

# **Multi-lingual Speech to Braille Script Converter using IoT and Arduino UNO**

*A Project Report Submitted for the Project-1*

*Of*

**Bachelor of Technology (Hons.)**

*By*

**Akshita Shrivastava (2021UGEC114)**

**Ayush Kumar Lal (2021UGEC090)**

**Khushal Kumar Rajak (2021UGEC080)**

**Gipashu Aryan (2021UGEC079)**

**G Likhith Reddy (2021UGEC078)**

**Pragati Dubey (2021UGEC008)**

Under the Supervision

*Of*

**B.N.S. Munda**

Associate Professor

Electronics and Communication Engineering



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION  
ENGINEERING**

National Institute of Technology, Jamshedpur-831014  
(April 2023)

# CANDIDATES' DECLARATION

We hereby declare that the project report entitled “Multi-lingual Speech to Braille Script Converter using IoT and Arduino UNO” is our work conducted under the supervision of Associate Prof. B.N.S. Munda, National Institute of Technology, Jamshedpur. We further declare that to the best of our knowledge. This report does not contain any part of work that has been submitted for the award of any degree either in this university or in another university / Deemed University without proper citation.

Student's Signature:

- |       |                                   |
|-------|-----------------------------------|
| (i)   | Akshita Shrivastava (2021UGEC114) |
| (ii)  | Ayush Kumar Lal (2021UGEC090)     |
| (iii) | Khushal Kr. Rajak (2021UGEC080)   |
| (iv)  | Gipashu Aryan (2021UGEC079)       |
| (v)   | G. Likhith Reddy (2021UGEC078)    |
| (vi)  | Pragati Dubey (2021UGEC008)       |

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date: \_\_\_\_\_

Supervisor: Prof. B.N.S. Munda

Designation: Associate Professor, Dept. of Electronics and Communication, National Institute of Technology, Jamshedpur.

# TABLE OF CONTENTS

1. <i>Candidates' Declaration</i>	ii
2. <i>Table of contents</i>	iii
<b>3. ABSTRACT</b>	1
<b>4. INTRODUCTION</b>	2
<b>5. MOTIVATION</b>	2
<b>6. OBJECTIVES</b>	3
<b>7. LITERATURE STUDY</b>	4
<b>8. TOOLS REQUIRED</b>	5
<b>9. WORK DONE SO FAR</b>	6
<b>10. RESULTS &amp; DISCUSSION</b>	7
<b>11. CONCLUSION &amp; FUTURE WORK</b>	8
<b>12. REFERENCES</b>	9

# ABSTRACT

The Braille system is being utilized for visually impaired individuals everywhere throughout the world. For a visually impaired person, this is the best way to get instructed. This project has been made thinking about the visually impaired individuals around us and to make their life simpler and more comfortable in the present education scenario and provide them with a helping hand. This is a push to carry a change to the entire Braille system. The device made is easy and effortless to use.

The main objective of the device is to take voice or text input and translate the input paragraph/statement into the text format of the Braille script in real-time. The device is proposed to be multilingual, meaning voice or text input can be in various languages, starting from English, the work can be extended to Hindi and other regional languages.

The IoT device proposed is expected to be cost-efficient and can be an excellent method for correspondence for visually impaired individuals in their everyday life. Cost-effectiveness and Reusability are the two major pillars of the project. Cost-effectiveness in comparison to the presently used printed Braille script, and Reusability as the device made can be used for various outputs repeatedly. Printed Braille is expensive and inaccessible to a large number of differently-abled people, in addition to that once a Braille script is printed it can't be used again to display other content, unlike our device.

In addition to providing a multilingual environment of the system in which the instructor can opt for text or voice input, the braille script generated can be readily tracked from an LCD screen by the instructor in English.

# INTRODUCTION

Visual impairment poses significant challenges for individuals in accessing information and communicating effectively. Braille, a tactile writing system, has been a crucial tool for individuals with visual impairments to read and write. However, there is a growing need for innovative and cost-effective solutions that leverage modern technology to enhance Braille accessibility.

According to the World Health Organization, an estimated 43 million people are living with severe blindness and 295 million people with moderate-to-severe blindness in the world, nearly 0.2% of the world's population – 16 million people are living with severe deafblindness and around 2% of the world population – 160 million suffer from acute deafblindness. In India, there are about 1 million cases of deafblindness, and about 60 million people are visually impaired including 8 million people suffering from severe blindness.

The growth in literacy among blind or visually impaired people has given them the ability to explore their issues and skills in this quick-paced world. Three categories—display, control, and content—can be used to characterise a learner's needs and preferences. While cutting-edge technology and the creation of diversified content for the blind have long been areas of study attention, innovative ideas and solutions also help increase the accessibility of blind individuals to higher education.

Blind or visually impaired people primarily rely on sounds and touch to communicate and navigate their environment, along with braille. A reading and writing system called Braille was developed and is now widely used to describe letters, numbers, and symbols as dots on embossed paper or other materials. Braille is utilised by blind persons as a knowledge access tool.

A Braille cell, depicted in Figure 1, is a specified area where the Braille characters are displayed. A 3x2 matrix-like arrangement of six dots arranged in two columns of three dots each makes up a Braille cell.

