

A Project Report on

SMART BRAILLE TECH

Submitted in partial fulfillment of the requirements for the award of the Degree of

Bachelor of Technology
in
Computer Science and Engineering

By

NANI SAMIREDDY

21A95A0517

KRISHI JAIN

20A91A05G0

SUBHRA KHANDA

20A91A05I6

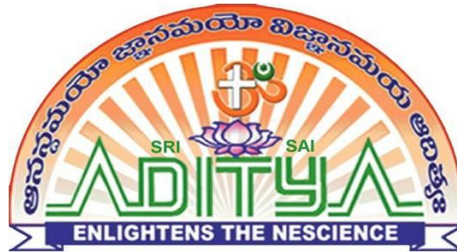
VASIREDDY PHANINDRA

20A91A05I9

Under the Esteemed Supervision of

Dr. Ravi Kishore Veluri (Ph.D)

Associate Professor



Department of Computer Science and Engineering

ADITYA ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Affiliated to JNTUK Kakinada, Accredited by NBA & NAAC with 'A++' Grade)

Aditya Nagar, ADB Road, Surampalem

2020 – 2024

ADITYA ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Affiliated to JNTUK Kakinada, Accredited by NBA & NAAC with 'A++' Grade)

Aditya Nagar, ADB Road, Surampalem

2020 – 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project work entitled “**SMART BRAILLE TECH**” is being submitted by

NANI SAMIREDDY

21A95A0517

KRISHI JAIN

20A91A05G0

SUBHRA KHANDA

20A91A05I6

VASIREDDY PHANINDRA

20A91A05I9

in partial fulfillment of the requirements for the award of degree of **B.Tech** in Computer Science and Engineering from **Jawaharlal Nehru Technological University Kakinada** is a record of bonafide work carried out by them at **Aditya Engineering College**

The results embodied in this Project report have not been submitted to any other University or Institute for the award of any degree or diploma.

PROJECT GUIDE

Dr.Ravi Kishore Veluri, Ph.D

HEAD OF THE DEPARTMENT

Dr.A.Vanathi, M.E., Ph.D

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project entitled “**SMART BRAILLE TECH** ” is genuine. This work has been submitted to the **ADITYA ENGINEERING COLLEGE**, Surampalem, permanently affiliated to **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA** in partial fulfillment of the **B.Tech** degree. We further declare that this project work has not been submitted in full or part of the award for any degree of this or any other educational institution.

By

NANI SAMIREDDY

21A95A0517

KRISHI JAIN

20A91A05G0

SUBHRA KHANDA

20A91A05I6

VASIREDDY PHANINDRA

20A91A05I9

ACKNOWLEDGEMENT

It is with immense pleasure that we would like to express our indebted gratitude to our project guide **Dr.Ravi Kishore Veluri, Ph.D, Associate Professor, Department of CSE** who has guided us a lot and encouraged us in every step of the project work, his valuable moral support and guidance throughout the project helped us to a greater extent.

Our sincere thanks to project coordinator **Mrs. T.Sudha Rani, M.Tech.,(Ph.D.), Associate Professor, Department of CSE** for providing great support and suggestions during our project work.

Our deepest thanks to our HOD **Dr. A.Vanathi, M.E., Ph.D, Associate Professor** for inspiring us all the way and for arranging all the facilities and resources needed for our project.

We wish to thank **Dr. S. Rama Sree, Professor** in CSE and Dean (Academics) for her support and suggestions during our project work.

We owe our sincere gratitude to **Dr. M.Sreenivasa Reddy, Principal** for providing great support and for giving us the opportunity of doing the project.

We are thankful to our **College Management** for providing all the facilities in time us for the completion of our project.

Not to forget, **Faculty, Lab Technicians, Non-teaching staff, and our friends** who have directly or indirectly helped and supported us in completing our project in time.



ADITYA ENGINEERING COLLEGE (A)

Aditya Nagar, ADB Road, Surampalem

VISION & MISSION OF THE INSTITUTE

Vision:

To emerge as a premier institute for quality technical education and innovation.

Mission:

M1: Provide learner centric technical education towards academic excellence

M2: Train on technology through collaborations

M3: Promote innovative research & development

M4: Involve industry institute interaction for societal needs

Sri Ravi

PRINCIPAL
PRINCIPAL

ADITYA ENGINEERING COLLEGE
SURAMPALEM - 533 437



ADITYA ENGINEERING COLLEGE (A)

Aditya Nagar, ADB Road, Surampalem

Department of Computer Science and Engineering

VISION & MISSION OF THE DEPARTMENT

VISION:

To emerge as a competent Centre of excellence in the field of Computer Science and Engineering for industry and societal needs.

MISSION:

- Impart quality and value based education.
- Inculcate the inter personal skills and professional ethics.
- Enable research through state-of-the-art infrastructure.
- Collaborate with industries, government and professional societies.

A handwritten signature in red ink, likely belonging to the Head of the Department, is positioned above the printed name.

Head of the Department

Head of the Department

Department of CSE

ADITYA ENGINEERING COLLEGE (A)



ADITYA ENGINEERING COLLEGE (A)

Aditya Nagar, ADB Road, Surampalem

Department of Computer Science and Engineering

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

Graduates of the Program will

- **PEO 1:** Adopt new technologies and provide innovative solutions.
- **PEO 2:** Be employable, become an entrepreneur or researcher for a successful career.
- **PEO 3:** Demonstrate interpersonal, multi-disciplinary skills and professional ethics to serve society.

A handwritten signature in red ink, appearing to be 'A. N. N.', is positioned above the printed name of the Head of the Department.

Head of the Department

Head of the Department

Department of CSE

ADITYA ENGINEERING COLLEGE (A)



ADITYA ENGINEERING COLLEGE (A)

Aditya Nagar, ADB Road, Surampalem

Department of Computer Science and Engineering

PROGRAM OUTCOMES (POs)

After successful completion of the program, the graduates will be able to

- PO 1 **Engineering Knowledge:** Apply knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.
- PO 2 **Problem Analysis:** Identify, formulate, research literature and analyze complex engineering problems, reaching substantiated conclusions using first principles of mathematics, natural sciences and engineering sciences.
- PO 3 **Design/Development of Solutions:** Design solutions for complex engineering problems and design systems, components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.
- PO 4 **Conduct Investigations of Complex Problems:** Conduct investigations of complex problems using research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of information to provide valid conclusions.
- PO 5 **Modern Tool Usage:** Create, select and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modelling, to complex engineering activities, with an understanding of the limitations.
- PO 6 **The Engineer and Society:** Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to professional engineering practice.
- PO 7 **Environment and Sustainability:** Understand the impact of professional engineering solutions in societal and environmental contexts and demonstrate knowledge of, and need for sustainable development.

- PO 8 **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of engineering practice.
- PO 9 **Individual and Teamwork:** Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.
- PO 10 **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- PO 11 **Project Management and Finance:** Demonstrate knowledge and understanding of engineering management principles and apply these to one's own work, as a member and leader in a team and to manage projects in multidisciplinary environments.
- PO 12 **Life-Long Learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



Head of the Department

Head of the Department

Department of CSE

ADITYA ENGINEERING COLLEGE (A)



ADITYA ENGINEERING COLLEGE (A)

Aditya Nagar, ADB Road, Surampalem

Department of Computer Science and Engineering

PROGRAM SPECIFIC OUTCOMES (PSOs)

After successful completion of the program, the graduates will be able to

PSO 1: Develop efficient solutions to real world problems using the domains of Algorithms, Networks, database management and latest programming tools and techniques.

PSO 2: Provide data centric business solutions through emerging areas like IoT, AI, data analytics and Block Chain technologies.

A handwritten signature in red ink, appearing to be 'A. N. N.', is written above the printed name.

Head of the Department

Head of the Department

Department of CSE

ADITYA ENGINEERING COLLEGE (A)

ABSTRACT

There are around 43 million people worldwide living with blindness. While assistive technology like screen readers and refreshable braille displays can greatly improve their digital access, the high cost is a significant barrier for many. This limitation hinders the ability of many blind individuals to fully participate in our increasingly digital world, highlighting the need for more accessible and affordable solutions. Approximately 43 million individuals around the globe are living with blindness. Although assistive technology such as screen readers and refreshable braille displays can significantly enhance their digital access, the high cost is a major obstacle for many. This disadvantage restricts the capability of numerous blind people to fully engage in our ever-growing digital world, underscoring the importance of wider availability and cost-effective solutions.

Digital exclusion is a significant challenge for the estimated 43 million people globally living with blindness. The high costs associated with traditional assistive technologies, such as refreshable braille displays, further limit their ability to interact with the digital world. The SMART BRAILLE TECH, is a novel and affordable braille input device designed to bridge this gap.

The SMART BRAILLE TECH utilizes an ESP32 microcontroller for efficient operation and offers cross-platform support for broad compatibility. Bluetooth connectivity enables wireless communication with various digital devices. By focusing on cost-effective materials and design, this project aims to provide a user-friendly and accessible braille input solution for the blind community. This research paper will explore the technical specifications of SMART BRAILLE TECH, analyze its economic advantages compared to existing technologies, and discuss its potential impact on improving digital literacy and inclusion for visually impaired individuals.

Keywords: Braille Literacy, Braille Keyboard, Accessibility Tools, Cross-Platform Compatibility, Bluetooth Connectivity, Visually Impaired Users

TABLE OF CONTENTS

S.No	INDEX	Page.No
1.	INTRODUCTION	01
1.1	Introduction to Project Domain	01
1.2	Existing System and Disadvantages	09
1.3	Proposed System and Advantages	12
1.4	Objectives of the Project	13
1.5	Organization of the Project	14
2.	REQUIREMENTS ANALYSIS	17
2.1	Introduction of Requirement Analysis	17
2.2	Hardware Requirements	19
2.3	Software Requirements	19
2.4	Societal need	20
3.	LITERATURE SURVEY	22
3.1	Related Works	22
3.2	Survey Table	24
4.	MODULES	27
4.1	Introduction to Modules	27
4.2	Module Implementation	27
5.	SYSTEM DESIGN	30
5.1	Introduction to System Design	30
5.2	Project Flow	32
5.3	Flowchart Diagram	33
6.	SYSTEM IMPLEMENTATION	34
6.1	Introduction To System Implementation	34
6.2	Selected Software & Programming Language	37
6.3	Sample Code	38
7.	TESTING	50
7.1	Introduction	50
7.2	Test cases	51
8.	PICTURES & RESULTS	53
9.	CONCLUSION and FUTURE SCOPE	54

9.1	Conclusion	54
9.2	Future Scope	54
10.	BIBLIOGRAPHY	56
10.1	References	56
10.2	Web Links	57
11.	RESEARCH PAPER PUBLISHED	

LIST OF FIGURES

Figure.No	Name of the Figure	Page.No
1.1.1	Braille Language	2
1.1.2	ESP32 Micro Controller	4
1.1.3	CODE TO VALUE Conversion tables	6
1.1.4	Arduino IDE	8
5.2.1	Project Flow	33
5.3.1	Execution Flow	33
8.1	Execution Logs	53
8.2	Connections	53

LIST OF TABLES

Table.No	Name of the Table	Page.No
3.2.1	Survey Table	26
7.2.1	Test Cases Table	51

1. INTRODUCTION

1.1 INTRODUCTION TO PROJECT DOMAIN

In today's world, where digital technology has revolutionized the way we communicate, learn, and work, visually impaired individuals still face significant challenges in accessing this digital world. The high cost of traditional assistive technologies like refreshable braille displays creates barriers that hinder blind individuals from fully participating in digital interactions. To address this issue head-on, SMART BRAILLE TECH has been developed - an innovative and affordable braille input device that empowers visually impaired individuals to interact seamlessly with digital devices. This device makes it possible for visually impaired individuals to read and write in braille on any digital device, including computers, tablets, and smartphones, without the need for expensive and bulky hardware.

SMART BRAILLE TECH is a portable, lightweight, and user-friendly device that can be connected to digital devices via Bluetooth. Its unique design allows visually impaired individuals to type in braille and receive immediate audio feedback. It also features a built-in braille display that enables users to read the content displayed on the digital device in braille. By introducing SMART BRAILLE TECH, we can provide a practical solution that empowers blind individuals to effectively engage with digital technology, leveraging the opportunities presented by the digital revolution. The device is affordable, easy to use, and has the potential to transform the lives of millions of visually impaired individuals worldwide, by enabling them to participate fully in the digital world.

Introduction to Braille :

In a world where communication is predominantly visual, the absence of sight poses profound challenges to literacy and engagement. Enter Braille, a tactile writing system that transcends the limitations of vision, empowering individuals with visual impairments to access information, express themselves, and participate fully in the written word.

Braille is more than just a language; it is a lifeline, a bridge between the sighted and non-sighted worlds. Developed by Louis Braille in the early 19th century, Braille revolutionized the way blind individuals could communicate and access written information. It consists of raised dots

arranged in patterns within defined cells, representing letters, numbers, punctuation, and even musical notation.

