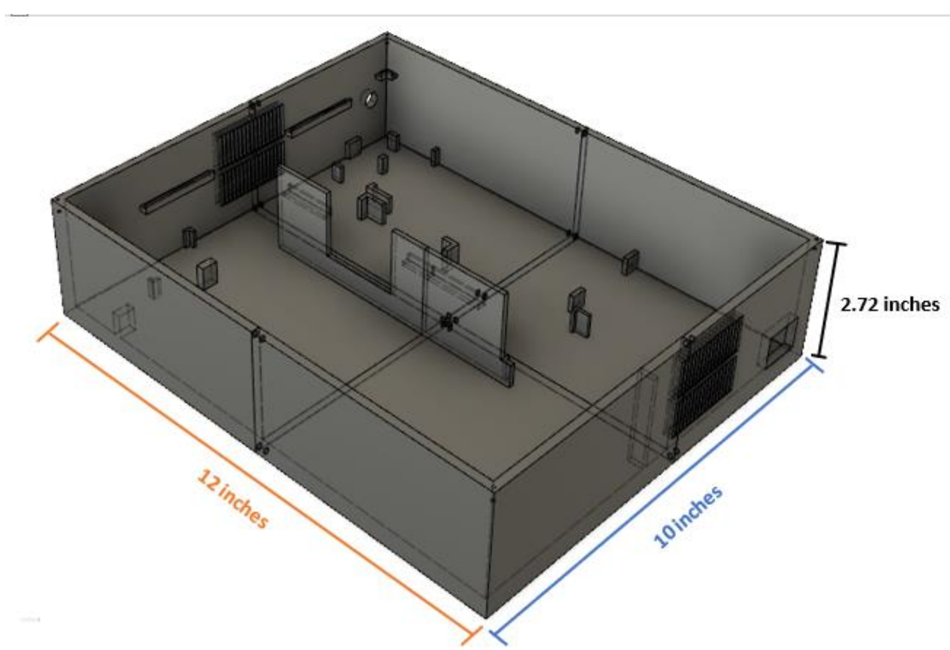


Figure 4. *Rear Side View for the Enclosure of the Chassis*

The enclosure of the chassis housed the components such as the Raspberry Pi, power supply, PCB, Arduino, and wirings. The enclosure itself measured 12 inches long, 10 inches wide, and 2.72 inches tall.

3.3.1.4 Construction of the device using push-pull solenoid actuators as braille cells and integrating the image, sound, and touch sensors

Setting up or constructing the device was the initial step, followed by testing the device's primary components and features. The incorporation of the components worked to the advantage of the proposed design and the functionality it possessed. After assembling all the necessary parts and accessories, it was time to check if the machine would function.

3.4 Designing and developing of a software application with a user-friendly Graphical User Interface (GUI) that integrates the text-to-braille, image-to-braille, and speech-to-braille functions all associated with the push buttons.

3.4.1 Construction of Graphical User Interface (GUI)

The application was written in the Python programming language. A graphical user interface (GUI) was built to incorporate the text-to-braille, speech-to-text, and image-to-braille functionalities. The text-to-braille and image-to-braille features were both connected to the Arduino program to control the solenoids. The speech-to-text function was used to convert speech input and display it on the screen. The microphone used was directly connected to the Audio Jack of the Raspberry Pi 4 Model B.

3.5 Developing a program that converts image-to-text using Tesseract OCR and speech-to-text using the Vosk API into a single program

3.5.1 Implementation of Tesseract for OCR

The optical character recognition engine Tesseract is available for multiple operating systems. It is open-source software distributed under the Apache License. Tesseract is capable of recognizing more than 100 languages and supports various image formats, including PNG, JPEG, and TIFF. It also supports Unicode (UTF-8) and different image formats. Due to its free availability and OCR capabilities, Tesseract was utilized as the algorithm for converting images into text.

3.5.2 Implementation of Vosk for Speech Recognition

Vosk is a speech-to-text software developed by Alpha Cephei. It is a voice recognition toolkit that is compatible with over 20 different languages and varieties of speech, including English, German, French, Spanish, Portuguese, Chinese, Russian, Turkish, Vietnamese, and more. The library provides ready-to-use speech recognition for various platforms, including mobile applications and Raspberry Pi. It includes accurate model sets, scripts, and processes. The researchers utilized this technology to convert speech into text at a faster rate, with lower cost, and increased accuracy.

3.5.3 Implementation of Tesseract and Vosk into a Single Program

The researchers created an application that was compatible with Raspberry Pi's operating system to combine Tesseract and Vosk API into a single software.

3.5.4 Text to Braille Conversion using Arduino.

The process of converting text to braille was carried out with the assistance of an Arduino. After completing the processing, the output of the device was in the form of braille, and the solenoid actuators were used to display it.

3.6 Assessment of the device functionality by actual testing it by blind people to deaf, mute, and unimpaired people.

3.6.1 Making of the survey in Likert scale form

In order to conduct the survey with the users, the researchers made use of a Likert scale form. Each individual chosen to participate received a questionnaire.

3.6.2 Gathering of the possible respondents for the survey

The evaluation and assessment of the device's functionality were conducted at P. Gomez Elementary School and Bagong Silang Elementary School. Both schools were urban public elementary schools managed by the Department of Education (DepED) and located at 1224 Pedro Guevarra St, Santa Cruz, Manila, 1000 Metro Manila, and Ph 1 Pkg 1 Langit Road Corner Old Zabarte Road, 176, Caloocan, 1428 Metro Manila, respectively.

3.6.3 Evaluating and assessing the device's functionality by actual usage

The blind individual communicated with deaf and/or mute people by allowing them to use the proposed device to test the accuracy of the results and determine if the communication could be successfully delivered.

3.6.4 Providing the respondents, the survey form

The questionnaire was distributed to the individuals who used the proposed device to evaluate their experience and assess the effectiveness of the communication process.

3.6.5 Gathering the completed survey form

Lastly, the researchers collected all the survey forms from the respondents after the questionnaire had been answered. They then analyzed and interpreted the responses.

3.7 Statistical Analysis

In this research study, the Likert scale was used as a statistical tool to gauge respondents' feelings by asking how strongly they were in agreement or disagreement with a given problem or statement. A Likert scale is a commonly used instrument for evaluating attitudes, knowledge, perceptions, values, and behavioral changes. A Likert-type scale is made up of a series of statements from which respondents can choose to rate their answers to evaluative questions. Because each user had a unique viewpoint about the product or device, researchers developed a weighted evaluation matrix to enable them to conduct a cost-benefit analysis in order to better meet the needs of all parties concerned. The researchers provided a questionnaire to be answered by the selected participants, and the combination of the results was carefully evaluated to provide effective solutions and improvements for future works.

3.8 Project Workplan

Table 9. *Gantt Chart of the Study*

BeReal Project Workplan													
ACTIVITIES		M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12
OBJECTIVE 1: Design and construct seven braille characters that utilized an electromagnetic push-pull solenoid as braille cells in developing a low-cost braille device that incorporates touch, speech, and an image sensor for data inputs.	Purchasing the push-pull solenoid actuators for braille cells of the device												
	Determining and purchasing the image, sound, and touch sensor for the device												
	Designing of the Braille Device												
	Construction of the device using push-pull solenoid actuators as braille cells and integrating the												
OBJECTIVE 2: Design and develop a software application with a user-friendly Graphical User Interface (GUI) that integrates the text-to-braille, image-to-braille, and speech-to-braille functions all associated with the push buttons.	Construction of Graphical User Interface (GUI)												
OBJECTIVE 3: Develop a program that converts image-to-text using Tesseract OCR and speech-to-text using the Vosk API into a single program.	Implementation of Tesseract for OCR												
	Implementation of Vosk for speech recognition												
	Integration of Tesseract and Vosk into a single program												
	Text to braille conversion using Arduino												
OBJECTIVE 4: Evaluation and Assessment of the device funcationality by actual testing it by blind, deaf and mute people.	Making of the survey in Likert scale form												
	Gathering of the possible respondents for the survey												
	Evaluating and assessing the device's functionality by actual usage												
	Providing the respondents, the survey form												
	Gathering the completed survey form												

Chapter 4

RESULTS AND DISCUSSIONS

This chapter contains the project's technical description, structural design, and findings analysis based on the tests performed.

4.1 Project Technical Description

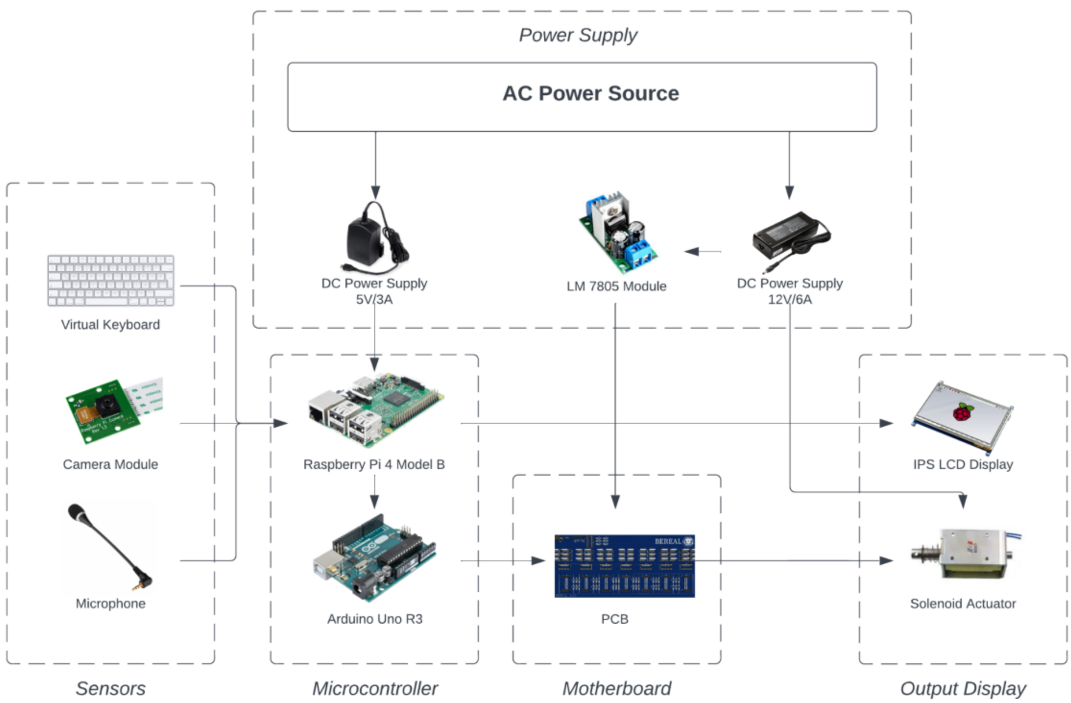


Figure 5. *Block diagram of the Study*

The overall system was depicted by the block diagram presented in Figure 5. The hardware part consisted of input sensors responsible for gathering input data through the virtual keyboard in IPS LCD Display, Raspberry Pi camera v1.3, and microphone. This data was processed by the software using Python language, Tesseract, and Vosk, respectively.

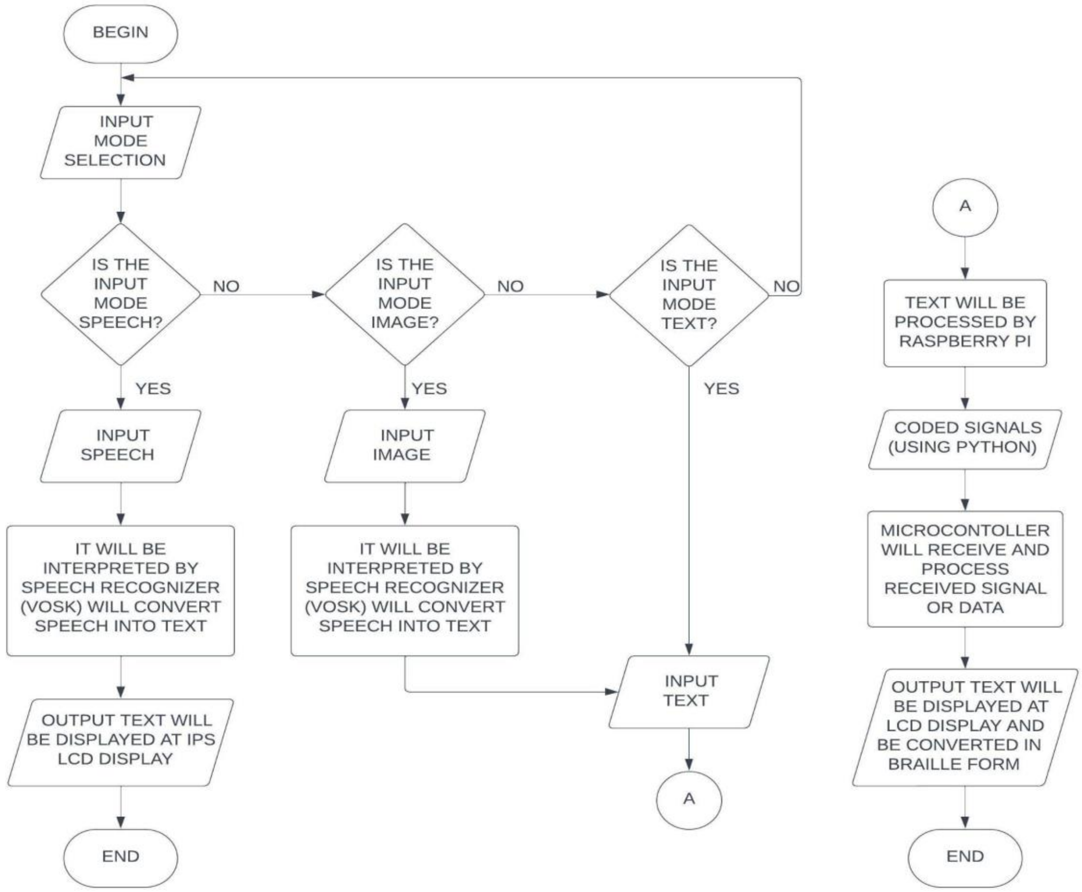


Figure 6. *Flowchart of the system*

The entire system was represented by the flowchart illustrated in Figure 6. It began with the input mode selection, associated with push buttons to choose between speech, image, and text input modes. If the Speech option was selected, it was interpreted by a speech recognizer (Vosk) to convert the speech into text, which was then displayed on an IPS LCD Display. If the Text option was selected, it was processed by Python to transfer the data to the Arduino Uno microcontroller through serial communication. Lastly, if the Image option was selected, it was processed by Tesseract for OCR Extraction to extract the text from the input image, and the text was then converted into braille characters.

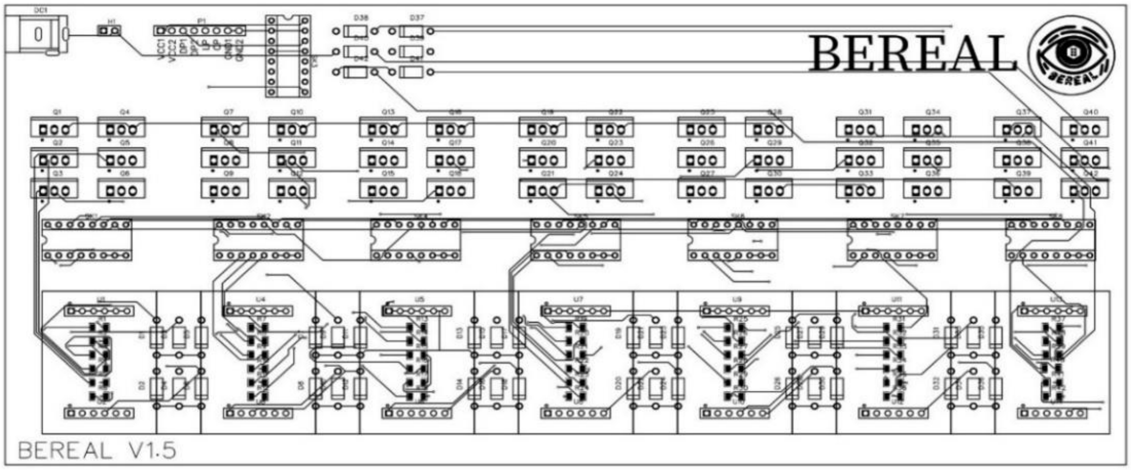


Figure 7. PCB Layout

This figure illustrates the printed circuit board (PCB) of the device. It represents the PCB layout corresponding to the schematic diagram shown in Figure 8.

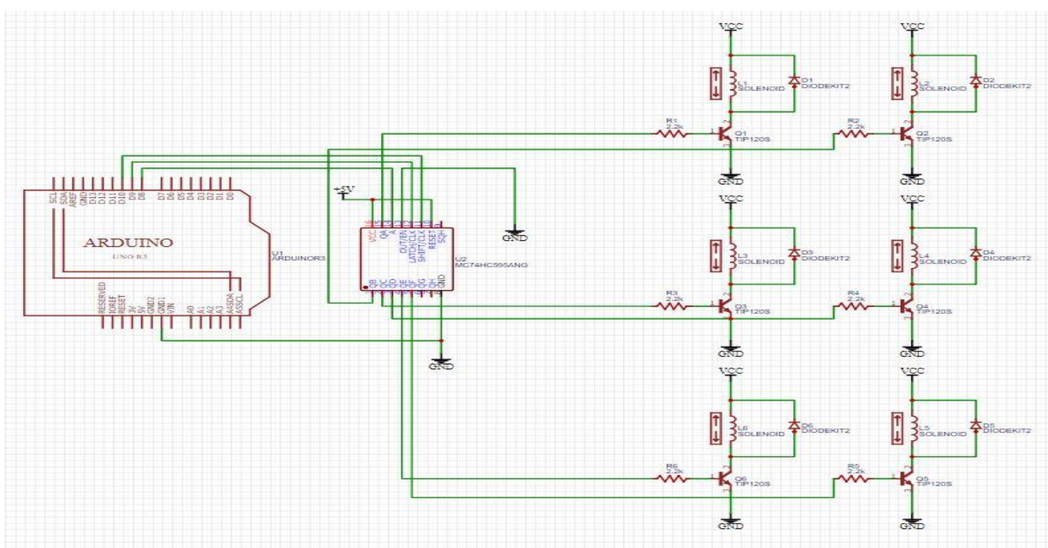


Figure 8. *Schematic diagram of each braille character*

This figure illustrates the connection between the Arduino, shift register, transistor, and solenoid. It shows the connection for 6 braille cells.

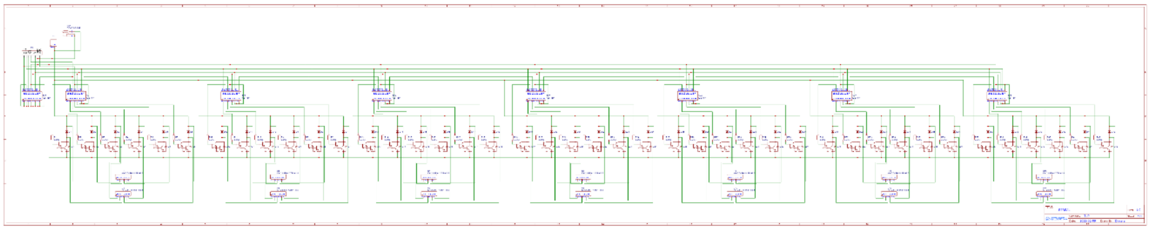


Figure 9. Schematic Diagram of the Whole Braille of the Whole Braille Character

Figure 10 displays the schematic diagram of the device, which incorporates the combination of all seven characters into a single circuit.

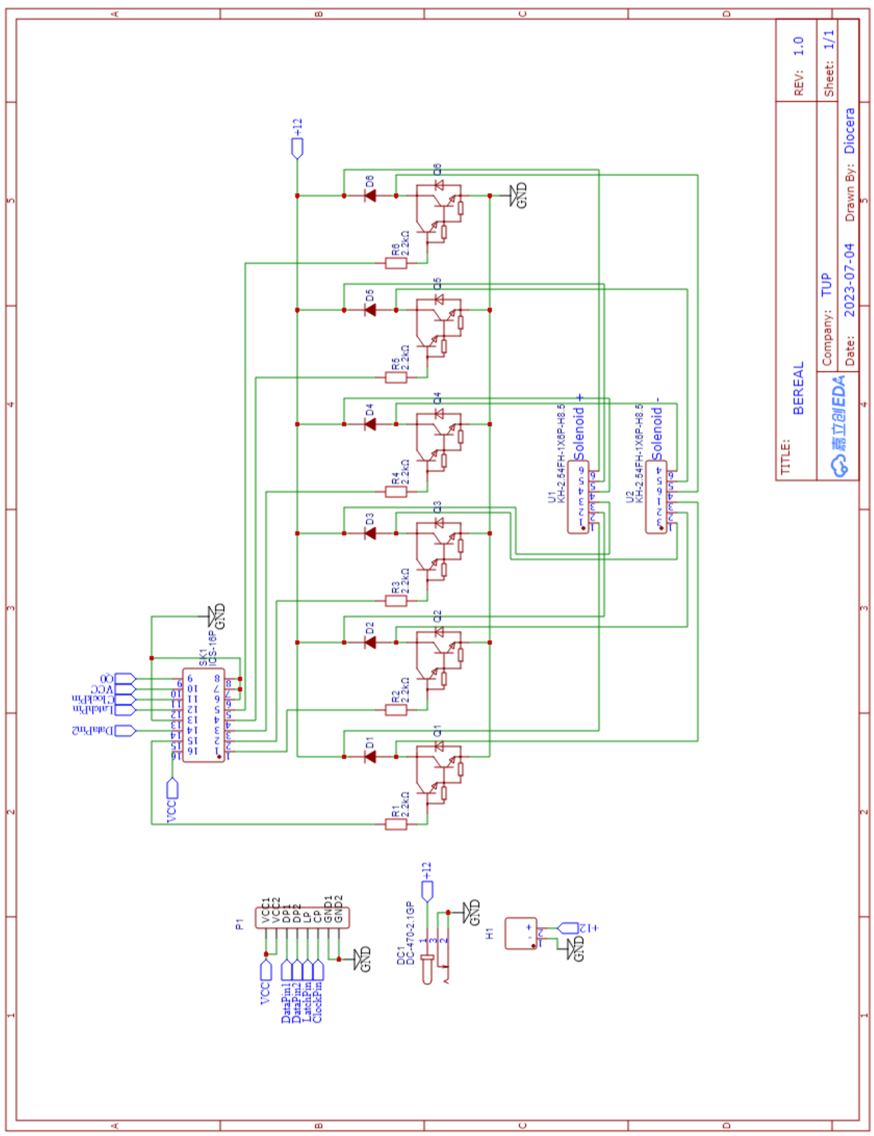


Figure 10.1. *Schematic of the 1st character*

Figure 10.1 shows the detailed connection of components within each character. In Figure (on sheet 1), the 1x8 pin header consists of 8 pins, each serving a specific purpose. The VCC 1 and 2 pins are connected to the LM 7805 module, supplying a positive 5V. The DP2 pin is connected to pin 8 of the Arduino. The LP pin and the CP pin are connected to pins 9 and 10 of the Arduino, respectively. GND 1 and 2 pins are connected to the ground. Additionally, the DC barrel jack and the 1x2 pin header are connected to the positive 12V. For the TIP120 transistor, the anode of the diode and the negative side of the solenoid are connected to the collector. The output pin of the shift register is connected in series with a 2.2k ohm resistor to the base of the transistor. The emitter is connected to ground. Regarding the shift register, pin 1, 2, 3, 4, 5, and 15 are connected to the bases of the transistors. Pin 16 and pin 10 are connected to VCC, pin 14 is connected to DataPin2, pin 12 is connected to LatchPin, pin 11 is connected to ClockPin, and pin 6, 7, 8, and 13 are connected to ground.

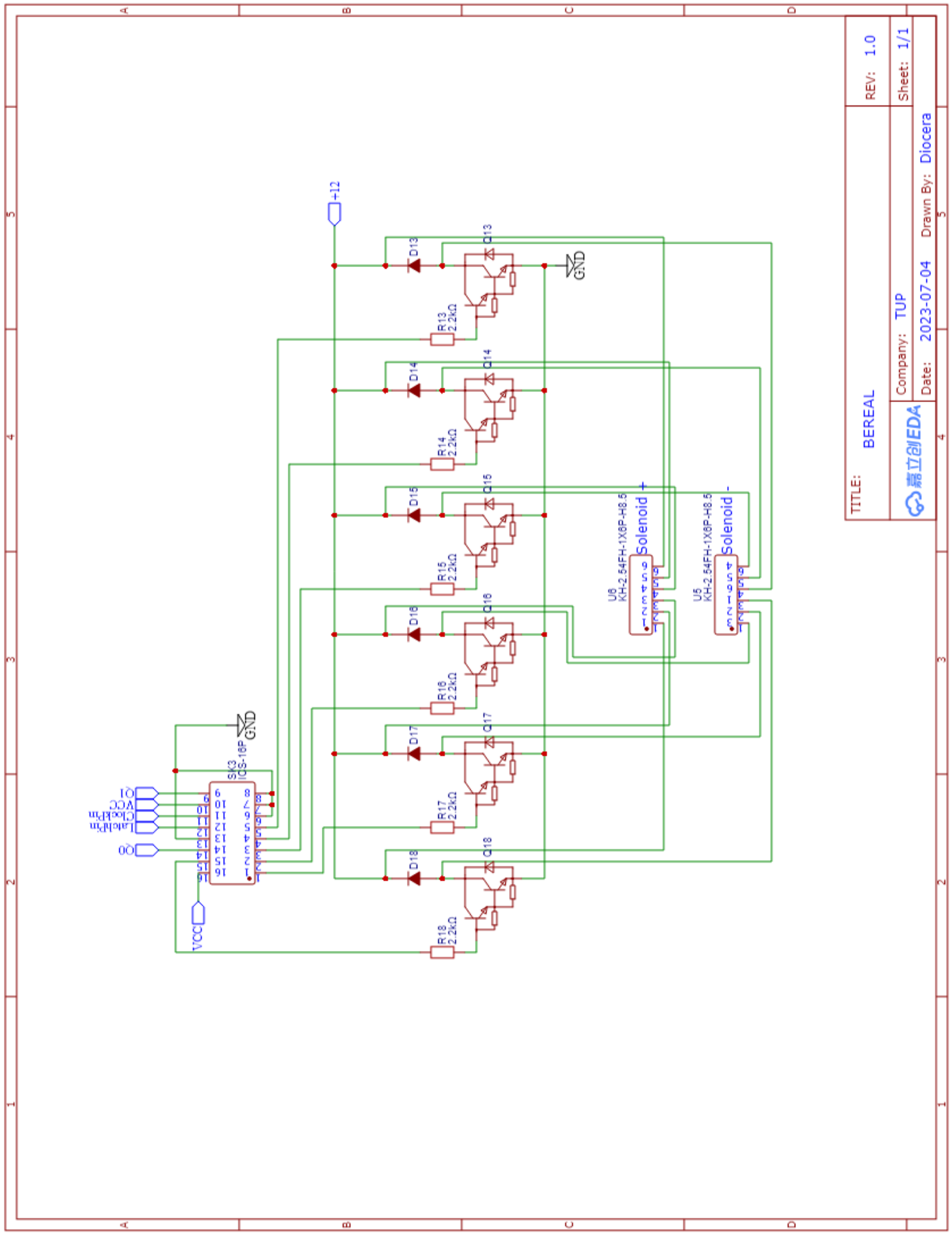


Figure 10.2. *Schematic* of *the* 2nd *character*

Figure 9.2 shows the same connections for the shift register and transistor. However, there are two changes in the connections. Pin 14 is now connected to Q0 instead of DataPin2, and pin 9 is connected to Q1 instead of Q0.

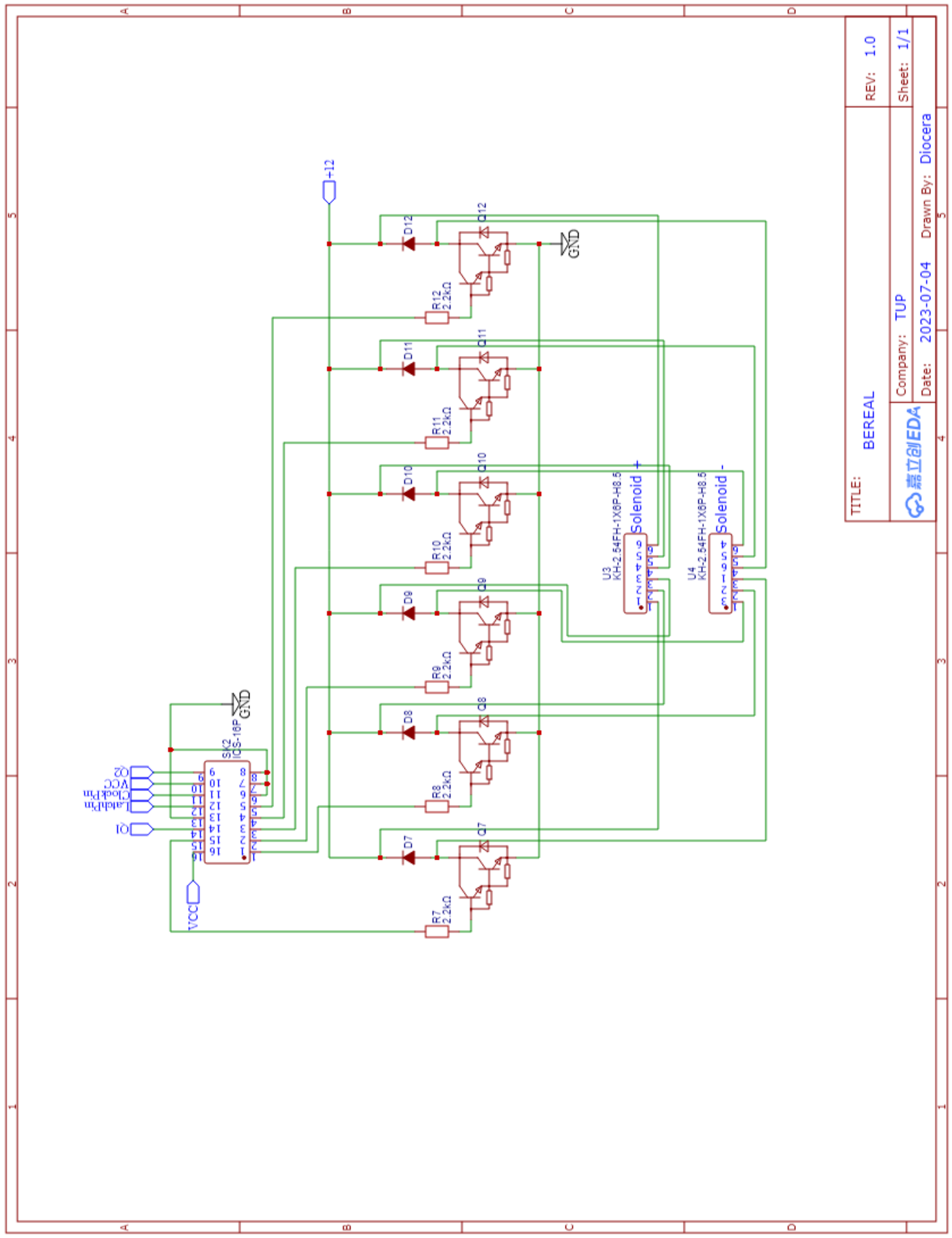


Figure 10.3. *Schematic of the 3rd character*

Figure 10.3 shows the same connections for the shift register and transistor. However, there are two changes in the connections. Pin 14 is now connected to Q1 instead of Q0, and pin 9 is connected to Q2 instead of Q1.