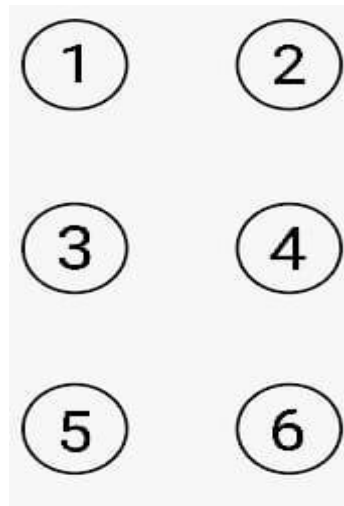


Figure 1: The Braille Cell  
Courtesy: Self-made image

The basic Braille system combines six raised dabs to create a possible combination of 64 different signs. Figure 2 depicts the Braille script used by visually impaired people.




















































































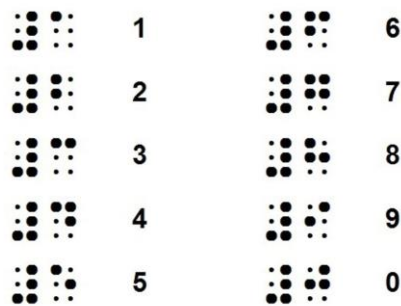
BRAILLE ALPHABET																																		
Alphabet												Punctuation																						
																																		
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	.	,	:	;	'	?	!	()	"	"	
																		Signs																
Y	Z	Capital Follows	Number Follows	And	For	Of	The	With	ch	gh	sh	th	wh	ed	er	ou	ow	Sounds																
Numbers																																		
																																		
0	1	2	3	4	5	6	7	8	9																									

Figure 2: The Braille Alphabet  
Courtesy: The Scottish Sun



*Figure 3: The Braille Numbers*  
*Courtesy: MWT Global*

The present Braille teaching procedure is a significant obstacle to Braille learners as it requires constant human effort. A few research projects have significant potential in light of the debate above, but they lack key essential qualities and capabilities that an e-book reader should have, such as:

- Transportable
- Reduced complexity
- Architecture in several languages
- Control of reading speed

So, there is a lot of room for the creation of a portable, affordable solution that also improves ergonomics and functionality. A refreshable Braille reader is a device that creates Braille dot patterns by electronically/mechanically raising or lowering pins to display information.

The reasons for the use of Arduino UNO include:

1. MCU architectures offer good flexibility as they can be interfaced with a lot of sensors easily.
2. The cost is fairly low for this Arduino model as compared to other microprocessors.

# MOTIVATION

The motivation behind undertaking this project stems from the pressing need to improve accessibility and communication for individuals with visual impairments. Visual impairment can limit an individual's ability to access information and communicate effectively, which can pose significant challenges in their daily lives. Braille, a tactile writing system, has long been recognized as a vital tool for individuals with visual impairments to read and write. However, there is a need for innovative and cost-effective solutions that leverage modern technology to enhance Braille accessibility and empower individuals with visual impairments.

- **Accessibility**: Braille is an essential tool for people with visual impairments, and providing a device that can convert text to Braille script in real-time can increase accessibility and inclusivity for this community.
- **Cost-effective**: Current Braille displays and books are expensive, and not all individuals with visual impairments can afford them. By using an Arduino-based device, the cost of creating a Braille display can be significantly reduced, making it more accessible for those who cannot afford expensive books.
- **Multilingual support**: By creating a multilingual speech/text-to-Braille converter, individuals who speak different languages can benefit from this technology, making it more inclusive and accessible to a wider range of people.
- **Educational tool**: The creation of a multilingual speech/text to Braille converter using IoT and Arduino can also serve as an educational tool for students.



- **Reusability**: This device has the potential for reusability. and can be used for different applications, and the same device can be used again and again for different text/speech input.
- **Scalability**: The device can be replicated on a larger scale and deployed in different locations, making it easier for people with visual impairments to access Braille script. This scalability can also benefit organizations and institutions that serve people with visual impairments.

The potential impact of this project is significant. It has the potential to improve the quality of life for individuals with visual impairments by enabling them to access information and communicate more effectively. The proposed system could be used in various applications, including reading books, newspapers, or documents, engaging in written communication through emails or messages, and accessing online content. Moreover, the project aligns with the principles of inclusivity, empowerment, and accessibility, promoting the rights and opportunities of individuals with visual impairments. The motivation behind this project lies in contributing to the field of assistive technology and making a meaningful difference in the lives of individuals with visual impairments by developing a speech-to-Braille conversion system using Arduino Uno.

# OBJECTIVES

1. The Phase-1 objective of this project is to develop an efficient model for converting text in English to Braille script.
2. Phase-2 includes upgrading the previous work and adding speech support to the initial model. Bluetooth module will be used for this purpose.
3. The first and second phases of work will be integrated to form a fully functional speech-to-Braille converter.
4. The same model then can be expanded to multiple languages which will make the model more effective and accessible. Languages like Hindi can be used as input.
5. Various other enhancements and developments can be made to make the model more user-friendly and efficient in the future.