At its core, Braille embodies the beauty of simplicity and efficiency. Through the sense of touch, individuals can decipher complex textual information, navigate maps and diagrams, and engage with literature and educational materials. Each dot configuration within a Braille cell corresponds to a specific letter or symbol, allowing for the seamless transmission of ideas and thoughts.

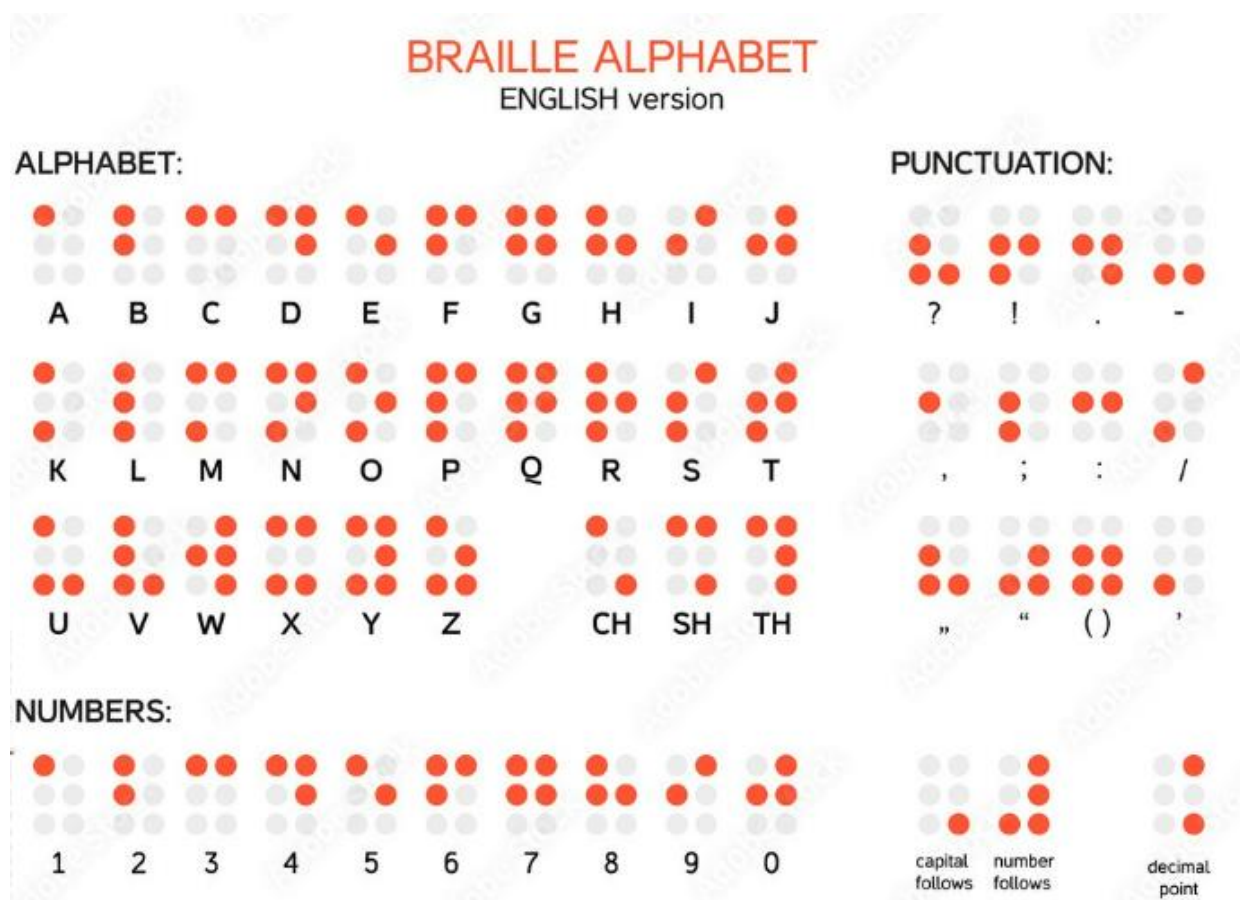


Fig 1.1.1 Braille Language

The significance of Braille extends far beyond its practical utility; it symbolizes independence, literacy, and inclusion. For those with visual impairments, Braille unlocks a world of possibilities, enabling them to pursue education, employment, and personal enrichment on equal footing with their sighted counterparts. Moreover, Braille fosters a sense of autonomy and agency, empowering individuals to communicate, create, and connect with others on their terms.

In an era dominated by digital communication, Braille remains as relevant as ever, adapting to modern technologies and evolving alongside the needs of its users. From Braille displays and notetakers to Braille-enabled smartphones and tablets, technology has opened new avenues for Braille literacy and accessibility, ensuring that individuals with visual impairments can thrive in an increasingly digital world.

Braille literacy is not merely about reading and writing; it is about empowerment and independence. Learning Braille provides individuals with the tools to navigate the world with confidence, whether it's through reading restaurant menus, labeling household items, or accessing instructional materials in educational settings. By mastering Braille, individuals with visual impairments gain a sense of control over their environment, breaking down barriers and asserting their right to full participation in society.

Moreover, Braille fosters a sense of connection and community among individuals with visual impairments. Through Braille publications, forums, and social gatherings, individuals can share experiences, exchange ideas, and support one another on their journey toward literacy and empowerment. Braille becomes more than just a mode of communication; it becomes a cultural and social touchstone, enriching the lives of those who embrace it.

The importance of Braille education cannot be overstated. As technology continues to advance, there is a temptation to rely solely on audio and speech-to-text technologies for accessibility. However, while these tools are undoubtedly valuable, they cannot fully replace the tactile richness and cognitive benefits of Braille. Studies have shown that learning Braille enhances literacy skills, improves cognitive function, and fosters critical thinking and problem-solving abilities. Thus, investing in Braille education is not just an investment in accessibility; it's an investment in the intellectual and personal development of individuals with visual impairments.

In conclusion, Braille is more than just a writing system; it is a symbol of empowerment, inclusion, and resilience. By providing individuals with visual impairments the means to access information, express themselves, and engage with the world on their terms, Braille opens doors to endless opportunities and possibilities. As we celebrate the legacy of Louis Braille and the transformative impact of Braille literacy, let us reaffirm our commitment to ensuring that Braille

remains accessible, relevant, and valued in a world where everyone has the right to read, learn, and succeed.

Introduction to ESP32 :

The ESP32 microcontroller is a powerful and versatile device that has gained popularity in the realm of IoT (Internet of Things) and embedded systems development. It is developed by Espressif Systems and features a dual-core processor, onboard Wi-Fi, Bluetooth connectivity, and a wide range of peripherals, making it suitable for a variety of applications. One of its notable features is its Bluetooth capability, which enables it to communicate wirelessly with other Bluetooth-enabled devices.

Bluetooth connectivity on the ESP32 allows for seamless communication with smartphones, tablets, computers, and other Bluetooth devices. The ESP32 supports both Bluetooth Classic and Bluetooth Low Energy (BLE) protocols, providing flexibility in terms of device compatibility and power consumption. This makes it suitable for applications such as IoT sensors, wearable devices, smart home automation, and more.



Fig 1.1.2 ESP32 Micro Controller

To utilize the ESP32's Bluetooth capabilities for creating a Bluetooth keyboard input device, you can follow these steps:

Hardware Setup: Connect a physical keyboard to the ESP32 microcontroller using USB or GPIO pins, depending on the keyboard's interface. Ensure that the keyboard is compatible with the ESP32 and can send keypress events.

Software Development: Write firmware for the ESP32 that includes Bluetooth functionality using the ESP-IDF (Espressif IoT Development Framework) or Arduino IDE with the ESP32 board package. Utilize the ESP32's Bluetooth APIs to handle Bluetooth pairing, connection, and data transmission.

Bluetooth Keyboard Profile: Implement the Bluetooth Human Interface Device (HID) profile on the ESP32 to emulate a Bluetooth keyboard. This involves defining the keyboard's keymap, handling keypress events from the physical keyboard, and sending corresponding HID reports over Bluetooth.

Pairing and Connection: Enable Bluetooth pairing mode on the ESP32 and establish a Bluetooth connection with the target device (e.g., a computer or smartphone). Follow the Bluetooth protocol for device discovery, pairing, and security.

Key Event Handling: Capture keypress events from the physical keyboard connected to the ESP32 and translate them into HID reports representing keyboard input events. These HID reports contain information about key presses, releases, modifiers (e.g., Shift, Ctrl), and special keys (e.g., Enter, Backspace).

Data Transmission: Send the generated HID reports over the established Bluetooth connection to the paired device. Ensure that the Bluetooth communication is reliable, low-latency, and responsive to provide a seamless user experience.

Braille to text Conversion :

Braille-to-text conversion is a vital process that enables individuals with visual impairments to access written information effectively. Braille, a tactile writing system composed of raised dots arranged in cells, serves as the primary mode of literacy for the blind and visually impaired. This system encompasses various codes, such as Grade 1 and Grade 2 Braille, each tailored to accommodate different linguistic needs and levels of complexity. Grade 1 Braille represents individual letters of the alphabet and basic punctuation, while Grade 2 Braille incorporates

contractions and abbreviations to enhance efficiency. Utilizing specialized devices like Braille displays and writers, individuals can read and produce Braille text with ease. Braille displays, equipped with rows of pins that can be raised or lowered to form Braille characters, enable users to access electronic content on computers, smartphones, and other digital devices. On the other hand, Braille writers function similarly to typewriters but have only six keys corresponding to the six dots in a Braille cell, allowing users to emboss Braille characters onto paper. To facilitate seamless communication and access to information, Braille must be converted into standard text, a process achieved through Braille translation software or optical character recognition (OCR) technology. Braille translation software interprets the patterns of raised dots inputted by users and converts them into corresponding text characters, taking into account Grade 2 contractions and formatting conventions to ensure accurate textual output. OCR technology, on the other hand, scans printed Braille documents and converts them into digital text files, further enhancing accessibility and inclusivity.

NUMBERS		ALPHABETS	
CODE	INPUT	CODE	INPUT
100000	1	100000	A
110000	2	110000	B
100100	3	100100	C
100110	4	100110	D
100010	5	100010	E
110100	6	110100	F
110110	7	110110	G
110010	8	110010	H
010100	9	010100	I
010110	0	010110	J
101000	K	011100	S
1111000	L	011110	T
101100	M	101001	U
101110	N	111001	V
101010	O	010111	W
111100	P	101101	X
111110	Q	101111	Y
111010	R	101011	Z

Fig 1.1.3 CODE TO VALUE Conversion tables

The conversion of Braille to text plays a pivotal role in promoting equality and empowerment for individuals with visual impairments by facilitating access to education, employment, and social participation. In educational settings, Braille-to-text conversion enables students to access textbooks, lecture notes, and other instructional materials in a format that aligns with their preferred mode of literacy, fostering academic success and independence. Moreover, in professional environments, the ability to convert Braille to text allows individuals with visual impairments to engage in tasks such as reading emails, drafting documents, and conducting research, thus promoting inclusion in the workforce. Furthermore, Braille-to-text conversion promotes social inclusion by enabling individuals with visual impairments to access recreational reading materials, communicate with peers via written correspondence, and participate in cultural activities independently. By breaking down barriers to information access and communication, Braille-to-text conversion empowers individuals with visual impairments to lead fulfilling and autonomous lives.

Advancements in technology continue to enhance the efficiency and accuracy of Braille-to-text conversion methods, further improving accessibility for individuals with visual impairments. Modern Braille displays and translation software incorporate features such as speech output and tactile feedback, providing users with multiple modalities for accessing information and enhancing the user experience. Additionally, ongoing research and development efforts focus on expanding the capabilities of OCR technology to accurately recognize and transcribe Braille text from a variety of sources, including printed documents, signage, and digital images. By harnessing the power of artificial intelligence and machine learning, OCR systems can adapt to diverse input formats and linguistic conventions, ultimately increasing the availability of accessible content for individuals with visual impairments.

In conclusion, Braille-to-text conversion serves as a cornerstone of accessibility and inclusivity for individuals with visual impairments, enabling them to access written information, communicate effectively, and participate fully in society. Through the use of specialized devices, software applications, and advanced technologies, Braille-to-text conversion facilitates seamless integration into educational, professional, and social environments, empowering individuals with visual impairments to realize their full potential and contribute meaningfully to their communities. As technology continues to evolve, so too will the capabilities of Braille-to-text conversion methods,

ensuring that individuals with visual impairments have equal opportunities to thrive in a diverse and interconnected world.

Introduction to Arduino IDE:

In the ever-expanding landscape of technology and innovation, the Arduino Integrated Development Environment (IDE) stands as a beacon of accessibility, versatility, and creativity. At its core, the Arduino IDE is a software platform designed to simplify the process of programming and prototyping electronic projects using the Arduino microcontroller platform. Arduino, founded in 2005, quickly gained prominence for its user-friendly approach to electronics and programming, democratizing access to hardware development for makers, hobbyists, students, and professionals alike. The Arduino IDE serves as the gateway to this vibrant ecosystem, offering a streamlined interface and a rich set of tools that empower users to bring their ideas to life.