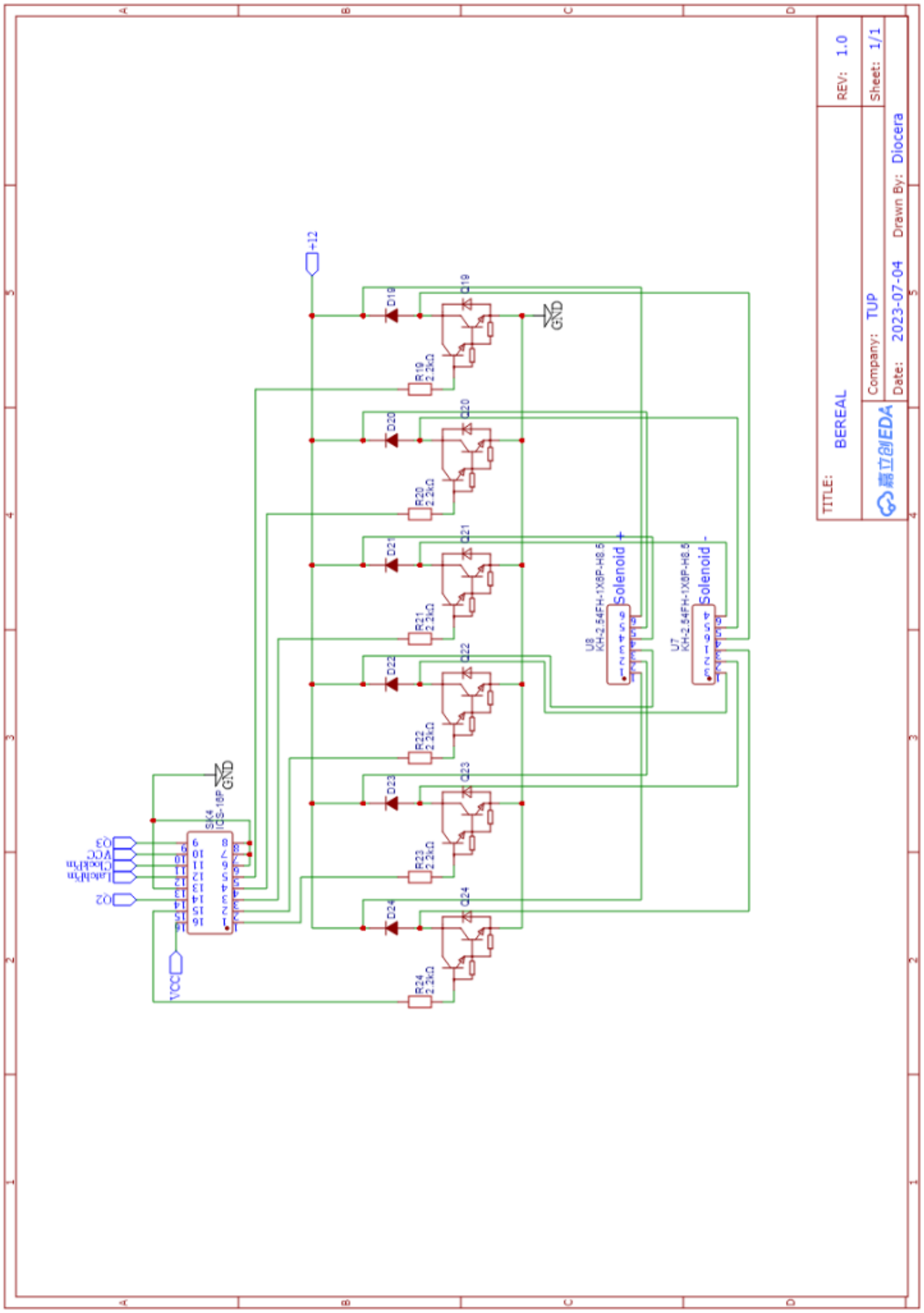


Figure 10.4. *Schematic of the 4th character*

Figure 10.4 shows the same connections for the shift register and transistor. However, there are two changes in the connections. Pin 14 is now connected to Q2 instead of Q1, and pin 9 is connected to Q3 instead of Q2.

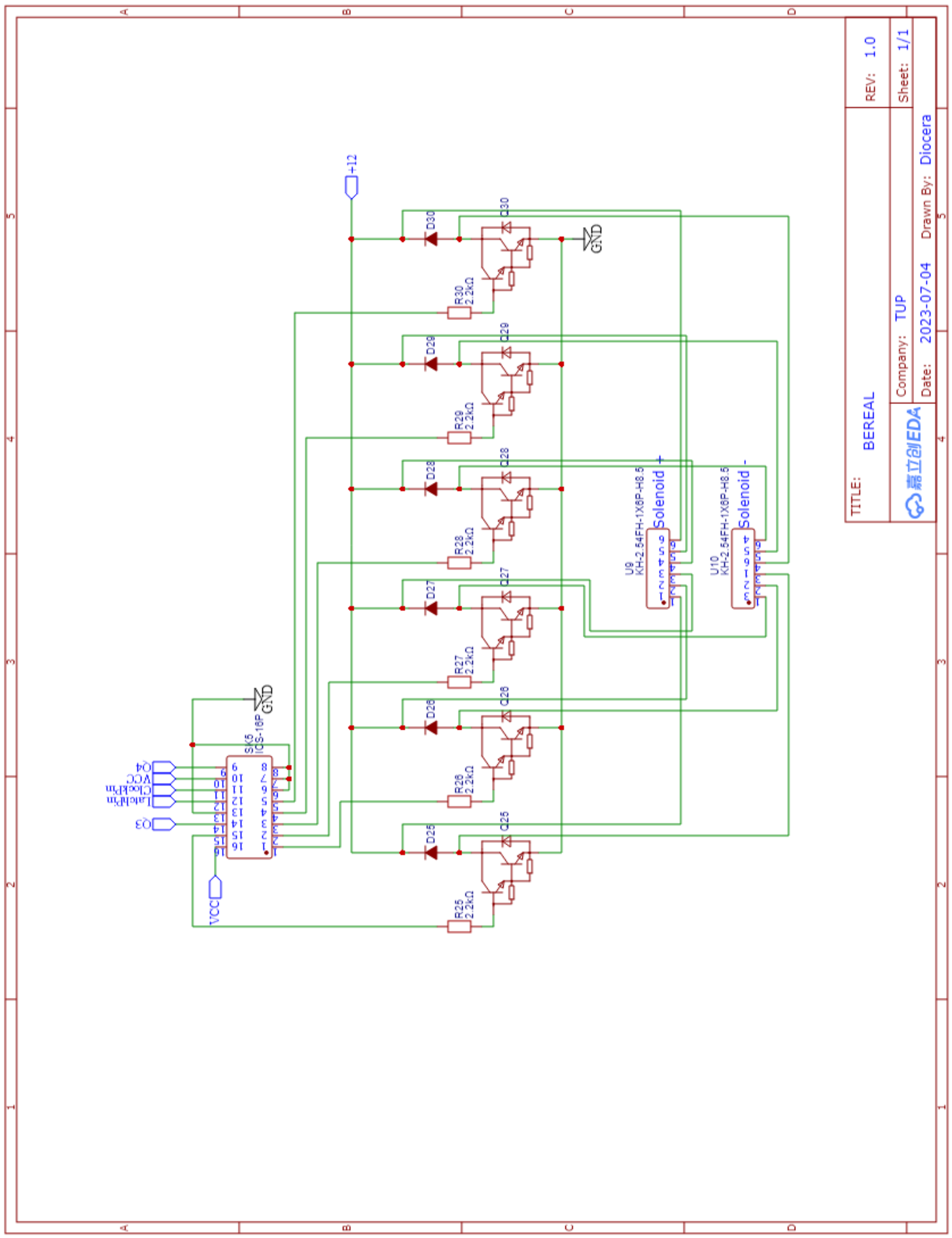


Figure 10.5. *Schematic of the 5th character*

Figure 9.5 shows the same connections for the shift register and transistor. However, there are two changes in the connections. Pin 14 is now connected to Q3 instead of Q2, and pin 9 is connected to Q4 instead of Q3.

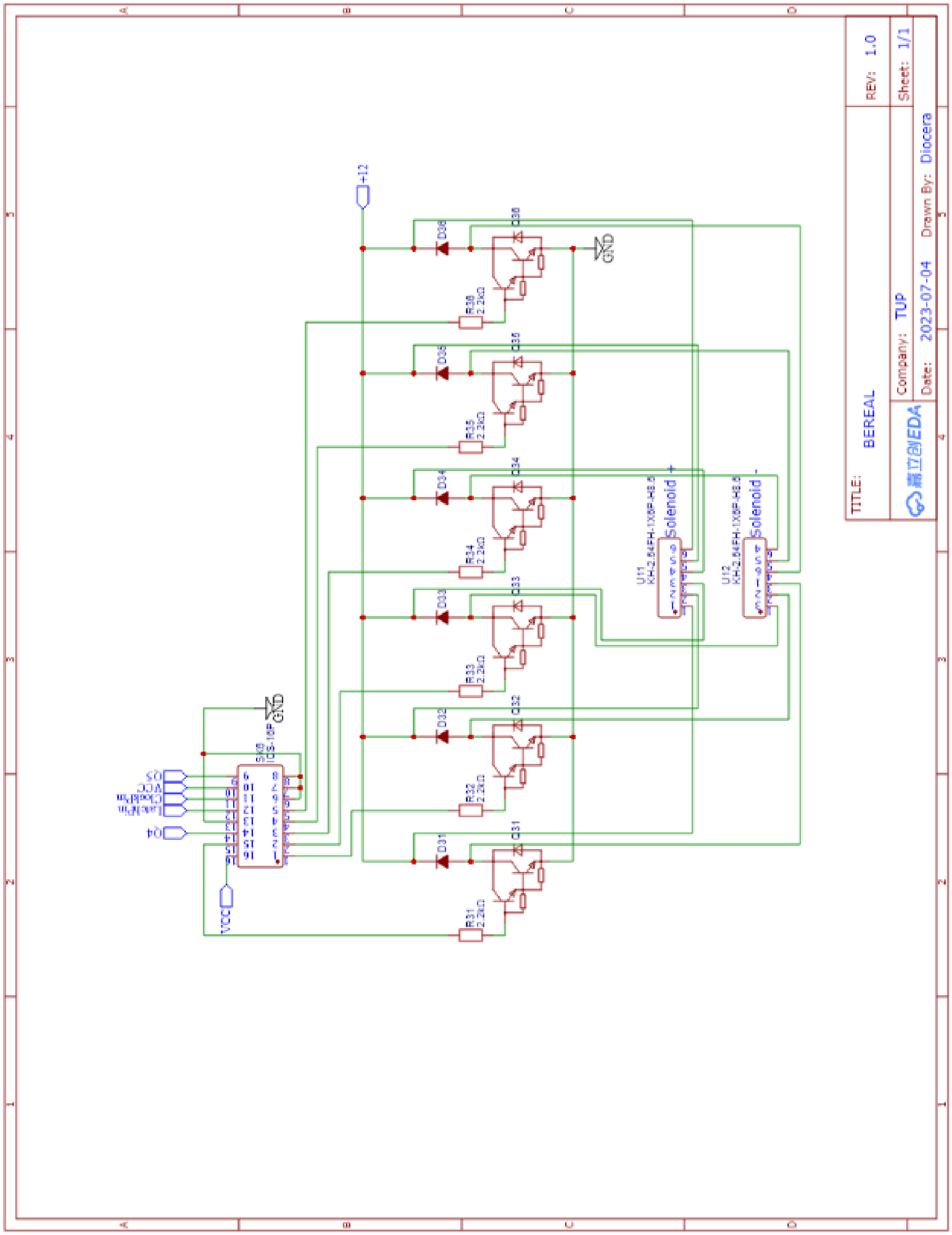


Figure 10.6. *Schematic of the 6th character*

Figure 10.6 shows the same connections for the shift register and transistor. However, there are two changes in the connections. Pin 14 is now connected to Q3 instead of Q2, and pin 9 is connected to Q4 instead of Q3.

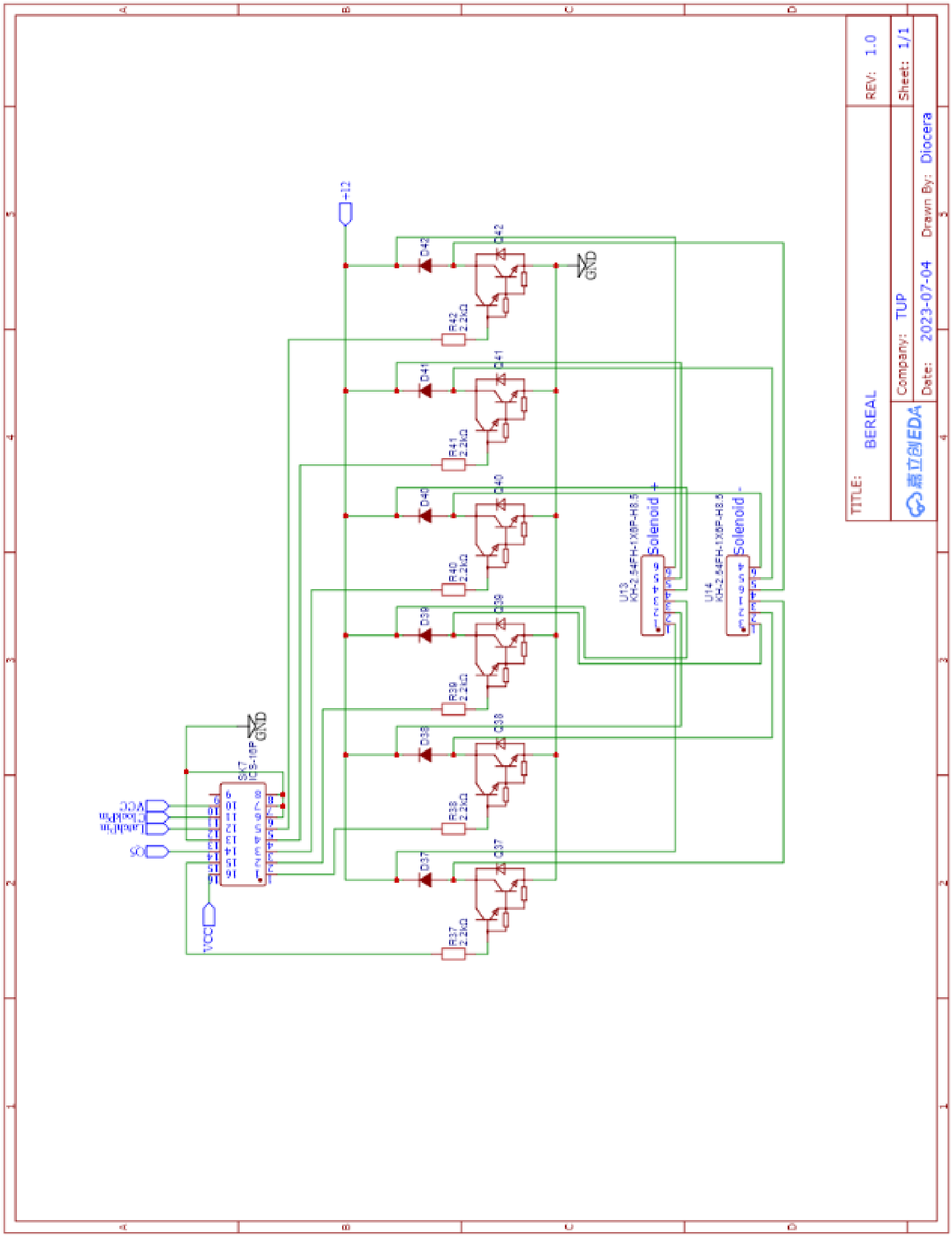


Figure 10.7. *Schematic of the 7th character*

Figure 10.7 shows the same connections for the shift register and transistor. However, there are two changes in the connections. Pin 14 is now connected to Q5 instead of Q3, and pin 9 is connected to ground instead of Q4.

4.1.1 Screenshots of the Software

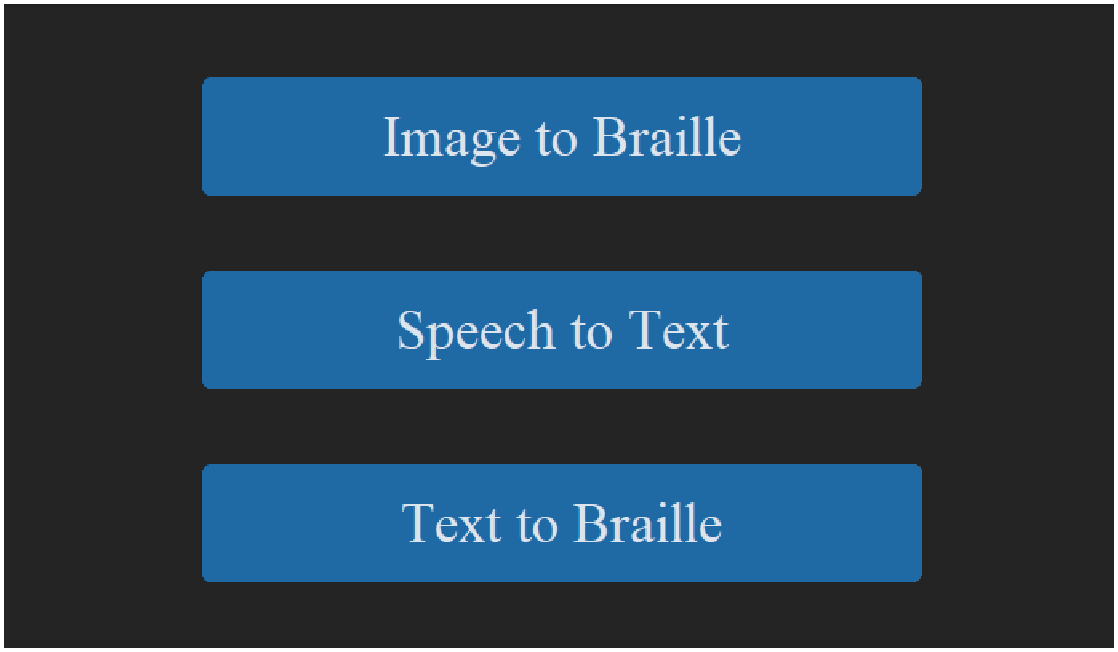


Figure 10. *Initial window for input mode selection*

Figure 10 illustrates the initial window for input mode selection of the system after booting it up. The initial window features three input mode options: image-to-braille, speech-to-braille, and text-to-braille.

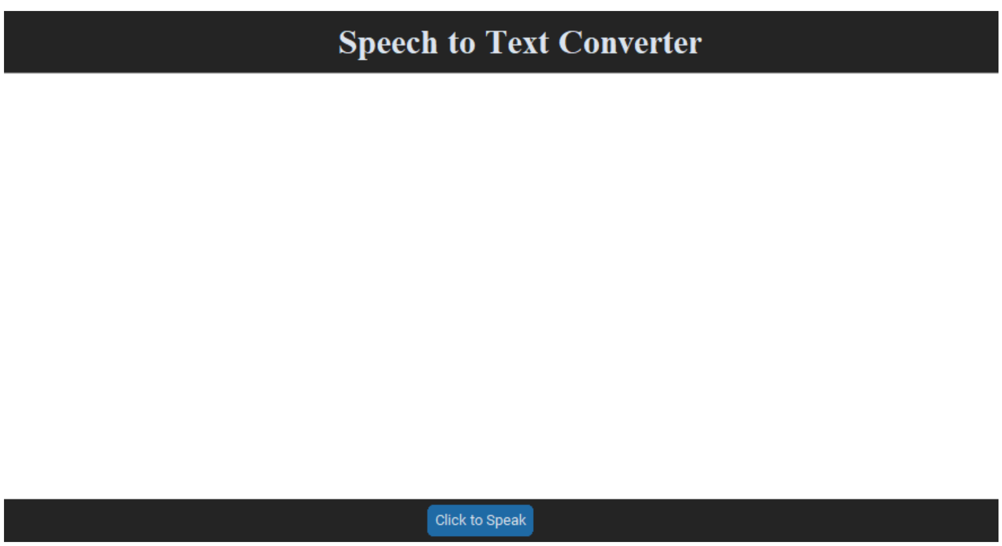


Figure 11. *GUI for Speech-to-Text Converter*

This figure depicts the graphical user interface of the speech-to-text converter. By pressing the "Speech to Text" button on the LCD screen or by pressing the red push button, the user will be redirected to the graphical user interface of the speech-to-text converter.

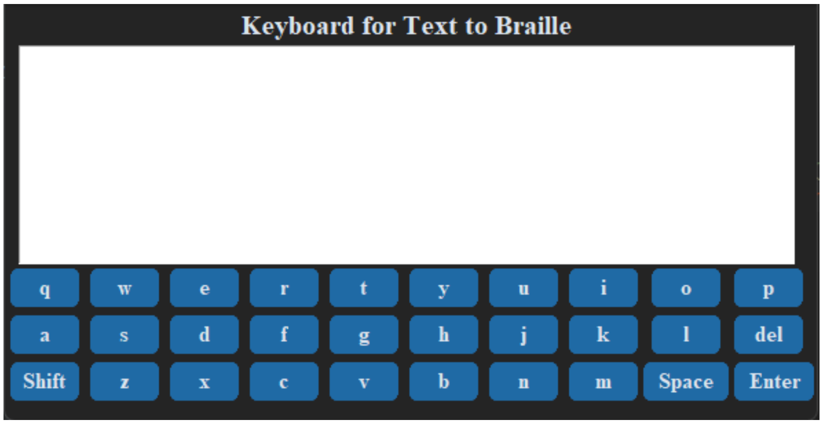


Figure 12. *GUI of Keyboard for Text to Braille*

This figure shows the graphical user interface of the text-to-braille feature. By pressing the "Text to Braille" button on the LCD screen or by pressing the blue push button, the user will be redirected to the graphical user interface of the text-to-braille converter. It features its own virtual keyboard for typing text.

Figure 13. *GUI for Image-to-Text*

This figure shows the graphical user interface of the image-to-braille feature. By pressing the "Image to Braille" button on the LCD screen or by pressing the white push button, the user will be redirected to the graphical user interface of the image-to-braille converter.

4.2 Project Limitations and Capabilities

The system's functionality was restricted to converting English inputs exclusively into Braille form. Specifically, the system employed the Grade 1 Braille system, which consisted of 3x2 dots and was designed to convert only the English alphabet into its corresponding Braille representation. One limitation of the system was its ability to display only seven-character words on the Braille display.

When utilizing the speech-to-text conversion feature, it was important to ensure clear diction and pronunciation of the words to achieve a high accuracy rate. For image detection, the image needed to be positioned approximately three to nine inches away from the camera. Additionally, the text within the image should not include numerals or cursive letters.

Furthermore, the text input feature was presently incapable of accepting numeric inputs. It is important to note that the solenoids had a restricted uptime, resulting in a shorter duration for reading the Braille output.

4.3 Project Structural Design

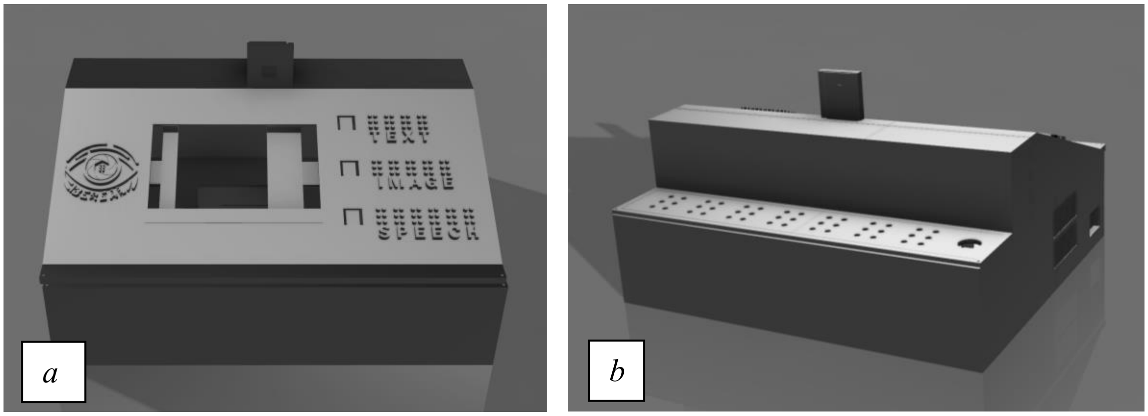


Figure 14. *3D Design of the Device: a.) Rear side view b.) Front view*

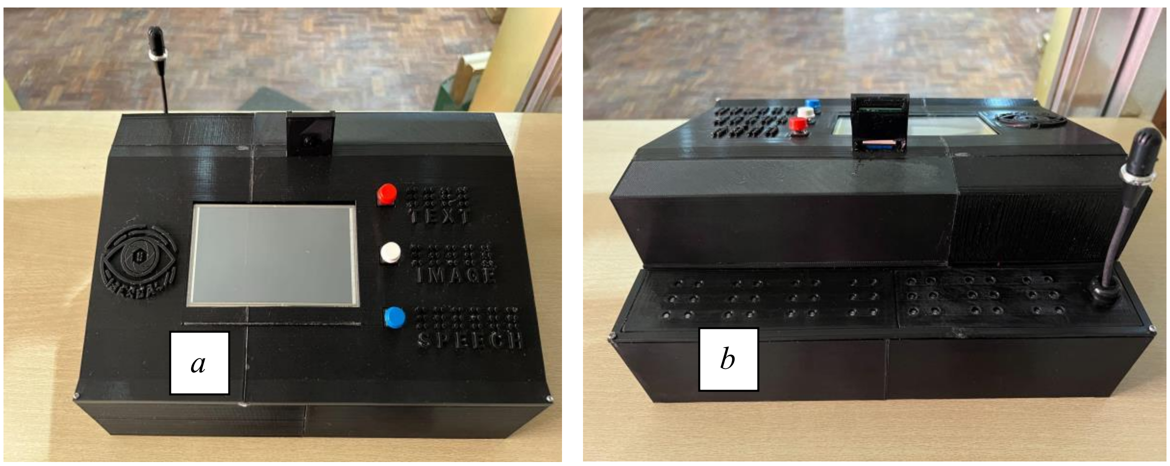


Figure 15. *Actual Prototype: a.) Back view b.) Front view*

Figure 15 showed the 3D design of the device casing, while figure 4.3.2 represented an actual prototype. The device had dimensions of 12x10x5 inches and was divided into two parts: the enclosure (as shown in figure 4.3d) and the top part (as shown in figure 4.3a). The enclosure part housed the Raspberry Pi 4b, Arduino Uno Microcontroller, and PCB. The top part was composed of the LCD Display, Solenoid Actuators, Microphone, Camera, and Pushbuttons. To achieve a lightweight design, it was planned to be printed using a filament-based 3D printer. Considering the size of the chassis, filament-based 3D printing was deemed the most suitable method for producing durable large objects.

4.4 Project Evaluation

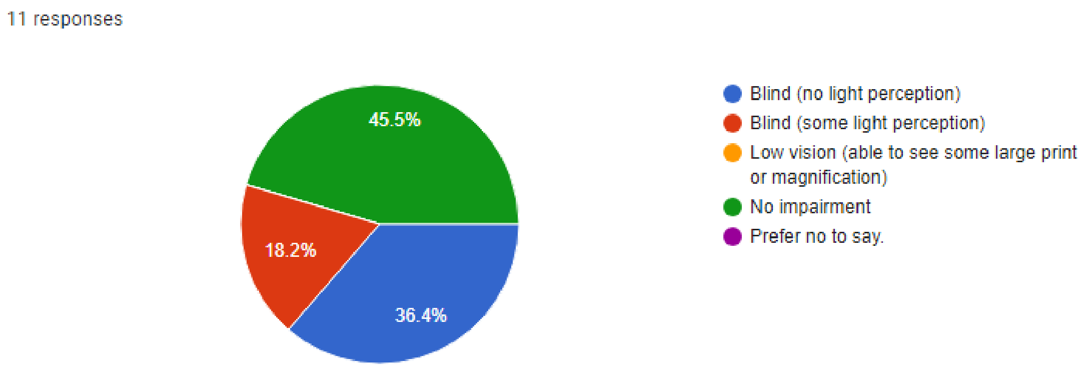


Figure 16. *Level of Vision Impairment*

This figure presented the distribution of impairments among the blind respondents. Out of the total participants, 36.4% were blind with no light perception, 18.2% had some light perception, and 45.5% reported no impairments. It is noteworthy that this last group consisted of experts in the field of teaching blind individuals.

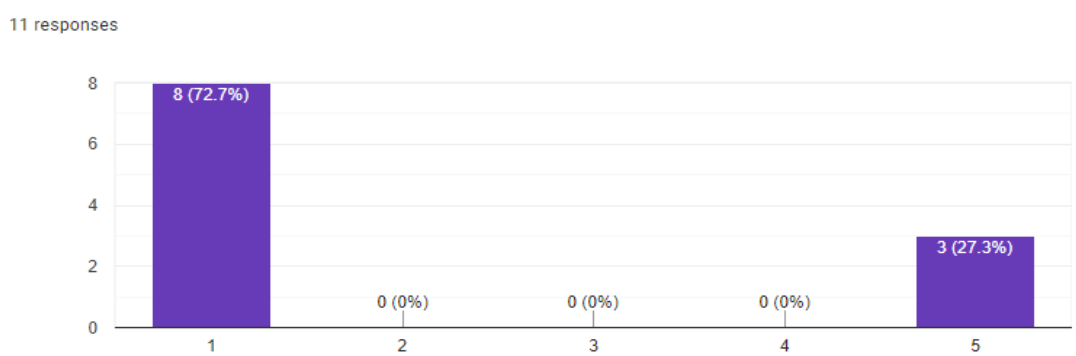


Figure 17. *Number of respondents who used a screen reader or a braille device before.*

This figure depicted that the majority of the respondents, specifically 72.7%, were not familiar with screen readers or refreshable braille devices. Only 27.3% of respondents were familiar with it.

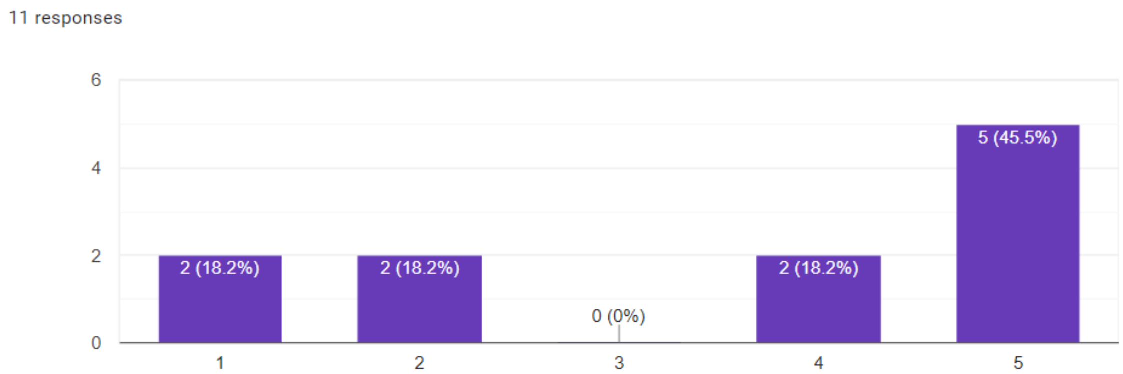


Figure 18. *Number of respondents who are familiar with Grade 1 braille alphabet.*

This figure presented the responses regarding familiarity with the Grade 1 braille alphabet among the respondents. It showed that 45.5% of the respondents strongly agreed that they were familiar with the Grade 1 braille alphabet, while 18.2% agreed that they were familiar with it. Additionally, 18.2% of the respondents disagreed that they were familiar with the Grade 1 braille alphabet, and 18.2% indicated that they were not familiar with it.

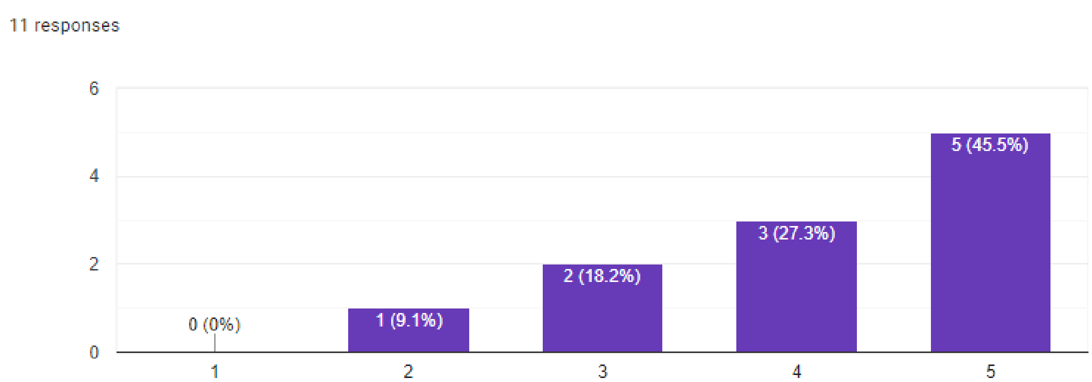


Figure 19. *The usefulness and relevance of speech-to-text conversion for communication purposes.*

This figure presented the responses regarding the usefulness and relevance of the speech-to-text and text-to-braille functions for communication purposes. It showed that 45.5% of the respondents strongly agreed and found it relevant, while 27.3%

agreed with it. Additionally, 18.2% of the respondents were neutral in their opinion, and 9.1% disagreed with its usefulness and relevance.