# LITERATURE STUDY

**Braille, the Language, Its Machine Translation and Display**

**KENNETH R. INGHAM**

**IEEE TRANSACTIONS ON MAN-MACHINE SYSTEMS, VOL.  
MMS-10, NO. 4, DECEMBER 1969**

## **History and Development in Technology**

- The American Printing House for the Blind in Louisville installed an IBM 709 computer that produced Grade 2 Braille on punched cards, which were then used to drive stereotyping machines. This process was cost-effective only for large-volume runs.
- Morrison of the Telephone Company Pioneers developed a modified teleprinter that embossed Braille on a paper tape.
- Gill, Blanco, Bellows, Dangel, and Blackman at M.I.T. built strip or belt Braille displays.
- These devices could have been used as displays for tape-recorded Braille.

These were some initial developments in the field of Braille printing technology. Although the focus was entirely on punching the papers as fast as possible, to eliminate the manual work.

## **Design and Implementation of Digital Braille System for the Visually Impaired**

**International Journal of Computer Trends and Technology**

**Volume 68 Issue 3, 80-83, March 2020**

- Companies overlook users with disabilities in favour of active users with higher purchasing power and digital literacy.

- Visually impaired individuals struggle to access good reading materials and the internet due to inaccessibility.
- Screen reading software like JAWS has revolutionized computer skills for visually impaired individuals, but it's expensive due to high license costs.
- The braille display system uses six electromagnetic solenoids representing the braille alphabet, connected to an Arduino Uno microcontroller.
- The system developed used rotary actuators and stepper motors to control eight pins in each braille cell, with four stepper motors and driver ICs controlled by an 8051 microcontroller.
- The use of driver ICs complicated the system.

## **Text to Braille Conversion System**

**Department of Electronics and Telecommunication Engineering, Symbiosis Institute of Technology, Pune, India**

### **Latest Projects**

- The research paper proposes the implementation of a prototype device that enables one-way communication between persons who are differently abled (deaf, dumb, and blind) and those who are not.
- The device will be a text-to-braille conversion system that uses Portable Technology and Arduino Circuit Boards to provide the means of communication.
- The code for the conversion to braille from text is written for the Arduino microcontroller that is used.
- The text is taken from a keyboard in real-time and the rotation of the motors creates the required braille impression that can be sensed by the user.

# TOOLS REQUIRED

1. **Arduino Uno**: Arduino Uno is an open-source microcontroller board that is designed for creating interactive electronic projects. It has digital input/output pins, analog input pins, and multiple communication interfaces. It can be easily extended with additional modules for additional functionality. The board contains 6 analog I/O pins and 14 digital I/O pins, six of which can be used for PWM output. It can be programmed using the Arduino IDE (Integrated Development Environment) with a type B USB connector.



*Figure 4: Arduino UNO  
Courtesy: Google images*

2. **16x2 LCD Character Module Display**: An LCD display is an electronic device that uses liquid crystal cells to display images and text. It is controlled by a microcontroller or control system and can display alphanumeric characters, symbols, or graphics, 16x2 LCD is required in the project.

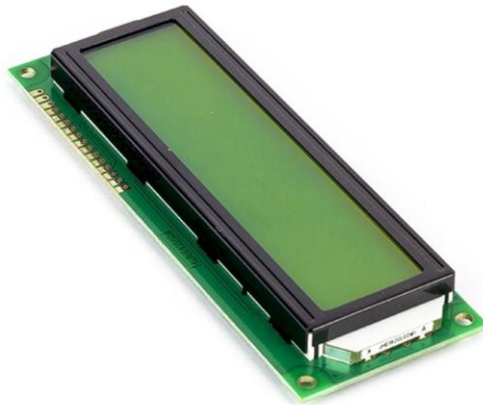


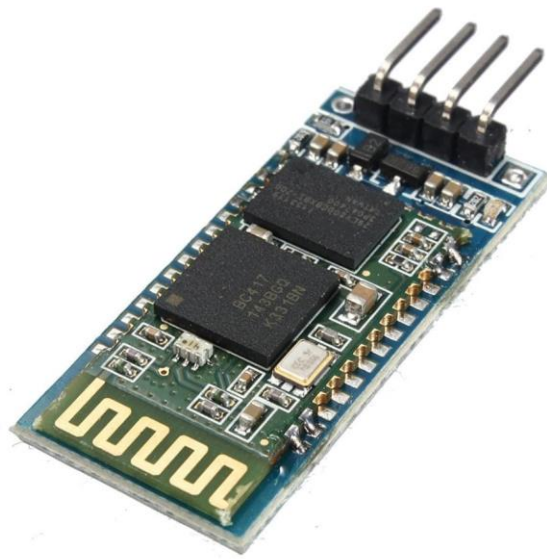
Figure 5: 16x2 LCD Character Module Display  
Courtesy: Google Images

3. **X-Axis Stepper Motor:** The X-axis stepper motor moves in the x-pivot. It is constrained by the stepper motor driver. The driver controls the moving way of the x-axis stepper motor.



Figure 6: X-axis Stepper Motor  
Courtesy: Google images

4. **Bluetooth Module:** A Bluetooth module is a small device that enables wireless communication between devices using Bluetooth technology. It typically contains a Bluetooth chip, antenna, and interface pins. The module can be configured and controlled using a microcontroller or computer and supports different modes and profiles for specific applications.