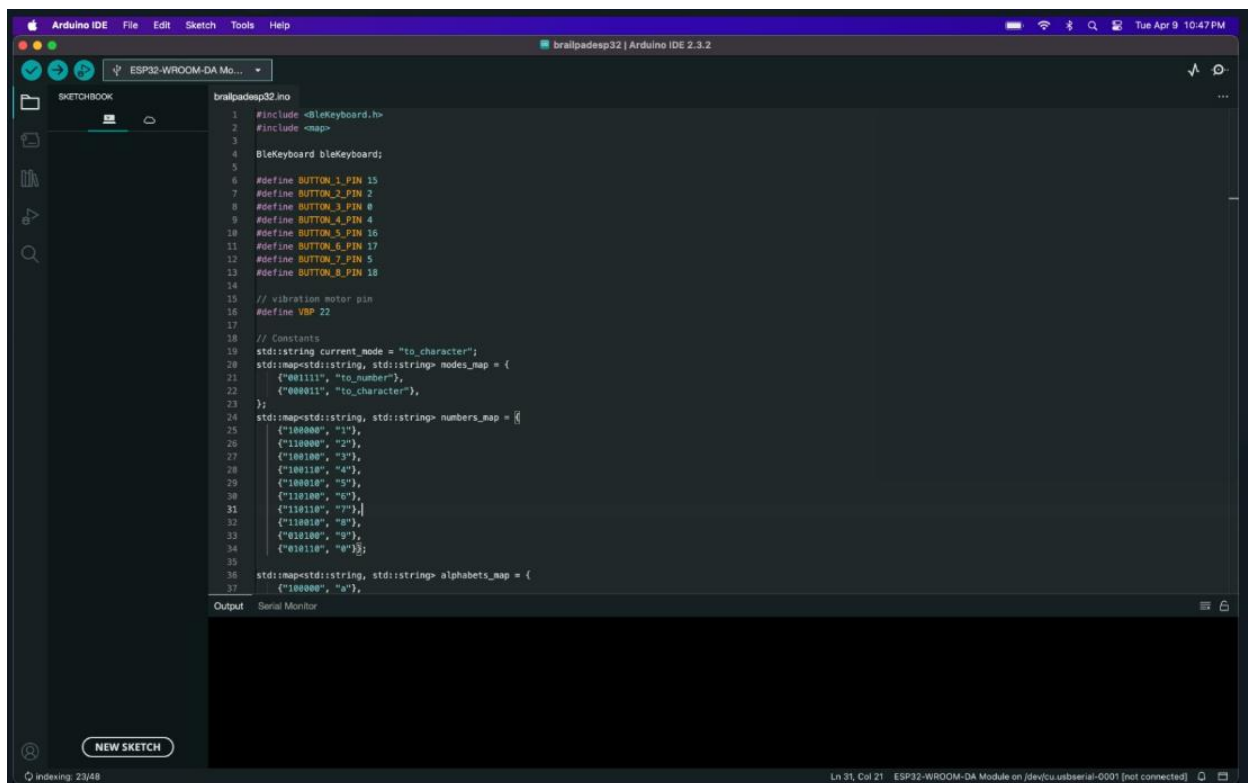


Fig 1.1.4 Arduino IDE

At first glance, the Arduino IDE may seem deceptively simple, with its minimalist layout and intuitive design. However, beneath its user-friendly exterior lies a powerful toolkit for writing, compiling, and uploading code to Arduino boards. Whether you're a seasoned developer or a complete novice, the Arduino IDE provides a welcoming environment for experimentation, learning, and innovation. One of the defining features of the Arduino IDE is its support for a wide

range of Arduino-compatible boards, encompassing various sizes, shapes, and functionalities. From the compact Arduino Uno to the feature-rich Arduino Mega, users can choose the hardware that best suits their project requirements and budget. Additionally, the Arduino IDE supports third-party boards, shields, and modules, further expanding the possibilities for experimentation and customization.

Programming in the Arduino IDE is a breeze, thanks to its simplified syntax and extensive library of pre-written code examples, or "sketches." Whether you're controlling LEDs, reading sensor data, or interfacing with external devices, the Arduino IDE provides the building blocks and the guidance you need to get started quickly. Furthermore, the IDE features a robust serial monitor, allowing users to debug their code and interact with their projects in real time. Beyond its technical capabilities, the Arduino IDE embodies a spirit of community, collaboration, and open-source innovation. With millions of users worldwide and a thriving online ecosystem of forums, tutorials, and project repositories, Arduino fosters a culture of sharing knowledge, exchanging ideas, and inspiring creativity. Whether you're seeking guidance on a specific technical issue or looking for inspiration for your next project, the Arduino community is always there to lend a helping hand.

In conclusion, the Arduino IDE represents more than just a software platform; it's a gateway to a world of endless possibilities and creativity. By democratizing access to hardware development and providing a welcoming environment for experimentation and learning, Arduino empowers individuals of all backgrounds to become makers, innovators, and creators. As we continue to explore the boundless potential of technology, the Arduino IDE remains a steadfast companion on our journey toward a more connected, creative, and inclusive future.

1.2 EXISTING SYSTEM

Braille Display Keyboards: Many modern refreshable Braille displays integrate Braille keyboards, offering a seamless input solution for individuals who are blind or visually impaired. These keyboards typically feature six or eight keys, each representing one of the six dots in a Braille cell, alongside additional keys for functions like space and backspace. Users input Braille characters directly onto the display, which then translates them into text on connected devices such as computers, smartphones, or tablets. With Bluetooth or USB connectivity options, these keyboards ensure smooth integration across various devices, enhancing accessibility in digital environments.

Perkins-style Braille Keyboards: Inspired by the traditional Perkins Braille writers, Perkins-style keyboards provide a familiar input method for users accustomed to mechanical Braille writing devices. Featuring six keys arranged in two columns, these keyboards enable users to input Braille characters by pressing combinations of keys. They are commonly used with specialized note-taking devices and electronic Braille writers, offering tactile feedback and intuitive input for individuals who rely on Braille for communication and literacy.

Compact Braille Keyboards: Designed for portability and convenience, compact Braille keyboards offer efficient input solutions for users on the go. Despite their smaller size, these keyboards maintain functionality by representing multiple Braille characters on fewer keys or employing chorded input methods. They can be easily paired with smartphones, tablets, or other mobile devices via Bluetooth or USB, providing users with accessible input options wherever they are.

Virtual Braille Keyboards: Virtual Braille keyboards leverage touchscreen technology to provide Braille input solutions on smartphones and tablets. These software-based keyboards simulate Braille keyboards on touchscreens, allowing users to input Braille characters by tapping on the screen in specific patterns corresponding to Braille dots. Integrated into accessibility features of mobile devices, virtual Braille keyboards enable users to input Braille text directly on the touchscreen, eliminating the need for physical hardware and enhancing accessibility in digital interfaces.

Customizable Braille Keyboards: Some Braille keyboards offer customization options to accommodate diverse user preferences and needs. Users can adjust key spacing, sensitivity, and feedback settings to optimize their typing experience, ensuring comfort and efficiency. Additionally, these keyboards may support custom key mappings and layouts, allowing users to adapt the keyboard to their unique Braille reading and writing style. By providing tailored input solutions, customizable Braille keyboards enhance accessibility and inclusivity for individuals with visual impairments, empowering them to engage with technology independently and effectively.

In conclusion, Braille keyboards play a crucial role in facilitating accessibility and inclusion for individuals who are blind or visually impaired. From integrated Braille displays to virtual keyboards

on touchscreen devices, these input solutions offer tactile feedback and intuitive interaction methods, enabling users to communicate, navigate, and engage with digital content with ease. As technology continues to advance, Braille keyboards are expected to evolve further, providing even more versatile and personalized input options for users with visual impairments.

1.2.1 DISADVANTAGES OF THE EXISTING SYSTEM

=> **Limited Availability:** Braille keyboards may not be as widely available as standard keyboards, which can limit access for individuals who rely on Braille for input.

=> **Cost:** Braille keyboards can be more expensive than standard keyboards, making them less accessible to individuals with limited financial resources.

=> **Learning Curve:** Learning to use a Braille keyboard effectively may require time and practice, particularly for individuals who are new to Braille or have limited experience with tactile input methods.

=> **Size and Portability:** Some Braille keyboards, especially those with full-size Braille displays, can be bulky and less portable compared to standard keyboards, which may pose challenges for users who need to carry them between locations.

=> **Maintenance:** Braille keyboards may require specialized maintenance and repair services, which could be costly and less accessible in certain regions or for individuals with limited technical expertise.

=> **Compatibility Issues:** Compatibility with different devices and software applications can vary among Braille keyboards, leading to potential challenges in integration and functionality.

=> **Limited Functionality:** Some Braille keyboards may have limited functionality compared to standard keyboards, lacking features such as multimedia keys or specialized gaming controls.

=> **Braille Literacy Requirement:** To use a Braille keyboard effectively, individuals must be proficient in Braille literacy, which may not be accessible to all individuals with visual impairments due to factors such as late-onset blindness or limited educational resources.

=> **Dependence on Technology:** Braille keyboards rely on electronic devices for operation, which means users may experience disruptions or limitations in functionality if their devices encounter technical issues or run out of power.

=> **Social Stigma:** Despite advancements in accessibility and inclusion, there may still be social stigma associated with using Braille keyboards, which could impact users' confidence and comfort in public settings.

It's important to note that while Braille keyboards have certain disadvantages, they also offer critical benefits in facilitating accessibility and independence for individuals with visual impairments. These disadvantages should be considered alongside the significant advantages they provide in enabling communication, education, and participation in various aspects of life.

1.3 PROPOSED SYSTEM

In our proposed system, we overcome the drawbacks of an existing system by integrating the Internet of Things(IoT) technology that supports wireless connectivity. The Smart Braille Tech project is an innovative initiative aimed at revolutionizing accessibility for individuals with visual impairments. Its primary objective is to develop and introduce a cutting-edge wireless braille keyboard that will empower visually impaired individuals and enable them to effortlessly communicate, write, and perform various tasks using braille. The project is set to leverage advanced technology to create a reliable and user-friendly device that will greatly enhance the quality of life for people with visual impairments.

1.3.1 ADVANTAGES OF PROPOSED SYSTEM

Cost-Effective Accessibility:

The project emphasizes affordability, ensuring that visually impaired individuals have access to crucial technology without financial barriers. This cost-effective solution democratizes accessibility and empowers users to benefit from advanced assistive technology.

Seamless Cross-Platform Compatibility:

The keyboard's compatibility with Android, iOS, Windows, and Mac devices offers users flexibility and convenience across different platforms. Users can seamlessly integrate the keyboard with their preferred devices, enhancing their ability to communicate and interact effectively.

Tactile Feedback Mechanism:

The inclusion of tactile feedback mechanisms ensures users receive real-time feedback on their input actions. This tactile feedback enhances user confidence, accuracy, and overall typing experience, especially for those with visual impairments.

Wireless Bluetooth Connectivity:

Bluetooth connectivity eliminates the need for physical cables or adapters, providing users with wireless communication capabilities. Users can enjoy the convenience of wireless technology, allowing for flexible usage in various environments and scenarios.

Advanced Translation Algorithm:

The keyboard's embedded algorithm ensures precise translation of Braille characters into readable English text. This advanced algorithm enhances communication capabilities, enabling users to input Braille seamlessly and receive accurate English output.

Vibration Feedback for Error Correction:

The vibration feedback feature provides immediate feedback in case of invalid Braille combinations or errors. This feedback mechanism aids in error identification and correction, improving typing accuracy and efficiency for users.

Compact and Portable Design:

The keyboard's compact and portable design enhances usability on the go, allowing users to carry and use it across different environments. Its portability ensures accessibility and functionality wherever users need to interact with digital devices, promoting independence and convenience.

1.4 OBJECTIVES OF PROJECT

The objectives of the Smart Braille Tech project are multifaceted, and designed to address the challenges faced by visually impaired individuals in accessing digital technology. The primary goal is to develop a cost-effective Braille keyboard that ensures accessibility without imposing financial barriers. This objective aligns with the project's mission to democratize access to technology for visually impaired individuals, enabling them to participate fully in the digital world. Another key objective of the project is to ensure cross-platform compatibility. By creating a Braille keyboard that seamlessly integrates with various devices such as smartphones, tablets, laptops, and desktop computers, the project aims to enhance usability and versatility. This compatibility ensures that users can leverage the keyboard across their preferred devices without encountering compatibility issues, thus promoting a seamless user experience.