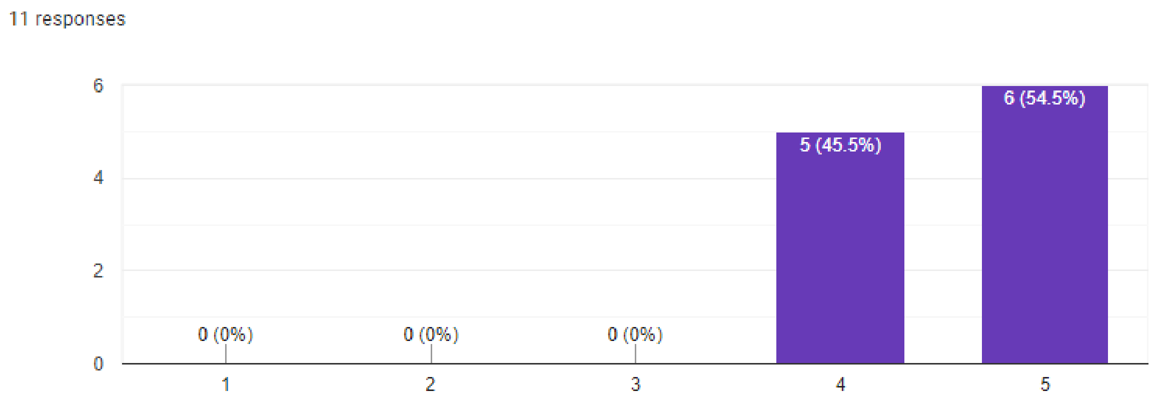


Figure 20. *The usefulness of image-to-braille conversion for learning purposes.*

This figure presented the responses regarding the usefulness of image-to-braille conversion for learning purposes. The data showed that 54.5% of the respondents strongly agreed and 45.5% agreed with it. This indicates that the image-to-braille conversion is perceived as useful for learning purposes by the respondents.

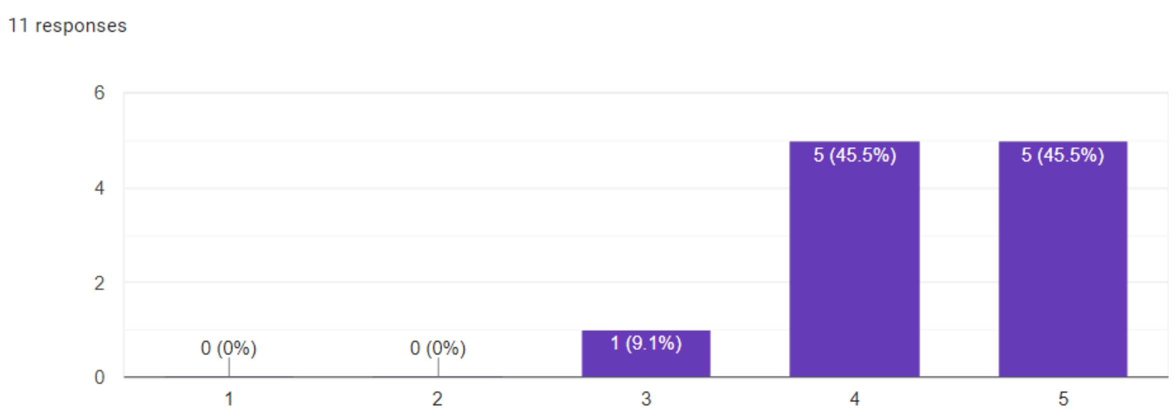


Figure 21. *Number of respondents who agreed that the device meets the needs of blind users.*

This figure presented the responses regarding the agreement on whether the device met the needs of blind users. The data showed that 45.5% of the respondents strongly agreed and 45.5% agreed that the device met the needs of the blind. Only 9.1% of the respondents were neutral in their opinion.

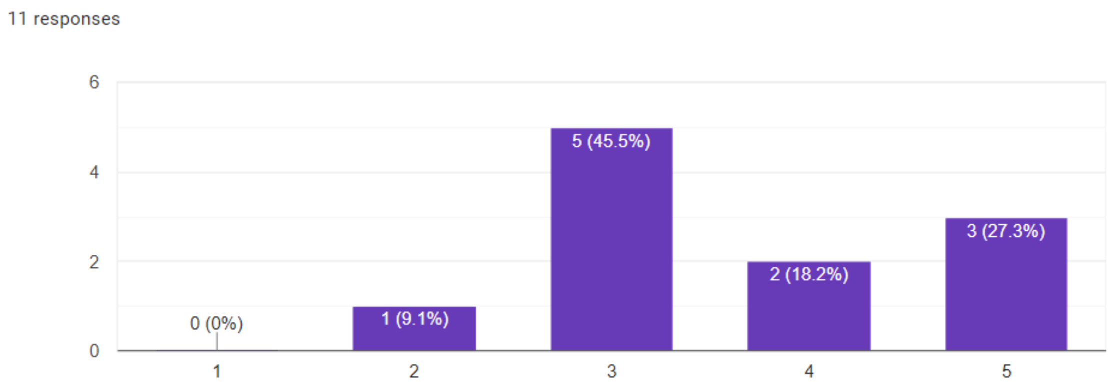


Figure 22. *Number of respondents satisfied with the braille output quality of the refreshable braille display.*

This figure indicated the level of user satisfaction with the device's functionality. It revealed that 27.3% of the respondents were highly satisfied with the braille output quality, while 18.2% expressed satisfaction. Additionally, 45.5% remained neutral in their opinion, and only 9.1% of the respondents were dissatisfied.

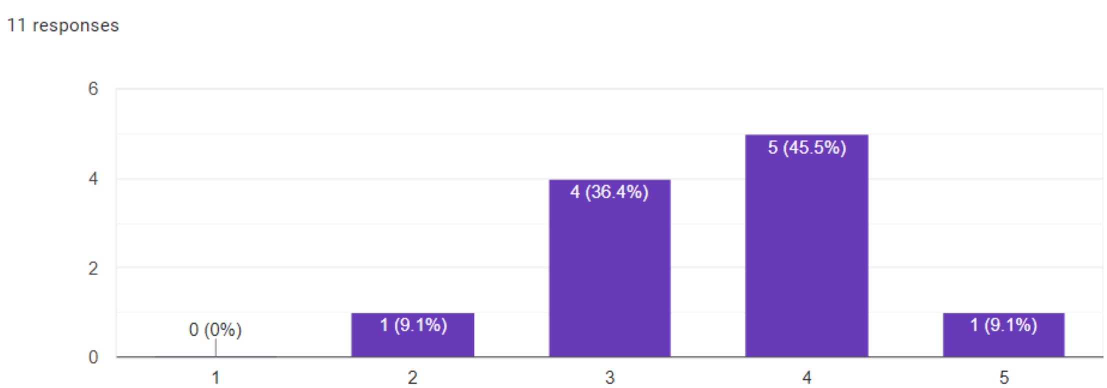


Figure 23. *Number of respondents satisfied with the user-friendliness of the refreshable braille display.*

This figure indicated the level of user satisfaction with the device's functionality. It revealed that 27.3% of the respondents were highly satisfied with the braille output quality, while 18.2% expressed satisfaction. Additionally, 45.5% remained neutral in their opinion, and only 9.1% of the respondents were dissatisfied.

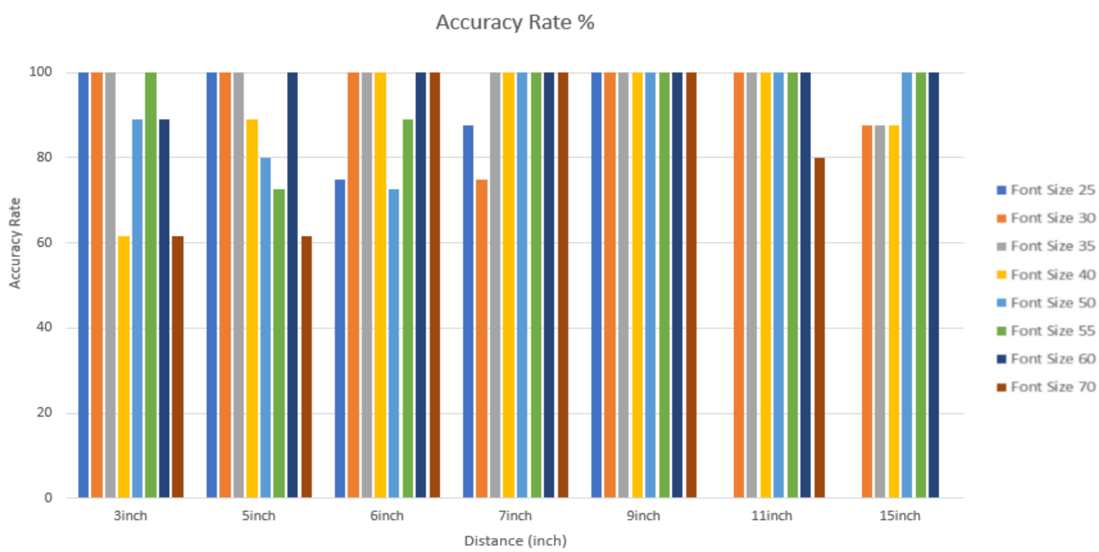


Figure 24. *Accuracy rate of image-to-text conversion*

By testing the phrase "thank you", the accuracy rates at different distances were measured as follows: At a distance of 3 inches, the accuracy rate was found to be 87.6%. When the distance was increased to 5 inches, the accuracy rate dropped slightly to 87.89%. Moving further to a distance of 6 inches, the accuracy rate improved to 92.08%. Notably, at a distance of 7 inches, the accuracy rate significantly increased to 95.3%. At a distance of 9 inches, the accuracy rate remained high at 100%. However, at an 11-inch distance, the accuracy rate declined to 85%. Finally, at a distance of 15 inches, the accuracy rate reached 70.31%.

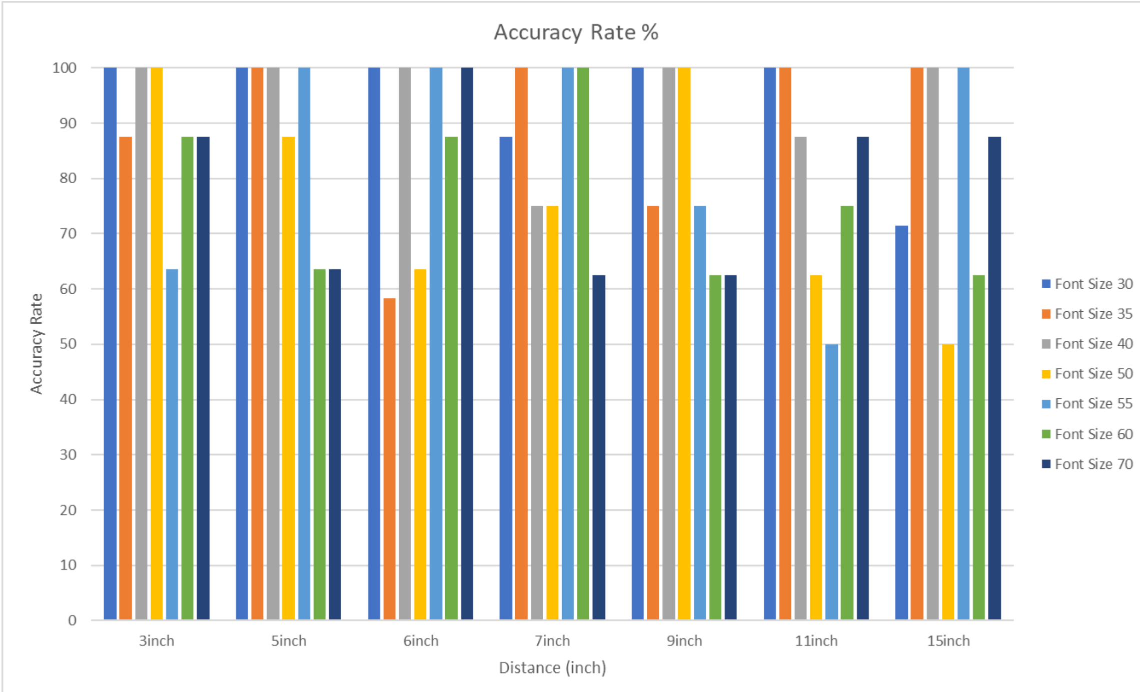


Figure 25. *Accuracy rate of image-to-text conversion*

Based on a comprehensive evaluation of accuracy rates at different distances using the word "welcome" as a test case, the following outcomes were observed. At a distance of 3 inches, the accuracy rate stood at 89.44%. Progressing to a 5-inch distance, the rate slightly decreased to 87.81%. Similarly, at a distance of 6 inches, the accuracy rate remained high at 87.06%. Notably, the rate experienced a slight decline to 85.71% at 7 inches. Further increasing the distance to 9 inches resulted in a reduction of the accuracy rate to 82.14%. The rate continued to decrease at an 11-inch distance, reaching 80.35%. Finally, at a distance of 15 inches, the accuracy rate settled at 81.62%

Figure 26. *Accuracy rate of image-to-text conversion*

Through rigorous testing of the phrase "Braille is a touch-based way used by the blind, deafblind, and visually impaired to read," accurate rates were measured at various distances. At a distance of 7 inches, the accuracy rate was determined to be 82.49%. However, as the distance increased to 9 inches, the accuracy rate slightly decreased to 80.88%. Notably, at an 11-inch distance, the accuracy rate experienced a more noticeable decline, reaching 73.25%. Finally, at a distance of 15 inches, the accuracy rate significantly dropped to 49.52%.

Figure 27. *Image-to-braille conversion*

The camera used was the Raspberry Pi Camera Module v1.3. During testing of the image-to-braille function, the researchers explored various formats, including different font sizes, font styles, and combinations of lowercase and uppercase alphabets. Additionally, they tested different distances at which an object should be positioned away from the camera lens. The results indicated that for optimal results, it is recommended to position the object at a distance ranging from three to nine inches from the lens, while setting the font size to a minimum of 25. This positioning ensures both accuracy and clarity in extracting the text.

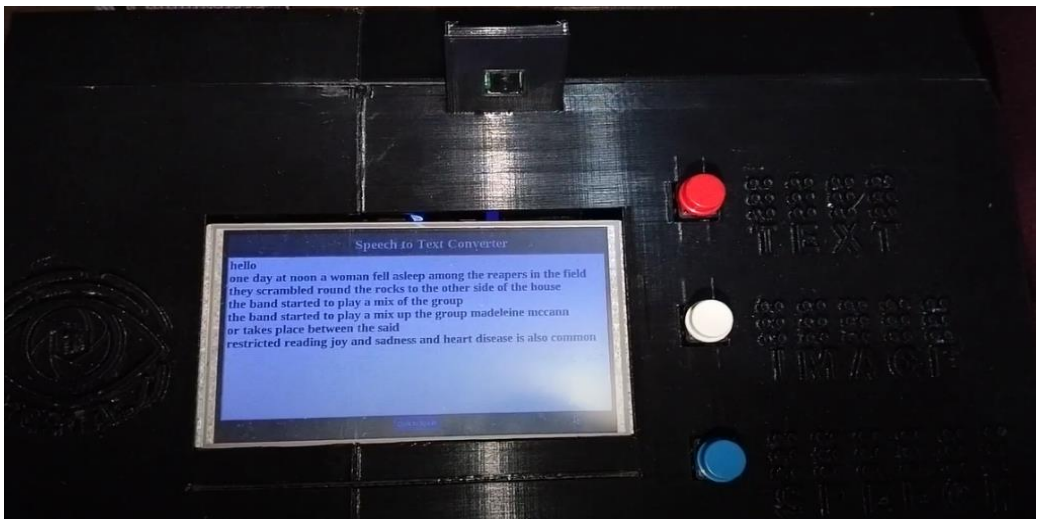


Figure 28. *Speech-to-text conversion*

The researchers used open-source datasets from the Common Voice project developed by Mozilla. These datasets also consisted of demographic metadata, such as age, gender, and accent, which could be utilized to enhance the accuracy of speech recognition engines during training. To evaluate the performance of the integrated Vosk speech recognition in the device, the researchers selected 50 random mp3 files for testing, which showed an accuracy rate of 88.896%.

The speech-to-text conversion feature was evaluated by participants who were deaf and blind. It is remarkable that these individuals were able to successfully communicate using the device. The blind participant utilized the microphone to speak, and the converted text output was displayed on the LCD screen, enabling the deaf participant to read it. In response, the deaf participant used the text-to-braille feature to generate a braille output. This effective interaction demonstrated the device's capability to facilitate communication between individuals with different sensory impairments.

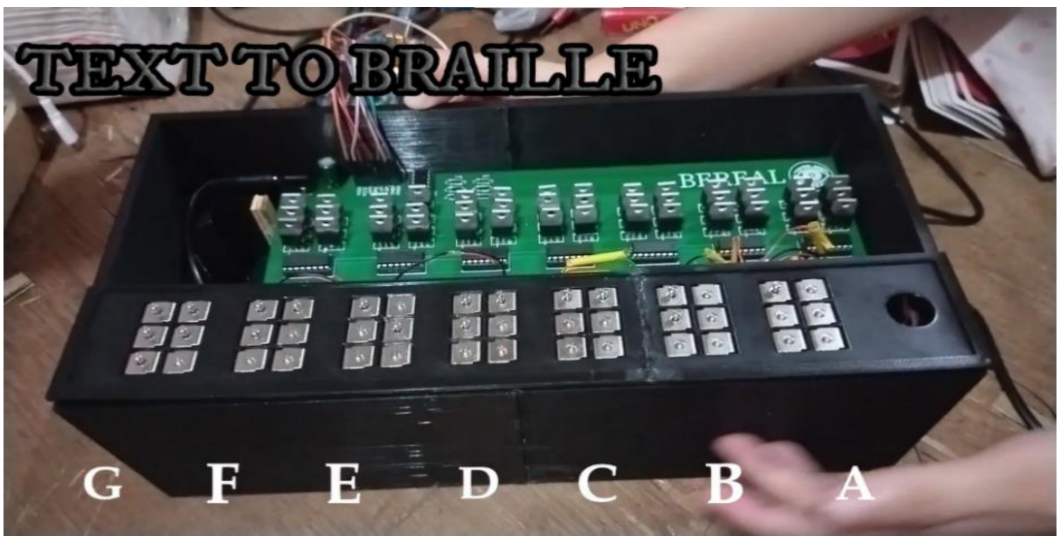


Figure 29. *Text-to-braille conversion*

The researchers developed a program for direct text-to-braille conversion, which generates braille output in a structured format. The braille output consists of three segments, each with a duration of 10 seconds, resulting in a total uptime of 30 seconds. Following each 10-second duration, there is a rest period of 3 seconds. Given that the respondents are elementary and beginner-level users of braille, the researchers have opted for a relatively extended uptime duration of the actuators. This decision aims to allow sufficient time for the respondents to read and comprehend the braille output accurately. However, it is worth noting that the uptime duration of the actuators can be adjusted based on the reading speed and proficiency level of the individual user in Braille. This flexibility allows for customization to accommodate users with varying reading abilities and ensure an optimal user experience.

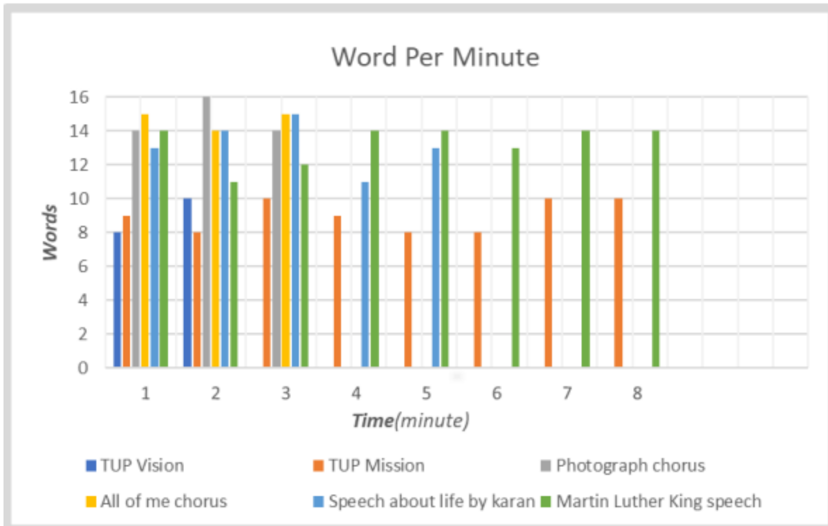


Figure 30. *Word Per Minute*

In addition, to determine the device's word-per-minute capacity, the researchers utilized various samples including TUP Vision, TUP Mission, the chorus of the songs "Photograph" and "All of Me," a speech about life by Karan, and a speech by Martin Luther King. Each set of words from the samples was presented with the braille raised for 4 seconds, allowing time for reading, followed by a 2-second rest period before flashing the next set of words. This testing methodology enabled the researchers to measure the device's efficiency in displaying and allowing users to read a specific number of words within a minute.

Table 10. *Table for Words Per Minute*

DATA INPUT	WORDS PER MINUTE							
	1	2	3	4	5	6	7	8
TUP Vision	8	10						
TUP Mission	9	8	10	9	8	8	10	10
Photograph Chorus	14	16	14					
All of Me Chorus	15	14	15					
Speech about Life by Karan	13	14	15	11	13			
Martin Luther King Speech	14	11	12	14	14	13	14	14

The table showed the accumulated word per minute of the device, whereas the data inputs were TUP Mission, TUP Vision, All of Me (lyric chorus), Photograph (lyric chorus), Martin Luther King speech, and Speech about Life by Karan.

Table 11. *Statistical Measure for Word per Minute*

Statistical Measure for Word per Minute	
Mode	14
Median	13
Standard Deviation	2.55

Based on the data presented in the graph, the device has a reading capacity of approximately 13 to 14 words per minute, with a standard deviation of 2.58. It is important to note that the maximum number of words per minute is 14, as most of the samples used in the testing had words with 4 or fewer characters. Therefore, the researchers have concluded that the number of characters per word in the samples is a significant factor in determining the device's word-per-minute capacity.

Table 12. *Comparison Table of the BeReal between the Commercially Available Braile Devices*

Braille Device	Total Cost	Braille Cells	Connectivity	Features
BEREAL	18,400 Php.	It has seven braille cell characters.	BeReal is a wired device that you just need to plug into a socket to function.	BeReal has a feature to translate speech to text, images to braille, and text to braille.
Smart Brailler	122,020 Php./ ($2,195)	It has 28 braille cells per line.	Smart Brailler can connect to devices like computers and tablets.	It has a conventional keyboard and an integrated speech recognition system.
Basic Braille (Handy Tech)	127,579 Php./ ($2,295)	It has 40 braille cell characters	Bluetooth connectivity, and support for multiple languages	Enables users to read digital text by showing braille characters on a line of tiny mechanical or electronic braille cells. Also, text-to-speech technology can be used by portable tech gadgets to change written text into synthesised voice.
Brailliant BI 40X	205,405 Php. ($3,695)	It has 40-cell braille display	Uniquely powered by Bluetooth 5 wireless technology.	The text-to-speech feature helps users transition from uncontracted to contracted braille by speaking the highlighted word and defining unknown words. Can personalise thumb keys, choose a braille table, and change the pace of the auto-scroll feature.

The table illustrates a comparison of the BEREAL Device with some of the commercially available braille devices in the market. The table clearly shows that the price of the BEREAL device is much cheaper compared to the traditional braille devices in the market.

4.5 Deployment Plan

4.5.1 Overview

In the Philippines, the education system for blind/visually impaired students is governed by the Department of Education (DepEd). DepEd has established specialized schools for students with blind/visual impairments, called Resource Rooms for the Blind (RRB) or Special Education (SPED) Centers. These centers provide a range of services to support visually impaired students in their education, such as assistive technology, Braille materials, and specialized instruction. The government also provides programs and services to support the education of visually impaired students in mainstream schools.

These programs include the provision of assistive technology, the production of Braille textbooks and learning materials, and training for teachers on inclusive education. However, despite the efforts of the government to provide access to education for visually impaired students, there are still challenges in the implementation of these programs. One challenge is the lack of specialized teachers and staff in mainstream schools who are trained in handling the needs of visually impaired students. Another challenge is the lack of resources and funding to support the production of Braille textbooks and other learning materials.

The proponents intend to provide a quality education for blind individuals, but it requires a range of strategies and interventions to ensure that their unique needs are met. Ensuring a high-quality education for blind individuals involves taking numerous factors into account such as access to assistive technology: Blind individuals need access to a range of assistive technologies to support their learning, such as screen readers, Braille displays, and tactile graphics. Access to Braille materials: The utilization of braille plays a vital role in enabling blind individuals to access written materials. Specialized instruction: Blind individuals may require specialized instruction in areas such as orientation and mobility, independent living skills, and assistive technology. Schools should ensure that blind students have access to specialized teachers who are trained in these areas.

4.5.2 Purpose

Braille is an important tool for blind individuals; it helps the blind to read with their fingers, providing blind individuals with a way to access written communication and communicate with others. The purpose of this research is to provide low-cost braille to ensure that blind people have equal access to opportunities and resources, regardless of their financial situation or geographic location. By lowering the cost of Braille materials and devices, it is possible to increase accessibility and promote inclusion for blind people all over the world.

4.5.3 Sites/ Seats/ Location

This project BEREAL will be deployed in P. Gomez Elementary School and Bagong Silang Elementary School. Where the researchers will have an actual testing of the device with the user of the device. After the actual testing of the device the researcher will provide a likert scale form that will be provided and answered by the user. The survey will be used to evaluate the device's functionality by actual usage. Also, the survey can be used for future plans.

Chapter 5

SUMMARY OF FINDINGS, CONCLUSIONS AND RECOMMENDATIONS

5.1 Summary Findings

On the initial deployment at P. Gomez Elementary School, where the device was tested and evaluated by a Grade 3 student and a professor who is teaching blind and braille systems. During the deployment, the uptime duration of braille cells was initially set at 4 seconds followed by 2 seconds rest before the next set of letters appeared. From this, the device is expected to display 13 to 14 words per minute as per testing different sets of paragraphs. However, upon the users' feedback, the 4 seconds uptime duration is not enough based on the following reasons: Firstly, it was noted that the users were relatively new in using refreshable braille displays. Secondly, it was observed that the spacing between each individual dot on our device's braille cells was comparatively wider in comparison to the conventional printed braille materials, which are characterized by a narrower spacing of dots. Lastly, the user was a beginner in terms of reading braille, whose reading speed and comprehension naturally require more time and patience in order to read each individual character or letter.

Hence to cope up with the problems encountered during initial testing, the researchers adjusted the code wherein the uptime duration of braille cells will be 10 seconds and will appear thrice with 3 seconds rest interval for the users to properly read the displayed message in braille form. With that, the researchers were able to lessen the duty cycle from 100% to 83.33% since the recommended working duration from the manufacturer of the solenoid is 5 seconds. These adjustments were implemented during the second deployment which is at Bagong Silang Elementary School. The device was tested and evaluated by Three Elementary Blind Students, Two Elementary Visually Impaired Students, Two Professors teaching Blind and Visually Impaired Students, One Professor teaching Deaf and Hearing-Impaired Students and One Elementary Deaf Student.

The test results indicate that the BEREAL device has great potential for being a replacement for brailler, particularly in educational settings. The device offers enhanced learning capabilities. Also, the Image-to-braille feature of the device allows blind people to access digital contents by scanning digital images using Tesseract which was implemented in the device to extract the text, and which then be converted into braille form. Furthermore, the test indicates that the BEREAL device has the potential to facilitate communication between individuals who are blind and those who are deaf/mute, thereby bridging a communication gap that was previously challenging to overcome. The communication between individuals who are blind and those who are deaf was made possible due to the integration of the speech-to-text and text-to-braille conversion features in the device. These findings highlight the promising prospects of the BEREAL device in improving accessibility and promoting effective communication among individuals with sensory impairments.

5.2 Conclusions

The researchers have conducted a study and arrived at the conclusion that this study addresses the communication challenges faced by blind individuals when interacting with deaf, mute, and unimpaired individuals by developing a portable device that converts text, speech, and images into a braille system. Therefore, the researchers arrived at the following conclusions:

1. The creation of seven braille characters was successful in developing a low-cost braille device as it included an increased number of braille cells. The use of push-pull solenoid actuators as braille cells in the development of a low-cost braille device was a significant aspect of this study. These actuators provide a tactile representation of text, enabling blind users to read and comprehend written information effectively. The approach of reducing manufacturing costs which includes buying and using cost-effective materials, resulted in the device to be more affordable than commercially available braille products such as Smart Brailler and Basic Braille (Handy Tech) which costs 122,020 Php and 127,579 Php, respectively.
2. The researchers have successfully developed a software application that integrated text-to-braille, speech-to-text, and image-to-braille functions, all of which were associated with the push buttons. A software application developed has a graphical user interface that integrates text-to-braille, image-to-braille, and speech-to-text into a single application, as shown in Figures 11–14. These modes of options are all associated with a push button. The software application

provided a user-friendly interface that enabled the user to easily understand and select the modes that were intended to be used.

3. The integration of Tesseract OCR and Vosk, open-source engines for text and speech recognition, respectively, allowed for the conversion of images containing text and spoken language into accessible formats. The implementation of Tesseract OCR and Vosk has shown significance as it provided a high accuracy when the object was positioned at a distance ranging from three to nine inches from the lens, while setting the font size to a minimum of 25 and an accuracy rate of 88.896%, respectively.
4. The Likert Scale was used by the researchers to measure the practicality and effectiveness of the device, extensive testing and evaluation were conducted with blind, mute, and deaf individuals. The Likert Scale shown an average of 4.044 conducted with 11 respondents, this data depicts the respondents were satisfied and agreed that the device was effective.

Overall, the objectives are satisfied and met. The BeReal device has proven to be a significant advancement in terms of accessibility for individuals who are both blind and have hearing or speech impairments. It offers a promising approach to enhance communication opportunities for blind individuals, enabling them to interact more effectively with the broader community. The proposed portable device, incorporating touch, speech, and image sensors, has the potential to bridge the communication gap and empower individuals with visual impairments to engage with others in a more inclusive and accessible manner.

5.3 Recommendations

Based on the feedback and suggestions of the respondents, the current prototype of the proposed modular low-cost refreshable braille display communication and learning device has room for enhancement in the following areas:

1. Add multiple languages especially Filipino Language for the speech-to-text option to allow individuals from different linguistic backgrounds to utilize the device effectively.
2. Create or use an actuator that is small to be able to produce narrow spacing for each dot which results in having more cells or alphabets to be catered.
3. Make the uptime duration of the braille cells longer for the blind or visually impaired people to have more time to read.
4. Replace with a longer mic for the user to eliminate the need to bend over or lean towards the microphone to ensure clear speech input.
5. Include an error message indicating the issue if the scanned text is not readable.
6. Add a feature where clicking a button enables the continuation of the word or sentence when the reader is confident that the text has been read.
7. Choose a much larger font size for the keyboard screen and the text displayed on the keyboard.

REFERENCES

[1] Department of Health, “PREVENTION OF BLINDNESS PROGRAM,” Department of Health, 2004. https://doh.gov.ph/national-prevention-of-blindness-program (accessed Jun. 06, 2022).

[2] M. Jaymalin, “Over 2 million Pinoys blind, sight-impaired,” The Philippine Star, p. 1, Aug. 06, 2017. [Online]. Available: https://www.philstar.com/headlines/2017/08/06/1726085/over-2-million-pinoys-blind-sight-impaired

[3] Dandona L, Dandona R, John RK., “Estimation of blindness in India from 2000 through 2020: implications for the blindness control policy.” The National medical journal of India vol. 14,6 (2001): 327-34.

[4] Hyperbraille: The project [Online]. Available: http://hyperbraille.de/project

[5] Better Health Channel, “Braille.” https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/braille (accessed Jun. 06, 2022).

[6] J. Pradeepkandhasamy, A. Priya, and P. Chellappan, “Voice O Braille,” Mater. Today Proc., Feb.2021.

[7] V. A. Devi, “Conversion of speech to Braille: Interaction device for visual and hearing impaired,” 2017.

[8] A. V. Hemanth, K. S. Bharadwaj, V. Prasanthi, M. E. Harikumar, and J. R. Gini, “Arduino and solenoid valve-based text to braille converter,” 2020.

[9] S. Sultana, A. Rahman, F. H. Chowdhury, and H. U. Zaman, “A novel Braille pad with dual text- to-Braille and Braille-to-text capabilities with an integrated LCD display,” in 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2017, 2018, vol. 2018-January.

[10] S. Hossain et al., “Text to Braille Scanner with Ultra Low Cost Refreshable Braille Display,” 2019.

[11] “The story of Louis braille: Inventor of the braille code,” Icoe.org. [Online]. Available: https://www.icoe.org/news/story-louis-braille-inventor-braille-code. [Accessed: 07-Jun-2022].

[12] G. Hudson-Searle, “From Louis Braille to leading edge technology that helps people see again,” Freedom after the sharks, 17-Jul-2017. [Online].Available: https://freedomafterthesharks.com/2017/07/17/louis-braille-leading-edge- technology-helps- people-see/. [Accessed: 07-Jun-2022].

[13] “Braille displays and notetakers,” RNIB - See differently, 21-Apr-2020. [Online]. Available: https://www.rnib.org.uk/sight-loss-advice/technology-and-useful-products/technology-resource- hub-latest-facts-tips-and-guides/braille-displays-and-notetakers. [Accessed: 07-Jun-2022].