*Figure 6: Bluetooth Module*  
*Courtesy: Google images*

- 5. Breadboard
- 6. LEDs
- 7. Resistors
- 8. Jumper Wires (F-F, M-M, F-M)

# WORK DONE SO FAR

## TEXT TO BRAILLE CODE USING 7 LEDs INCLUDING BUZZER LED

```
/*
Group-3
AKSHITA SHRIVASTAVA      2021UGEC114
AYUSH KUMAR LAL          2021UGEC090
KHUSHAL KUMAR RAJAK      2021UGEC080
GIPASHU ARYAN            2021UGEC079
LIKHIT REDDY             2021UGEC078
PRAGATI DUBEY            2021UGEC008
```

### Arduino ASCII- Braille Translator

This project code will use six small solenoids/electromagnets arranged in three rows of two columns to operate a Braille dot matrix.

Any ASCII character sent to Arduino via COM port will be "translated" in a timed sequence of Braille symbols by configuring, one character at a time, the dot matrix, so that the "dots" (solenoids) corresponding to the current ASCII character are raised for 500 msec before passing to the subsequent character.

When an un-translatable character is sent, a short buzz is emitted.

#### ==== DOT NUMBERING ====

Dots are not numbered according to the Braille convention, but are conventionally numbered from left to right, then from top to bottom:

```
1  2
3  4
5  6
```

As an example, showing letter "b" (asterisk represent a high dot):

```
*_
*_
--
```

will require raising dots number 1 and 3

#### ==== DOT CONFIGURATION ====

The dot configuration will be stored as a series of six binary values. As an example, according to the preceding explanation, the configuration for **13**



"b" is 101000.

As long as configurations contain just 6 bits of information, they can be stored in a single char that, translated bitwise (least significant bit at right), corresponds to the needed configuration.

So, 101000 (configuration for "b"), corresponding to DEC 40 and HEX 28, will be represented by the ASCII character "(".

Summarizing, considering that the ASCII code for "b" is 98D (62H) I will say that

```
myBrailleDots[98] = 40;
```

meaning that the binary sextet corresponding to the letter "b" is 101000.

When needed, a bitwise AND with increasing powers of two (from 1 to 32) will reveal which dots shall be raised: this is accomplished by iterating through a 6-bit mask:

```
*/
```

```
// Declared a char array with one index for every possible ASCII
```

```
byte/character
```

```
byte myBrailleDots[256];
```

```
int firstOutputPin = 2; // pin corresponding to least significant bit
```

```
int buzzerPin = 8; // Buzzer on pin 8
```

```
byte matrixPoints = 0; // byte that will store the point matrix configuration  
// for a specific ASCII character
```

```
byte inByte;
```

```
byte mask = 1; //our bitmask
```

```
void setup() {
```

```
/*
```

```
Temporarily assign 99 to every possible ASCII byte/character
```

```
All the characters in the input string will decode to "99" by default
```

```
ASCII requires 7 bits to represent a character and one bit more is required  
to work with the computer. i.e.,  $2^8=256$ 
```

```
*/
```

```
for (int i = 0; i < 256; i = i + 1) {
```

```
    myBrailleDots[i] = 99;
```

```
}
```

```
// Now, only for the ASCII characters with a corresponding Braille  
character.
```

```
//Assign the corresponding Braille Dot configuration
```

```
myBrailleDots[32] = 0; // blank is 000000
```

```
myBrailleDots[33] = 14; // exclamation mark is 001110
```

```
myBrailleDots[34] = 7; // double quote is 000111
```

```
myBrailleDots[34] = 2; // single quote is 000010
```

```
myBrailleDots[40] = 15; // left parenthesis is 001111
```

```
myBrailleDots[41] = 15; // right parenthesis is 001111
```

```
myBrailleDots[44] = 8; // comma is 001000
```

```
myBrailleDots[46] = 13; // period is 001101
```

```

myBrailleDots[48] = 28; // 0 is 011100
myBrailleDots[49] = 32; // 1 is 100000
myBrailleDots[50] = 40; // 2 is 101000
myBrailleDots[51] = 48; // 3 is 110000
myBrailleDots[52] = 52; // 4 is 110100
myBrailleDots[53] = 36; // 5 is 100100
myBrailleDots[54] = 56; // 6 is 111000
myBrailleDots[55] = 60; // 7 is 111100
myBrailleDots[56] = 44; // 8 is 101100
myBrailleDots[57] = 24; // 9 is 011000
myBrailleDots[58] = 12; // colon is 001100
myBrailleDots[59] = 10; // semicolon is 001010
myBrailleDots[63] = 11; // question mark is 001011
myBrailleDots[65] = 32; // A is 100000
myBrailleDots[66] = 40; // B is 101000
myBrailleDots[67] = 48; // C is 110000
myBrailleDots[68] = 52; // D is 110100
myBrailleDots[69] = 36; // E is 100100
myBrailleDots[70] = 56; // F is 111000
myBrailleDots[71] = 60; // G is 111100
myBrailleDots[72] = 44; // H is 101100
myBrailleDots[73] = 24; // I is 011000
myBrailleDots[74] = 28; // J is 011100
myBrailleDots[75] = 34; // K is 100010
myBrailleDots[76] = 42; // L is 101010
myBrailleDots[77] = 50; // M is 110010
myBrailleDots[78] = 54; // N is 110110
myBrailleDots[79] = 38; // O is 100110
myBrailleDots[80] = 58; // P is 111010
myBrailleDots[81] = 62; // Q is 111110
myBrailleDots[82] = 46; // R is 101110
myBrailleDots[83] = 26; // S is 011010
myBrailleDots[84] = 30; // T is 011110
myBrailleDots[85] = 35; // U is 100011
myBrailleDots[86] = 43; // V is 101011
myBrailleDots[87] = 29; // W is 011101
myBrailleDots[88] = 51; // X is 110011
myBrailleDots[89] = 55; // Y is 110111
myBrailleDots[90] = 39; // Z is 100111
myBrailleDots[97] = 32; // A is 100000
myBrailleDots[98] = 40; // B is 101000
myBrailleDots[99] = 48; // C is 110000
myBrailleDots[100] = 52; // D is 110100
myBrailleDots[101] = 36; // E is 100100
myBrailleDots[102] = 56; // F is 111000
myBrailleDots[103] = 60; // G is 111100
myBrailleDots[104] = 44; // H is 101100
myBrailleDots[105] = 24; // I is 011000