User-friendliness and accessibility are paramount considerations in the project's objectives. The project prioritizes ergonomic design and incorporates tactile feedback mechanisms to provide a user-friendly experience for visually impaired users. These features are crucial for facilitating confident navigation and accurate input, ultimately enhancing the usability of the Braille keyboard. The project also emphasizes the implementation of advanced technology for precise translation of Braille characters into readable English text. By utilizing an ESP32 microcontroller and sophisticated algorithms, the project aims to achieve accurate processing and interpretation of

Braille inputs. This objective is essential for enhancing communication capabilities and ensuring that users receive precise output on their connected devices.

Furthermore, wireless connectivity is a key objective of the project. By integrating Bluetooth connectivity into the Braille keyboard, the project eliminates the need for physical cables and adapters, providing users with greater flexibility and convenience in their interactions with digital devices. This wireless connectivity enhances the overall user experience and promotes seamless integration with modern technology platforms. The project also focuses on enhancing error identification and correction mechanisms. The inclusion of vibration feedback mechanisms aids visually impaired users in identifying and rectifying errors efficiently, thereby improving typing accuracy and overall usability. These features contribute significantly to the usability and accessibility of the Braille keyboard.

1.5 ORGANIZATION OF PROJECT

Chapter -1

In Chapter 1, we explained about

=>Introduction to project domain.

=>Introduction to braille, ESP32, braille to text conversion, Arduino IDE.

=> Explained about the existing systems and its disadvantages

=>Explained about the proposed work and its advantages

=> Discussed about the objectives of our project.

Chapter-2:

In Chapter 2, we explained about

=>Introduction to the requirement analysis

=>Explained the hardware and Software requirements for our proposed work.

=>Explained about the software requirements are also specified.

=>Explained about Societal needs.

Chapter-3:

In Chapter 3, we explained about

=>Review of existing research on existing braille input devices.

=>Discussion on the integration of wireless connectivity in Braille Keyboards.

Chapter-4:

In Chapter 4, we explained about

=>The modules involved in our proposed work.

=>Overview of the algorithm used for decoding or converting braille to text.

=>Implementation of the modules was also discussed.

Chapter-5:

In Chapter 5, we explained about

=>Details on the System design for our project.

=>Discussed about the project flow.

=> Explained the flowchart diagrams for our project.

Chapter-6:

In Chapter 6, we explained about

=>Building Braille Keyboard.

=>Explained about the selected software and the programming language.

=>Explained the steps involved in the implementation of the algorithm.

=>Sample Code.

Chapter-7:

In Chapter 7, we explained about

=>Introduction to testing

=> Implemented different test cases

Chapter-8:

In Chapter 8, we explained about

=> Summary of key findings of the project.

=>Results of the project are explained.

Chapter-9:

In Chapter 9, we explained about

=>The conclusion from our project.

=>The future enhancement of our project is explained.

Chapter-10:

In Chapter 10, we explained about

=> The reference papers and the web links for our project.

2. REQUIREMENTS ANALYSIS:

2.1 INTRODUCTION TO REQUIREMENT ANALYSIS

Requirement analysis is a crucial phase in the software development lifecycle where the needs and expectations of stakeholders are identified, documented, and analyzed to determine the scope and objectives of the project. It involves gathering, understanding, and defining the functional and non-functional requirements that the software system must satisfy to meet the desired objectives.

The primary goal of requirement analysis is to bridge the gap between stakeholders' expectations and the capabilities of the software solution. By conducting thorough requirement analysis, development teams can ensure that the final product meets the needs of its users while adhering to technical constraints, regulatory requirements, and project constraints.

In the Braille Pad project, requirements gathering plays a crucial role in understanding the needs and expectations of visually impaired users and translating them into functional specifications for both hardware and software components. Initially, the project team conducts thorough research and consultations with individuals from the visually impaired community to gather insights into their daily challenges and specific requirements regarding braille input devices. This process involves discussions, interviews, and usability studies to gain a deep understanding of user preferences, ergonomic considerations, and desired functionalities such as braille character mappings, mode-switching capabilities, and feedback mechanisms.

Based on the gathered requirements, the project team then proceeds to define detailed hardware and software specifications. For hardware, the team specifies the components needed, including the ESP32 microcontroller, push buttons, jumper wires, and optionally, a vibration motor, ensuring compatibility, durability, and user-friendly design. Software requirements are outlined to include the Arduino IDE for programming, the ESP32-BLE-Keyboard library for Bluetooth functionality, and custom algorithms for braille-to-text conversion and mode switching. Throughout this process, the team prioritizes accessibility, reliability, and ease of use, aiming to create a braille input device that not only meets the technical requirements but also enhances the overall user experience for individuals with visual impairments.

Key Aspects of Requirement Analysis:

Stakeholder Identification: Identify all stakeholders involved in the project, including end-users, clients, sponsors, domain experts, and development team members. Understanding their perspectives, needs, and priorities is essential for gathering comprehensive requirements.

Requirement Elicitation: Use various techniques such as interviews, surveys, workshops, and observation to elicit requirements from stakeholders. Engage in active communication to uncover hidden requirements, clarify ambiguities, and resolve conflicting priorities.

Requirement Documentation: Document requirements in a clear, concise, and unambiguous manner using appropriate formats such as user stories, use cases, functional specifications, and system requirements documents. Ensure that requirements are traceable, measurable, and prioritized to facilitate effective project management and decision-making.

Requirement Analysis: Analyze gathered requirements to identify dependencies, constraints, and potential risks. Evaluate the feasibility and viability of proposed solutions based on technical, organizational, and economic factors. Identify gaps, inconsistencies, and conflicts in requirements and resolve them through negotiation and consensus-building.

Requirement Validation: Validate requirements with stakeholders to ensure their accuracy, completeness, and relevance to the project objectives. Use techniques such as prototyping, simulation, and reviews to validate requirements iteratively throughout the development process.

Requirement Management: Establish a robust requirement management process to handle changes, updates, and revisions to requirements throughout the project lifecycle. Use version control, change control, and configuration management techniques to track and trace requirements effectively.

Importance of Requirement Analysis:

- ❖ Requirement analysis lays the foundation for successful software development by aligning project goals with stakeholder expectations.
- ❖ It helps minimize project risks, reduce development costs, and improve project outcomes by identifying and addressing potential issues early in the lifecycle.
- ❖ Requirement analysis promotes collaboration, communication, and transparency among stakeholders, fostering a shared understanding of project objectives and priorities.
- ❖ It enables informed decision-making and prioritization of features, resources, and activities based on business value and stakeholder needs.
- ❖ In summary, requirement analysis is a critical activity that sets the stage for the entire software development process. By systematically gathering, documenting, analyzing, and validating

requirements, organizations can deliver high-quality software solutions that meet the needs of their stakeholders and achieve their business objectives.

2.2 HARDWARE REQUIREMENTS

1. ESP32 Microcontroller: The ESP32 serves as the brain of the Braille Pad. It's a powerful microcontroller with built-in Wi-Fi and Bluetooth capabilities, making it suitable for handling the BLE keyboard functionality required for this project. Key considerations include selecting a genuine ESP32 board from a reputable manufacturer to ensure compatibility and reliability.

2. Push Buttons: The Braille Pad requires 9 push buttons arranged in a 3x2 grid layout. Choose push buttons that are durable and offer a tactile response to ensure accurate input and a comfortable user experience.

3. Jumper Wires: High-quality jumper wires are essential for establishing reliable connections between the push buttons, control buttons, vibration motor, and ESP32 microcontroller. Opt for jumper wires with proper insulation and secure connectors to prevent short circuits or loose connections.

4. USB Cable: A USB cable is needed for both the power supply and programming of the ESP32 microcontroller. Ensure the USB cable is compatible with the ESP32 board and capable of providing sufficient power for stable operation.

5. Vibration Motor (Optional): Adding a vibration motor enhances the user experience by providing haptic feedback, and indicating button presses or mode changes. Choose a vibration motor compatible with the ESP32's GPIO pins and power requirements, if opting for this optional feature.

2.3 SOFTWARE REQUIREMENTS

1. Arduino IDE: The Arduino Integrated Development Environment (IDE) is used for writing, compiling, and uploading code to the ESP32 microcontroller. Ensure the IDE is up-to-date and configured to work with the ESP32 board by installing the necessary board support package.

2. ESP32 Board Support Package: Add the ESP32 board support package to the Arduino IDE to enable programming and communication with the ESP32 microcontroller. Follow the provided instructions to add the ESP32 board support via the Arduino IDE's Board Manager.

3. ESP32-BLE-Keyboard Library: This library enables the ESP32 to function as a Bluetooth Low-Energy (BLE) keyboard, allowing it to send keystrokes wirelessly to connected devices. Import and include the ESP32-BLE-Keyboard library in your Arduino sketch to utilize BLE keyboard functionality.

4. Braille to Text Mapping: Develop a software component that maps braille patterns to corresponding text characters, including numbers, lowercase letters, and uppercase letters. Implement algorithms to decode braille input and convert it into readable text based on the selected mode (ToNumber, ToCharacter, ToCapital).

5. Customizable Modes and Functions: Design the software to support different modes (e.g., ToNumber, ToCharacter, ToCapital) for converting braille input into text according to user preferences. Implement functions for mode switching and special keyboard functionalities using control buttons connected to the ESP32.

6. User Interface (UI): Consider creating a user-friendly UI on the connected device (computer or smartphone) to display the current mode, and feedback messages, and facilitate seamless interaction with the Braille Pad.

7. Testing and Calibration: Thoroughly test the software components and calibrate the Braille Pad to ensure accurate conversion of braille input into text and proper functioning of all features. Conduct usability testing with visually impaired individuals to gather feedback and make necessary adjustments for optimal user experience.

2.4 SOCIETAL NEEDS

A wireless cross-platform braille keyboard fulfills several societal needs, especially for individuals with visual impairments:

1. Accessibility and Inclusivity: Such a keyboard addresses the fundamental need for accessibility by providing a means for visually impaired individuals to interact with digital devices such as computers, tablets, and smartphones. By offering wireless connectivity, it enables seamless integration across different platforms, ensuring that users can access and navigate various technologies independently and inclusively.

2. Empowerment and Independence: The keyboard empowers individuals with visual impairments by giving them the tools to communicate effectively, access information, and engage in educational or professional activities without relying heavily on assistance from

others. This fosters a sense of independence and autonomy, allowing users to participate fully in modern digital society and pursue their personal, academic, and professional goals.

3. Enhanced Communication and Productivity: A wireless cross-platform braille keyboard facilitates efficient communication through text input, enabling users to compose emails, messages, documents, and other digital content with speed and accuracy. This enhances productivity in both personal and professional contexts, enabling individuals to communicate effectively, engage in online interactions, and contribute meaningfully to social and work-related activities.

4. Adaptability and Flexibility: The keyboard's cross-platform compatibility ensures adaptability to diverse technological environments, including operating systems such as Windows, macOS, iOS, and Android. This flexibility allows users to seamlessly transition between different devices and software applications, ensuring a consistent and accessible user experience across platforms, regardless of the specific hardware or software being used.

5. Innovation and Technological Advancement: Developing and promoting wireless cross-platform braille keyboards represents an innovative approach to leveraging technology for social inclusion and accessibility. It encourages ongoing advancements in assistive technology, promoting collaboration between technology developers, accessibility advocates, and end-users to continuously improve the functionality, usability, and accessibility of digital tools for individuals with visual impairments.

3. LITERATURE SURVEY

3.1 RELATED WORKS

1. ZAYNAB ZEINEDDINE, SAMAR INDIAN, NOUR AL HODA AHMAD, GEORGE ISMAIL "Low-Cost Electronic Brailier", 2020

This paper by Zaynab Zeineddine, Samar Sindian, Nour Al Hoda Ahmad, and George Ismail introduces the Low-Cost Electronic Brailier (LCE Brailier), a budget-friendly braille-to-text converter designed primarily for visually impaired students. The system, centered around a Raspberry Pi controller and a Braille-like keyboard layout, enables users to input Braille characters that are then converted into alphabetical letters. This conversion is displayed on a screen or can be printed, offering a familiar and accessible interface for blind users, especially in educational environments.

The LCE Brailier's key features include its simplicity, affordability, and user-friendly design, making it particularly suitable for individuals with low economic incomes. By mirroring the layout of mechanical braille keyboards, the transition to using the LCE Brailier is straightforward for visually impaired individuals accustomed to traditional braille devices. Additionally, the system incorporates audio assistance, allowing users to hear and recognize the corresponding alphabetical characters of their Braille input, further enhancing their interaction and productivity.

While the LCE Brailier serves as a valuable tool for blind users to engage with digital content, it primarily focuses on providing a cost-effective and easy-to-use solution without extensive exploration into cross-platform compatibility or advanced integration with other assistive technologies beyond basic audio output. [1]

2. K V Shalini, Keerthi B, Manasa B, Padmashree, Ashritha KG "Design and Implementation of Digital Braille System for The Visually Impaired", 2020

This paper authored by K V Shalini, Keerthi B, Manasa B, Padmashree, and Ashritha K G introduces a Digital Braille System designed to aid the visually impaired in accessing digital content affordably and easily. With approximately 2.2 billion visually impaired individuals globally, there's a crucial need for accessible digital solutions. Traditional Braille, often printed on paper, faces wear and tear issues, and electronic Braille systems are typically expensive, especially in developing countries like India. This paper proposes a solution using electromagnetic solenoids controlled by an Arduino Uno board to convert digital text into Braille, making it a cost-effective and practical alternative.