[14] S. Sultana, A. Rahman, F. H. Chowdhury and H. U. Zaman, "A novel Braille pad with dual text- to- Braille and Braille-to-text capabilities with an integrated LCD display," 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), 2017, pp. 195-200, doi: 10.1109/ICICICT1.2017.8342559.

[15] B. Bhanushali, A. Dhoot, P. Gandhi, and K. Mehta, “Refreshable Braille Displays,” Int. J. Comput. Appl., vol. 180, no. 37, pp. 1–4, 2018.

[16] L. Kumar, “Refreshable Braille Display: Function, benefits and purchasing decision,” WeCapable, 10-Jul-2021. [Online]. Available: https://wecapable.com/refreshable-braille-display/. [Accessed: 07-Jun 2022]

[17] M. E. Adnan, N. M. Dastagir, J. Jabin, A. M. Chowdhury and M. R. Islam, "A cost effective electronic braille for visually impaired individuals," 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), 2017, pp. 175-178, doi: 10.1109/R10-HTC.2017.8288932.

[18] “‘What is a Computer Keyboard? - Parts, Layout & Functions.,’” Study.com, 2014. https://study.com/academy/lesson/what-is-a-computer-keyboard-parts-layout-functions.html (accessed Jun. 06, 2022).

[19] “Touch screen,” Computer Hope, 2022. https://www.computerhope.com/jargon/t/toucscre.htm (accessed Jun. 06, 2022).

[20] K. Jindal, A. Mattoo, and B. Kumar, “Braille decoding device employing microcontroller,” Int. J.Recent Technol. Eng., vol. 8, no. 2 Special Issue 11, 2019.

[21] S. Ramachandran and N. Rajan, "Communication Device for the Visual and Hearing Impaired Persons to Convert Braille Characters to English Text," pp. 587–592, 2021.

[22] K. K. Kishore, G. Prudhvi, and M. Naveen, "Braille script to voice conversion," Proc. Int. Conf. Comput. Methodol. Commun. ICCMC 2017, vol. 2018-January, no. Iccmc, pp. 1080–1082, 2018.

[23] V. Papadimitriou and V. Argyropoulos, "Tracing the effectiveness of braille reading patterns in individuals with blindness: Handedness and error analysis," Br. J. Vis. Impair., vol. 38, no. 2, pp. 209–221, 2020.

[24] Guerreiro, João & Gonçalves, Daniel & Marques, D & Guerreiro, Tiago & Nicolau, Hugo & Montague, Kyle. (2013). The Today and Tomorrow of Braille Learning. 10.1145/2513383.2513415.

[25] L. Triyono, T. R. Yudantoro, S. Sukamto, and I. Hestinigsih, "VeRO: Smart home assistant for blind with voice recognition," IOP Conf. Ser. Mater. Sci. Eng., vol. 1108, no. 1, p. 012016, 2021.

[26] L. A. D. Arbes, J. M. J. Baybay, J. E. E. Turingan, and M. J. C. Samonte, "Tagalog text-to-braille translator tactile story board with 3D printing," IOP Conf. Ser. Mater. Sci. Eng., vol. 482, no. 1, 2019.

[27] D. Lee and J. Cho, "Automatic Object Detection Algorithm-Based Braille Image Generation System for the Recognition of Real-Life Obstacles for Visually Impaired People," Sensors, vol. 22, no. 4, 2022.

[28] S. Manoharan, "A Smart Image Processing Algorithm for Text Recognition, Information Extraction and Vocalization for the Visually Challenged," J. Innov. Image Process., vol. 1, no. 01, pp. 31–38, 2019.

[29] K. Smelyakov, A. Chupryna, D. Yeremenko, A. Sakhon, and V. Polezhai, "Braille Character Recognition Based on Neural Networks," Proc. 2018 IEEE 2nd Int. Conf. Data Stream Min. Process. DSMP 2018, pp. 509–513, 2018.

[30] S. Kumar., S. Mathew., N. Anumula., K. S. Chandra. Portable Camera-Based Assistive Device for Real-Time Text Recognition on Various Products and Speech Using Android for Blind People. In: Saini, H.S., Singh, R.K., Tariq Beg, M., Sahambi,

J.S. (eds) Innovations in Electronics and Communication Engineering. Lecture Notes in Networks and Systems, vol 107, pp 437–448. 2020.

[31] K. A. Moharkar, “Review on Different Microcontroller Boards Used in IoT,” Int. J. Res. Appl. Sci. Eng. Technol., vol. 10, no. 1, 2022

[32] M. M. Garcillanosa, K. N. T. Apuyan, A. M. Arro, and G. G. Ascan, “Audio-assisted standalone microcontroller-based Braille System Tutor for Grade 1 Braille symbols,” 2017

[33] N. A. Z. N. M. Rahimi, N. H. H. M. Hanif, and Z. Janin, “Identification of hijaiyah letters using wirelessly controlled Arabic braille module,” Indonesian Journal of Electrical Engineering and Computer Science, vol. 17, no. 2. pp. 826–832, 2019

[34] S. Pérez-Vicente and M. Expósito Ruiz, “Descriptive statistics,” Allergol. Immunopathol. (Madr)., vol. 37, no. 6, pp. 314–320, 2009.

[35] R. C. Richey, “Developmental Research: The Definition and Scope,” 1994 Natl. Conv. teh Assoc. fr Educ. Commun. Technol., pp. 714–720, 1994, [Online]. Available: http://files.eric.ed.gov/fulltext/ED373753.pdf

[36] J. L. Dela Cruz, J. A. D. Ebreo, R. A. J. P. Inovejas, A. R. C. Medrano, and A. A. Bandala, “Development of a text to braille interpreter for printed documents through optical image processing,” HNICEM 2017 - 9th Int. Conf. Humanoid, Nanotechnology, Inf. Technol. Commun. Control. Environ. Manag., vol. 2018-January, pp. 1–6, 2017.

[37] R. G. De Luna, “A Tesseract-based Optical Character Recognition for a Text-to-Braille Code Conversion,” vol. 10, no. 1, 2020.

[38] D. Sporici, E. Cuşnir, and C. A. Boiangiu, “Improving the accuracy of Tesseract 4.0 OCR engine using convolution-based preprocessing,” Symmetry (Basel)., vol. 12, no. 5, 2020.

[39] C. Adjetey and K. S. Adu-Manu, “Content-based Image Retrieval using Tesseract OCR Engine and Levenshtein Algorithm,” Int. J. Adv. Comput. Sci. Appl., vol. 12, no. 7, 2021.

[40] E. B. BOZTEPE, B. KARAKAYA, B. KARASULU, and İ. ÜNLÜ, “An Approach for Audio-Visual Content Understanding of Video using Multimodal Deep Learning Methodology,” Sak. Univ. J. Comput. Inf. Sci., vol. 5, no. 2, 2022.

[41] C. W. An and S. Assistant, “Offline Speech to Text Engine for Delimited Context,” 2022.

[42] F. P. Comas, “Language grounding for robotics,” 2022

[43] C. Clausner, A. Antonacopoulos, and S. Pletschacher, “Efficient and effective OCR engine training, Int. J. Doc. Anal. Recognit., vol. 23, no. 1, 2020.

[44] T. F. Pereira et al., “A web-based Voice Interaction framework proposal for enhancing Information Systems user experience,” Procedia Comput. Sci., vol. 196, pp. 235–244, 2021.

[45] S. Cakic, T. Popovic, S. Sandi, S. Krco, and A. Gazivoda, “The Use of Tesseract OCR Number Recognition for Food Tracking and Tracing,” 2020 24th Int. Conf. Inf. Technol. IT 2020, no. February, 2020.

[46] I. S. Akila, B. Akshaya, S. Deepthi, and P. Sivadharshini, “A Text Reader for the Visually Impaired using Raspberry Pi,” Proc. 2nd Int. Conf. Comput. Methodol. Commun. ICCMC 2018, no. Iccmc, pp. 778–782, 2018.

[47] M. Z. M. Nizam, S. M. Saad, M. A. Suhaimi, M. Hussein, S. Z. A. Rahim, and Z. A. Ahmad, “Development of electromagnetic actuator for braille cell,” 2021, p. 020203

[48] United Nations. (n.d.). The 17 goals | sustainable development. United Nations. https://sdgs.un.org/goals.

ANNEX I
BILL OF MATERIALS

BILL OF MATERIALS

Item #	Unit	Item Description	Qty.	Unit Cost	Total Cost
1	pc	Raspberry Pi 4 Model B (4gb ram)	1	5,400.00	5,400.00
2	pc	Arduino Uno R3	1	400.00	400.00
3	pc	Electromagnetic Push-Pull Solenoid Actuator	42	65.83	2,765.00
4	pc	Raspberry Pi Camera Module V1.3	1	235.00	235.00
5	pc	12V 6A Power Supply	1	500.00	500.00
6	pc	LM7805 Module	1	100.00	100.00
7	pc	PCB Fabrication and Assembly (*42pcs ¼ watts 2.2k ohm Resistors, 42pcs 1n4004 Diode, 1pc 1x8 female pin header, 14pcs 1x6 female pin header and 8pcs 16 pins IC socket*)	1	3,296.00	3,296.00
8	pc	TIP120 Power Darlington Transistor	42	565.00	565.00
9	pc	5.5mmx2.1mm DC Barrel Jack	1	74.00	74.00
10	pc	Raspberry Pi 4 5V 3A Type C Power Adapter	1	250.00	250.00
11	pc	Esun PLA+ Black Filament	1	800.00	800.00
12	pc	3D Printing Service	1	800.00	800.00
13	pc	Toggle Switch	1	22.00	22.00
14	pc	74HC595 8-bit Shift Register	7	25.00	175.00
15	pc	3.5mm Flexible Microphone	1	99.00	99.00
16	pc	5” HDMI Touch Screen LCD Display Monitor	1	2,205.00	2,205.00
17	pc	Pushbuttons	4	22.00	88.00
18	pc	64gb SD Card	1	500.00	500.00
19	pc	Micro HDMI to HDMI Cable	1	79.00	79.00
20	pc	Male USB A to Male USB B Cable	1	50.00	50.00
21	pc	White Small Acrylic Paint	1	25.00	25.00
22	pc	Terminal Block	1	20.00	20.00
23	pc	40pcs Female-Female Jumper Wire	1	80.00	80.00
24	pc	40pcs Male-Male Jumper Wire	2	65.00	130.00
25	pc	5 meters Solid Copper Wire	1	100.00	100.00
				TOTAL	**18,758**

ANNEX II
BEREAL CODES

FOR SOLENOIDS CODES

Code:

```
// Pin definitions
const int dataPin = 8;
const int clockPin = 10;
const int latchPin = 9;
const int numRegisters = 1;
const int numCharacters = 6;

// Braille code for each character
byte brailleCodes[] = {
  B000001, // A
  B000011, // B
  B100001, // C
  B110001, // D
  B010001, // E
  B100011, // F
  B110011, // G
  B010011, // H
  B100010, // I
  B110010, // J
  B000101, // K
  B000111, // L
```

```
  B100101, // M
  B110101, // N
  B010101, // O
  B100111, // P
  B110111, // Q
  B010111, // R
  B100110, // S
  B110110, // T
  B001101, // U
  B001111, // V
  B111010, // W
  B101101, // X
  B111101, // Y
  B011101, // Z
};
void setup() {
  // Set pin modes
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(latchPin, OUTPUT);
  // Initialize serial communication
  Serial.begin(9600);
}
```

```
void loop() {
  // Read input from serial
  if (Serial.available() > 0) {
    // Read the input string
    String inputString = Serial.readString();
    // Reverse the input string
    String reversedString = "";
    for (int i = inputString.length() - 1; i >= 0; i--) {
      reversedString += inputString.charAt(i);
    }
    // Convert each character to Braille code and display on Braille display
    for (int i = 0; i < reversedString.length(); i++) {
      // Convert character to Braille code
      byte brailleCode = getBrailleCode(reversedString.charAt(i));

      // Shift out the Braille code to the shift registers
      for (int j = 0; j < numRegisters; j++) {
        shiftOut(dataPin, clockPin, MSBFIRST, brailleCode);
      }
      // Latch the data to the output
      digitalWrite(latchPin, HIGH);
      delay(1);
      digitalWrite(latchPin, LOW);
```

```
    }
  }
}
byte getBrailleCode(char inputChar) {
  // Convert input character to uppercase
  inputChar = toupper(inputChar);
  // Check if input character is a letter
  if (inputChar >= 'A' && inputChar <= 'Z') {
    // Return the corresponding Braille code
    return brailleCodes[inputChar - 'A'];
  } else {
    // Return blank code for non-letter characters
    return B00000000;
  }
}
String reverseString(String inputString) {
  String reversedString = "";
  // Loop through each character in the string
  for (int i = inputString.length() - 1; i >= 0; i--) {
    reversedString += inputString[i];
  }
  return reversedString;
}
```

BEREAL CODE FOR GUI

```
#!/usr/bin/env python

import tkinter

import os

import sys

import RPi.GPIO as GPIO

import subprocess

import pygame


# Set the PYTHONPATH environment variable

os.environ['PYTHONPATH'] = '/home/brbr/.local/lib/python3.9/site-
packages/customtkinter'


# Import the module

sys.path.insert(0, '/home/brbr/.local/lib/python3.9/site-packages/customtkinter')

import customtkinter


GPIO.setmode(GPIO.BCM)

buttontext = 13

buttonimage = 5

buttonspeech = 6

GPIO.setup(buttontext, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(buttonspeech, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buttonimage, GPIO.IN, pull_up_down=GPIO.PUD_UP)

customtkinter.set_appearance_mode("dark")  # Modes: system (default), light, dark
customtkinter.set_default_color_theme("blue")  # Themes: blue (default), dark-blue,
green

class BrailleSystem():
    def __init__(self, master=None):
        self.master = master
        self.tl_bg = "#000000"
        self.tl_bg2 = "#000000"
        self.tl_fg = "#000000"
        self.font = "Bodoni"
        # Initialize Pygame
        pygame.init()

        # Load the intro video
        self.intro_video = pygame.mixer.music
        # self.intro_video.load("/home/brbr/intro.mp4")
        #
        # # Play the intro video
        # self.intro_video.play()
```

```
        #Image
        self.buttonimg = customtkinter.CTkButton(root, text="Image to Braille", font=("Palatino", 35), width=450, height=75, command=self.ImageBraille)
        self.buttonimg.place(relx= 0.5, rely= 0.2, anchor=tkinter.CENTER)

        #Audio
        self.buttonaud = customtkinter.CTkButton(root, text="Speech to Text", font=("Palatino", 35), width=450, height=75, command=self.AudioBraille)
        self.buttonaud.place(relx= 0.5, rely= 0.5, anchor=tkinter.CENTER)

        #Keyboard
        self.buttontxt = customtkinter.CTkButton(root, text="Text to Braille", font=("Palatino", 35), width=450, height=75, command=self.KeyboardBraille)
        self.buttontxt.place(relx= 0.5, rely= 0.8, anchor=tkinter.CENTER)

        # Schedule the stop of intro video after it finishes
        self.master.after(0, self.check_intro_status)

    def check_intro_status(self):
        if not pygame.mixer.music.get_busy():
            self.stop_intro()
        self.master.after(1000, self.check_intro_status)
```

```
def play_intro(self):
    # Play the intro video
    self.intro_video.play()
def stop_intro(self):
    # Stop the intro video
    self.intro_video.stop()


def AudioBraille(self):
    self.stop_intro()
    self.buttonimg.destroy()
    self.buttonaud.destroy()
    self.buttontxt.destroy()
    os.system('python berealspeech.py')
    self.master.withdraw()
    os._exit(0)


def KeyboardBraille(self):
    self.stop_intro()
    self.buttonimg.destroy()
    self.buttonaud.destroy()
    self.buttontxt.destroy()
    os.system('python berealtext.py')
```

```
        self.master.withdraw()
        os._exit(0)

    def ImageBraille(self):
        self.stop_intro()
        self.buttonimg.destroy()
        self.buttonaud.destroy()
        self.buttontxt.destroy()
        os.system('python berealimage.py')
        self.master.withdraw()
        os._exit(0)

    @staticmethod
    def check_buttontext(pin):
        if GPIO.input(buttontext) == False:
            a.KeyboardBraille()

    @staticmethod
    def check_buttonspeech(pin):
        if GPIO.input(buttonspeech) == False:
            a.AudioBraille()

    @staticmethod
```

```
    def check_buttonimage(pin):
        if GPIO.input(buttonimage) == False:
            a.ImageBraille()


GPIO.add_event_detect(buttontext, GPIO.FALLING,
callback=BrailleSystem.check_buttontext, bouncetime=100)
GPIO.add_event_detect(buttonspeech, GPIO.FALLING,
callback=BrailleSystem.check_buttonspeech, bouncetime=100)
GPIO.add_event_detect(buttonimage, GPIO.FALLING,
callback=BrailleSystem.check_buttonimage, bouncetime=100)


root = customtkinter.CTk()
root.geometry("800x480")
a = BrailleSystem(root)
# root.attributes('-fullscreen', True)
root.geometry('700x300')
root.configure(bg='black')
root.title("Bereal")
root.mainloop()
root.destroy()
```

BEREAL IMAGE-TO-BRAILLE CODE

```
import cv2

import pytesseract

from PIL import Image

import tkinter as tk

import RPi.GPIO as GPIO

import os

import time

import serial

import subprocess


pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR'


# Set up GPIO

GPIO.setmode(GPIO.BCM)

buttontext = 13

cameracapture = 5

buttonspeech = 6

buzzer_pin = 26

GPIO.setup(buttontext, GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(buttonspeech, GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(cameracapture, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(buzzer_pin, GPIO.OUT)


# Capture image from camera
cap = cv2.VideoCapture(0)


# Click counter
clicks = 0
def print_text_with_buzzer(text):
    textt = text
    substrings = [textt[i:i + 7] for i in range(0, len(textt), 7)]
    output = ""
    for s in substrings:
        s += " " * (7 - len(s))
        output += s + "\n"
    outputt = [output[i:i + 8] for i in range(0, len(output), 8)]
    # ser = serial.Serial('/dev/ttyACM0', 9600)


    try:
        if ser.is_open:
            print("Serial connection opened successfully")
            # Send "......." as the first input
            # ser.write(".......".encode())
            print(".......")
```

```
time.sleep(2)

for i in range(len(outputt)):
    for j in range(3):
        if i != 0 or j != 0:
            ser.write(".......".encode())
            print(".......")
            time.sleep(3)
        if len(outputt[i].strip()) > 0:
            print(outputt[i].strip())
            ser.write(outputt[i].encode())
            time.sleep(3)
        else:
            print(" ")
        if outputt[i].strip() != "" and outputt[i] != ".......":
            time.sleep(5)
        else:
            print(outputt[i])
            ser.write(".......".encode())
            print(".......")
            time.sleep(3)
        if i != len(outputt) - 1:
            time.sleep(3)
```

```
            ser.write(".......".encode())  # Send "......." as the last encode for the serial
            print(".......")
            time.sleep(3)
        else:
            print("Failed to open serial connection")
    except serial.SerialException as e:
        print("Error opening serial connection:", e)
    finally:
        ser.close()
    # Turn on the buzzer for 0.2 seconds
    GPIO.output(buzzer_pin, GPIO.HIGH)
    time.sleep(0.2)
GPIO.add_event_detect(cameracapture, GPIO.FALLING, callback=button_clicked,
bouncetime=200)
class ImageSystem():
    def __init__(self, master=None):
        self.master = master
    def AudioBraille(self):
        print('audo braille open')
        cv2.destroyAllWindows()
        os.system('python berealspeech.py')  # execute the system call to open the new
Python file
```

```
        # self.master.destroyALLWindows()  # destroy the current GUI window
        os._exit(0)
    def KeyboardBraille(self):
        print('keyboard braille open')
        cv2.destroyAllWindows()
        os.system('python berealtext.py')   # execute the system call to open the new Python
file
        # self.master.destroyALLWindows()
        os._exit(0)
    @staticmethod
    def check_buttontext(pin):
        print('click text')
        if GPIO.input(buttontext) == False:
            a.KeyboardBraille()
    @staticmethod
    def check_buttonspeech(pin):
        print('click speech')
        if GPIO.input(buttonspeech) == False:
            a.AudioBraille()
    def destroy_windows(self):
        # Destroy all OpenCV windows
        cv2.destroyAllWindows()
        # Release the camera resources
```

```
        cap.release()
# Create an instance of ImageSystem
a = ImageSystem(cv2)
GPIO.add_event_detect(buttonspeech, GPIO.FALLING,
callback=a.check_buttonspeech, bouncetime=100)
GPIO.add_event_detect(buttontext, GPIO.FALLING, callback=a.check_buttontext,
bouncetime=100)
# Create an instance of ImageSystem
while True:
    ret, image = cap.read()
    image = cv2.flip(image, 1)
    # Create a named window with full screen property
    cv2.namedWindow("Capturing Image", cv2.WND_PROP_FULLSCREEN)
    cv2.setWindowProperty("Capturing Image", cv2.WND_PROP_FULLSCREEN,
cv2.WINDOW_FULLSCREEN)
    # Show the captured video
    cv2.imshow("Capturing Image", image)
    # Wait for user input
    key = cv2.waitKey(1)
    if clicks == 2:
        # Preprocess the image
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        gray = cv2.medianBlur(gray, 3)
```

```
        gray = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)[1]
        # Save the preprocessed image
        cv2.imwrite(r"/home/brbr/bereal/captured_preprocessed.png", gray)
        # Perform OCR on the image
        text =
pytesseract.image_to_string(Image.open(r"/home/brbr/bereal/captured_preprocessed.p
ng"), lang='eng')


        # Print the recognized text
        print_text_with_buzzer(text)
        clicks = 0
    elif key == ord('q'):
        break
    elif a.check_buttonspeech(buttonspeech) or a.check_buttontext(buttontext):
        cv2.destroyAllWindows()
        # Exit the while loop if either button is clicked twice
        clicks = 0
        break
# Release resources
cap.release()
a = ImageSystem(cv2)
cv2.destroyAllWindows()
```

BEREAL SPEECH-TO-BRAILLE CODE

```
from tkinter import *
from vosk import Model, KaldiRecognizer
import time
import pyaudio
import RPi.GPIO as GPIO
import os
import sys
import subprocess
os.environ['PYTHONPATH'] = '/home/brbr/.local/lib/python3.9/site-
packages/customtkinter'

# Import the module
sys.path.insert(0, '/home/brbr/.local/lib/python3.9/site-packages/customtkinter')
import customtkinter
import customtkinter as tk
GPIO.setmode(GPIO.BCM)
buttontext = 13
buttonimage = 5
speak = 6
buzzer_pin = 26
GPIO.setup(buttontext, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(buttonimage, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(speak, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buzzer_pin, GPIO.OUT)
customtkinter.set_appearance_mode("dark")  # Modes: system (default), light, dark
customtkinter.set_default_color_theme("blue")  # Themes: blue (default), dark-blue,
green
speechtotextwindow = customtkinter.CTk()
model = Model(r'C:\Users\Owner\PycharmProjects\BEREAL REAL NA REAL\vosk-
model-small-en-us-0.15\vosk-model-small-en-us-0.15')
recognizer = KaldiRecognizer(model, 16000)
mic = pyaudio.PyAudio()
stream = mic.open(format=pyaudio.paInt16, channels=1, rate=16000, input=True,
frames_per_buffer=8192)
clicks = 0
recording = False
def print_text_with_buzzer(text):
   print(text)
   GPIO.output(buzzer_pin, GPIO.HIGH)
   time.sleep(0.5)
   GPIO.output(buzzer_pin, GPIO.LOW)


def button_clicked(channel):
   global clicks, recording, scheduled
```

```
    clicks += 1
    if clicks == 2:
        if not recording:
            recording = True
            recognizer.FinalResult()  # Stop the recognizer before starting a new recording
            print('double click')
            record_voice()
        clicks = 0
        stream.stop_stream()
        print("Streaming stopped")  # Add this line to indicate that streaming has stopped
    else:
        # Schedule the function to run after 0.5 seconds (in case of double-click)
        scheduled = speechtotextwindow.after(500, check_clicks)
        recognizer.FinalResult()  # Clear the recognizer's result
GPIO.add_event_detect(speak, GPIO.FALLING, callback=button_clicked,
bouncetime=200)
def check_clicks():
    global clicks
    if clicks == 1:
        print("Button clicked 1 time")
    clicks = 0  # Reset clicks variable after processing single-click
def record_voice():
    global clicks, recording
```

```
    clicks = 0  # Reset clicks variable
    print("Recording started")  # Debugging line
    recognizer.FinalResult()  # Clear the recognizer's result
    stream.start_stream()
    while recording:
        data = stream.read(4096)
        if recognizer.AcceptWaveform(data):
            result = recognizer.Result()
            text = result
            print_text_with_buzzer(text[14:-3])
            textarea.insert(END, text[14:-3] + '\n')
            textarea.see(END)
            recording = False
            return (text[14:-3] + '\n')
    result = None
def update_scroll_region(event):
    textarea.yview_moveto(1.0)
Label = tk.CTkLabel(speechtotextwindow, text="Speech to Text Converter",
font=('Times New Roman', 30, 'bold'), padx=2,
                    pady=12).grid(row=0, columnspan=12)
textarea = Text(speechtotextwindow, width=65, height=12, font=('Times New Roman',
20, 'bold'), padx=2, pady=2)
textarea.grid(row=2, columnspan=12)
```

```
textarea.bind('<Configure>', update_scroll_region)

buttons = ['Click to Speak']

varRow = 2

varColumn = 5

for button in buttons:

   command = lambda x=button: button_clicked()

   if button == "Click to Speak":

      tk.CTkButton(speechtotextwindow, text=button, width=60,

command=command).grid(row=4, column=5, padx=5, pady=5)

class SpeechSystem():

   def __init__(self, master=None):

      self.master = master

   def KeyboardBraille(self):

      os.system('python berealtext.py')   # execute the system call to open the new Python

file

      self.master.withdraw()  # destroy the current GUI window

      os._exit(0)

   def ImageBraille(self):

      os.system('python berealimage.py')  # execute the system call to open the new

Python file

      self.master.withdraw()

      os._exit(0)

   @staticmethod
```

```
    def check_buttontext(pin):
        if GPIO.input(buttontext) == False:
            a.KeyboardBraille()
    @staticmethod
    def check_buttonimage(pin):
        if GPIO.input(buttonimage) == False:
            a.ImageBraille()
a = SpeechSystem(speechtotextwindow)
GPIO.add_event_detect(buttontext, GPIO.FALLING, callback=a.check_buttontext,
bouncetime=100)
GPIO.add_event_detect(buttonimage, GPIO.FALLING,
callback=a.check_buttonimage, bouncetime=100)
speechtotextwindow.title('Speech-to-Text')
speechtotextwindow.geometry("800x480")
# speechtotextwindow.attributes('-fullscreen', True)
speechtotextwindow.update()
speechtotextwindow.geometry('700x350')
speechtotextwindow.mainloop()
speechtotextwindow.destroy()
```

BEREAL TEXT-TO-BRAILLE

```
#try:
#    import tkinter as tk
#except ImportError:
#    import tkinter as tk


import customtkinter as tk
import customtkinter
from tkinter.scrolledtext import ScrolledText
from functools import partial
from tkinter import*
import tkinter
from tkinter import ttk
import RPi.GPIO as GPIO
import os
import time
# import serial
import sys
import  subprocess
os.environ['PYTHONPATH'] = '/home/brbr/.local/lib/python3.9/site-
packages/customtkinter'
```

```
# Import the module
sys.path.insert(0, '/home/brbr/.local/lib/python3.9/site-packages/customtkinter')
import customtkinter
import customtkinter as tk
GPIO.setmode(GPIO.BCM)
buttonimage = 5
buttonspeech = 6
buzzer_pin = 26
GPIO.setup(buttonspeech, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buttonimage, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buzzer_pin, GPIO.OUT)
customtkinter.set_appearance_mode("dark")  # Modes: system (default), light, dark
customtkinter.set_default_color_theme("blue")  # Themes: blue (default), dark-blue,
green
Keyboard_App = customtkinter.CTk()  # create CTk window like you do with the Tk
window
Keyboard_App.title('Keyboard')
Keyboard_App.geometry('700x480')
# Keyboard_App.attributes('-fullscreen', True)
# Keyboard_App.state('normal')


def print_text_with_buzzer(text):
    textt = text
```

```
substrings = [textt[i:i + 7] for i in range(0, len(textt), 7)]
output = ""
for s in substrings:
    s += " " * (7 - len(s))
    output += s + "\n"
outputt = [output[i:i + 8] for i in range(0, len(output), 8)]
# ser = serial.Serial('/dev/ttyACM0', 9600)


try:
    if ser.is_open:
        print("Serial connection opened successfully")
        # Send "......." as the first input
        # ser.write(".......".encode())
        print(".......")
        time.sleep(2)
        for i in range(len(outputt)):
            for j in range(3):
                if i != 0 or j != 0:
                    ser.write(".......".encode())
                    print(".......")
                    time.sleep(3)
                if len(outputt[i].strip()) > 0:
                    print(outputt[i].strip())
```

```
            ser.write(outputt[i].encode())
            time.sleep(3)
          else:
            print(" ")
          if outputt[i].strip() != "" and outputt[i] != ".......":
            time.sleep(5)
          else:
            print(outputt[i])
            ser.write(".......".encode())
            print(".......")
            time.sleep(3)
          if i != len(outputt) - 1:
            time.sleep(3)
      ser.write(".......".encode())  # Send "......." as the last encode for the serial
      print(".......")
      time.sleep(3)
    else:
      print("Failed to open serial connection")
except serial.SerialException as e:
    print("Error opening serial connection:", e)
finally:
    ser.close()
# Turn on the buzzer for 0.2 seconds
```

```
    GPIO.output(buzzer_pin, GPIO.HIGH)
    time.sleep(0.2
def select(value):
    if value == ' Space ':
        textarea.insert(INSERT, ' ')
    elif value == 'Tab':
        textarea.insert(INSERT, '\t')
    elif value == 'del':
        i = textarea.get(1.0, END)
        newtext = i[:-2]
        textarea.delete(1.0, END)
        textarea.insert(INSERT, newtext)
    else:
        textarea.insert(INSERT, value)
    textarea.focus_set()
label1 = tk.CTkLabel(Keyboard_App, text="Keyboard for Text to Braille",
font=('Palatino',30,'bold'),pady=10).grid(row=0,columnspan=10)
textarea = Text(Keyboard_App, width=55, height=7,
font=('Palatino',20,'bold'),padx=15,pady=10)
textarea.grid(row=1,columnspan=10)
buttons = [
    'q','w','e','r','t','y','u','i','o','p',
    'a','s','d','f','g','h','j','k','l','del',
```

```
    'shift','z','x','c','v','b','n','m',
    ' Space '
]
varRow = 2
varColumn= 0
def send_message(event=None):
    message = textarea.get("1.0", END)
    textarea.delete(1.0, END)
    print_text_with_buzzer(message)
for button in buttons:
    command = lambda x=button: select(x)
    if button != " Space ":
        tk.CTkButton(Keyboard_App, text=button, width=70,
command=command,height=60, font=('Palatino',23,'bold')).grid(row=varRow,
column=varColumn, padx=4, pady=5)
    if button == " Space ":
        tk.CTkButton(Keyboard_App, text=button, width=70,
command=command,height=60, font=('Palatino',20,'bold')).grid(row=varRow,
column=varColumn, padx=0, pady=5)
    varColumn += 1
    if varColumn > 9 and varRow == 2:
        varColumn = 0
        varRow += 1
```

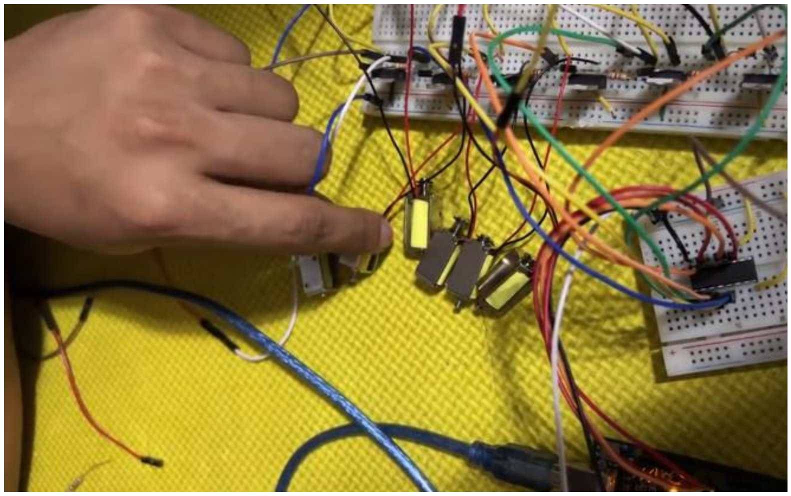
```
    if varColumn > 9 and varRow == 3:
        varColumn = 0
        varRow += 1
class TextSystem():
    def __init__(self, master=None):
        self.master = master
    def AudioBraille(self):
        os.system('python berealspeech.py')   # execute the system call to open the new
Python file
        self.master.withdraw()
        os._exit(0)
    def ImageBraille(self):
        os.system('python berealimage.py')  # execute the system call to open the new
Python file
        self.master.withdraw()
        os._exit(0)
    #
    @staticmethod
    def check_buttonspeech(pin):
        if GPIO.input(buttonspeech) == False:
            a.AudioBraille()

    @staticmethod
```