```

```

myBrailleDots[106] = 28; // J is 011100
myBrailleDots[107] = 34; // K is 100010
myBrailleDots[108] = 42; // L is 101010
myBrailleDots[109] = 50; // M is 110010
myBrailleDots[110] = 54; // N is 110110
myBrailleDots[111] = 38; // O is 100110
myBrailleDots[112] = 58; // P is 111010
myBrailleDots[113] = 62; // Q is 111110
myBrailleDots[114] = 46; // R is 101110
myBrailleDots[115] = 26; // S is 011010
myBrailleDots[116] = 30; // T is 011110
myBrailleDots[117] = 35; // U is 100011
myBrailleDots[118] = 43; // V is 101011
myBrailleDots[119] = 29; // W is 011101
myBrailleDots[120] = 51; // X is 110011
myBrailleDots[121] = 55; // Y is 110111
myBrailleDots[122] = 39; // Z is 100111

pinMode(buzzerPin, OUTPUT); //if un-translatable character come pin 8 will
buzz
pinMode(2, OUTPUT);          //pin 2 indicate the least significant bit
pinMode(3, OUTPUT);
pinMode(4, OUTPUT);
pinMode(5, OUTPUT);
pinMode(6, OUTPUT);
pinMode(7, OUTPUT);          //pin 7 indicate the most significant bit

//This tells the Arduino to get ready to exchange messages with
//the Serial Monitor at a data rate of 9600 bits per second.
Serial.begin(9600);

Serial.println("ASCII - Braille Arduino Converter"); //Print on serial
monitor
Serial.println("LED test - begin"); //when Arduino start displaying the
character it will print
digitalWrite(2,HIGH);
digitalWrite(3,HIGH);
digitalWrite(4,HIGH);
digitalWrite(5,HIGH);
digitalWrite(6,HIGH);
digitalWrite(7,HIGH);
delay(3000);
digitalWrite(2,LOW);
digitalWrite(3,LOW);
digitalWrite(4,LOW);
digitalWrite(5,LOW);
digitalWrite(6,LOW);
digitalWrite(7,LOW);

```

```

    Serial.println("LED test - end");
    Serial.println("Type some character: it will be transmitted to Arduino and
displayed on a Braille 2 x 3 matrix");
}

```

```

void loop() {
    // Braille print data only when you receive data:
    if (Serial.available() > 0) {
        // read the incoming byte:
        inByte = Serial.read();
        // say what you got:
        Serial.print("Received (inByte): ");
        Serial.println(inByte);
        // Translate inByte in matrix points
        Serial.print("Matrix points variable (myBrailleDots[inByte]): ");
        Serial.println(myBrailleDots[inByte]);

        // Braille print only admissible characters
        //The unadmissible ones decode to 99
        if (myBrailleDots[inByte] == 99) // if unadmissible
        {
            Serial.println("Not a translatable character");
            digitalWrite(buzzerPin,HIGH); // buzz
            delay(250);
            digitalWrite(buzzerPin,LOW); // stop buzzing
        }
        else{
            int thisPin = 2;
            for (mask = 000001; mask<64; mask <= 1) {
                Serial.print("thisPin = ");
                Serial.println(thisPin);
                if (myBrailleDots[inByte] & mask){ // if bitwise AND
resolves to true
                    Serial.print("AND successful, put pin on!");
                    Serial.println(mask);
                    digitalWrite(thisPin,HIGH);
                }
                else{ //if bitwise and resolves to false
                    Serial.print("AND unsuccessful, put pin off!");
                    Serial.println(mask);
                    digitalWrite(thisPin,LOW);
                }
                thisPin = thisPin + 1;
            }
            delay(1000); // allow 1 sec before passing to next character
        }
    }
}

```

## CODE IMPLEMENTATION

1. First of all, we have declared a character array with one index for every possible ASCII byte/character.
2. Set pin 2 of Arduino for the least significant bit and pin 3,4,5, 6,7 subsequently for higher bits, and pin 8 for the buzzer if a non-translatable character comes then the buzzer led will blink.
3. We have assigned most of the ASCII characters to our braille dot configuration.
4. If character B was taken as an input, then we have assigned 40 at index 66 in the character array, this is because B in the ASCII table comes at 66 and 40 because its binary representation is 101000, as pin 2 represents the least significant bit then and LSB is 0 here so led at pin 2 will be off, LEDs at pin 3,4, and 6 will be off and LEDs at pin 5 and 7 will glow.
5. If any non-translatable character comes then led at pin blink 2 times.
6. This code read 9600 bits per second as it is the standard number to work with Arduino.
7. We are giving input through the serial monitor.  
Serial.available(), check whether the input is given or not if any input is given to the serial monitor then the loop started.
8. As input is given any character then it will be converted to the ASCII number of that character and will check in our array that at index 99 present of some other value, if 99 is present then it is a non-translatable character and buzzer led start blinking.
9. We set starting pin 2 in this loop and starting the loop of masking from 000001 and on each iteration 1 shift to the left. And corresponding to that bit, the assigned LED will glow.

# ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Figure 7-ASCII TABLE  
Courtesy:Wikimedia.org

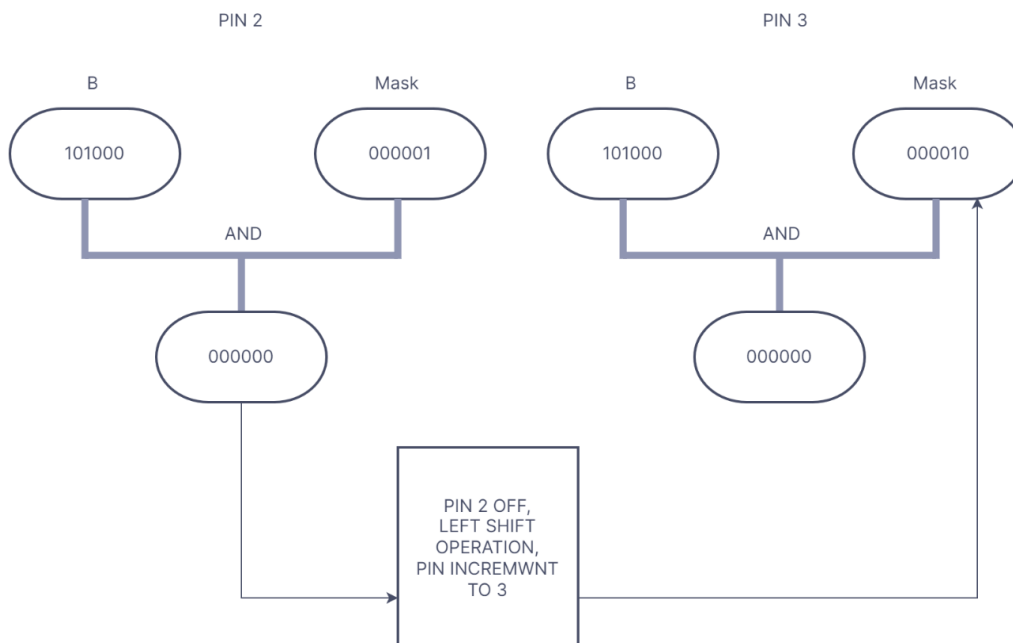


Figure 8- For Loop Flow Chart  
Courtesy: Self-Made using Zen Flowchart

# RESULTS & DISCUSSIONS

Phase 1 of the project has been successfully implemented using Arduino UNO, resistors, and LEDs. Presently LEDs are used instead of motors for displaying the braille script.

Arduino IDE is used for the development of code for the Arduino UNO and breadboard, LEDs, resistors are used for the hardware implementation of the text-to-braille converter.

In the circuit diagram below, pin 7 is connected to LED 1, pin 6 to LED 2, pin 5 to LED 3, pin 4 to LED 4, pin 3 to LED 5, pin 2 to LED 6, and a common ground to the negative terminal of the LEDs. LED 1 corresponds to cell 1 of the Braille cell, LED 2 corresponds to cell 2 of the Braille cell, and so on till cell 6.

The buzzer LED is the LED 7 which is denoted by the red colour in the circuit given below.