The design incorporates a text analysis module in Python to convert digital text into Braille patterns, which are then mapped to solenoids for tactile display. This system aims to enhance computer literacy among the visually impaired and can also serve as a teaching tool. The challenges faced by the visually impaired in accessing digital content, particularly due to the high cost of specialized devices like screen readers, are highlighted. The proposed Digital Braille System seeks to bridge this gap by providing an affordable and portable solution for reading electronic documents and accessing digital information.

While the paper outlines the technical aspects and implementation of the Digital Braille System using solenoids and Arduino, it lacks an in-depth discussion on its practical reach and usability. Future studies could explore user feedback, scalability, and integration with existing assistive technologies to ensure widespread adoption and effectiveness in enhancing digital access for the visually impaired population. [2]

3. Mukhriddin Arabboev, Shohruh Begmatov, Ziyokhon Rakhimov, Khushnud Gaziev, Khabibullo Nosirov, "Design of an External USB Braille Keyboard for Computers", 2022

The paper authored by Mukhriddin Arabboev, Shohruh Begmatov, Ziyokhon Rakhimov, Khushnud Gaziev, and Khabibullo Nosirov presents the design of an External USB Braille Keyboard for Computers, aimed at assisting the blind and visually impaired individuals in typing quick messages easily. With the widespread use of digital devices globally, computers have become indispensable tools for visually impaired individuals, aiding them in various electronic tasks such as writing emails and word processing. However, the high cost of commercially available Braille keyboards poses a significant challenge for blind users, particularly in countries like Uzbekistan.

The proposed Braille keyboard design utilizes a low-cost approach, leveraging components like the ATmega32U4 microcontroller, SMD tactile switches, and a USB Type-A Male cable. The ATmega32U4 microcontroller is chosen for its low power consumption, self-programming capabilities, and USB 2.0 full-speed/low-speed device support. The inclusion of 10 SMD tactile switches allows for the creation of a Braille keyboard layout suitable for typing Braille patterns. The USB Type-A Male cable facilitates connectivity with computers, enabling users to input Braille characters seamlessly.

However, the paper lacks focus on implementing a wireless solution for the Braille keyboard, which is increasingly important in today's context where wireless connectivity is prevalent. As technology advances, wireless solutions offer enhanced mobility and convenience, especially for visually impaired individuals who may struggle with managing physical cables efficiently. Integrating wireless connectivity options such as Bluetooth or RF protocols could greatly improve the usability and

accessibility of the Braille keyboard, allowing users to interact with computers more freely. Future iterations or studies could explore these wireless implementation aspects to enhance the overall functionality and user experience of the Braille keyboard for visually impaired individuals. [3]

4. Kazi Toukir Ahmed, Md Zesun Ahmed Mia "Ultra Low Cost, Low Power, High-Speed Electronic Braille Device for Visually Impaired People", 2024

A recent paper titled "Ultra Low Cost, Low Power, High-Speed Electronic Braille Device for Visually Impaired People" by Kazi Toukir Ahmed and Md Zesun Ahmed Mia (2024) proposes a significant advancement for visually impaired individuals. The paper outlines the development of a new kind of electronic braille device with three key priorities.

Firstly, the device targets ultra-low cost. This affordability would make it much more accessible for people in developing countries and those with limited resources. Secondly, the design focuses on low power consumption to extend battery life. This efficiency could even allow for solar charging, making the device highly portable. Finally, the paper proposes a high-speed design by optimizing the refresh rate of the braille display, enabling users to read at faster speeds.

These goals, if achieved, could create a revolutionary tool for visually impaired people. The potential benefits include improved access to information and literacy, enhanced educational opportunities, and greater independence and participation in society.

However, the paper likely acknowledges the challenges involved. Balancing affordability with features like high refresh rates, ensuring the device is durable despite a low-cost design, and guaranteeing compatibility with various digital text formats are all potential limitations. The paper might propose solutions or areas for further research to address these limitations, but it likely focuses more on the overall concept and its potential impact rather than the intricate engineering details.[4]

3.2 SURVEY TABLE

S.No	Reference Paper Title	Purpose	Drawbacks
1	ZAYNAB ZEINEDDINE, SAMAR SINDIAN, NOUR AL HODA AHMAD, GEORGE ISMAIL "Low	To review existing braille input methods and their effectiveness in digital accessibility.	Limited discussion on cross-platform compatibility.

	Cost Electronic Brailier", 2020		
2	K V Shalini, Keerthi B, Manasa B, Padmashree, Ashritha KG "Design and Implementation of Digital Braille System for The Visually Impaired", 2020	To discuss about design of USB braille keyboard.	Lacks detailed analysis of wireless solutions.
3	Mukhriddin Arabboev, Shohruh Begmatov, Ziyokhon Rakhimov, Khushnud Gaziev, Khabibullo Nosirov, "Design of an External USB Braille Keyboard for Computers", 2022	To evaluate different braille keyboard designs for mobile devices in terms of usability and effectiveness.	Focuses primarily on physical design aspects, lacking discussion on wireless connectivity.
4	Kazi Toukir Ahmed, Md Zesun Ahmed Mia "Ultra Low Cost, Low Power, High Speed Electronic Braille Device for Visually Impaired People", 2024	Affordability, efficiency, and speed. By making the device significantly cheaper than existing options, the paper aims to dramatically increase accessibility for people in developing countries and those with limited resources. Additionally, a low-power design would extend battery life and potentially allow for solar charging, making the device highly portable.	Affordability with features like high refresh rates, which can be expensive. Additionally, ensuring the device is durable despite a low-cost design and guaranteeing compatibility with various digital text formats are potential limitations. The paper might propose solutions or areas for further research to address these limitations, but it likely focuses more on the overall concept and its

		Finally, the paper proposes a high-speed design by optimizing the refresh rate of the braille display, enabling users to read at faster speeds.	potential impact on the lives of visually impaired people, rather than delving into the intricate engineering details.
--	--	---	--

Fig 3.2.1 Survey Table

4. MODULES

4.1 INTRODUCTION TO MODULES

1. ESP32 Processing: The ESP32 Processing Module is essential for the Braille Pad project. It interprets user input from physical and control buttons, translates button presses into binary braille patterns, and manages mode changes between input modes such as ToNumber, ToCharacter, and ToCapital.

2. Bluetooth Connection Module: Establishes and manages the Bluetooth Low Energy (BLE) connection between the ESP32 microcontroller and external devices

3. User Input (Braille) Module: Captures and processes user input in braille format from the physical braille pad buttons.

4. Translation Module: Converts braille input into corresponding text characters or control commands based on the current mode and user preferences

4.2 MODULE IMPLEMENTATION

1. ESP32 Processing Module: The ESP32 Processing Module forms the backbone of the Braille Pad project, orchestrating crucial tasks that enable seamless user interaction and efficient data processing. At its core, this module is responsible for reading and interpreting user input from the braille pad's physical buttons and control buttons. The `decode_braille_input` function within this module plays a pivotal role in this process by translating physical button presses into binary braille patterns, which are essential for subsequent translation into readable text or keyboard commands. Additionally, the module manages mode changes based on user preferences, allowing users to switch between input modes such as ToNumber, ToCharacter, and ToCapital effortlessly.

Furthermore, the ESP32 Processing Module integrates functionalities for providing haptic feedback through the vibration motor, enhancing the interactive experience for visually impaired users. The `send_haptic_feedback` function generates tactile confirmation signals, ensuring that users receive feedback for their actions on the braille pad. This module also handles the transmission of keyboard outputs over Bluetooth Low Energy (BLE), enabling wireless connectivity with external devices like computers or smartphones. Overall, the ESP32 Processing Module serves as the brains behind the braille pad's functionality, combining input processing, mode management, haptic feedback, and wireless communication to deliver a user-friendly and accessible device.

2. Bluetooth Connection Module: The Bluetooth Connection Module is crucial for establishing and maintaining reliable communication between the Braille Pad and external devices over Bluetooth Low Energy (BLE). Its primary objective is to ensure seamless wireless connectivity, allowing users to transmit converted text inputs from the braille pad to connected devices such as computers or smartphones. This module leverages the `BleKeyboard` library to initialize and manage the BLE keyboard service, providing a standardized interface for transmitting keyboard inputs wirelessly. Functions like `bleKeyboard.isConnected()` enable the module to monitor Bluetooth connectivity status, ensuring that the braille pad remains connected to the desired device.

Moreover, the Bluetooth Connection Module includes error-handling mechanisms, such as the `disconnectBluetooth` function, which manages disconnection scenarios gracefully. In the event of a Bluetooth disconnection, this function initiates a restart of the ESP32 microcontroller to facilitate reconnection, ensuring uninterrupted usability of the braille pad. By handling Bluetooth communication complexities and maintaining a stable connection, this module enhances the overall reliability and usability of the Braille Pad, empowering visually impaired individuals with seamless wireless interaction capabilities.

3. User Input (Braille) Module: The User Input Module serves as the interface between the physical interaction on the braille pad and the digital processing capabilities of the ESP32 microcontroller. Its primary function is to process user input from the physical braille pad buttons and translate these inputs into digital binary braille patterns. The `decode_braille_input` function within this module reads the digital states of the braille input buttons (`BRAILLE_INPUT_1` to `BRAILLE_INPUT_6`) and constructs binary strings representing the braille characters entered by the user. This binary representation is fundamental for subsequent translation into readable text or keyboard commands, ensuring accurate interpretation of user input.

Additionally, the User Input Module incorporates debounce mechanisms and input validation to enhance input accuracy and reliability. By filtering out spurious signals and ensuring consistent input processing, this module contributes to the overall usability and effectiveness of the braille pad for visually impaired users. Its efficient handling of braille input forms the foundation for the Translation Module to interpret and convert user input into actionable outputs, ultimately enabling seamless interaction with digital devices via the braille pad.

4. Translation Module: The Translation Module is a critical component of the Braille Pad project, responsible for converting binary braille patterns generated by the User Input Module into corresponding textual characters or commands. This module acts as a bridge between the raw braille input and meaningful output, aligning user input with the selected input mode (ToNumber, ToCharacter, ToCapital). One of its primary functions is to map binary braille patterns to numerical digits, lowercase and uppercase letters, as well as special keyboard commands, as defined in the project's mappings. For example, when in ToNumber mode, the module translates specific binary patterns into numerical digits (1-9, 0), facilitating numerical input for users.

The Translation Module integrates logic and lookup tables, such as `numbers_map` and `alphabets_map`, to perform accurate conversions based on the current input mode. For instance, in ToCharacter mode, binary patterns representing braille characters are translated into corresponding lowercase letters (a-z), enabling users to input text seamlessly. Similarly, in ToCapital mode, the module converts braille patterns into uppercase letters (A-Z), ensuring compatibility with various text input scenarios. This flexibility in translation allows visually impaired users to interact effectively with digital devices through the braille pad, enhancing accessibility and usability.

Furthermore, the Translation Module includes error-checking mechanisms to handle invalid or unrecognized input gracefully. It ensures that only valid braille patterns corresponding to mapped characters or commands are processed, maintaining accuracy and reliability in the translation process. By efficiently translating braille input into actionable outputs, the module contributes significantly to the functionality and user experience of the Braille Pad, empowering visually impaired individuals to communicate and navigate digital platforms with ease.

5. SYSTEM DESIGN

5.1 INTRODUCTION TO SYSTEM DESIGN

The system design of the Braille Pad for the Visually Impaired involves the integration of hardware components, software modules, and communication protocols to create a functional and accessible device. The primary goal of the system design is to enable visually impaired users to input braille characters seamlessly, convert them into readable text, and transmit the text wirelessly to external devices using Bluetooth Low Energy (BLE) technology. Below is an overview of the key aspects of the system design:

Hardware Components

The hardware components of the Braille Pad include:

- 1. ESP32 Microcontroller:** The central processing unit responsible for reading user input from the braille pad buttons, processing braille patterns, managing modes, providing haptic feedback, and transmitting keyboard inputs over BLE.
- 2. Braille Input Buttons:** Consisting of a 3x2 grid of push buttons representing braille dots, used for inputting braille characters.
- 3. Control Buttons:** Additional buttons for mode switching (ToNumber, ToCharacter, ToCapital) and special keyboard functions.
- 4. Vibration Motor:** Provides haptic feedback to the user, enhancing the interactive experience.
- 5. Jumper Wires and USB Cable:** Connect the components and provide power and programming interface to the ESP32.