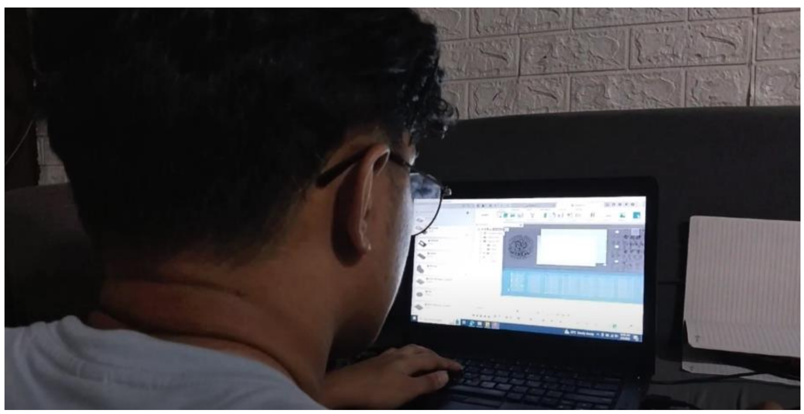
```
    def check_buttonimage(pin):
        if GPIO.input(buttonimage) == False:
            a.ImageBraille()
a = TextSystem(Keyboard_App)
#
GPIO.add_event_detect(buttonspeech, GPIO.FALLING,
callback=a.check_buttonspeech, bouncetime=100)
GPIO.add_event_detect(buttonimage, GPIO.FALLING,
callback=a.check_buttonimage, bouncetime=100)
Enter = tk.CTkButton(Keyboard_App, text=' Enter ', width=70,height=60,
font=('Palatino',23,'bold'), command=send_message)
Enter.place(x=735, y=415)
Keyboard_App.bind('<Return>', send_message)
Keyboard_App.mainloop()
```

ANNEX III

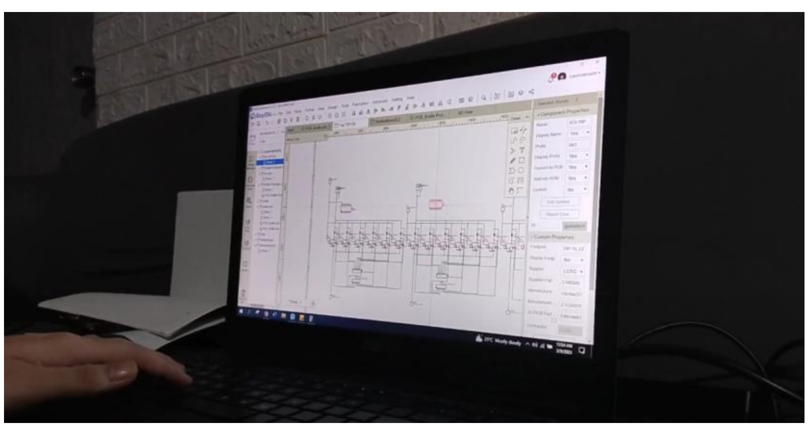
PROJECT DOCUMENTATION



Calibrating of electromagnetic push-pull solenoid actuator



3D designing of chassis



Designing of PCB layout

Testing of image to braille function

Project Deployment at Bagong Silang Elementary School

Project Deployment at Bagong Silang Elementary School

Project Deployment at Bagong Silang Elementary School

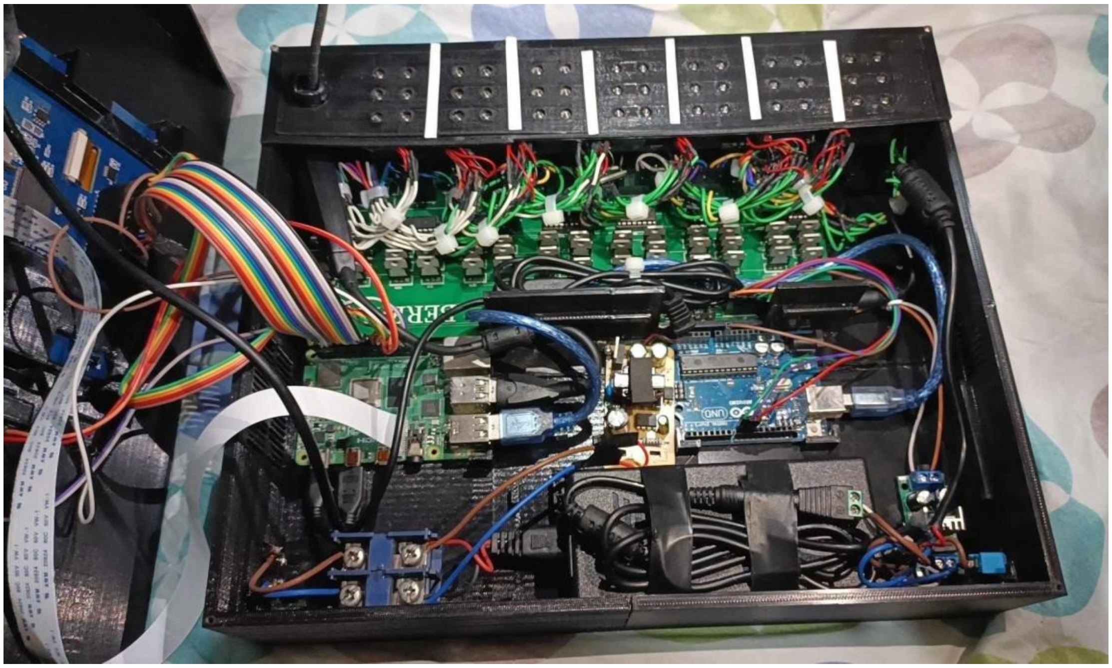
Project Deployment at P.Gomez Elementary School

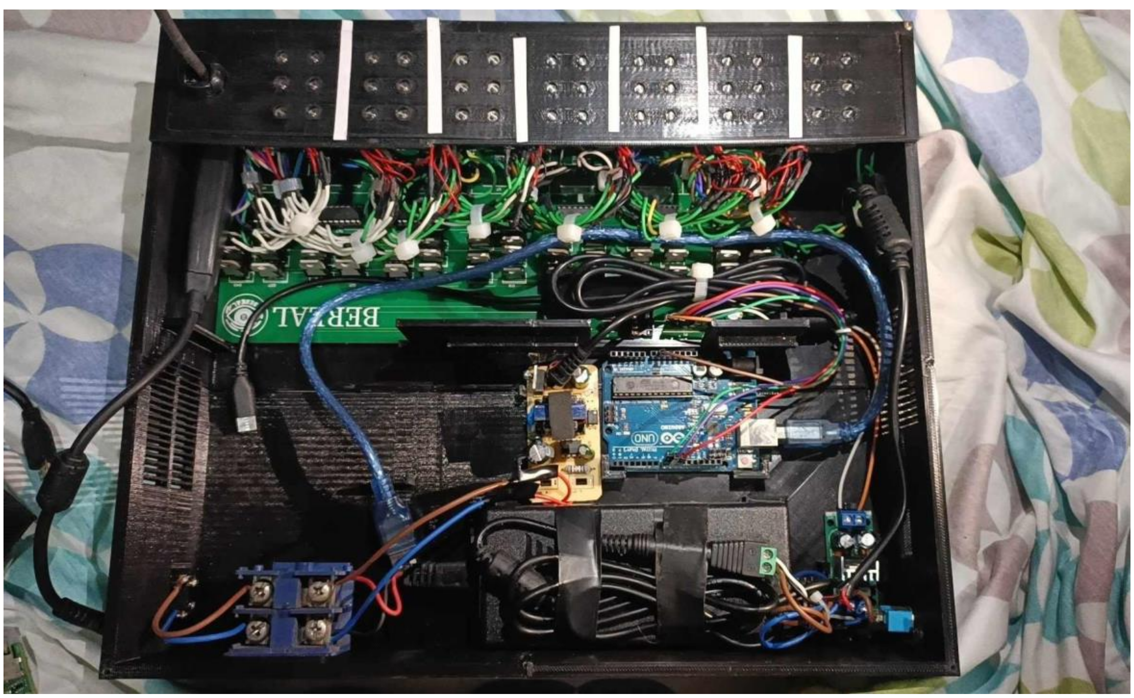
Progress Defense

Pre-Final Defense

Final Defense

The Connection Inside the BeReal





Whiteboard that will be used for Image-toText purposes

ANNEX IV

EVALUATION FORM

Republic of the Philippines
TECHNOLOGICAL UNIVERSITY OF THE PHILIPPINES
COLLEGE OF ENGINEERING
Ayala Blvd., corner San Marcelino St., Ermita, 1000 Manila
Fax: (62-2) 521-4063, Tel. Nos. (63-2) 523-2293, Trunkline: 302-7750 loc. 112
Website: http://www.tup.edu.ph

BEREAL SURVEY FORM

BEREAL: A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Please take a few minutes to complete the survey form below. Read and answer the question truthfully. All the information gathered in this survey will remain confidential and will only use for research purposes. Thank you very much!

Part I. Demographic Information

1. Name (Optional)

   ____

2. Age (Optional)

   ____

3. Gender (Optional)

   ____

4. What is your current occupation?
   - Student
   - Employed
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception)
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No

7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?					
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?					
3	Can you read Grade 2 braille?					
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?					
5	Do you find the conversion of image-to-braille to be useful for learning purposes?					
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?					
7	Did the device meet the needs of blind users?					

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?					
2	How satisfied are you with the refreshable braille display's user-friendliness?					

3	How satisfied are you with the refreshable braille display's durability?					
4	How clear were the instructions for using the product/service?					
5	How relevant and useful is the product/service for blind people?					
6	How satisfied are you with the overall quality of our device?					

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?					
2	How likely are you to recommend our device to a friend or colleague?					

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

____

____

____

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)
   Mark Daniel
2. Age (Optional)
   7
3. Gender (Optional)
   Male
4. What is your current occupation?
   - Student /
   - Employed
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception) /
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No /
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before /

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Strongly Disagree
1	Have you used a screen reader or braille device before?					✓
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?				✓	
3	Can you read Grade 2 braille?				✓	
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?		✓			
5	Do you find the conversion of image-to-braille to be useful for learning purposes?		✓			
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?				✓	
7	Did the device meet the needs of blind users?		✓			

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?			✓		
2	How satisfied are you with the refreshable braille display's user-friendliness?			✓		
3	How satisfied are you with the refreshable braille display's durability?		✓			
4	How clear were the instructions for using the product/service?	✓				
5	How relevant and useful is the product/service for blind people?	✓				
6	How satisfied are you with the overall quality of our device?		✓			

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?		✓			
2	How likely are you to recommend our device to a friend or colleague?	✓				

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

The size of the braille cell is too big compared to the printed document.

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)
   JM
2. Age (Optional)
   11
3. Gender (Optional)
   Male
4. What is your current occupation?
   - Student /
   - Employed
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception)
   - Blind (some light perception) /
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No /
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before /

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?					✓
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?	✓				
3	Can you read Grade 2 braille?	✓				
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?	✓				
5	Do you find the conversion of image-to-braille to be useful for learning purposes?	✓				
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?	✓				
7	Did the device meet the needs of blind users?	✓				

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?			✓		
2	How satisfied are you with the refreshable braille display's user-friendliness?				✓	
3	How satisfied are you with the refreshable braille display's durability?		✓			
4	How clear were the instructions for using the product/service?			✓		
5	How relevant and useful is the product/service for blind people?		✓			
6	How satisfied are you with the overall quality of our device?		✓			

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?			✓		
2	How likely are you to recommend our device to a friend or colleague?	✓				

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

- Duration of uptime, lack of time to read the message
- The user was distracted to the sudden up and down of the actuator

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)
   Mica
2. Age (Optional)
   13
3. Gender (Optional)
   Female
4. What is your current occupation?
   - Student /
   - Employed
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception)
   - Blind (some light perception) /
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No /
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before /

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?					/
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?	/				
3	Can you read Grade 2 braille?					/
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?			/		
5	Do you find the conversion of image-to-braille to be useful for learning purposes?		/			
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?			/		
7	Did the device meet the needs of blind users?	/				

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?	/				
2	How satisfied are you with the refreshable braille display's user-friendliness?			/		
3	How satisfied are you with the refreshable braille display's durability?		/			
4	How clear were the instructions for using the product/service?			/		
5	How relevant and useful is the product/service for blind people?		/			
6	How satisfied are you with the overall quality of our device?			/		

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?		/			
2	How likely are you to recommend our device to a friend or colleague?		/			

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

Braillet is easier compared to BEREAL because braillet has smaller braille cell which is easier to read the message.

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)
   Carlito
2. Age (Optional)
   15
3. Gender (Optional)
   Male
4. What is your current occupation?
   - Student ✓
   - Employed
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception) ✓
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No ✓
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before ✓

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?					✓
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?	✓				
3	Can you read Grade 2 braille?					✓
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?	✓				
5	Do you find the conversion of image-to-braille to be useful for learning purposes?	✓				
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?				✓	
7	Did the device meet the needs of blind users?	✓				

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?	✓				
2	How satisfied are you with the refreshable braille display's user-friendliness?	✓				
3	How satisfied are you with the refreshable braille display's durability?	✓				
4	How clear were the instructions for using the product/service?	✓				
5	How relevant and useful is the product/service for blind people?	✓				
6	How satisfied are you with the overall quality of our device?	✓				

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?	✓				
2	How likely are you to recommend our device to a friend or colleague?	✓				

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

- make it commercially available in shoppee /lazada

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)
   Angel
2. Age (Optional)
   16
3. Gender (Optional)
   Female
4. What is your current occupation?
   - Student /
   - Employed
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception) /
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No /
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before /

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?					✓
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?		✓			
3	Can you read Grade 2 braille?					✓
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?			✓		
5	Do you find the conversion of image-to-braille to be useful for learning purposes?		✓			
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?				✓	
7	Did the device meet the needs of blind users?	✓				

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?				✓	
2	How satisfied are you with the refreshable braille display's user-friendliness?			✓		
3	How satisfied are you with the refreshable braille display's durability?	✓				
4	How clear were the instructions for using the product/service?	✓				
5	How relevant and useful is the product/service for blind people?			✓		
6	How satisfied are you with the overall quality of our device?	✓				

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?	✓				
2	How likely are you to recommend our device to a friend or colleague?	✓				

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

confused on the movement of the actuators because they are familiar with the embossed braille material.

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)
   Clint person Rendon
2. Age (Optional)
   11
3. Gender (Optional)
   Male
4. What is your current occupation?
   - Student /
   - Employed
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception) /
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No /
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before /

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?					✓
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?		✓			
3	Can you read Grade 2 braille?		✓			
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?		✓			
5	Do you find the conversion of image-to-braille to be useful for learning purposes?		✓			
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?				✓	
7	Did the device meet the needs of blind users?		✓			

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?	✓				
2	How satisfied are you with the refreshable braille display's user-friendliness?		✓			
3	How satisfied are you with the refreshable braille display's durability?		✓			
4	How clear were the instructions for using the product/service?		✓			
5	How relevant and useful is the product/service for blind people?		✓			
6	How satisfied are you with the overall quality of our device?	✓				

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?	✓				
2	How likely are you to recommend our device to a friend or colleague?	✓				

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

- Add another language (Filipino)
- duration of uptime, lack of time

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)
   Sienna
2. Age (Optional)
   ____
3. Gender (Optional)
   Female
4. What is your current occupation?
   - Student
   - Employed ✓
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception)
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No ✓
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before ✓

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?	✓				
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?	✓				
3	Can you read Grade 2 braille?	✓				
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?	✓				
5	Do you find the conversion of image-to-braille to be useful for learning purposes?	✓				
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?			✓		
7	Did the device meet the needs of blind users?		✓			

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?			✓		
2	How satisfied are you with the refreshable braille display's user-friendliness?			✓		
3	How satisfied are you with the refreshable braille display's durability?	✓				
4	How clear were the instructions for using the product/service?		✓			
5	How relevant and useful is the product/service for blind people?		✓			
6	How satisfied are you with the overall quality of our device?		✓			

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?		✓			
2	How likely are you to recommend our device to a friend or colleague?		✓			

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

The 4 seconds uptime duration of the braille dots is not enough. Make it 30 seconds above for the students to read it properly.

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)

   ____

2. Age (Optional)

   ____

3. Gender (Optional)

   Female

4. What is your current occupation?
   - Student
   - Employed /
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception)
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No /
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before /

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?	✓				
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?	✓				
3	Can you read Grade 2 braille?	✓				
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?				✓	
5	Do you find the conversion of image-to-braille to be useful for learning purposes?	✓				
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?	✓				
7	Did the device meet the needs of blind users?			✓		

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?		✓			
2	How satisfied are you with the refreshable braille display's user-friendliness?		✓			
3	How satisfied are you with the refreshable braille display's durability?		✓			
4	How clear were the instructions for using the product/service?	✓				
5	How relevant and useful is the product/service for blind people?		✓			
6	How satisfied are you with the overall quality of our device?		✓			

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?	✓				
2	How likely are you to recommend our device to a friend or colleague?		✓			

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

The device will be very useful for the blind learners for instructional materials purposes specifically for beginning readers. Make the braille dots smaller for the blind to easily read the dots. And more cells to make it possible to use it for communication purposes.

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

BEREAL:

A Modular Low-Cost Two-Way Refreshable Braille Display Communication and Learning Device for Blind Person

Thank you for participating in this survey about our refreshable braille display. Your feedback is important to us and will help us improve our work. Please take a few minutes to complete this survey.

Part I. Demographic Information

1. Name (Optional)

   ____

2. Age (Optional)

   ____

3. Gender (Optional)

   Female

4. What is your current occupation?
   - Student
   - Employed ✓
   - None
   - Others
5. What is your level of vision impairment?
   - Blind (no light perception)
   - Blind (some light perception)
   - Low Vision (able to see some large print or magnification)
   - Prefer not to say.
6. Have you ever used a refreshable braille display before?
   - Yes
   - No ✓
7. How long have you been using a refreshable braille display?
   - Less than 6 months
   - 6 months to 1 year
   - 1 to 3 years
   - 3 to 5 years
   - More than 5 years
   - Never used one before ✓

Part II. Product Satisfaction

	Survey Questions	Strongly Agree	Agree	Unsure	Disagree	Disagree
1	Have you used a screen reader or braille device before?	✓				
2	Are you familiar with both Grade 1 and Grade 2 braille alphabets?	✓				
3	Can you read Grade 2 braille?	✓				
4	Do you find the conversion of speech-to-text and text-to-braille to be useful and relevant for communication purposes?	✓				
5	Do you find the conversion of image-to-braille to be useful for learning purposes?	✓				
6	Are you knowledgeable about assistive technologies such as screen readers and braille devices?	✓				
7	Did the device meet the needs of blind users?	✓				

Part III. Customer Satisfaction

	Survey Questions	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
1	How satisfied are you with the refreshable braille display's braille output quality?		✓			
2	How satisfied are you with the refreshable braille display's user-friendliness?		✓			
3	How satisfied are you with the refreshable braille display's durability?	✓				
4	How clear were the instructions for using the product/service?	✓				
5	How relevant and useful is the product/service for blind people?		✓			
6	How satisfied are you with the overall quality of our device?	✓				

Part IV. Likelihood to Recommend

	Survey Questions	5	4	3	2	1
1	How likely are you to continue using our device in the future?		✓			
2	How likely are you to recommend our device to a friend or colleague?	✓				

Part V. Feedback

Please provide any additional feedback or comments you may have about the device in the space below.

Make it smaller since students are more familiar with embossed braille materials. Provide more character of braille cells.

Thank you for completing this survey. Your feedback is valuable to us and will help us improve our device.

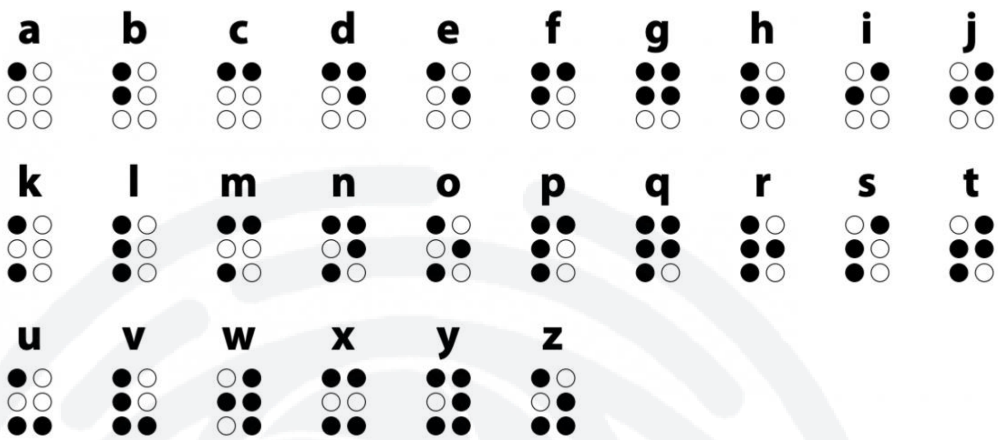
ANNEX V
BEREAL USER'S MANUAL

BEREAL USER'S MANUAL

VERSION 1.0
AUGUST 2023

USER'S MANUAL

BRAILLE ALPHABET



ACTUAL PROTOTYPE

FUNCTIONS

1 Touchscreen LCD - This will serve as the screen for displaying the output of speech to text, and keyboard for typing the text.

2. Push Buttons - There are **three** colored buttons on the Bereal with their different functionalities:

- **Red** - Speech to Text Conversion
- **Blue** - Text to Braille Conversion

- **White** - Scanned Images to Braille Conversion

3.Camera - Used for scanning images.

4. Braille Cells - Depending on the user's input, the braille cells will move upwards to show the converted output.

5. Microphone - Used for speech purposes.

USER GUIDE

To turn on the Bereal:

- Plug the power supply.
- Make sure to toggle on the switch. It is located at the right side of the device. This helps to turn on the braille cells.
  - The upward position corresponds to the activation or "ON" state, whereas the downward position represents the deactivation or "OFF" state of the toggle.
- Once the power supply is plugged, the touchscreen LCD will automatically light up.

How it works:

- Once the Bereal is open, the user has an option to choose whether to use the touchscreen LCD or press the colored pushbuttons on how they want to use the device.

> **For Speech to Text Conversion (RED):**

- Use the microphone to record the speech.
- Double tap the button to record for speech to text conversion.
- The text output will be displayed on the touchscreen LCD.

> **For Text to Braille Conversion (BLUE):**

- Use the touchscreen LCD as a keyboard to type the text.
- The output will be displayed using the braille cells. This output will be shown three times with an uptime of ten seconds and a delay of three seconds.

> **For Scanned Images to Braille (WHITE):**

- Use the camera to scan the images.

- Double tap the button to capture the image for image to braille conversion.
- Once scanned, the text will be displayed on the touchscreen LCD.
- The whiteboard provided is intended for users to write their input and scan it using the camera. The whiteboard is equipped with guidelines to assist users in their writing process.

To turn off the Bereal:

- Toggle off the switch
- Unplug the power supply

SAFETY REMINDERS

1 Refrain from using the device for 30 minutes.

2. Avoid sudden plugging and unplugging of the power supply.

3. After using the device, allow it to rest for 15 to 30 minutes before using it again.

ANNEX VI

MANUAL ON HOW TO DUPLICATE THE PROTOTYPE

MANUAL ON HOW TO DUPLICATE THE PROTOTYPE

VERSION 1.0
AUGUST 2023

MANUAL ON HOW TO DUPLICATE THE PROTOTYPE

PART I - DESIGN

1 Hardware

1.1 Circuit Design

Circuit design involves the selection and arrangement of electronic components to create a functional circuit capable of performing a desired task. Designers also need to consider factors such as power consumption. The use of computer-aided design (CAD) tools is helpful in designing a schematic circuit and simulating it before physical implementation. EasyEDA is the one used in this project among the various CAD tools available.

To design the PCB used in this project, it is necessary to break down the procedure into step-by-step stages.

- Initial Planning - defining the functionality of the design, and considering the components that will be used.
- Schematic Design - Schematic Design involves creating a schematic diagram, selecting, and connecting the electronic components while ensuring proper functionality.
- Routing - once the components are placed and connected, the designer proceeds with routing the connections on the PCB.

- Design Rule Check (DRC) - After completing the routing, a DRC is performed using the CAD tool. This automated check verifies if the design conforms to specific manufacturing and electrical design rules.
- Gerber File Generation - Once the design passes the DRC, the designer generates Gerber files, which are industry-standard files that contain the PCB design information required for manufacturing.
- PCB Fabrication - The Gerber files are sent to a PCB fabrication service or manufacturer to produce a prototype PCB.
- Final Product - The design files are sent to the manufacturer for mass production, and the finished PCBs are assembled into the final product.

1.1.1 Circuit Materials

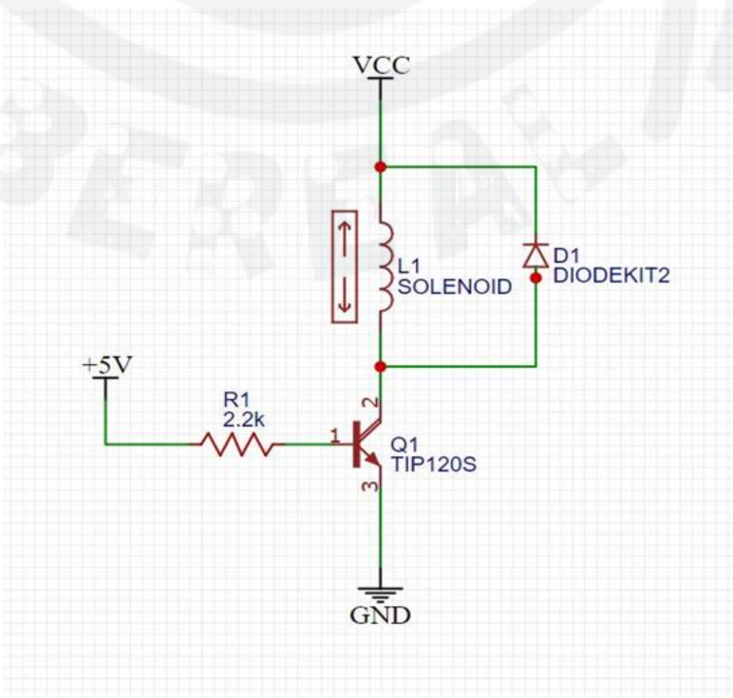
- **TIP120 -** The TIP120 transistor is a type of NPN Darlington Power Transistor commonly used for controlling motors, solenoids, or high-power LEDs in medium to high-power electronic applications. It has the capability to handle loads up to 60V and can handle peak currents of 8A, with a continuous current rating of 5A.
- **1N4004 -** The 1N4004 is a rectifier diode designed to handle currents up to a maximum of 1A, and it has the ability to withstand peak currents of up to 30A.
- **2.2k Ω -** A 2.2k ohm resistor is an electronic component that offers a resistance of 2.2 kilo-ohms, or 2200 ohms.
- **74HC595 -** The 74HCT595 is an integrated circuit that combines an 8-bit shift register with 3-state output registers. It is commonly used in

digital electronics to store and shift 8 bits of data while also providing the ability to control the output states of the registers, allowing for efficient data sharing and reduced pin count in communication systems.

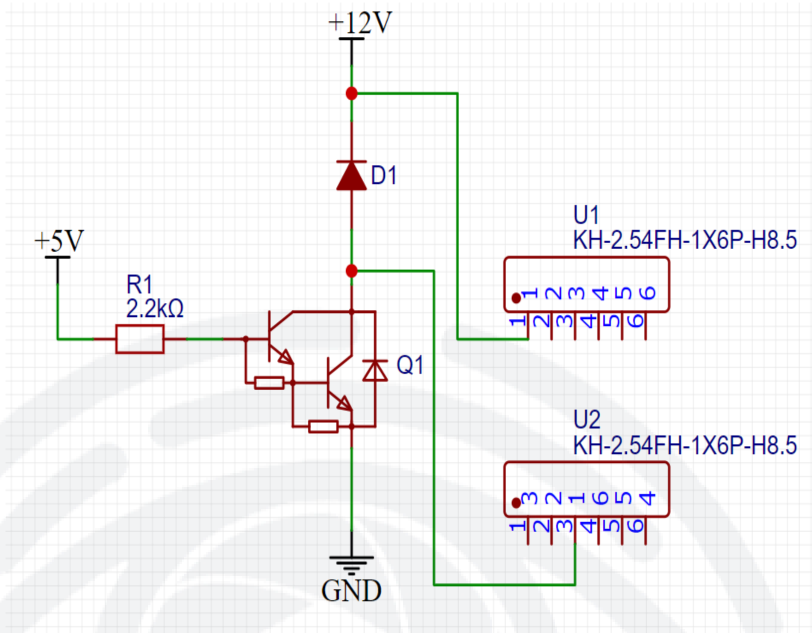
- **Female Pin Header -** A female pin header refers to a connector or socket that has a row of receptacles or sockets designed to receive male pins. Female pin headers are often used on printed circuit boards (PCBs) to create interfaces for external connections or to enable plug-in modules to be attached securely.

1.1.2 Procedure

1. Install EasyEDA to design the schematic and PCB.
2. This is the general circuit connection of a push-pull solenoid incorporating a 1N4004 diode and TIP120 transistor.



3. Replace the solenoid with the 1x6 female pin header.

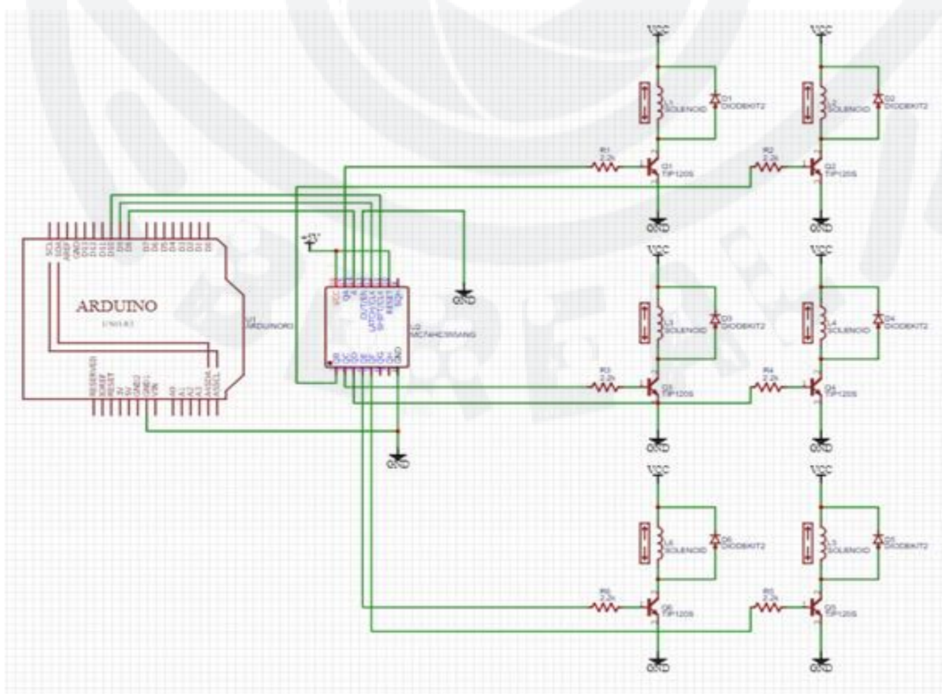


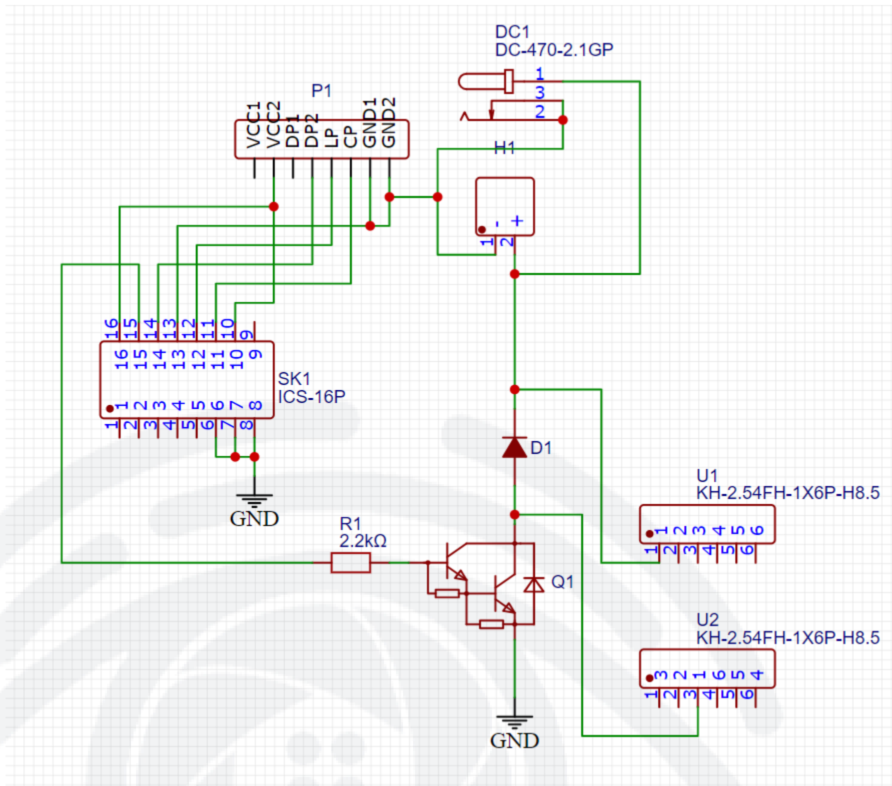
4. Connect the shift register to the circuit described above or simply follow the figure below and connect the 1x8 female pin header, which will serve as the receptacle for the power supply and Arduino. Additionally, add another 1x2 female pin header for the power supply and a DC barrel jack.