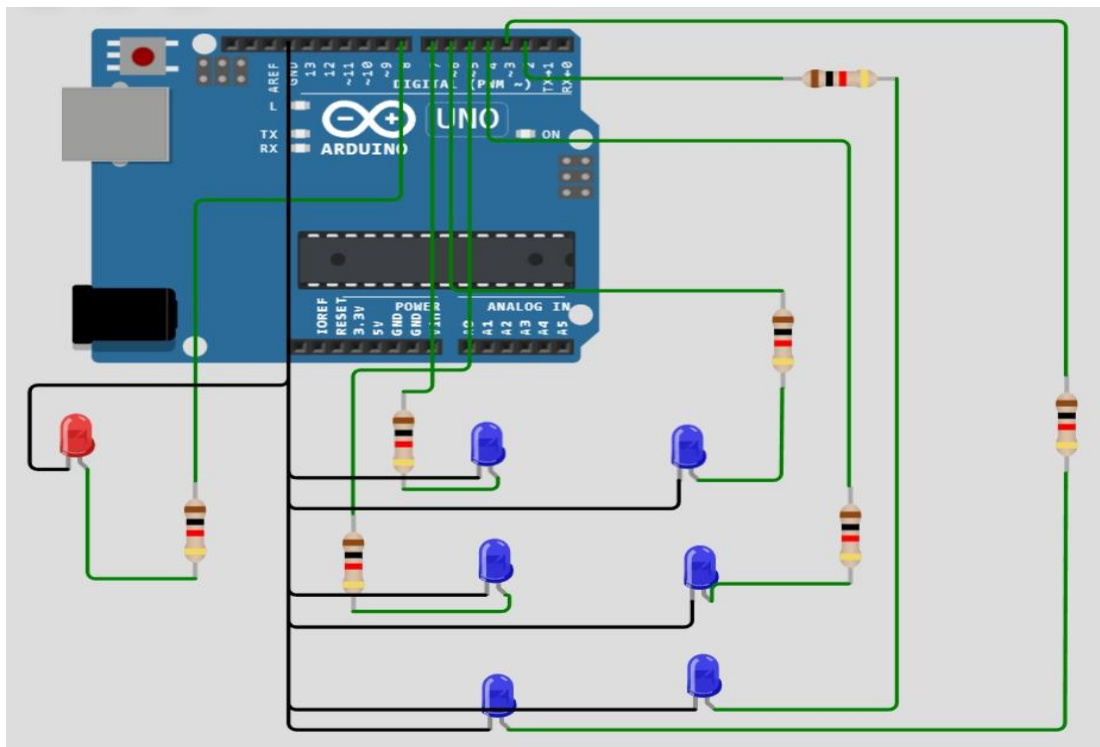


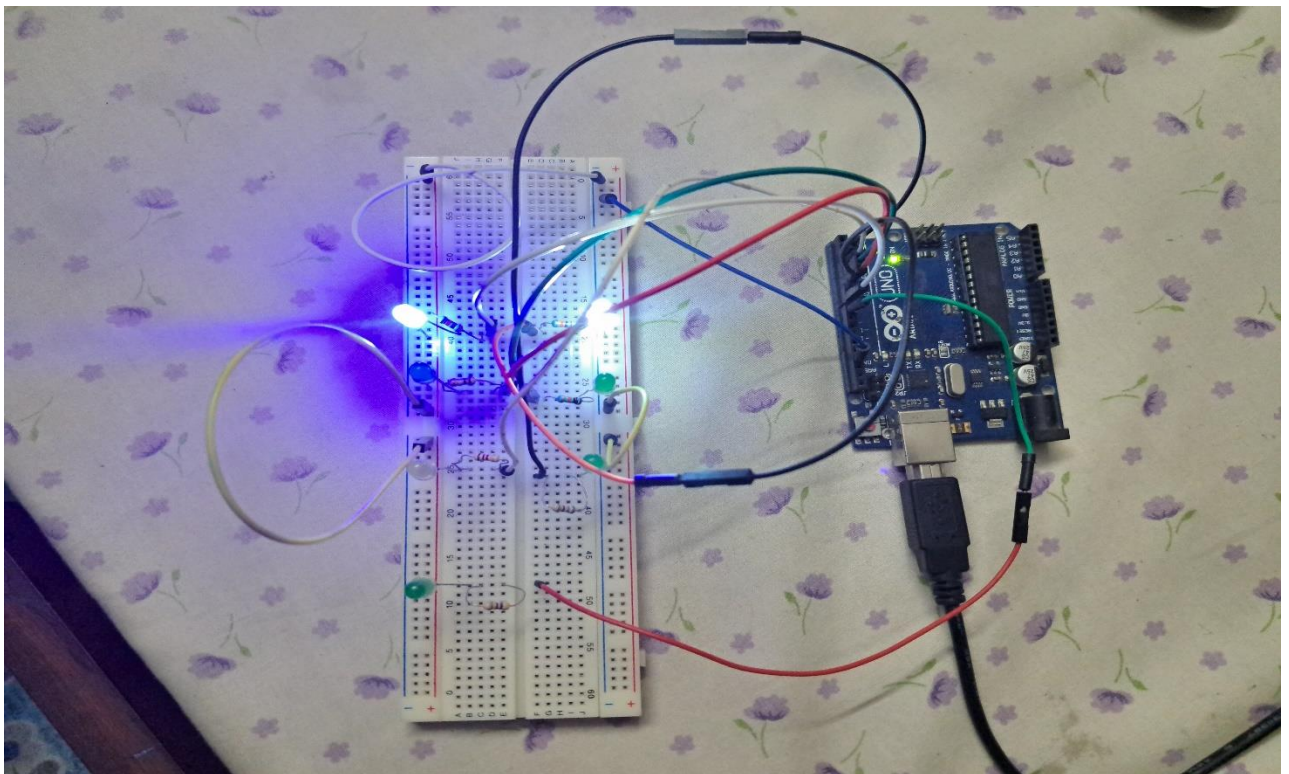
Figure 7: The implementation of the Text to Braille conversion on an online simulator ([www.wokwi.com](http://www.wokwi.com))





*Figure 8: the implementation of the text-to-braille conversion on a physical Arduino UNO*

The circuit processes the given text and displays the braille script through the LED.



*Figure 9: The result of text-to-braille conversion when the text input is "c"*



Discussion of the working of the project:

- The user gives the input through the serial monitor of the Arduino IDE.
- Then the Arduino processes the text and converts it to braille using binary conversion.
- The LED will glow as given by the binary number processed by the Arduino.
- The LED will display the corresponding braille character.

# CONCLUSIONS & FUTURE WORK

Phase 1 of the project i.e., Text to Braille script conversion has been implemented and phase 2 of the project i.e., Speech to Text to Braille script conversion will be implemented.