Software Modules

The software architecture of the Braille Pad comprises several key modules:

- 1. ESP32 Processing Module:** Responsible for handling user input, mode switching, haptic feedback control, and BLE keyboard service implementation. Includes functions for decoding braille input, managing input modes, and transmitting keyboard outputs.

2. Bluetooth Connection Module: Manages Bluetooth Low Energy connectivity, initializes and manages the BLE keyboard service using the BleKeyboard library, monitors connection status, and handles disconnection scenarios.

3. User Input (Braille) Module: Interprets physical button presses from the braille pad, translates them into binary braille patterns, validates the input, and interfaces with the Translation Module.

4. Translation Module: Converts binary braille patterns into readable text or keyboard commands based on the selected input mode (ToNumber, ToCharacter, ToCapital). Includes mappings for numbers, lowercase and uppercase letters, and special characters.

Communication Protocols

The Braille Pad utilizes Bluetooth Low Energy (BLE) as the primary communication protocol for wireless connectivity with external devices. The ESP32 microcontroller implements the BLE Keyboard service, allowing the converted text from braille input to be sent as keyboard inputs to connected devices such as computers or smartphones. This enables users to input text using the braille pad and interact with digital platforms seamlessly.

System Workflow

1. User Interaction: The visually impaired user interacts with the Braille Pad by pressing buttons on the 3x2 grid to input braille characters. Control buttons are used for mode switching and special functions.

2. Input Processing: The ESP32 Processing Module reads the digital states of the braille input buttons and control buttons, decodes braille input into binary patterns, manages input modes, and provides haptic feedback.

3. Translation and Conversion: The Translation Module translates binary braille patterns into corresponding text characters or keyboard commands based on the selected mode (ToNumber, ToCharacter, ToCapital).

4. Bluetooth Connectivity: The Bluetooth Connection Module establishes and manages BLE communication, allowing the converted text to be transmitted wirelessly to connected devices as keyboard inputs.

5. External Device Interaction: The converted text from braille input appears as keyboard inputs on the connected external device, enabling the visually impaired user to input text and interact with digital platforms effectively.

System Design Benefits

Accessibility: The system design caters specifically to the needs of visually impaired individuals, providing an accessible and user-friendly interface for text input and device interaction.

Wireless Connectivity: Utilizing Bluetooth Low Energy enables wireless connectivity with external devices, enhancing mobility and convenience for users.

Customization: The modular software design allows for the customization of braille mappings, input modes, and functionality, catering to individual user preferences and requirements.

Reliability: Error-checking mechanisms, mode validation, and Bluetooth connection management ensure reliable and uninterrupted operation of the Braille Pad system.

5.2 PROJECT FLOW:

The project flow of the Braille Pad involves several interconnected processes that ensure efficient operation and usability for visually impaired users. The flow starts with the user inputting braille characters through the 3x2 grid of braille input buttons. These button presses are decoded by the ESP32 microcontroller, which processes the binary braille patterns based on the current input mode (ToNumber, ToCharacter, ToCapital). The Translation Module converts these patterns into numerical digits, lowercase or uppercase letters, or special commands, depending on the selected mode. Simultaneously, the system provides haptic feedback through the vibration motor to confirm user actions.

Once the braille input is processed and translated, the system establishes a Bluetooth Low Energy (BLE) connection with external devices like computers or smartphones using the BLE Keyboard service. The translated text or commands are then transmitted wirelessly as keyboard inputs to the connected device, enabling users to input text, navigate interfaces, or perform actions remotely. The system flow ensures seamless interaction between the braille pad and external devices, enhancing accessibility and enabling visually impaired individuals to engage with digital platforms effectively.

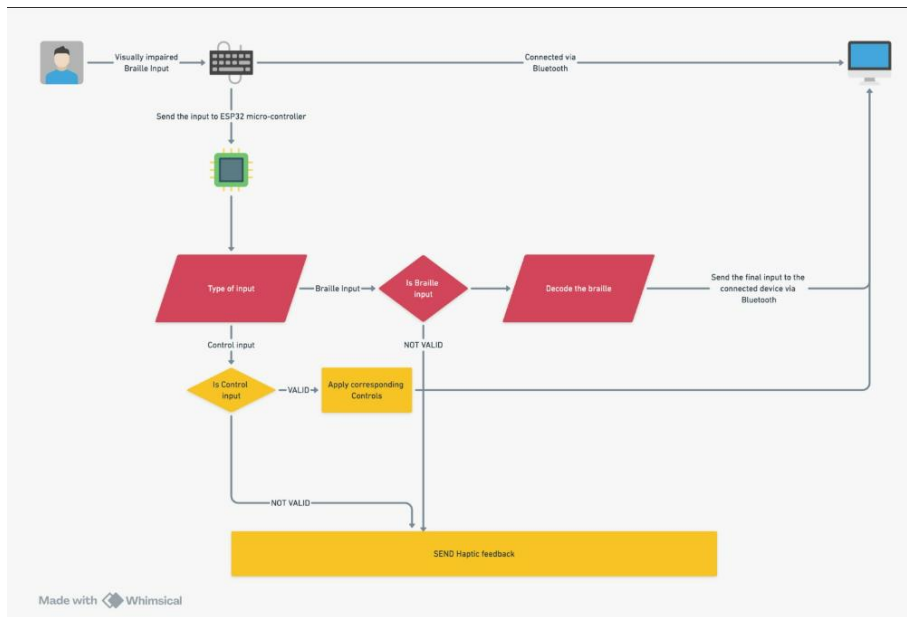


Fig 5.2.1 Project Flow

5.3 FLOW CHART DIAGRAM

The following diagram is the flowchart diagram of the project which shows the sequence of actions involved in the execution of the proposed model:

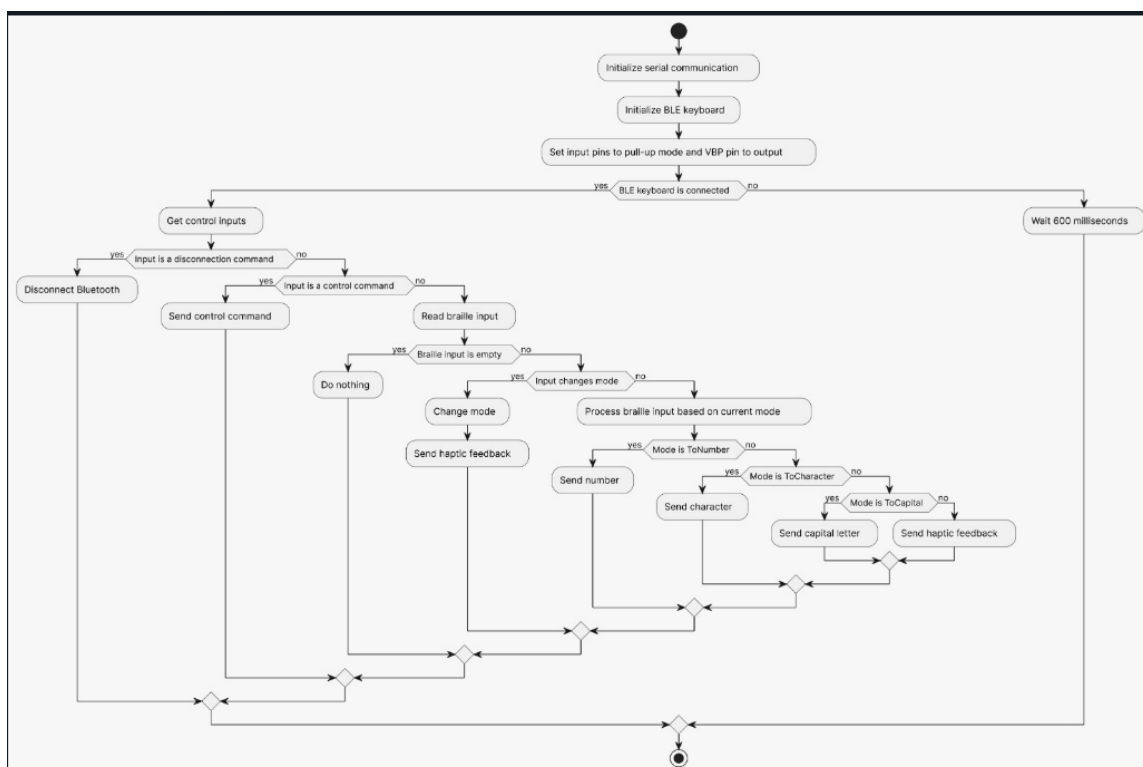


Fig 5.3.1 Execution Flow

6. SYSTEM IMPLEMENTATION

6.1 INTRODUCTION TO SYSTEM IMPLEMENTATION

To implement the Braille Pad for the Visually Impaired, follow these detailed steps for hardware setup, connection, programming, and system operation:

Hardware Setup

1. Components Required: ESP32 microcontroller board: This serves as the central processing unit for the braille pad, handling input processing, translation, and Bluetooth communication. It is a low-cost, low-power system-on-a-chip microcontroller that supports both Wi-Fi and Bluetooth.

=> 9 push buttons for braille input: These buttons form a 3x2 grid for users to input braille characters. The buttons are made of high-quality materials and are designed for long-term durability.

=> Jumper wires for connections: These are used to connect the push buttons and control buttons to the ESP32 pins. The jumper wires have a secure and reliable connection and are available in different lengths.

=> USB cable for power and programming: This connects the ESP32 to a computer for power supply and code uploading. A high-quality USB cable is recommended for a stable power supply and data transmission.

=> Optional: Vibration motor for haptic feedback: This enhances the user experience by providing tactile feedback for button presses. The vibration motor is lightweight, compact, and easy to install.

2. Connection Instructions: Connect the push buttons to the ESP32 as specified in the project documentation:

=> BRAILLE_INPUT_1 to BRAILLE_INPUT_6 connected to pins 15, 2, 0, 12, 14, 27, respectively. These pins are used to read the digital states of the push buttons and decode the braille input.

=> CONTROL_BUTTON_1 to CONTROL_BUTTON_3 connected to pins 4, 16, 26, respectively. These pins are used to switch between input modes.

=> Vibration Motor (VBP) connected to pin 22 for haptic feedback. This pin is used to control the vibration motor for providing tactile feedback.

=> Use jumper wires to make these connections, ensuring a secure and reliable connection.

Software Setup and Programming

1. Arduino IDE Setup: If you haven't already, install the Arduino IDE on your computer from the official Arduino website. The Arduino IDE is an open-source integrated development environment used for programming microcontrollers.

=> Open the Arduino IDE and navigate to File > Preferences. In the Preferences window, add the ESP32 board support URL (https://dl.espressif.com/dl/package_esp32_index.json) to the Additional Board Manager URLs. This allows the Arduino IDE to support ESP32 microcontroller boards.

=> Go to Tools > Board > Boards Manager, search for "esp32," and install the ESP32 board package. This installs the necessary drivers and tools for programming the ESP32 microcontroller board.

=> Once installed, select the appropriate ESP32 board from the Tools > Board menu. This ensures that the Arduino IDE uses the correct board settings for programming.

2. Library Installation: Download the necessary libraries provided in the project repository, such as ESP32-BLE-Keyboard.zip. These libraries are code modules that provide additional functionality to the main program.

=> Import these libraries into the Arduino IDE by going to Sketch > Include Library > Add .ZIP Library and selecting the downloaded ZIP file. This makes the libraries available for use in the main program.

3. Programming the ESP32: Open the Braille Pad project in the Arduino IDE by navigating to File > Open. This opens the main program code.

=> Connect your ESP32 board to the computer using a USB cable. This provides power and data transmission for programming.

=> Select the correct board and port from the Tools menu in the Arduino IDE. This ensures that the Arduino IDE communicates with the correct board.

=> Click the upload button (right arrow icon) to compile the code and upload it to the ESP32 board. This transfers the program code to the ESP32 microcontroller board.

System Operation and Working

1. Power On and Initialization: After programming the ESP32, power on the braille pad and ESP32 board. This initializes the hardware and prepares it for operation.

=> The ESP32 initializes and starts executing the uploaded code. This sets up the system for braille input processing.

2. Braille Input Processing: Use the 3x2 grid of push buttons to input braille characters. Each push button corresponds to one or more dots in a braille cell.

=> The ESP32 reads the digital states of these input pins (BRAILLE_INPUT_1 to BRAILLE_INPUT_6) to decode the braille input. Each button press corresponds to a binary digit (1 or 0), forming a 6-bit binary pattern representing a braille character.

=> The braille input is processed by the ESP32 microcontroller board, which translates it into the desired output format.

3. Mode Selection: Utilize the control buttons (CONTROL_BUTTON_1 to CONTROL_BUTTON_3) to switch between input modes. The three input modes are:

=> ToNumber: Converts braille input to numbers. This mode is used to input numerical data.

=> ToCharacter: Converts braille input to lowercase characters. This mode is used to input text data in lowercase letters.

=> ToCapital: Converts braille input to capital letters. This mode is used to input text data in capital letters.