   For Shift Register:

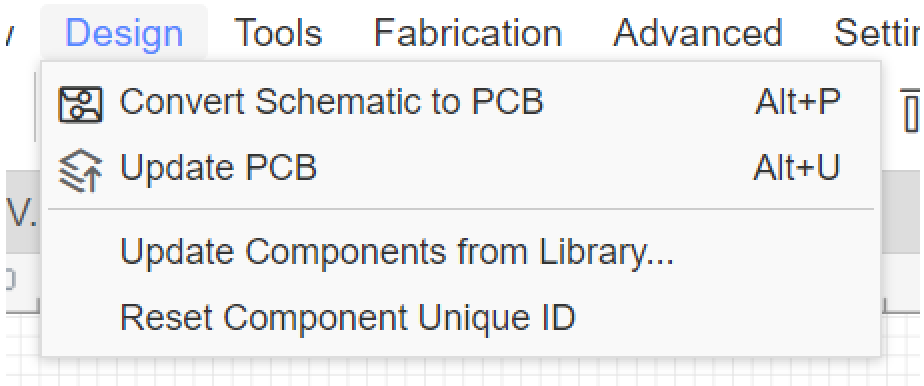
   - Connect the VCC to 5V.
   - Connect the Serial Data Input of the shift register to pin 8 of the Arduino.
   - Connect the Shift Register Clock In to pin 9 of the Arduino.
   - Connect the Storage Register Clock In to pin 10 of the Arduino.

- Connect the Master Reset to 5V.
- Connect Q7' of the shift register to the Data Input of the next shift register.
- Connect Q0 to the first solenoid circuit.
- Connect Q1 to the second solenoid circuit.
- Connect Q2 to the third solenoid circuit.
- Connect Q3 to the fourth solenoid circuit.
- Connect Q4 to the fifth solenoid circuit.
- Connect Q5 to the sixth solenoid circuit.
- Repeat this process to complete the remaining 42 solenoid circuits.

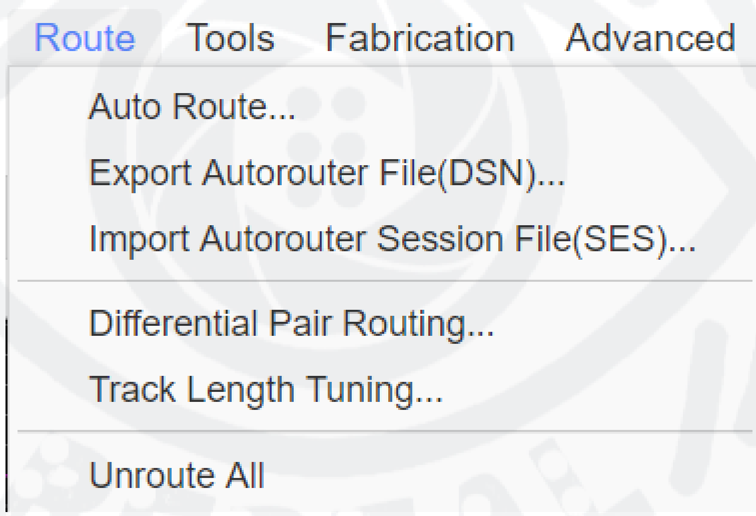




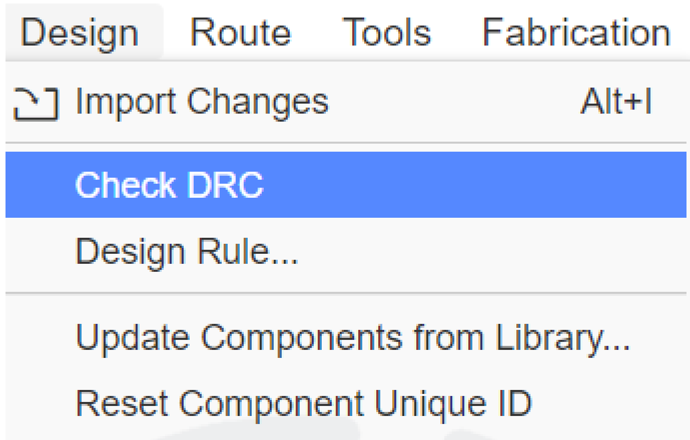
5. Connect the 6 outputs of the shift register to the resistor of each TIP120 transistor. The transistor will serve as a switch for the solenoid. Connect pin 9 of the shift register to the data input of the next shift register. Repeat the same process until you complete the shift register needed for the solenoid used.
6. Check the connections of the schematic diagram.
7. To convert the schematic to PCB, go to Design > Convert Schematic to PCB > Yes, Check Nets, or simply use the hotkey Alt+P



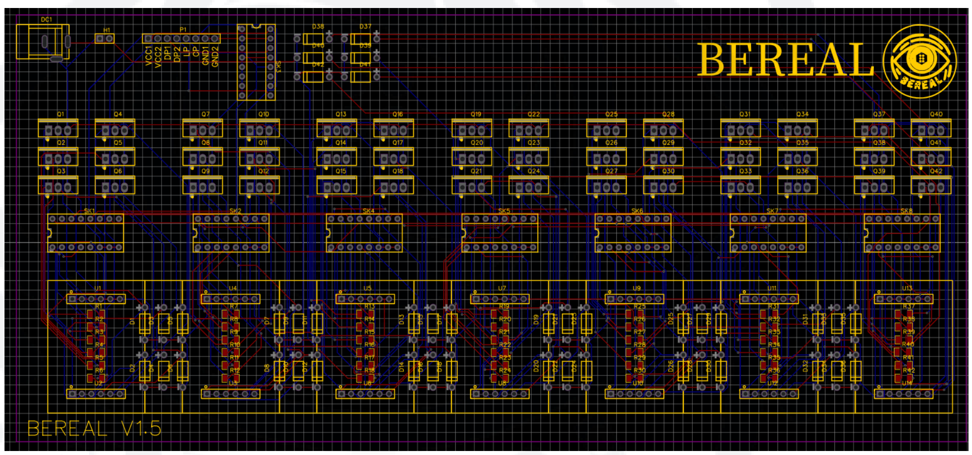
8. Arrange the position of the components and adjust the PCB board size. After that, go to Route > Auto Route.



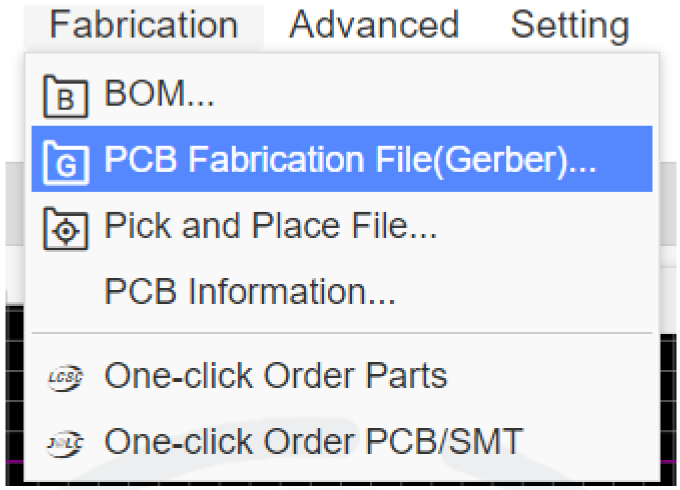
9. After routing the connections, check the design using the DRC checker. Go to Design > Check DRC.



10. After completing the PCB design and checking the design rules, we can now proceed to generate the Gerber file.



11. To generate the Gerber file, go to Fabrication > PCB Fabrication File(Gerber) > Yes, Check DRC > Generate Gerber.

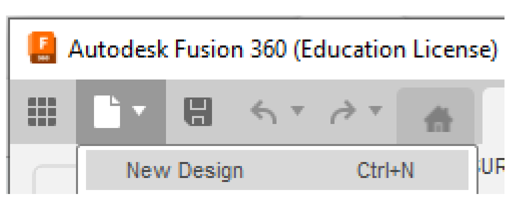


12. Send the Gerber file to your desired manufacturer and provide instructions regarding the desired quantity of PCBs and any additional requirements or specifications you may have.

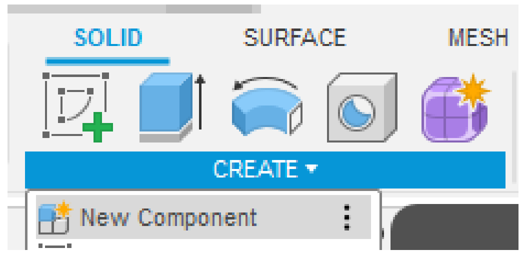
1.2 3D Design

1.2.1 3D Chassis Materials

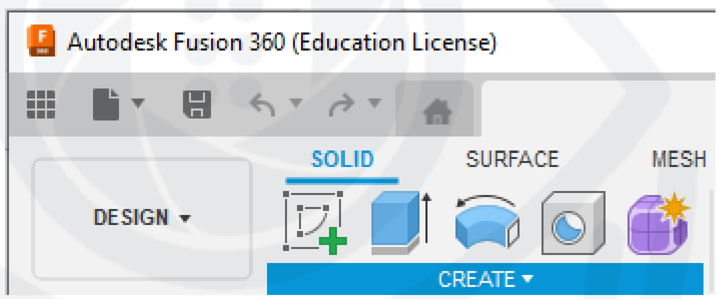
- The software used to design the chassis is Autodesk Fusion 360. You can get the software from the official Autodesk website and follow the instructions provided during the installation process.
- Next step is to create a new design file by clicking on "New Design" from the startup screen or select File from the menu bar, then choose "New Design".



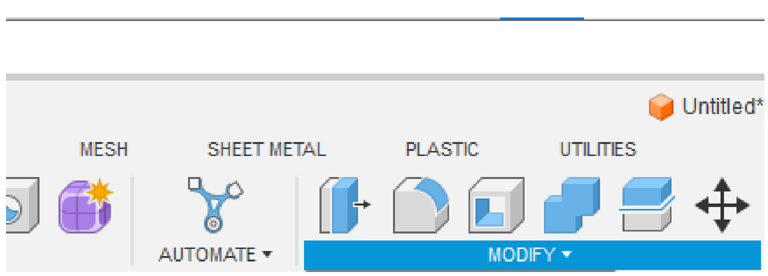
- Click the Create toolbar and select New Component. This will allow you to separate the design into different components or parts.



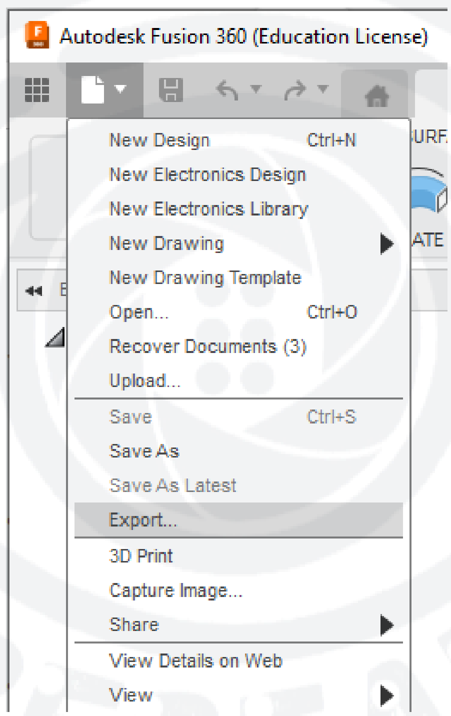
- In designing, access different shapes by clicking on "Create" in the toolbar at the top of the window which will appear a drop-down menu, and you can choose from options like Box, Cylinder, Sphere, etc. The device dimension is 12x6x6 inches.



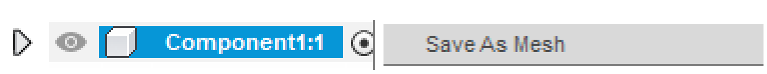
- After creating a shape, you can modify it to create your desired design by clicking the "Modify" toolbar. You can use commands like Extrude, Loft, Fillet, and more commands to create and modify your design.



- To save your design, click on the "Save" icon in the toolbar or select "File" from the menu bar and choose "Save". If you want to export your work file, click “Export”. This will allow you to name your file and select the desired file location on your computer. Then, you can choose file type fusion Archive Files (.f3d) for editable fusion 360 3D Designs.



- If you’re finished in designing, you can convert your file to STL File (.stl) by right clicking the desired component and clicking the option “Save as Mesh” then a new window will appear and just click “OK”. STL File is used for 3D Printing purposes.



2 Software

2.1 Arduino

This study uses Arduino Uno R3 to receive the data from the raspberry pi to control the solenoids.

Code:

```
// Pin definitions
const int dataPin = 8;
const int clockPin = 10;
const int latchPin = 9;
const int numRegisters = 1;
const int numCharacters = 6;

// Braille code for each character
byte brailleCodes[] = {
  B000001, // A
  B000011, // B
  B100001, // C
  B110001, // D
  B010001, // E
  B100011, // F
  B110011, // G
  B010011, // H
  B100010, // I
  B110010, // J
```

```
  B000101, // K
  B000111, // L
  B100101, // M
  B110101, // N
  B010101, // O
  B100111, // P
  B110111, // Q
  B010111, // R
  B100110, // S
  B110110, // T
  B001101, // U
  B001111, // V
  B111010, // W
  B101101, // X
  B111101, // Y
  B011101, // Z
};

void setup() {
  // Set pin modes
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(latchPin, OUTPUT);

  // Initialize serial communication
```

```
  Serial.begin(9600);
}

void loop() {
  // Read input from serial
  if (Serial.available() > 0) {
    // Read the input string
    String inputString = Serial.readString();

    // Reverse the input string
    String reversedString = "";
    for (int i = inputString.length() - 1; i >= 0; i--) {
      reversedString += inputString.charAt(i);
    }

    // Convert each character to Braille code and display on Braille display
    for (int i = 0; i < reversedString.length(); i++) {
      // Convert character to Braille code
      byte brailleCode = getBrailleCode(reversedString.charAt(i));

      // Shift out the Braille code to the shift registers
      for (int j = 0; j < numRegisters; j++) {
        shiftOut(dataPin, clockPin, MSBFIRST, brailleCode);
      }
```

```
      // Latch the data to the output
      digitalWrite(latchPin, HIGH);
      delay(1);
      digitalWrite(latchPin, LOW);
    }
  }
}
byte getBrailleCode(char inputChar) {
  // Convert input character to uppercase
  inputChar = toupper(inputChar);

  // Check if input character is a letter
  if (inputChar >= 'A' && inputChar <= 'Z') {
    // Return the corresponding Braille code
    return brailleCodes[inputChar - 'A'];
  } else {
    // Return blank code for non-letter characters
    return B00000000;
  }
}

String reverseString(String inputString) {
  String reversedString = "";

  // Loop through each character in the string
```

```
    for (int i = inputString.length() - 1; i >= 0; i--) {
      reversedString += inputString[i];
    }
    return reversedString;
  }
```

2.2 Raspberry Pi

In this study, the Raspberry Pi used was Raspberry Pi 4 Model B. In order to start using Raspberry Pi with your codes you need first to set up the said material.

- Raspberry Pi 4B
- 64 GB or larger SD card
- 5" Touchscreen IPS LCD Display for Raspberry Pi
- Microphone usb
- Raspberry Pi Camera module v1.3
- HDMI cable
- Necessary cables and wires

Process:

1. Obtain a 64 GB or larger SD card for the Raspberry Pi.

2. Insert the SD card into a computer or laptop.

3. Download the 64-bit OS for Raspberry Pi from the official website.

4. Use the Raspberry Pi app to write the OS to the SD card.

5. Remove the SD card from the computer and insert it into the Raspberry Pi.

6. Power on the Raspberry Pi and ensure that the OS is successfully installed.

7. Connect a 5 inch LCD screen to the Raspberry Pi using the appropriate GPIO pins.

5 inch LCD pins	**Raspberry pi 4 pins**	**Description**
1, 17	3.3V	Power positive (3.3V power input)
2, 4	5V	Power positive (5V power input)
3, 5, 7, 8, 10, 12, 13, 15, 16, 18, 24	NC	NC
11	Backlight Control	Control the backlight through pin 11
6, 9, 14, 20, 25	GND	Ground
19	TP_SI	SPI data input of Touch Panel
21	TP_SO	SPI data output of Touch Panel
22	TP_IRQ	Touch panel interrupt, low level while the touch panel detects touching
23	TP_SCK	SPI clock of touch panel
26	TP_CS	Touch panel chip selection, low active

8. Connect the HDMI connector to both the screen and the Pi.

9. Check the LCD touchscreen if it is working. if not do this

- Access the configuration file named "config.txt" located in the root directory.

- Append the provided code to the end of the "config.txt" file, and then save and exit the file.

```
hdmi_group=2
hdmi_mode=87
hdmi_cvt 800 480 60 6 0 0 0
hdmi_drive=1
dtoverlay=ads7846,cs=1,penirq=25,penirq_pull=2,speed=50000,keep_vref_on=0,swapxy=0,pmax=255,xohms=150,xmin=2
00,xmax=3900,ymin=200,ymax=3900
```

- The calibration of the display can be performed using the xinput-calibrator tool.
- Please execute the provided command to install the necessary software:

```
sudo cp -rf /usr/share/X11/xorg.conf.d/10-evdev.conf /usr/share/X11/xorg.conf.d/45-evdev.conf
sudo nano /usr/share/X11/xorg.conf.d/99-calibration.conf
```

- Add the following code to 99-calibration.conf:

```
Section "InputClass"
        Identifier      "calibration"
        MatchProduct    "ADS7846 Touchscreen"
        Option   "Calibration"   "208 3905 288 3910"
        Option   "SwapAxes"      "0"
        Option "EmulateThirdButton" "1"
        Option "EmulateThirdButtonTimeout" "1000"
        Option "EmulateThirdButtonMoveThreshold" "300"
EndSection
```

- After rebooting, the touch functionality should work normally under regular conditions. However, it's worth noting that for screens with different resistance properties, the default calibration parameters may not provide optimal accuracy.

- You can perform touch calibration by clicking the Raspberry Pi icon on the taskbar, selecting Preferences -> Calibrate Touchscreen, and following the displayed prompts.
- Upon completing the calibration process, the resulting touch data will be displayed. If you wish to preserve these touch values, you can replace the data in the "Calibration" option within the "99-calibration.conf" file with the corresponding values obtained from the calibration process.

10. To identify the connected microphone on your Raspberry Pi, please connect the microphone and run the following command to terminal to view all available recording devices:

    arecord -l

11. From this command, you should see something like below appear in the terminal.

    card 1: Microphone [Yeti Stereo Microphone], device 0: USB Audio [USB Audio] ***Subdevices: 1/1***
Subdevice #0: subdevice #0

12. The two things you need to pay attention to is the card number and the device number. If nothing appears, then make sure you have plugged your device in properly.
13. To start modifying the file, please execute the following command:

    nano /home/pi/.asoundrc

14. Within the specified file, input the following text. Ensure that you replace "[card number]" and "[device number]" with the respective values you obtained during the step of this process.

```
pcm.!default {
    type asym
    capture.pcm "mic"
}
pcm.mic {
    type plug
    slave {
        pcm "hw:[card number],[device number]"
    }
}
```

15. Once done, save the file by pressing CTRL + X, followed by Y, then enter

16. Connect your camera to the Raspberry Pi.

17. Open the terminal and execute the following command to edit the "config.txt" file using the nano text editor.

```
sudo nano /boot/config.txt
```

18. Locate the line that contains "#display_auto_detect=". The "#" symbol signifies that the line is commented out. Remove the "#" symbol from the beginning of the line to uncomment it.

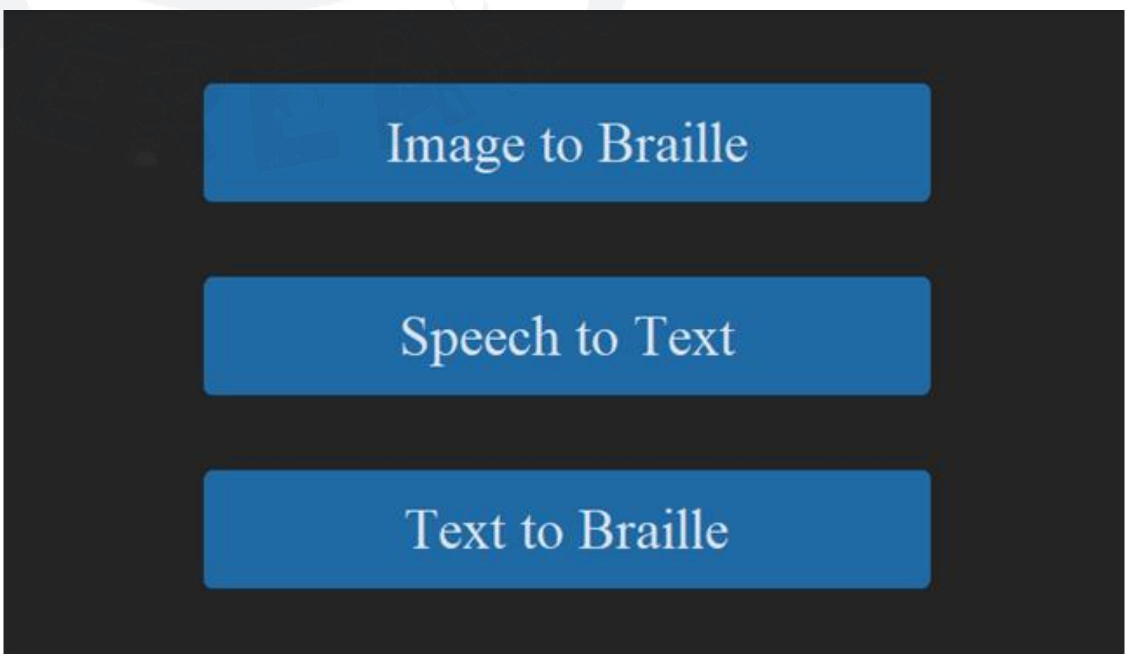
19. Add the following code at the end of the file:

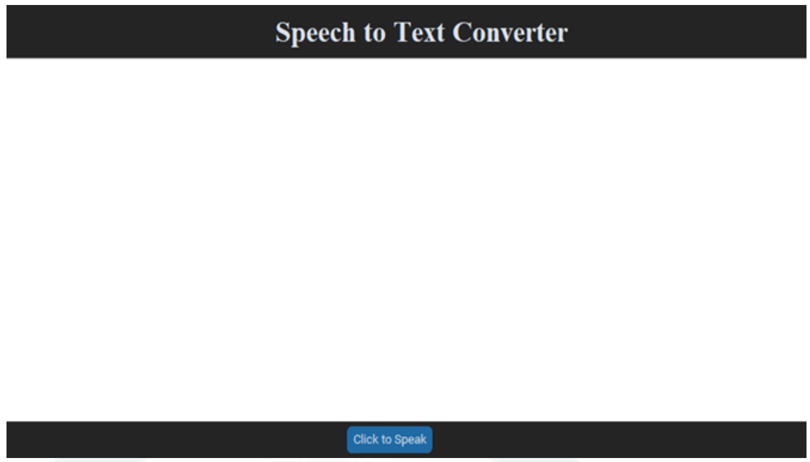
start_x=1

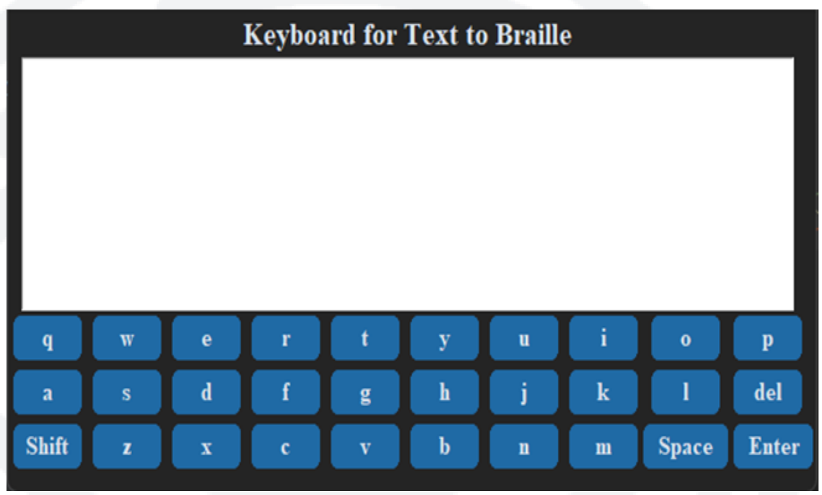
20. Save the changes by pressing "Ctrl + O" and then press "Enter". Exit the nano editor by pressing "Ctrl + X". Reboot your Raspberry Pi for the changes to take effect.

2.2.1 GUI

After setting up your raspberry pi, the program was done using Python. For a more user-friendly interface, the device has a GUI (Graphical User Interface) which allows the user to access and navigate through the device more easily.




Speech to Text Converter
Click to Speak


Keyboard for Text to Braille
q w e r t y u i o p
a s d f g h j k l del
Shift z x c v b n m Space Enter


Sample text here

BEREAL CODE FOR GUI

```
#!/usr/bin/env python
import tkinter
import os
import sys
import RPi.GPIO as GPIO
import subprocess
import pygame

# Set the PYTHONPATH environment variable
os.environ['PYTHONPATH'] = '/home/brbr/.local/lib/python3.9/site-
packages/customtkinter'

# Import the module
sys.path.insert(0, '/home/brbr/.local/lib/python3.9/site-packages/customtkinter')
import customtkinter

GPIO.setmode(GPIO.BCM)
buttontext = 13
buttonimage = 5
buttonspeech = 6
GPIO.setup(buttontext, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(buttonspeech, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buttonimage, GPIO.IN, pull_up_down=GPIO.PUD_UP)

customtkinter.set_appearance_mode("dark")  # Modes: system (default), light, dark
customtkinter.set_default_color_theme("blue")  # Themes: blue (default), dark-blue,
green

class BrailleSystem():
    def __init__(self, master=None):
        self.master = master
        self.tl_bg = "#000000"
        self.tl_bg2 = "#000000"
        self.tl_fg = "#000000"
        self.font = "Bodoni"
        # Initialize Pygame
        pygame.init()

        # Load the intro video
        self.intro_video = pygame.mixer.music
        # self.intro_video.load("/home/brbr/intro.mp4")
        #
        # # Play the intro video
        # self.intro_video.play()
```

```
        #Image
        self.buttonimg = customtkinter.CTkButton(root, text="Image to Braille",
font=("Palatino", 35), width=450, height=75, command=self.ImageBraille)
        self.buttonimg.place(relx= 0.5, rely= 0.2, anchor=tkinter.CENTER)

        #Audio
        self.buttonaud = customtkinter.CTkButton(root, text="Speech to Text",
font=("Palatino", 35), width=450, height=75, command=self.AudioBraille)
        self.buttonaud.place(relx= 0.5, rely= 0.5, anchor=tkinter.CENTER)

        #Keyboard
        self.buttontxt = customtkinter.CTkButton(root, text="Text to Braille",
font=("Palatino", 35), width=450, height=75, command=self.KeyboardBraille)
        self.buttontxt.place(relx= 0.5, rely= 0.8, anchor=tkinter.CENTER)

        # Schedule the stop of intro video after it finishes
        self.master.after(0, self.check_intro_status)

    def check_intro_status(self):
        if not pygame.mixer.music.get_busy():
            self.stop_intro()
        self.master.after(1000, self.check_intro_status)
```

```
def play_intro(self):
    # Play the intro video
    self.intro_video.play()
def stop_intro(self):
    # Stop the intro video
    self.intro_video.stop()

def AudioBraille(self):
    self.stop_intro()
    self.buttonimg.destroy()
    self.buttonaud.destroy()
    self.buttontxt.destroy()
    os.system('python berealspeech.py')
    self.master.withdraw()
    os._exit(0)

def KeyboardBraille(self):
    self.stop_intro()
    self.buttonimg.destroy()
    self.buttonaud.destroy()
    self.buttontxt.destroy()
    os.system('python berealtext.py')
```

```
        self.master.withdraw()
        os._exit(0)

    def ImageBraille(self):
        self.stop_intro()
        self.buttonimg.destroy()
        self.buttonaud.destroy()
        self.buttontxt.destroy()
        os.system('python berealimage.py')
        self.master.withdraw()
        os._exit(0)

    @staticmethod
    def check_buttontext(pin):
        if GPIO.input(buttontext) == False:
            a.KeyboardBraille()

    @staticmethod
    def check_buttonspeech(pin):
        if GPIO.input(buttonspeech) == False:
            a.AudioBraille()

    @staticmethod
```

```
    def check_buttonimage(pin):
        if GPIO.input(buttonimage) == False:
            a.ImageBraille()


GPIO.add_event_detect(buttontext, GPIO.FALLING,
callback=BrailleSystem.check_buttontext, bouncetime=100)
GPIO.add_event_detect(buttonspeech, GPIO.FALLING,
callback=BrailleSystem.check_buttonspeech, bouncetime=100)
GPIO.add_event_detect(buttonimage, GPIO.FALLING,
callback=BrailleSystem.check_buttonimage, bouncetime=100)


root = customtkinter.CTk()
root.geometry("800x480")
a = BrailleSystem(root)
# root.attributes('-fullscreen', True)
root.geometry('700x300')
root.configure(bg='black')
root.title("Bereal")
root.mainloop()
root.destroy()
```