## Phase 1: Text to Braille script conversion

The text is converted to braille script by the Arduino UNO. The braille script is displayed by the network of the LEDs.



Figure 10: Flow chart explaining phase 1(text to braille script conversion)

## Phase 2: Speech to Text to Braille script conversion

The speech is converted to text. The converted text is received by Arduino UNO through a Bluetooth module.

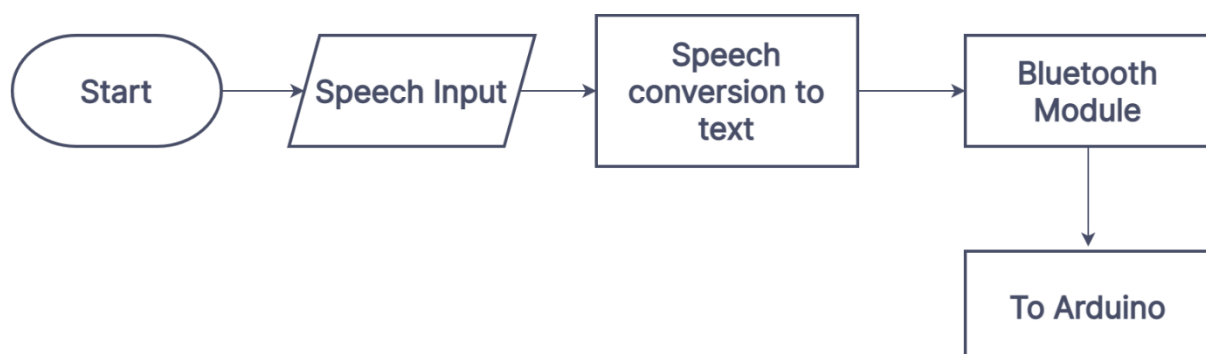


Figure 11: Flowchart explaining phase 2 (speech-to-text to braille script conversion)

Phase 1 and Phase 2 are integrated for the final device to convert speech to Braille script. The flowchart below explains the conversion of speech to Braille. The speech is converted to text which is received by Arduino UNO through a Bluetooth module. The text/string received by the Arduino UNO is converted to Braille script. The braille script is then displayed by the network of LEDs.

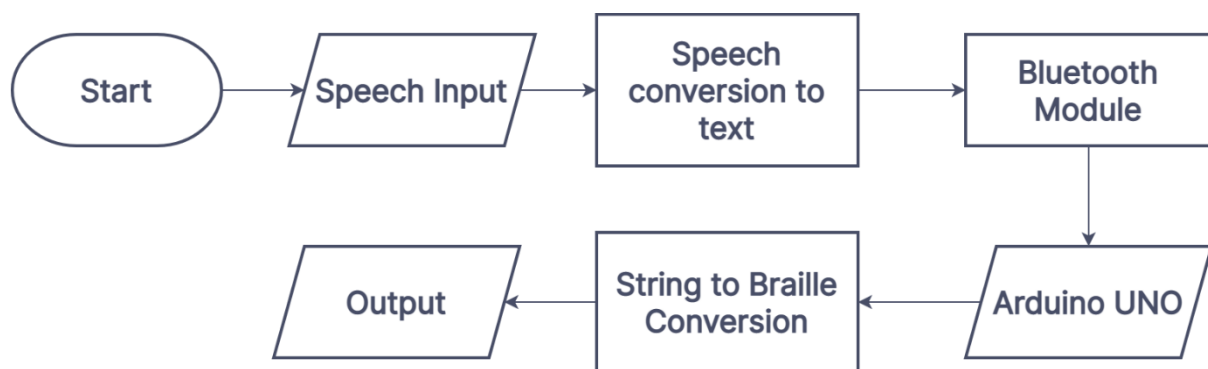


Figure 12: Flowchart explaining the integration of Phase 1 and Phase 2

# REFERENCES

1. Braille, the Language, Its Machine Translation and Display KENNETH R. INGHAM IEEE TRANSACTIONS ON MAN-MACHINE SYSTEMS, VOL. MMS-10, NO. 4, DECEMBER 1969
2. International Journal of Computer Trends and Technology Volume 68 Issue 3, 80-83, March 2020
3. The perception and use of technology within braille instruction: A preliminary study of braille teaching professionals Natalie Martiniello and Walter Wittich Université de Montréal, Canada; CRIR/MAB-Mackay Rehabilitation Centre du CIUSSS du Centre-Ouest-de-l'Île-deMontréal, Canada
4. World Health Organization, Available: <https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment>
5. World Health Organization, "Global data on visual impairment", Available: <https://www.who.int/blindness/publications/globaldata/en/>
6. J.L. Dela Cruz, J.A.D. Ebreo, R.A.J.P. Inovejas, A.R.C. Medrano and A. A. Bandala, "Development of a text to braille interpreter for printed documents through optical image processing," IEEE 9th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), 2017
7. Text to Braille Conversion System Department of Electronics and Telecommunication Engineering, Symbiosis Institute of Technology, Pune, India
8. S. Shokat, R. Riaz, S. S. Rizvi, K. Khan, F. Riaz and S. J. Kwon, "Analysis and Evaluation of Braille to Text Conversion Methods," Mobile Information Systems, 2020
9. Wikipedia article "Braille", Available: <https://en.wikipedia.org/wiki/Braille>