4. Bluetooth Connectivity: Connect the ESP32 to a Bluetooth-enabled device (e.g., computer, smartphone) using the BLE Keyboard service. This establishes a wireless connection between the braille pad and the connected device.

=> The ESP32 functions as a BLE keyboard, sending the translated text or commands wirelessly as keyboard inputs to the connected device. This allows the user to input text or commands using the braille pad.

5. Haptic Feedback: The vibration motor (VBP) provides haptic feedback to the user, confirming button presses and mode changes. This enhances the interactive experience and provides tactile cues for user interactions.

=> The vibration motor is controlled by the ESP32 microcontroller board, which sends signals to the VBP pin (pin 22) to control the vibration. The vibration strength can be adjusted as per user preference.

6.2 SELECTED SOFTWARE AND PROGRAMMING LANGUAGE

The software and programming language selected for the Braille Pad project is as follows:

Programming Language:

C++ (Arduino-based) Language:

C++ was chosen as the programming language for the Braille Pad project due to several key advantages it offers. First and foremost, C++ strikes a good balance between performance and ease of development, making it well-suited for microcontroller programming. Its low-level control capabilities are essential for interacting with hardware components such as buttons and sensors, which are integral to the functionality of the braille pad. Additionally, C++ supports high-level abstractions, allowing developers to write efficient and structured code while leveraging the power of object-oriented programming paradigms. Furthermore, since Arduino libraries and frameworks are primarily based on C++, using C++ ensures compatibility and seamless integration with the Arduino ecosystem, including libraries for Bluetooth communication, input/output operations, and more.

Software Environment:

Integrated Development Environment (IDE): Arduino IDE: The Arduino Integrated Development Environment (IDE) was selected as the software environment for developing the Braille Pad project due to its user-friendly nature and strong integration with the ESP32 microcontroller. Arduino IDE provides a straightforward development environment that is accessible to both beginners and experienced developers, thanks to its intuitive interface and simplified workflow. Its built-in editor, compiler, and uploader streamline the development process, allowing developers to write, compile, and upload code to the ESP32 board with ease. Additionally, Arduino IDE offers a rich collection of libraries and examples specifically designed for hardware interfacing and IoT projects, which significantly accelerates development time and

simplifies tasks such as Bluetooth communication, input/output handling, and sensor integration. Overall, Arduino IDE's robust features and compatibility with ESP32 make it an ideal choice for developing the Braille Pad with efficiency and effectiveness.

BleKeyboard Library:

The BleKeyboard library is specifically designed for ESP32-based projects that require Bluetooth keyboard functionality. It simplifies the process of emulating a keyboard using BLE, making it an essential component for the Braille Pad project. The library handles the Bluetooth communication protocols, allowing the ESP32 to act as a BLE keyboard seamlessly.

6.3 SAMPLE CODE

```
#include <BleKeyboard.h>
#include <map>
#include <ESP.h>

BleKeyboard bleKeyboard;

// Define input pins for Braille inputs
#define BRAILLE_INPUT_1 15
#define BRAILLE_INPUT_2 2
#define BRAILLE_INPUT_3 0
#define BRAILLE_INPUT_4 12
#define BRAILLE_INPUT_5 14
#define BRAILLE_INPUT_6 27

// Define control button input pins
#define CONTROL_BUTTON_1 4
#define CONTROL_BUTTON_2 16
#define CONTROL_BUTTON_3 26

// Vibration motor pin
#define VBP 22

// Enum to represent different modes
```

```

enum class Mode
{
    ToNumber,
    ToCharacter,
    ToCapital
};

// Initialize current mode to ToCharacter
Mode current_mode = Mode::ToCharacter;

// Map to store the conversion between braille patterns and modes
std::map<std::string, Mode> modes_map = {
    {"001111", Mode::ToNumber},
    {"000011", Mode::ToCharacter},
    {"000001", Mode::ToCapital},
};

// Control Inputs map
std::map<std::string, int> controls_map = {
    {"100", KEY_TAB},
    {"010", 0x20},
    {"001", KEY_BACKSPACE},
};

// Map to store the conversion between braille patterns and numbers
std::map<std::string, std::string> numbers_map = {
    {"100000", "1"},
    {"110000", "2"},
    {"100100", "3"},
    {"100110", "4"},
    {"100010", "5"},
    {"110100", "6"},
    {"110110", "7"},
    {"110010", "8"},
};

```

```

    {"010100", "9"},
    {"010110", "0"},
};

// Map to store the conversion between braille patterns and alphabets
std::map<std::string, std::string> alphabets_map = {
    {"100000", "a"},
    {"110000", "b"},
    {"100100", "c"},
    {"100110", "d"},
    {"100010", "e"},
    {"110100", "f"},
    {"110110", "g"},
    {"110010", "h"},
    {"010100", "i"},
    {"010110", "j"},
    {"101000", "k"},
    {"111000", "l"},
    {"101100", "m"},
    {"101110", "n"},
    {"101010", "o"},
    {"111100", "p"},
    {"111110", "q"},
    {"111010", "r"},
    {"011100", "s"},
    {"011110", "t"},
    {"101001", "u"},
    {"111001", "v"},
    {"010111", "w"},
    {"101101", "x"},
    {"101111", "y"},
    {"101011", "z"},
};

```

```

/**
 * @brief Set the input pins to pull-up mode and the vibration motor pin to output.
 *
 * This function sets the input pins BRAILLE_INPUT_1 to BRAILLE_INPUT_6 and
 * CONTROL_BUTTON_1 to CONTROL_BUTTON_2
 * to INPUT_PULLUP mode, and sets the VBP pin to OUTPUT mode for controlling the
 * vibration motor.
 *
 * @return void
 */
void set_input_pins_pullup()
{
    pinMode(BRAILLE_INPUT_1, INPUT_PULLUP);
    pinMode(BRAILLE_INPUT_2, INPUT_PULLUP);
    pinMode(BRAILLE_INPUT_3, INPUT_PULLUP);
    pinMode(BRAILLE_INPUT_4, INPUT_PULLUP);
    pinMode(BRAILLE_INPUT_5, INPUT_PULLUP);
    pinMode(BRAILLE_INPUT_6, INPUT_PULLUP);
    pinMode(CONTROL_BUTTON_1, INPUT_PULLUP);
    pinMode(CONTROL_BUTTON_2, INPUT_PULLUP);
    pinMode(CONTROL_BUTTON_3, INPUT_PULLUP);
    pinMode(VBP, OUTPUT);
}

/**
 * @brief Send haptic feedback by toggling the vibration motor pin.
 *
 * This function toggles the VBP pin HIGH for a brief duration to simulate haptic feedback,
 * then brings it LOW again.
 *
 * @return void
 */
void send_haptic_feedback()
{

```



```

Serial.println("Sending Haptic Feedback");
digitalWrite(VBP, HIGH);
delay(200);
digitalWrite(VBP, LOW);
}

/**
 * @brief Decode the braille input by reading the digital states of input pins.
 *
 * This function reads the digital states of the input pins BRAILLE_INPUT_1 to
BRAILLE_INPUT_6
 * and constructs a binary string representing the braille input.
 *
 * @return std::string - The braille input as a binary string.
 */
std::string decode_braille_input()
{
    std::string braille_input = "";
    braille_input += digitalRead(BRAILLE_INPUT_1) == LOW ? "1" : "0";
    braille_input += digitalRead(BRAILLE_INPUT_2) == LOW ? "1" : "0";
    braille_input += digitalRead(BRAILLE_INPUT_3) == LOW ? "1" : "0";
    braille_input += digitalRead(BRAILLE_INPUT_4) == LOW ? "1" : "0";
    braille_input += digitalRead(BRAILLE_INPUT_5) == LOW ? "1" : "0";
    braille_input += digitalRead(BRAILLE_INPUT_6) == LOW ? "1" : "0";
    return braille_input;
}

/**
 * @brief Get the control inputs by reading the digital states of control button pins.
 *
 * This function reads the digital states of the control button pins CONTROL_BUTTON_1 to
CONTROL_BUTTON_3
 * and constructs a binary string representing the control input.
 *

```

```

* @return std::string - The control input as a binary string.
*/
std::string get_controls_input()
{
    std::string controls_input = "";
    controls_input += digitalRead(CONTROL_BUTTON_1) == LOW ? "1" : "0";
    controls_input += digitalRead(CONTROL_BUTTON_2) == LOW ? "1" : "0";
    controls_input += digitalRead(CONTROL_BUTTON_3) == LOW ? "1" : "0";
    return controls_input;
}

/**
* @brief Change the current mode based on the input braille pattern.
*
* This function checks the input braille pattern and changes the current mode accordingly.
* If the pattern corresponds to a mode switch, it toggles between ToCharacter and ToCapital
modes.
*
* @param braille_input - The input braille pattern to analyze for mode change.
* @return void
*/
void change_mode(std::string braille_input)
{
    if (braille_input != "000000" && modes_map.find(braille_input) != modes_map.end())
    {
        Serial.println("Changing mode");
        // Toggle the mode between ToCharacter and ToCapital
        if ((current_mode == Mode::ToCharacter || current_mode == Mode::ToCapital) &&
(modes_map[braille_input] == Mode::ToCharacter || modes_map[braille_input] ==
Mode::ToCapital))
        {
            current_mode = current_mode == Mode::ToCharacter ? Mode::ToCapital :
Mode::ToCharacter;
        }
    }
}

```

```

    else
    {
        current_mode = modes_map[braille_input];
    }
}

/**
 * @brief Send braille input as keyboard output based on the current mode.
 *
 * This function processes the braille input based on the current mode and sends the
 * corresponding
 * keyboard output using the BLE keyboard.
 *
 * @return void
 */
void process_braille_input()
{
    // Check if the input is a control key
    std::string controls_input = get_controls_input();

    // Check if it is a disconnection input.
    if (controls_input == "111")
    {
        disconnectBluetooth();
        return;
    }

    // Check for other control combinations
    if (controls_map.find(controls_input) != controls_map.end())
    {
        bleKeyboard.write(static_cast<uint8_t>(controls_map[controls_input]));
        Serial.println(controls_input.c_str());
        return;
    }
}

```

```

}

std::string braille_input = decode_braille_input();
if (braille_input == "000000")
{
    return;
}

// Log the braille input and current mode
Serial.println("Braille Input");
Serial.println(braille_input.c_str());

// Check if the input is a mode change
if (modes_map.find(braille_input) != modes_map.end())
{
    change_mode(braille_input);
    send_haptic_feedback();
    return;
}

// Switch statement based on the current mode
switch (current_mode)
{
case Mode::ToNumber:
    /**
     * @brief Send braille input as numbers.
     */
    if (numbers_map.find(braille_input) != numbers_map.end())
    {
        const char *number_str = numbers_map[braille_input].c_str();
        size_t len = strlen(number_str);
        for (size_t i = 0; i < len; i++)
        {
            bleKeyboard.write(static_cast<uint8_t>(number_str[i]));

```

```

    }
    return;
}
send_haptic_feedback();
break;

case Mode::ToCharacter:
/**
 * @brief Send braille input as characters.
 */
if (alphabets_map.find(braille_input) != alphabets_map.end())
{
    const char *char_str = alphabets_map[braille_input].c_str();
    size_t len = strlen(char_str);
    for (size_t i = 0; i < len; i++)
    {
        bleKeyboard.write(static_cast<uint8_t>(char_str[i]));
    }
    return;
}
send_haptic_feedback();
break;

case Mode::ToCapital:
/**
 * @brief Send braille input as capital characters.
 */
if (alphabets_map.find(braille_input) != alphabets_map.end())
{
    const char *capital_str = alphabets_map[braille_input].c_str();
    size_t len = strlen(capital_str);
    for (size_t i = 0; i < len; i++)
    {
        bleKeyboard.write(static_cast<uint8_t>(toupper(capital_str[i])));
    }
}

```

```

    }
    return;
}
send_haptic_feedback();
break;

default:
    send_haptic_feedback();
    break;
}
}

/**
 * @brief Disconnect Bluetooth if connected.
 *
 * This function checks if the BLE keyboard is connected and disconnects it if so.
 *
 * @return void
 */
void disconnectBluetooth()
{
    if (bleKeyboard.isConnected())
    {
        // Delay before restarting to ensure proper disconnection
        delay(600);
        // Restart the ESP32 to disconnect Bluetooth
        ESP.restart();
    }
}

/**
 * @brief Setup function to initialize serial communication, pins, and BLE keyboard.
 *
 * This function is called once during startup to initialize serial communication,

```

```

* set input pins to pull-up mode, and start the BLE keyboard service.
*
* @return void
*/
void setup()
{
  // Initialize serial communication
  Serial.begin(115200);
  Serial.println("Starting BLE work!");
  // Set input pins to pull-up mode and VBP pin to output
  set_input_pins_pullup();
  // Start the BLE keyboard service
  bleKeyboard.begin();
}

/**
* @brief Loop function to check for Bluetooth connection and send braille input.
*
* This function is called repeatedly in a loop. It checks if the BLE keyboard is connected,
* and if so, it sends the braille input based on the current mode.
* If the keyboard is not connected, it waits for 600 milliseconds before checking again.
*
* @return void
*/
void loop()
{
  if (bleKeyboard.isConnected())
  {
    Serial.println("Bluetooth connected");
    // Process braille input if Bluetooth is connected
    process_braille_input();
  }
  else
  {

```

```
Serial.println("Bluetooth Not connected");  
Serial.println("Waiting 600ms...");  
}  
  
// Wait for 600 milliseconds before checking again  
delay(600);  
}
```


7. TESTING

7.1 INTRODUCTION

UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

FUNCTIONAL TEST

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In

addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Overall, testing plays a critical role in ensuring the quality, reliability, and performance of software products, helping to deliver a better user experience and achieve business objectives.