BEREAL IMAGE-TO-BRAILLE CODE

```
import cv2

import pytesseract

from PIL import Image

import tkinter as tk

import RPi.GPIO as GPIO

import os

import time

import serial

import subprocess


pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR'


# Set up GPIO

GPIO.setmode(GPIO.BCM)

buttontext = 13

cameracapture = 5

buttonspeech = 6

buzzer_pin = 26

GPIO.setup(buttontext, GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(buttonspeech, GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(cameracapture, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(buzzer_pin, GPIO.OUT)


# Capture image from camera
cap = cv2.VideoCapture(0)


# Click counter
clicks = 0
def print_text_with_buzzer(text):
    textt = text
    substrings = [textt[i:i + 7] for i in range(0, len(textt), 7)]
    output = ""
    for s in substrings:
        s += " " * (7 - len(s))
        output += s + "\n"
    outputt = [output[i:i + 8] for i in range(0, len(output), 8)]
    # ser = serial.Serial('/dev/ttyACM0', 9600)


    try:
        if ser.is_open:
            print("Serial connection opened successfully")
            # Send "......." as the first input
            # ser.write(".......".encode())
            print(".......")
```

```
time.sleep(2)


for i in range(len(outputt)):
    for j in range(3):
        if i != 0 or j != 0:
            ser.write(".......".encode())
            print(".......")
            time.sleep(3)
        if len(outputt[i].strip()) > 0:
            print(outputt[i].strip())
            ser.write(outputt[i].encode())
            time.sleep(3)
        else:
            print(" ")
        if outputt[i].strip() != "" and outputt[i] != ".......":
            time.sleep(5)
        else:
            print(outputt[i])
            ser.write(".......".encode())
            print(".......")
            time.sleep(3)
        if i != len(outputt) - 1:
            time.sleep(3)
```

```
        ser.write(".......".encode())  # Send "......." as the last encode for the serial
        print(".......")
        time.sleep(3)
      else:
        print("Failed to open serial connection")
    except serial.SerialException as e:
      print("Error opening serial connection:", e)
    finally:
      ser.close()
    # Turn on the buzzer for 0.2 seconds
    GPIO.output(buzzer_pin, GPIO.HIGH)
    time.sleep(0.2)
GPIO.add_event_detect(cameracapture, GPIO.FALLING, callback=button_clicked,
bouncetime=200)
class ImageSystem():
    def __init__(self, master=None):
      self.master = master
    def AudioBraille(self):
      print('audo braille open')
      cv2.destroyAllWindows()
      os.system('python berealspeech.py')  # execute the system call to open the new
Python file
```

```
        # self.master.destroyALLWindows()  # destroy the current GUI window
        os._exit(0)
    def KeyboardBraille(self):
        print('keyboard braille open')
        cv2.destroyAllWindows()
        os.system('python berealtext.py')   # execute the system call to open the new Python
file
        # self.master.destroyALLWindows()
        os._exit(0)
    @staticmethod
    def check_buttontext(pin):
        print('click text')
        if GPIO.input(buttontext) == False:
            a.KeyboardBraille()
    @staticmethod
    def check_buttonspeech(pin):
        print('click speech')
        if GPIO.input(buttonspeech) == False:
            a.AudioBraille()
    def destroy_windows(self):
        # Destroy all OpenCV windows
        cv2.destroyAllWindows()
        # Release the camera resources
```

```
        cap.release()
# Create an instance of ImageSystem
a = ImageSystem(cv2)
GPIO.add_event_detect(buttonspeech, GPIO.FALLING,
callback=a.check_buttonspeech, bouncetime=100)
GPIO.add_event_detect(buttontext, GPIO.FALLING, callback=a.check_buttontext,
bouncetime=100)
# Create an instance of ImageSystem
while True:
    ret, image = cap.read()
    image = cv2.flip(image, 1)
    # Create a named window with full screen property
    cv2.namedWindow("Capturing Image", cv2.WND_PROP_FULLSCREEN)
    cv2.setWindowProperty("Capturing Image", cv2.WND_PROP_FULLSCREEN,
cv2.WINDOW_FULLSCREEN)
    # Show the captured video
    cv2.imshow("Capturing Image", image)
    # Wait for user input
    key = cv2.waitKey(1)
    if clicks == 2:
        # Preprocess the image
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        gray = cv2.medianBlur(gray, 3)
```

```
        gray = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)[1]
        # Save the preprocessed image
        cv2.imwrite(r"/home/brbr/bereal/captured_preprocessed.png", gray)
        # Perform OCR on the image
        text =
pytesseract.image_to_string(Image.open(r"/home/brbr/bereal/captured_preprocessed.p
ng"), lang='eng')


        # Print the recognized text
        print_text_with_buzzer(text)
        clicks = 0
    elif key == ord('q'):
        break
    elif a.check_buttonspeech(buttonspeech) or a.check_buttontext(buttontext):
        cv2.destroyAllWindows()
        # Exit the while loop if either button is clicked twice
        clicks = 0
        break
# Release resources
cap.release()
a = ImageSystem(cv2)
cv2.destroyAllWindows()
```

BEREAL SPEECH-TO-BRAILLE CODE

```
from tkinter import *
from vosk import Model, KaldiRecognizer
import time
import pyaudio
import RPi.GPIO as GPIO
import os
import sys
import subprocess
os.environ['PYTHONPATH'] = '/home/brbr/.local/lib/python3.9/site-
packages/customtkinter'

# Import the module
sys.path.insert(0, '/home/brbr/.local/lib/python3.9/site-packages/customtkinter')
import customtkinter
import customtkinter as tk
GPIO.setmode(GPIO.BCM)
buttontext = 13
buttonimage = 5
speak = 6
buzzer_pin = 26
GPIO.setup(buttontext, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(buttonimage, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(speak, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buzzer_pin, GPIO.OUT)
customtkinter.set_appearance_mode("dark")  # Modes: system (default), light, dark
customtkinter.set_default_color_theme("blue")  # Themes: blue (default), dark-blue,
green
speechtotextwindow = customtkinter.CTk()


model = Model(r'C:\Users\Owner\PycharmProjects\BEREAL REAL NA REAL\vosk-
model-small-en-us-0.15\vosk-model-small-en-us-0.15')
recognizer = KaldiRecognizer(model, 16000)
mic = pyaudio.PyAudio()
stream = mic.open(format=pyaudio.paInt16, channels=1, rate=16000, input=True,
frames_per_buffer=8192)


clicks = 0
recording = False
def print_text_with_buzzer(text):
    print(text)
    GPIO.output(buzzer_pin, GPIO.HIGH)
    time.sleep(0.5)
    GPIO.output(buzzer_pin, GPIO.LOW)
def button_clicked(channel):
```

```
    global clicks, recording, scheduled
    clicks += 1
    if clicks == 2:
        if not recording:
            recording = True
            recognizer.FinalResult()  # Stop the recognizer before starting a new recording
            print('double click')
            record_voice()
        clicks = 0
        stream.stop_stream()
        print("Streaming stopped")  # Add this line to indicate that streaming has stopped
    else:
        # Schedule the function to run after 0.5 seconds (in case of double-click)
        scheduled = speechtotextwindow.after(500, check_clicks)
        recognizer.FinalResult()  # Clear the recognizer's result
GPIO.add_event_detect(speak, GPIO.FALLING, callback=button_clicked,
bouncetime=200)
def check_clicks():
    global clicks
    if clicks == 1:
        print("Button clicked 1 time")
    clicks = 0  # Reset clicks variable after processing single-click
def record_voice():
```

```
    global clicks, recording
    clicks = 0  # Reset clicks variable
    print("Recording started")  # Debugging line
    recognizer.FinalResult()  # Clear the recognizer's result
    stream.start_stream()
    while recording:
        data = stream.read(4096)
        if recognizer.AcceptWaveform(data):
            result = recognizer.Result()
            text = result
            print_text_with_buzzer(text[14:-3])
            textarea.insert(END, text[14:-3] + '\n')
            textarea.see(END)
            recording = False
            return (text[14:-3] + '\n')
    result = None
def update_scroll_region(event):
    textarea.yview_moveto(1.0)
Label = tk.CTkLabel(speechtotextwindow, text="Speech to Text Converter",
font=('Times New Roman', 30, 'bold'), padx=2,
                pady=12).grid(row=0, columnspan=12)
textarea = Text(speechtotextwindow, width=65, height=12, font=('Times New Roman',
20, 'bold'), padx=2, pady=2)
```

```
textarea.grid(row=2, columnspan=12)
textarea.bind('<Configure>', update_scroll_region)
buttons = ['Click to Speak']
varRow = 2
varColumn = 5
for button in buttons:
    command = lambda x=button: button_clicked()
    if button == "Click to Speak":
        tk.CTkButton(speechtotextwindow, text=button, width=60,
command=command).grid(row=4, column=5, padx=5, pady=5)
class SpeechSystem():
    def __init__(self, master=None):
        self.master = master
    def KeyboardBraille(self):
        os.system('python berealtext.py')   # execute the system call to open the new Python
file
        self.master.withdraw()  # destroy the current GUI window
        os._exit(0)
    def ImageBraille(self):
        os.system('python berealimage.py')  # execute the system call to open the new
Python file
        self.master.withdraw()
        os._exit(0)
```

```
    @staticmethod
    def check_buttontext(pin):
        if GPIO.input(buttontext) == False:
            a.KeyboardBraille()
    @staticmethod
    def check_buttonimage(pin):
        if GPIO.input(buttonimage) == False:
            a.ImageBraille()
a = SpeechSystem(speechtotextwindow)
GPIO.add_event_detect(buttontext, GPIO.FALLING, callback=a.check_buttontext,
bouncetime=100)
GPIO.add_event_detect(buttonimage, GPIO.FALLING,
callback=a.check_buttonimage, bouncetime=100)
speechtotextwindow.title('Speech-to-Text')
speechtotextwindow.geometry("800x480")
# speechtotextwindow.attributes('-fullscreen', True)
speechtotextwindow.update()
speechtotextwindow.geometry('700x350')
speechtotextwindow.mainloop()
speechtotextwindow.destroy()
```

BEREAL TEXT-TO-BRAILLE

```
#try:
#    import tkinter as tk
#except ImportError:
#    import tkinter as tk


import customtkinter as tk
import customtkinter
from tkinter.scrolledtext import ScrolledText
from functools import partial
from tkinter import*
import tkinter
from tkinter import ttk
import RPi.GPIO as GPIO
import os
import time
# import serial
import sys
import  subprocess
os.environ['PYTHONPATH'] = '/home/brbr/.local/lib/python3.9/site-
packages/customtkinter'
```

```
# Import the module
sys.path.insert(0, '/home/brbr/.local/lib/python3.9/site-packages/customtkinter')
import customtkinter
import customtkinter as tk
GPIO.setmode(GPIO.BCM)
buttonimage = 5
buttonspeech = 6
buzzer_pin = 26
GPIO.setup(buttonspeech, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buttonimage, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(buzzer_pin, GPIO.OUT)
customtkinter.set_appearance_mode("dark")  # Modes: system (default), light, dark
customtkinter.set_default_color_theme("blue")  # Themes: blue (default), dark-blue,
green
Keyboard_App = customtkinter.CTk()  # create CTk window like you do with the Tk
window
Keyboard_App.title('Keyboard')
Keyboard_App.geometry('700x480')
# Keyboard_App.attributes('-fullscreen', True)
# Keyboard_App.state('normal')


def print_text_with_buzzer(text):
    textt = text
```

```
substrings = [textt[i:i + 7] for i in range(0, len(textt), 7)]
output = ""
for s in substrings:
    s += " " * (7 - len(s))
    output += s + "\n"
outputt = [output[i:i + 8] for i in range(0, len(output), 8)]
# ser = serial.Serial('/dev/ttyACM0', 9600)


try:
    if ser.is_open:
        print("Serial connection opened successfully")
        # Send "......." as the first input
        # ser.write(".......".encode())
        print(".......")
        time.sleep(2)
        for i in range(len(outputt)):
            for j in range(3):
                if i != 0 or j != 0:
                    ser.write(".......".encode())
                    print(".......")
                    time.sleep(3)
                if len(outputt[i].strip()) > 0:
                    print(outputt[i].strip())
```

```
            ser.write(outputt[i].encode())
            time.sleep(3)
          else:
            print(" ")
          if outputt[i].strip() != "" and outputt[i] != ".......":
            time.sleep(5)
          else:
            print(outputt[i])
            ser.write(".......".encode())
            print(".......")
            time.sleep(3)
          if i != len(outputt) - 1:
            time.sleep(3)
      ser.write(".......".encode())  # Send "......." as the last encode for the serial
      print(".......")
      time.sleep(3)
    else:
      print("Failed to open serial connection")
except serial.SerialException as e:
    print("Error opening serial connection:", e)
finally:
    ser.close()
# Turn on the buzzer for 0.2 seconds
```

```
    GPIO.output(buzzer_pin, GPIO.HIGH)
    time.sleep(0.2
def select(value):
    if value == ' Space ':
        textarea.insert(INSERT, ' ')
    elif value == 'Tab':
        textarea.insert(INSERT, '\t')
    elif value == 'del':
        i = textarea.get(1.0, END)
        newtext = i[:-2]
        textarea.delete(1.0, END)
        textarea.insert(INSERT, newtext)
    else:
        textarea.insert(INSERT, value)
    textarea.focus_set()
label1 = tk.CTkLabel(Keyboard_App, text="Keyboard for Text to Braille",
font=('Palatino',30,'bold'),pady=10).grid(row=0,columnspan=10)
textarea = Text(Keyboard_App, width=55, height=7,
font=('Palatino',20,'bold'),padx=15,pady=10)
textarea.grid(row=1,columnspan=10)
buttons = [
    'q','w','e','r','t','y','u','i','o','p',
    'a','s','d','f','g','h','j','k','l','del',
```

```
    'shift','z','x','c','v','b','n','m',
    ' Space '
]
varRow = 2
varColumn= 0
def send_message(event=None):
    message = textarea.get("1.0", END)
    textarea.delete(1.0, END)
    print_text_with_buzzer(message)
for button in buttons:
    command = lambda x=button: select(x)
    if button != " Space ":
        tk.CTkButton(Keyboard_App, text=button, width=70,
command=command,height=60, font=('Palatino',23,'bold')).grid(row=varRow,
column=varColumn, padx=4, pady=5)
    if button == " Space ":
        tk.CTkButton(Keyboard_App, text=button, width=70,
command=command,height=60, font=('Palatino',20,'bold')).grid(row=varRow,
column=varColumn, padx=0, pady=5)
    varColumn += 1
    if varColumn > 9 and varRow == 2:
        varColumn = 0
        varRow += 1
```

```
    if varColumn > 9 and varRow == 3:
        varColumn = 0
        varRow += 1
class TextSystem():
    def __init__(self, master=None):
        self.master = master
    def AudioBraille(self):
        os.system('python berealspeech.py')   # execute the system call to open the new
Python file
        self.master.withdraw()
        os._exit(0)
    def ImageBraille(self):
        os.system('python berealimage.py')  # execute the system call to open the new
Python file
        self.master.withdraw()
        os._exit(0)
    #
    @staticmethod
    def check_buttonspeech(pin):
        if GPIO.input(buttonspeech) == False:
          a.AudioBraille()


    @staticmethod
```

```
    def check_buttonimage(pin):
        if GPIO.input(buttonimage) == False:
            a.ImageBraille()
a = TextSystem(Keyboard_App)
#
GPIO.add_event_detect(buttonspeech, GPIO.FALLING,
callback=a.check_buttonspeech, bouncetime=100)
GPIO.add_event_detect(buttonimage, GPIO.FALLING,
callback=a.check_buttonimage, bouncetime=100)
Enter = tk.CTkButton(Keyboard_App, text=' Enter ', width=70,height=60,
font=('Palatino',23,'bold'), command=send_message)
Enter.place(x=735, y=415)
Keyboard_App.bind('<Return>', send_message)
Keyboard_App.mainloop()
Keyboard_App.destroy()
```

PART II - ASSEMBLY

1 Components and Materials

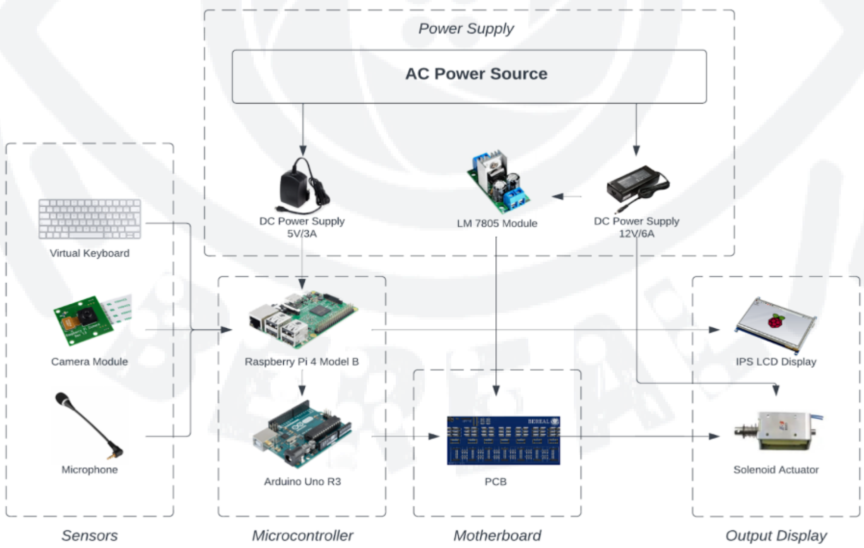
- Raspberry Pi 4 Model B
- Arduino Uno R3
- 5” Touchscreen IPS LCD Display for Raspberry Pi
- Raspberry Pi Camera Module V1.3
- Microphone
- Linear Push-Pull Solenoid
- 3D Printed Chassis
- PCB
- Power Supply 12V/6A
- Power Supply 5V/3A
- LM 7805 Module
- USB A to USB B cable
- Jumper Wires
- Terminal Block
- Push Buttons

2 Raspberry Pi and Arduino

The Raspberry Pi 4 Model B is a single-board computer that offers a lot of capabilities that can be used in various projects. The Raspberry Pi is similar to a real-life computer and acts like a minicomputer. Arduino, on the other hand, is an open-source platform that combines software and hardware. The Arduino board can be programmed using the Arduino IDE. To connect to the outside world, the sensors are connected through the I/O pins of the board, and the software is used to write programs to interact with the sensors.

This project utilizes the Raspberry Pi 4 Model B and the Arduino Uno R3. The Raspberry Pi 4 Model B focuses on processing the camera, microphone, and virtual keyboard inputs, whereas the Arduino Uno R3 focuses on controlling the solenoids. The program in Python, which is run by the Raspberry Pi 4 Model B, converts all the input into text format, and it will send the text output to the Arduino to control the solenoids. To connect this, use a USB 2.0 Cable Type A/B; it will allow data transfer from the Raspberry Pi to the Arduino.

3 Power Supply and Wirings



The block diagram shows the connections between the Raspberry Pi, Arduino, PCB, power supply, sensors, LCD display, and solenoids.

Power Supply:

- Connect the 12V/6A power supply to the DC barrel jack of the motherboard and the LM7805 module.
- Connect the output of the LM 7805 module to the VCC2 pin of the motherboard.
- Connect the 5V/3A to the Raspberry Pi 4 Model B.
- Connect the AC of the 12V/6A and 5V/3A power supply to the terminal block.

Raspberry Pi 4 Model B:

- Connect the Raspberry Pi Camera Module V1.3 to the dedicated slot for the camera on the Raspberry Pi.
- The microphone is connected to an adapter and connects it to the USB A port.
- Connect the LCD to the Raspberry Pi using HDMI cable. Connect a 5 inch LCD screen to the Raspberry Pi using the appropriate GPIO pins.

5 inch LCD pins	**Raspberry pi 4 pins**	**Description**
1, 17	3.3V	Power positive (3.3V power input)
2, 4	5V	Power positive (5V power input)
3, 5, 7, 8, 10, 12, 13, 15, 16, 18, 24	NC	NC
11	Backlight Control	Control the backlight through pin 11
6, 9, 14, 20, 25	GND	Ground
19	TP_SI	SPI data input of Touch Panel
21	TP_SO	SPI data output of Touch Panel
22	TP_IRQ	Touch panel interrupt, low level while the touch panel detects touching
23	TP_SCK	SPI clock of touch panel
26	TP_CS	Touch panel chip selection, low active

- There are 3 Push Buttons on the Device that are connected to Raspberry pi.
  - Button for text is connected to GPIO 13
  - Button for image is connected to GPIO 5
  - Button for speech is connected to GPIO 6

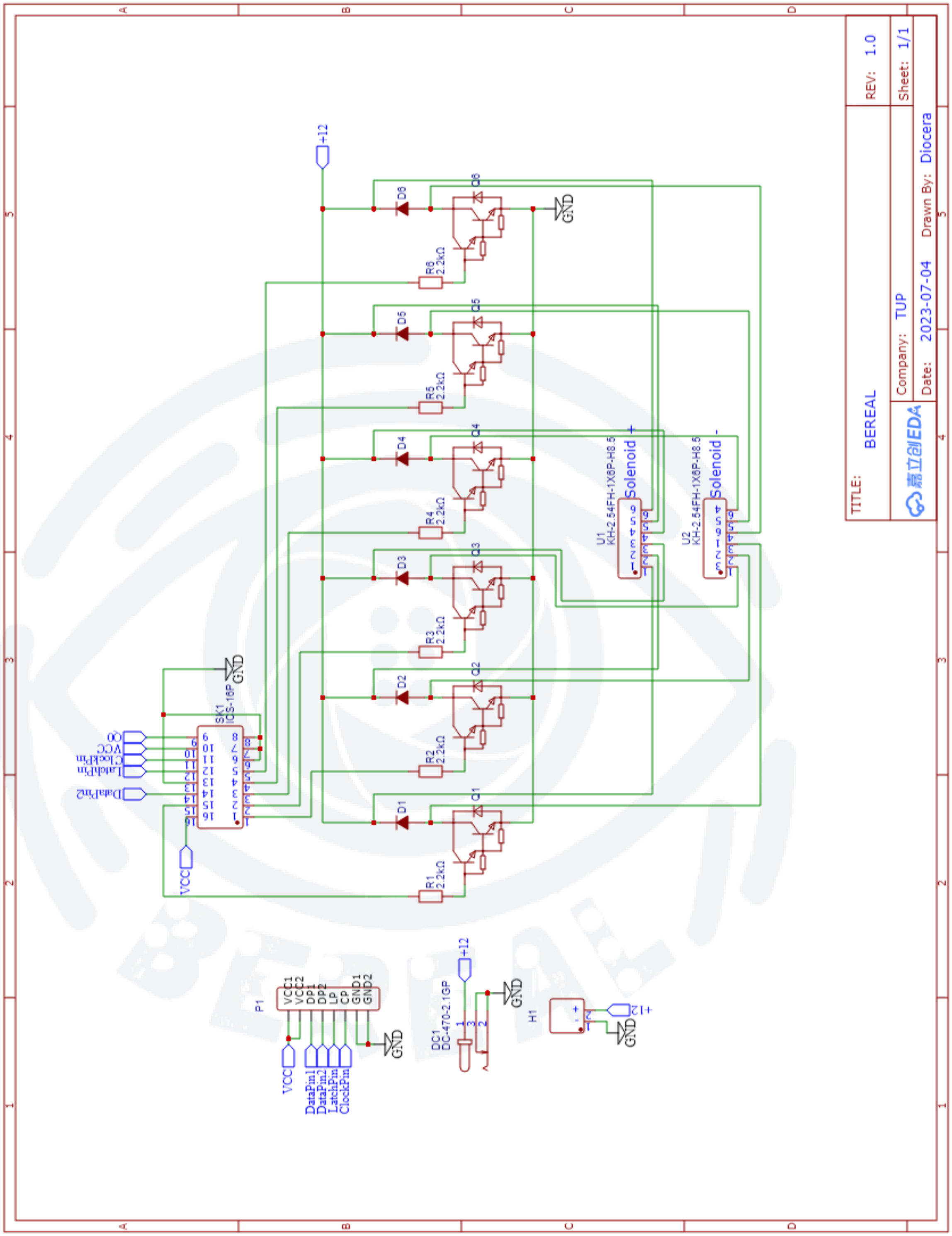
- To connect the Raspberry Pi and Arduino Uno R3 using USB 2.0 Cable Type A/B, connect the USB A part of the cable to the USB A port.

Arduino Uno R3:

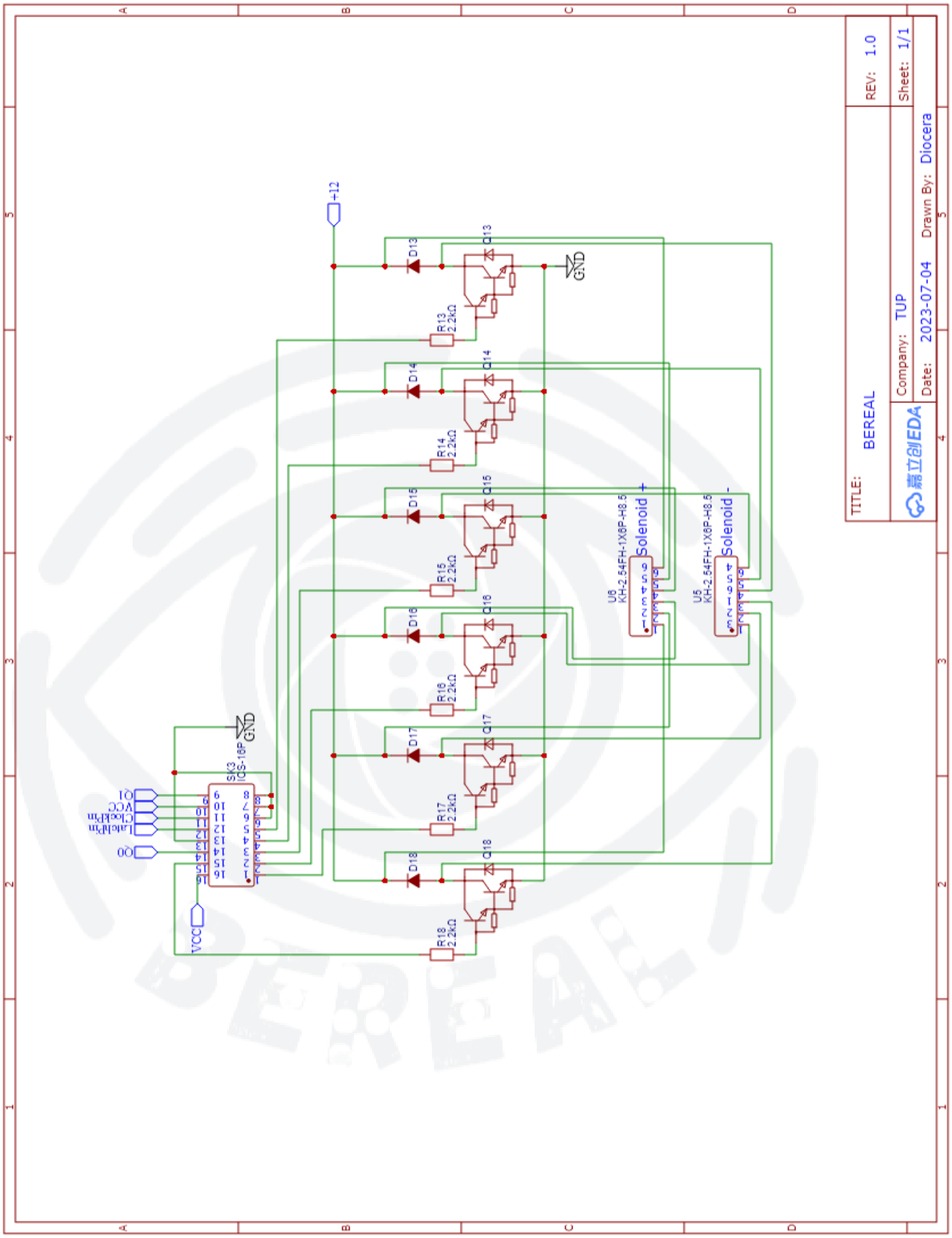
- Connect the USB B cable from the Raspberry Pi to the USB B port of the Arduino.
- Connect the output of the LM7805 to the DC barrel jack of the Arduino.
- Connect the DP2 pin of the motherboard to pin 8 of the Arduino.
- Connect the LP pin of the motherboard to pin 9 of the Arduino.
- Connect the CP pin of the motherboard to pin 9 of the Arduino.

Part III Manufacturing

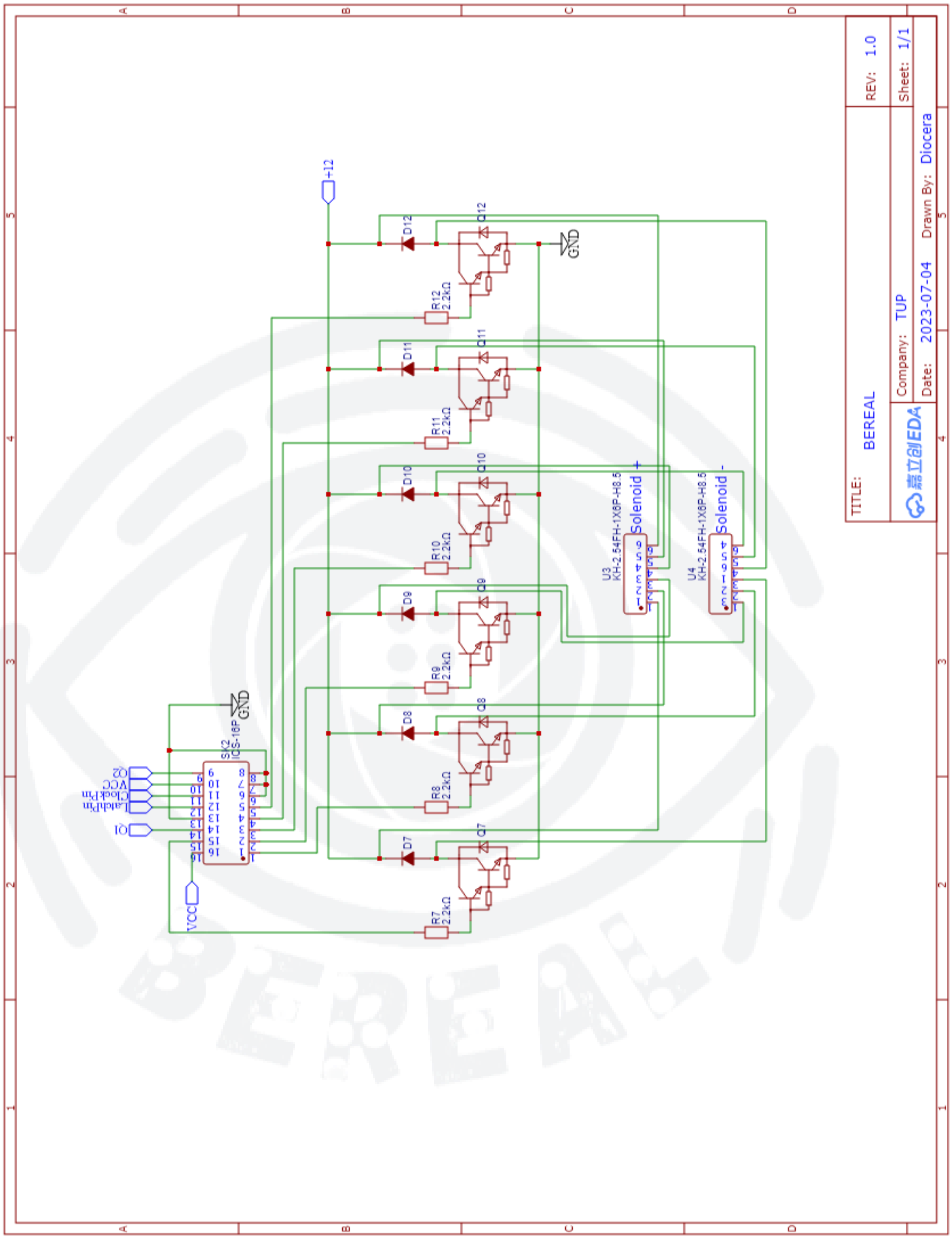
1 Schematic Layout



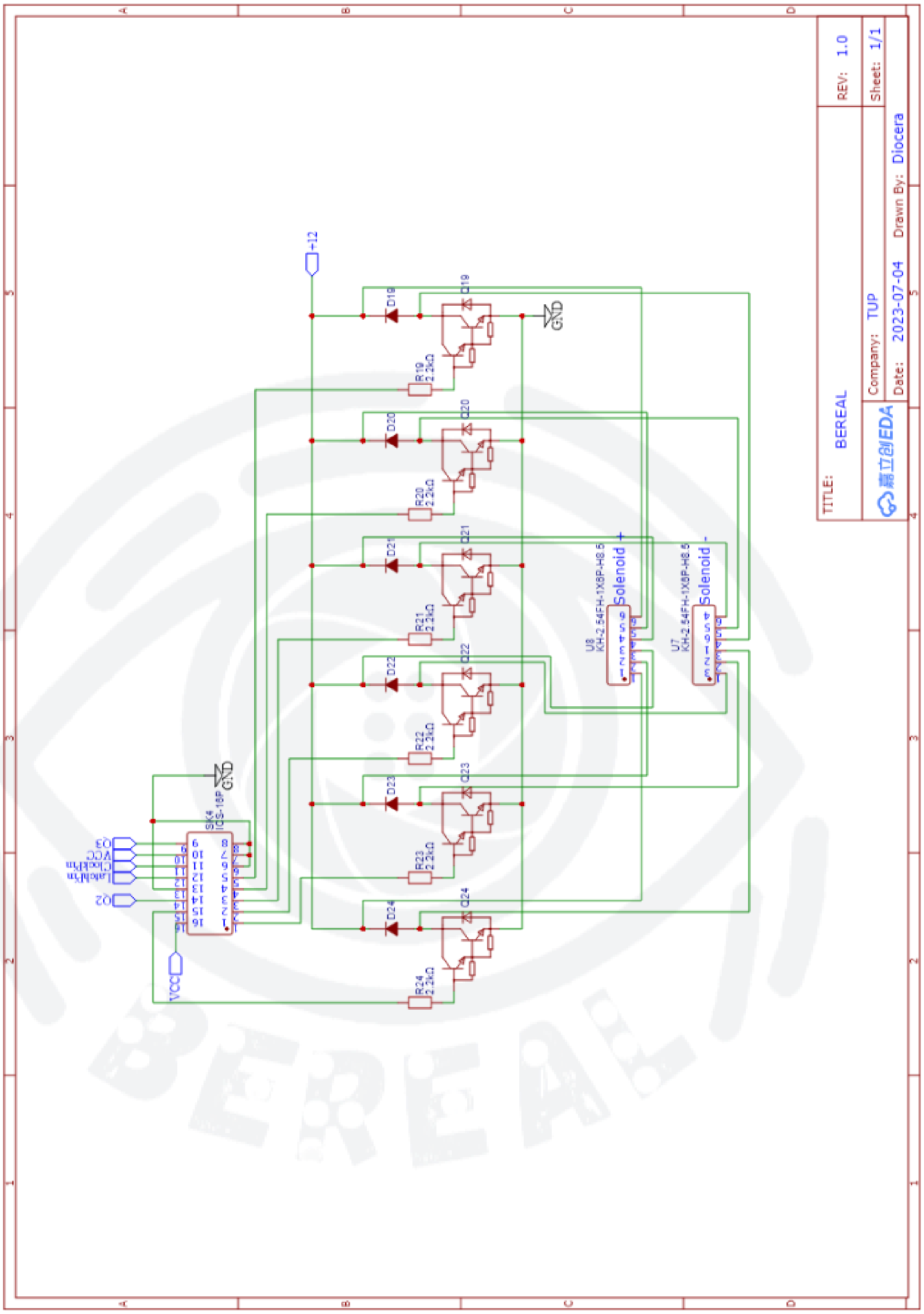
Schematic of the 1st character



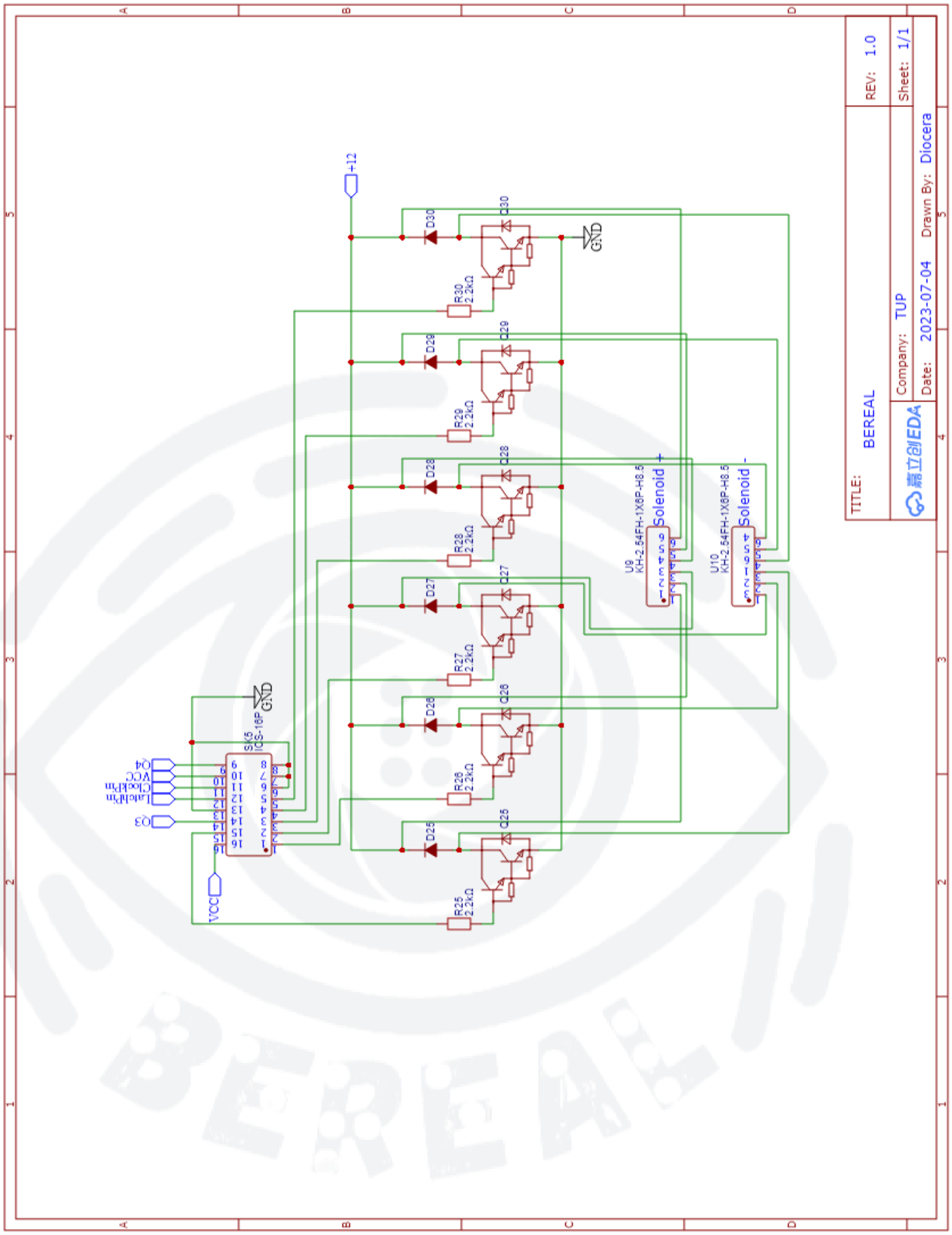
Schematic of *the* 2nd *character*



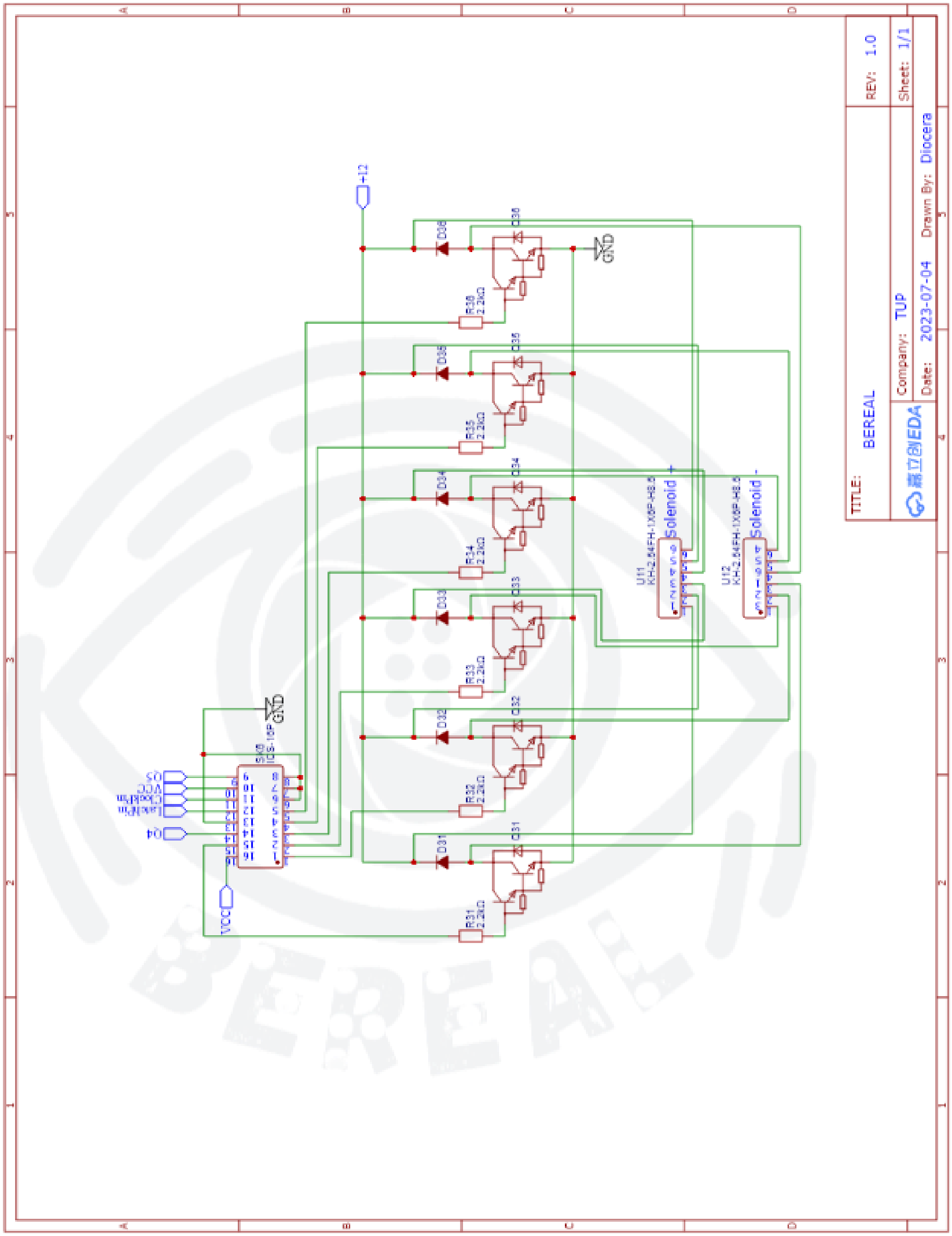
Schematic of the 3rd character



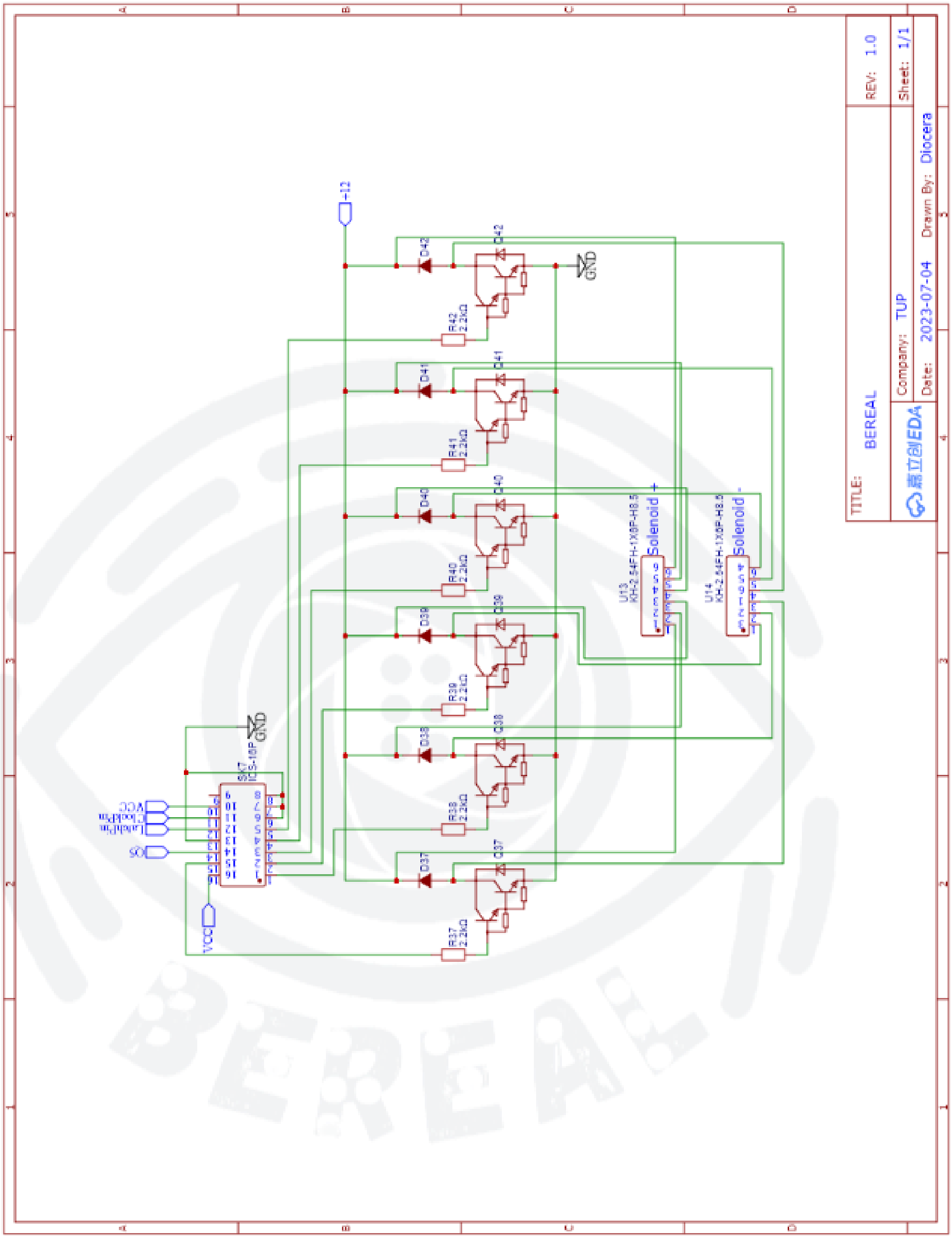
Schematic of the 4th character



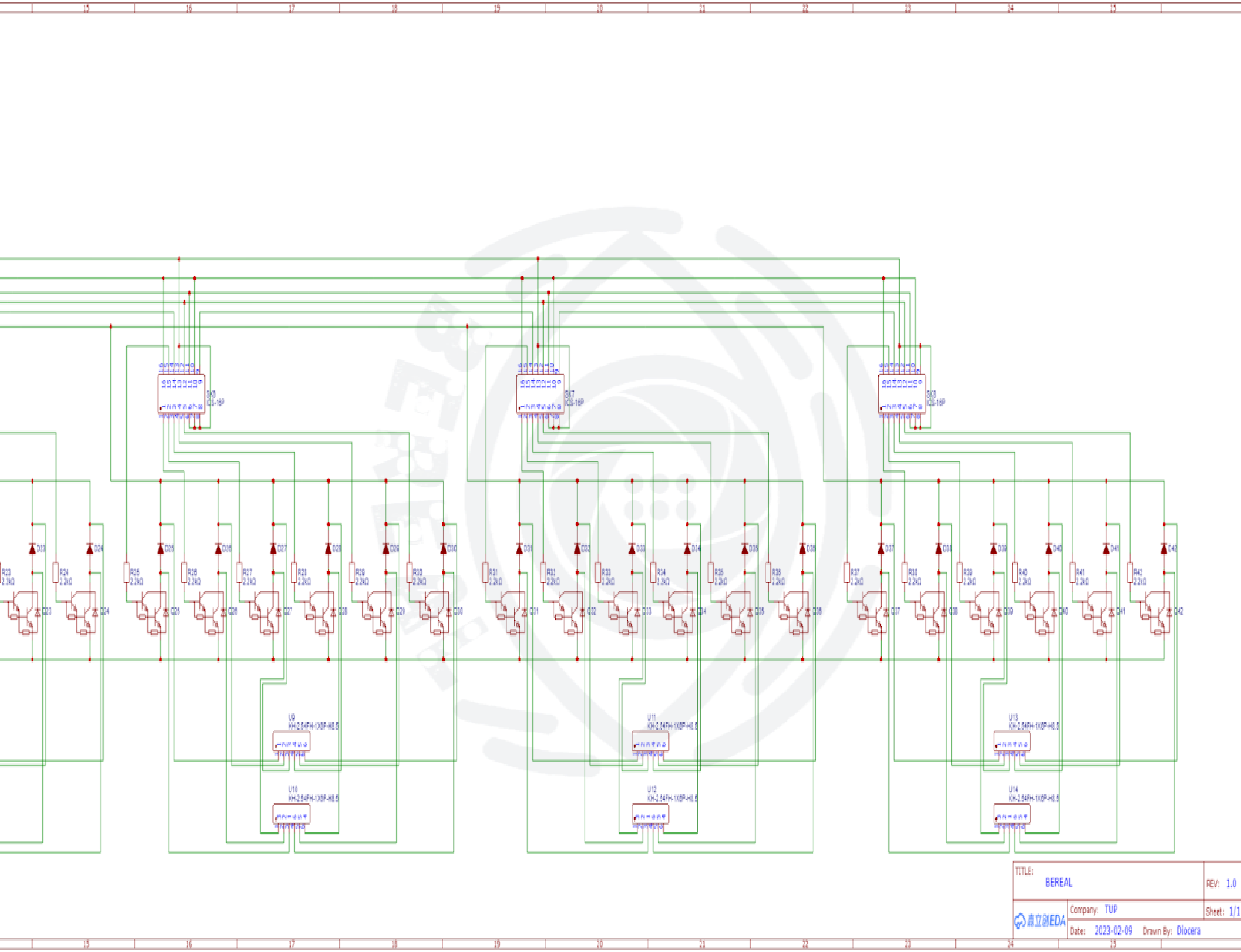
Schematic of the 5th character

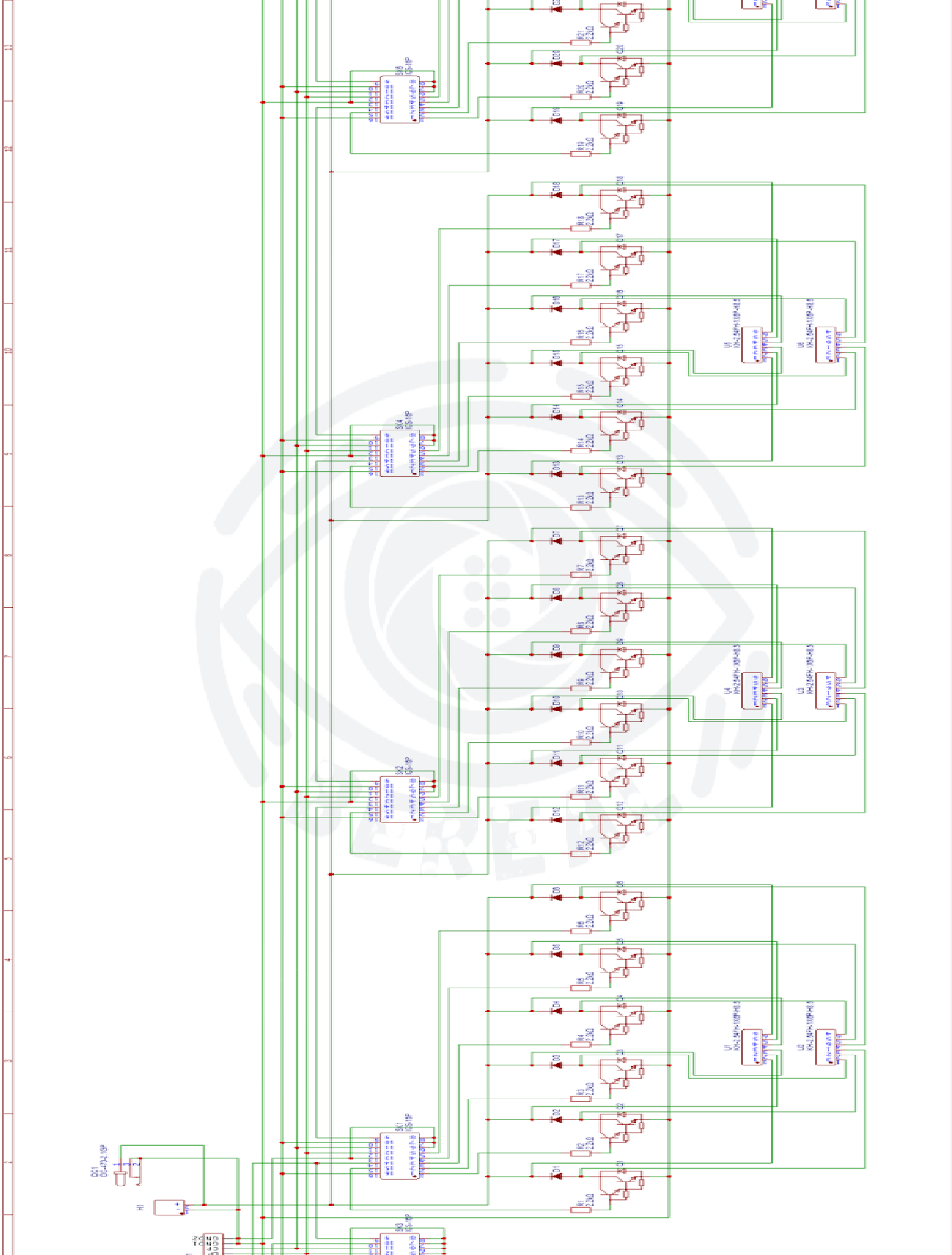


Schematic of the 6th character

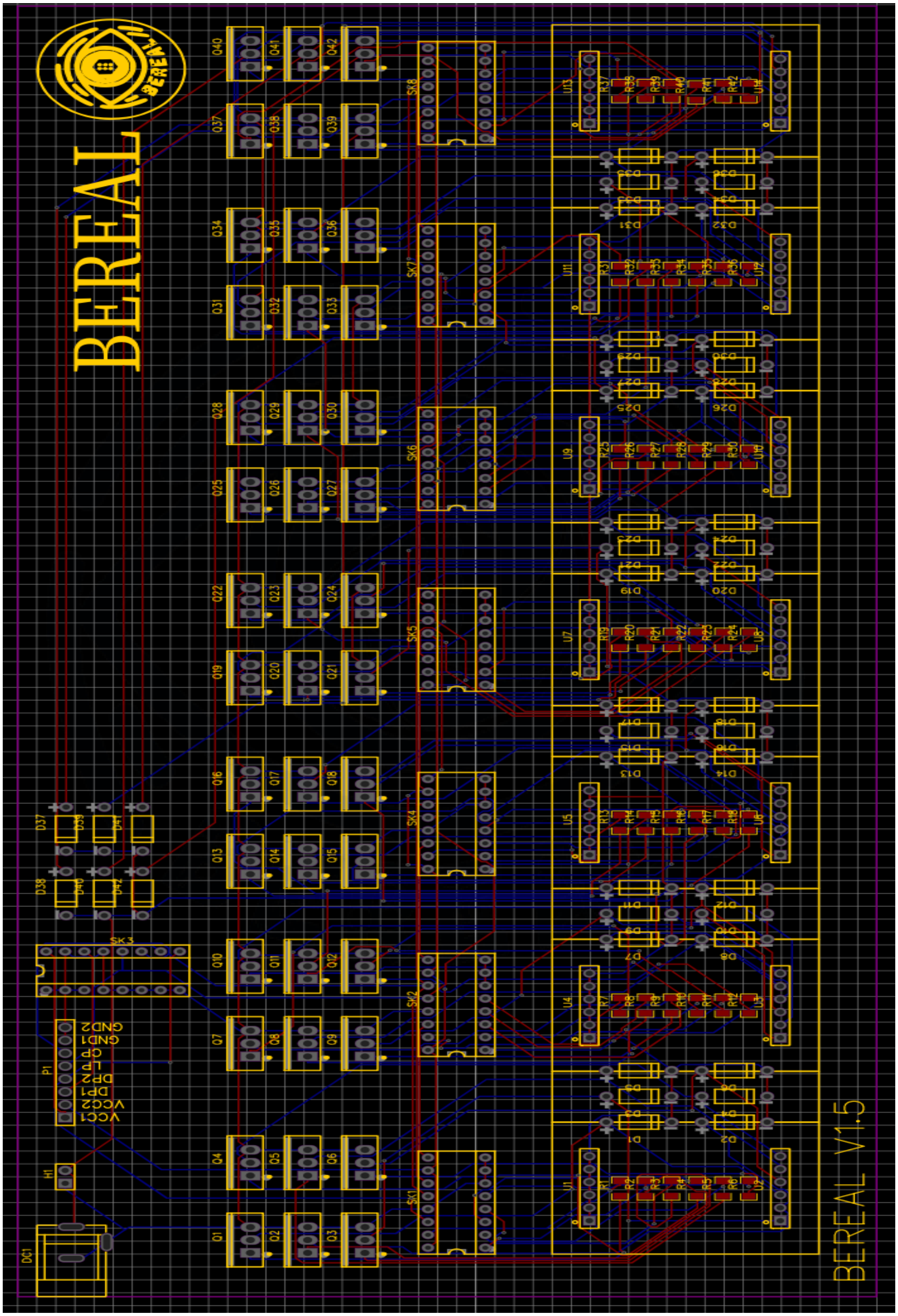


Schematic of the 7th character

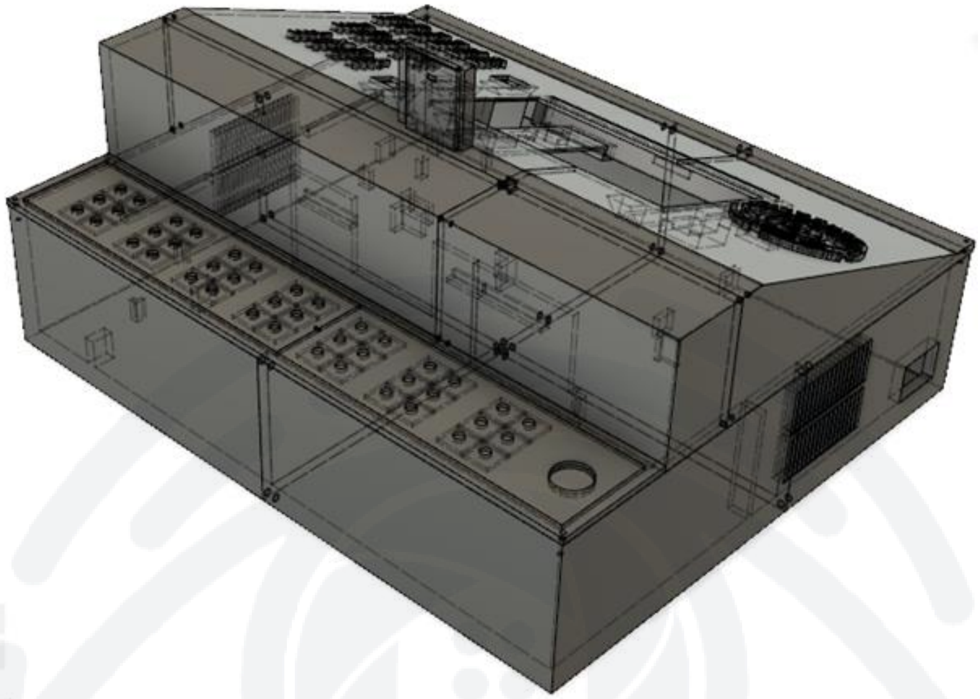

TITLE: BEREAL
REV: 1.0
Company: TUP
Sheet: 1/1
Date: 2023-02-09
Drawn By: Diocera
立创EDA



2 PCB Layout



3 3D Printing



ANNEX VII
PROPONENT’S INFORMATION

Mark Pangantihon

ELECTRONICS ENGINEERING STUDENT

Contacts

Las Piñas City, Metro Manila

pangantihonmarky18@gmail.com

0951-923-6589

Skills

- Literate in Microsoft Office Applications
- Computer Literate
- Basic Knowledge in 3D Designing and Printing
- Basic Programming in Python Language
- Basic Programming in Arduino
- Basic Knowledge in IC Designing
- Creative

About

Currently seeking Looking for a Job where I can apply my knowledge and skills in Microelectronics (Designing and layouting of an IC, Prototype Fabrication, etc.) and to work with an environment that can help me to grow and learn as individual while helping the company to achieve its goals.

Education History

TERTIARY

Bachelor of Science in Electronics Engineering

Technological University of the Philippines – Manila

2019-2023

SECONDARY

Science, Technology, Engineering and Mathematics

San Juan De Dios Educational Foundation Inc. - College

2017-2019

Work Experience

On-the-Job Training (OJT)

Commsec Inc.

2259 Aurora 1303 Pasay National Capital Region

2022

Work Immersion

San Juan De Dios Hospital

2772-2774 Roxas Boulevard 1300 Pasay City Metro Manila

2019

Mary Loise Borjal

ELECTRONICS ENGINEERING STUDENT

Contacts

Manila , Metro Manila

maryloiseborjal4@gmail.com

0905-303-7737

Skills

- Proficiency in statistical software (Python, R, Sql)
- Data cleaning and preprocessing
- PCB layout design,
- Basic Programming in C++
- NI Multisim
- Cadence Virtuoso
- Arduino
- Cisco Packet Tracer
- Amazon Pay-Per-Click campaigns
- Keyword Research

About

Motivated and career-driven Electronics Engineer with a relentless passion for leveraging technical expertise and abilities to contribute to the development and innovation of electronic systems and IT solutions. Equipped with a solid foundation in electrical engineering principles and a diverse skill set, I am dedicated to pushing the boundaries of technological advancements and driving positive change in the field.

Education History

TERTIARY

Bachelor of Science in Electronics Engineering

Technological University of the Philippines – Manila

2019-2023

SECONDARY

Science, Technology, Engineering and Mathematics

Immaculate Conception Polytechnic

2017-2019

Work Experience

On-the-Job Training (OJT)

STELSEN INC.
Makati City, Metro Manila
2022

Amazon PPC VA

VAA Philippines (Simpliworks)
Quezon City, Metro Manila
2023

Michael Conrad Diocera

ELECTRONICS ENGINEERING STUDENT

Contacts

Caloocan City, Metro Manila

dioceramichaelconrad@gmail.com

0921-284-8046

Skills

- Literate in Microsoft Office Applications
- PCB Layout Design
- Python Programming
- Basic Programming in C++
- NI Multisim
- Cadence Virtuoso
- Arduino
- Cisco Packet Tracer

About

As an electronics engineer, I am highly motivated and career-driven, seeking a challenging position where I can passionately utilize my technical expertise and abilities. I am passionate about leveraging my technical expertise and abilities to contribute to the development and innovation of electronic systems and IT solutions.

Education History

TERTIARY

Bachelor of Science in Electronics Engineering

Technological University of the Philippines – Manila

2019-2023

SECONDARY

Science, Technology, Engineering and Mathematics

La Consolacion College Manila

2017-2019

Work Experience

On-the-Job Training (OJT)

MELHAM CONSTRUCTION CORPORATION
Quezon City, Metro Manila
2022

Work Immersion

STCW ADMINISTRATION OFFICE – MARINA CENTRAL
Manila, Metro HXPF+Q2C Marina Central Office, Taft Ave, Port Area, Manila
2022

Jetlee Dumandan

ELECTRONICS ENGINEERING STUDENT

Contacts

Parañaque City, Metro Manila

dumandanjetlee@gmail.com

0945-230-9374

Skills

- Literate in Microsoft Office Applications
- Computer Literate
- Basic Knowledge in 3D Designing and Printing
- Basic Knowledge in PCB Assembly and Production
- Basic Knowledge in IC Designing
- Cadence Virtuoso
- NI Multisim

About

He is a responsible and hardworking person. He is also a mature team worker and adaptable to all challenging situations. Beside all of that he is an aspiring electronics engineer who's ready to commit and apply his knowledge in electronics applications and pcb fabrications.

Education History

TERTIARY

Bachelor of Science in Electronics Engineering

Technological University of the Philippines – Manila

2019-2023

SECONDARY

Science, Technology, Engineering and Mathematics

Parañaque National Hign School - Main

2017-2019

Work Experience

Part Time Helper

Eastern Integrated System & Automation (EISA)
Suite 319 Republic SuperMarket Bldg., Soler St.
cor. Rizal Ave., Sta. Cruz, Manila, Philippines
May 2023 - Present

Employee

Special Program for Emplyment of Students (SPES)
Parañaque City
June 2019 - August 2019

On-the-Job Training (OJT)

Commsec Inc.
2259 Aurora 1303 Pasay National Capital Region
2022

Jessica Gamil

ELECTRONICS ENGINEERING STUDENT

Contacts

Quezon City, Metro Manila

jessicagamil15@gmail.com

0999-5482-150

Skills

- Literate in Microsoft Office Applications
- Computer Literate
- Basic Knowledge in 3D Designing and Printing
- CISCO Packet Tracer
- Adobe Photoshop
- Python Programming
- Basic Knowledge in Arduino
- NI Multisim
- Cadence Virtuoso

About

An electronics engineer who's ready to contribute my expertise and knowledge in electrical applications such as troubleshooting, testing, and solving technical problems.

Education History

TERTIARY

Bachelor of Science in Electronics Engineering

Technological University of the Philippines – Manila
2019-2023

SECONDARY

Science, Technology, Engineering and Mathematics

Polytechnic University of the Philippines
2017-2019

Work Experience

On-the-Job Training (OJT)

ONE COMMERCE (INT'L) CORPORATION
Mandaluyong, Metro Manila
2022

Work Immersion

BUREAU OF AGRICULTURAL RESEARCH (DA-BAR)
P. Tuazon Blvd, Quezon City, Metro Manila
April - May 2019

POLYTECHNIC UNIVERSITY OF THE PHILIPPINES
Anonas, Sta. Mesa, Maynila, Kalakhang Maynila
2018

Jewelle Milles Reyes

ELECTRONICS ENGINEERING STUDENT

Contacts

Muntinlupa City, Metro Manila

jewellereyes01@gmail.com

0947-407-1562

Skills

- Literate in Microsoft Office Applications
- Basic Programming in Python Language
- Familiarity with IC Designing Fundamentals
- Basic Knowledge in using Cadence Virtuoso
- Proficiency in using Cisco Packet Tracer
- Experience with NI Multism

About

Highly motivated and enthusiastic Electronics Engineering graduate specializing in Microelectronics. A quick learner with a strong work ethic, eager to contribute to a dynamic engineering team and offer technical skills to drive product innovation and client satisfaction.

Education History

TERTIARY

Bachelor of Science in Electronics Engineering

Technological University of the Philippines – Manila

2019-2023

SECONDARY

Science, Technology, Engineering and Mathematics

Adamson University

2017-2019

Work Experience

Executive Virtual Assistant

Direct Client
New Jersey, USA
2022-Present

Client Support Representative

Synchrony Financial
Manila, Philippines
2021-2022