7.2 TEST CASES

Test case id	Description	Steps	Test Data	Expected Result	Actual Result	Test Status
#BK001	Verifying Bluetooth connection among various devices.	-Connect to a windows computer. -Disconnect from windows computer and connect to android. -Disconnect from android and connect to MacBook.	NA	Should connect to different devices without any connection issues.	Established connection for different devices without any issues.	Pass




#BK002	Input Alphabets in braille.	-Connect to a device using Bluetooth. -Input braille.	 Input: ”100000”	A	A	Pass
#BK003	Input Number in braille.	-Connect to a device using Bluetooth. -Input braille.	 Input: ”001111 100010”	5	5	Pass
#BK004	Input Symbol in braille.	-Connect to a device using Bluetooth. -Input braille.	 Input: ”010000”	,	,	Pass

Fig 7.2.1 Test Cases Table

8. PICTURES AND RESULTS

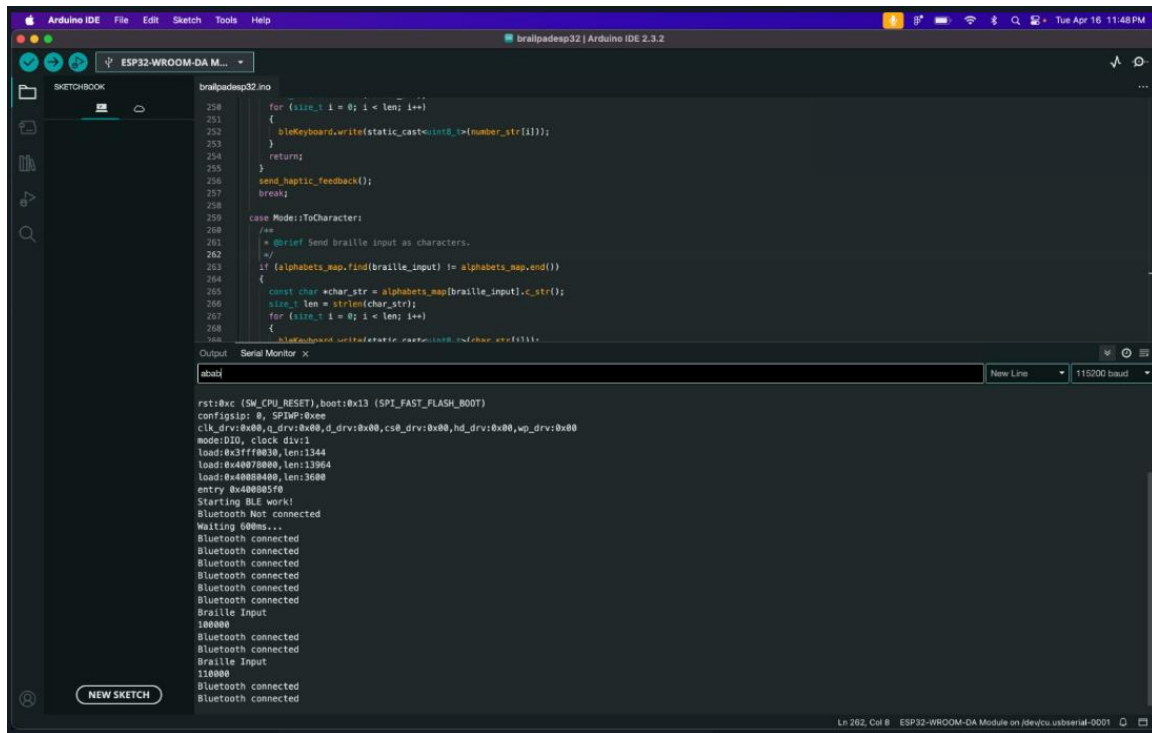


Fig 8.1 Execution Logs

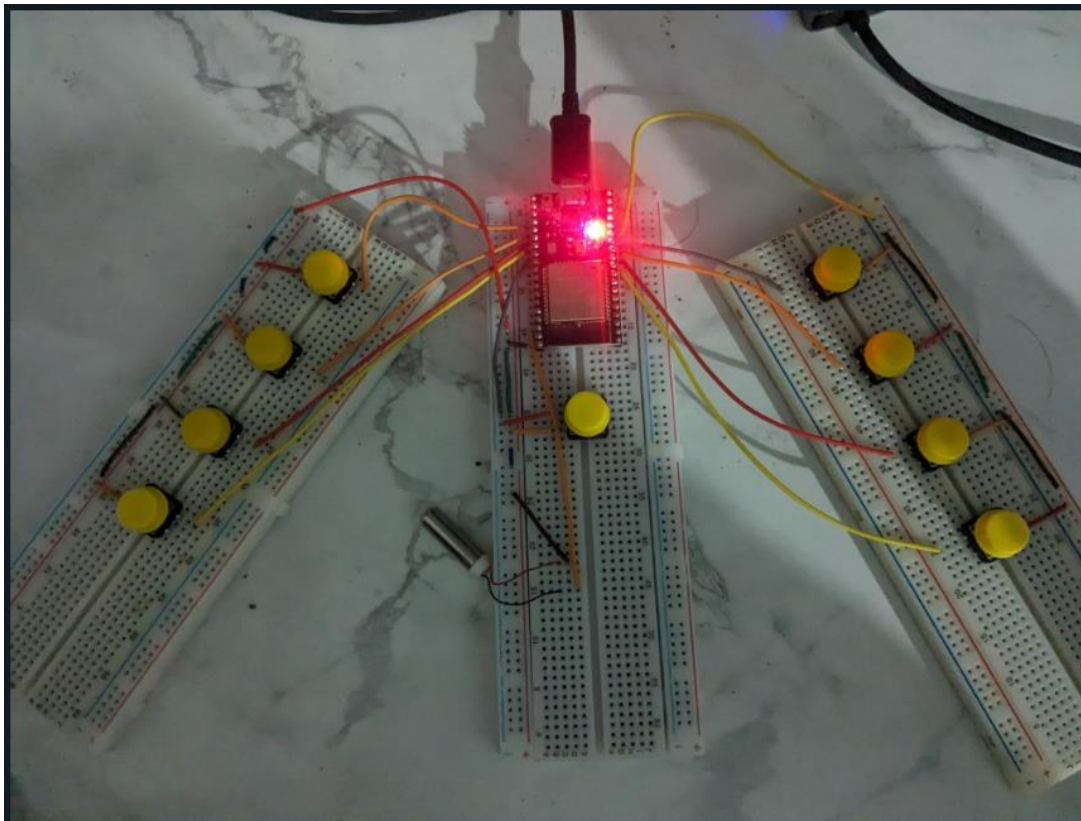


Fig 8.2 Connections

9. CONCLUSION AND FUTURE SCOPE

9.1 CONCLUSION

The new Braille keyboard is a remarkable advancement in accessibility technology, specifically designed to cater to the needs of visually impaired individuals. It is a cost-effective and versatile keyboard that works seamlessly with different devices like smartphones, tablets, laptops, and desktop computers. The keyboard is designed to be user-friendly and accessible, with tactile feedback keys that provide a comfortable typing experience. Its compact layout makes it easy to carry around, making it an excellent option for individuals on the go.

Moreover, the Braille keyboard can be used across different platforms, ensuring that users can integrate it into their daily activities with ease. The keyboard's algorithm, which is embedded in the ESP32 microcontroller, accurately translates Braille characters into English text, making it easier for users to communicate with others.

One of the most amazing features of the Braille keyboard is its vibration feedback mechanism, which helps users detect and correct errors. The keyboard vibrates when an incorrect key is pressed, allowing users to identify and correct errors quickly. This makes the keyboard an excellent option for individuals who are learning Braille or those who are not yet proficient in it.

Overall, the Braille keyboard is an incredible technological advancement that empowers visually impaired individuals to communicate and interact with digital devices more effectively. It promotes inclusivity and accessibility in the digital age, ensuring that everyone can enjoy the benefits of technology.

9.2 FUTURE SCOPE

As technology continues to advance, we are constantly looking for ways to make digital devices more accessible to everyone. One group that has faced challenges when it comes to using such technology is the visually impaired community. However, with the implementation of smart braille technology, we can open up a whole new world of possibilities for these individuals.

One way we can improve the experience for visually impaired users is by enhancing the features of the braille keyboard. In addition to the standard braille input, we can incorporate

additional features such as haptic feedback or voice recognition. These features will not only make the keyboard more user-friendly but will also make it accessible to a wider range of users with different levels of vision impairment.

Moreover, we can partner with assistive technology developers to seamlessly integrate the braille keyboard with screen readers, magnifiers, and other assistive tools. This will ensure that visually impaired users have access to a complete suite of assistive technologies that work together seamlessly, providing them with a more comprehensive and accessible experience.

Finally, we can expand language support and localization options to cater to diverse user populations worldwide. This will make sure that the device meets the needs of users in different regions and environments. By improving the accessibility and usability of digital devices for visually impaired individuals, we can help them to live more productive and fulfilling lives.

10.BIBLIOGRAPHY

10.1 REFERENCES

1. ZAYNAB ZEINEDDINE, SAMAR SINDIAN, NOUR AL HODA AHMAD, GEORGE ISMAIL "Low Cost Electronic Brailier", 2020
2. K V Shalini, Keerthi B, Manasa B, Padmashree, Ashritha KG "Design and Implementation of Digital Braille System for The Visually Impaired", 2020
3. Mukhriddin Arabboev, Shohruh Begmatov, Ziyokhon Rakhimov, Khushnud Gaziev, Khabibullo Nosirov, "Design of an External USB Braille Keyboard for Computers", 2022
4. Kazi Toukir Ahmed, Md Zesun Ahmed Mia "Ultra Low Cost, Low Power, High Speed Electronic Braille Device for Visually Impaired People", 2024
5. T. Ribeiro AlvesC. Pinto PereiraT. Cerqueira de Jesus "LITERA BRAILLE: PROTOTYPING AND DEVELOPMENT OF LOW-COST DEVICE BASED ON BRAILLE TYPEWRITER", 2023
6. Sana Shokat, Rabia Riaz, Sanam Shahla Rizvi, Khalil Khan, Farina Riaz, and Se Jin "Analysis and Evaluation of Braille to Text Conversion Methods", 2020
7. Shohruh Begmatov, Mukhriddin Arabboev, Sardor Vakhkhobov, ^[1]_{SEP}"Design and development of a folding type braille keyboard", 2023
8. John Lloyd Suarez, Danielle Joy Pueblo, Jamaica Pablita Padida, Justine Quilantang
A Cost-Effective 6-Key Wooden Braille Keyboard for Visually Impaired Individuals", 2023
9. Joy Protim; Abu Talha Md. Abdullah; Ahmed Iqbal Pritom; Mehrab Zaman Chowdhury
"BADHON: A High Performing Keyboard Layout for Physically Impaired People", 2019
10. Nikhil Khare, Om Sharan, Manikandan J "Design and Development of a Digital Scribe for Visually Challenged Students", 2020

10.2 WEB LINKS

- <https://www.w3schools.com/cpp/>
- <https://www.arduino.cc/en/software/>
- <https://randomnerdtutorials.com/getting-started-with-esp32/>
- <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/get-started/index.html>
- <https://ieeexplore.ieee.org/abstract/document/9633940>
- https://www.researchgate.net/profile/Mukhriddin-Arabboev/publication/374234840_Design_and_development_of_a_folding_type_braille_keyboard/links/65156718f91aee386e744f72/Design-and-development-of-a-folding-type-braille-keyboard.pdf
- <https://www.hindawi.com/journals/misy/2020/3461651/>
- <https://library.iated.org/view/RIBEIROALVES2023LIT>
- https://www.researchgate.net/profile/Md-Zesun-Ahmed-Mia-2/publication/378857244_Ultra_Low_Cost_Low_Power_High_Speed_Electronic_Braille_Device_for_Visually_Impaired_People/links/65edd4019ab2af0ef8ac847f/Ultra-Low-Cost-Low-Power-High-Speed-Electronic-Braille-Device-for-Visually-Impaired-People.pdf