Early Learning Braille Block Language System

Mr. Atif Saeed, Vaughn College of Aeronautics & Technology

Atif Saeed is a Junior in Mechatronic Engineering at Vaughn College of Aeronautics and Technology. He is also the Vice President of Vaughn College's robotics team along with an active participant in both Society of Women Engineers and Unmanned Aerial Club at Vaughn College. He currently holds a FAA Airframe and Powerplant and also a FCC GROL. His interests include aerospace, aeronautics, and automotive.

Ms. Niki T. Taheri, Vaughn College of Aeronautics & Technology

Niki Taheri, Senior Mechatronics Engineering student at Vaughn College of Aeronautics and Technology. She is the President of Vaughn College's Society of Women Engineers chapter and Secretary of the Robotics Team at Vaughn College.

Early Learning Braille Block Language System

Introduction

For people who are blind or visually impaired, it is a difficult task to learn how to read. There is equipment that is available for purchase to learn how to read. These devices can be expensive and not easily accessible to all income ranges. Additionally, most of these devices are large and unaccommodating for children. The objective of this project is to develop an inexpensive, small, user-friendly braille cell learning device.

Background Research

Braille is a language that can be read by using fingers to touch a series of raised dots created by Louis Braille and is used by the blind community [1]. The symbols are formed inside braille cells which are made up of 6 raised dots in three parallel rows each having two dots. A cell can be used to represent either a letter, number or punctuation mark. An example of this can be seen in Figure 1.

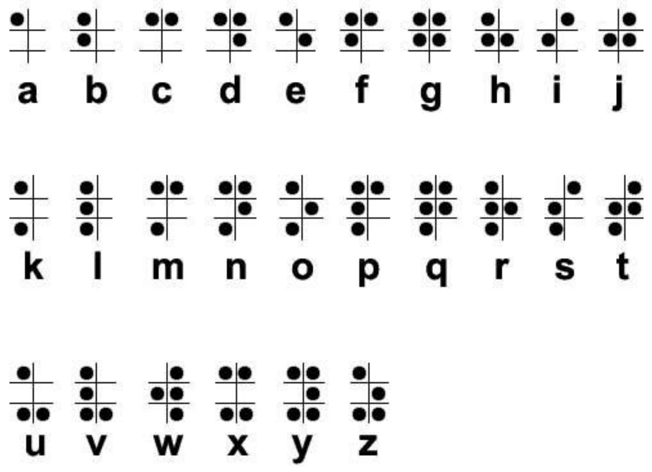

Figure 1. Braille Alphabet courtesy of https://en.m.wikipedia.org/wiki/File:Braille_alfabet.jpg

According to the National Federation of the Blind, 63,357 students have been reported blind throughout the United States. The study states that only 7.8% of these students are braille readers, and 32.7% of students are non-readers [2]. This small percentage of braille readers and a large percentage of nonreaders could be explained due to the high cost of learning tools.

Supplying the Department of Education with an affordable option would allow them to accommodate a larger percentage of students with the opportunity to learn braille.

Design Concepts

In braille, each cell is made up of six raised dots. A variable pattern of the six dots is raised and represent each letter of the alphabet. Using this knowledge, it was determined that each dot needs to be represented by a physical actuator. To turn the device on, it must be plugged into a wall outlet. A microphone is programmed to trigger the actuators which represent the letter spoken by the user.

The design for the braille learning block is fairly simple, the structure of the Braille Block is a small cube. This cube represents a singular braille cell, which includes six dots that raise or lower to represent the letter spoken by the user. Each dot is actuated using a small 5V solenoid.

Working Mechanism of Design

The design uses a total of six small solenoids, each solenoid represents a dot in the braille cell. The solenoids are placed in a three by two configuration to represent the Braille cell. There are 26 different possibilities for displaying the letters and ten possibilities for representing the numbers.

Operation of the device is as follow:

1. To turn the device ON the device must be plugged into a wall outlet
2. Say the desired letter
3. Actuators will display the desired letter
4. Repeat
5. To Turn OFF the device, the device must be unplugged from the wall

Manufactured Parts

The case for the braille learning block was designed in Inventor. The design consists of two parts: the lid and the body. The entire block is 74mm long, 36mm wide, and 36mm tall and is large enough for small children to manipulate. All electrical components, including the microcontroller, the actuators, the circuit, and the voice recognition module are housed inside the 3D printed block.

The lid of the braille block can be seen in Figure 2. This lid is specially designed for the actuators to rise up from of the box to form the braille cell. Small caps are 3D printed and placed on top of the actuators to allow for a more raised dot. The lid is attached to the base of the braille block using two screws on each end of the box, this will cover all the electronic components.

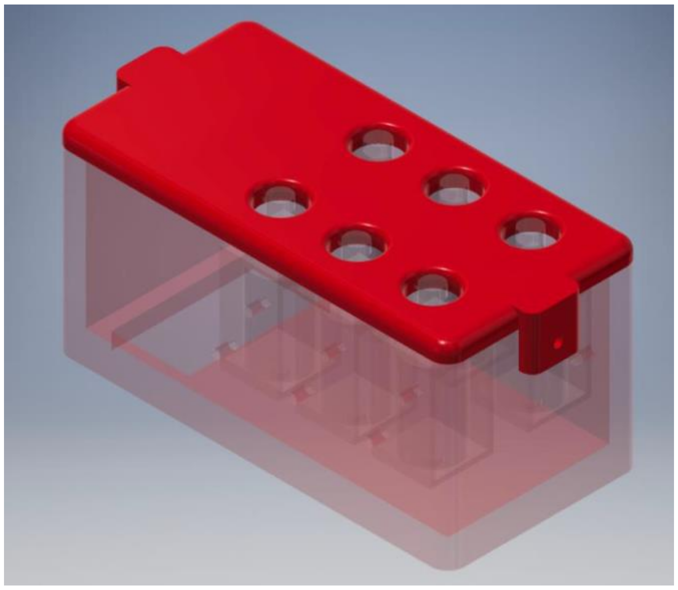

Figure 2. Lid of Braille Block

In Figure 3, the body of the braille block can be seen. Screw holes on the side of the block help mount the actuators. The braille block contains six actuators that are placed inside to make up a single braille cell. Underneath the actuators, the remaining circuit will be located.

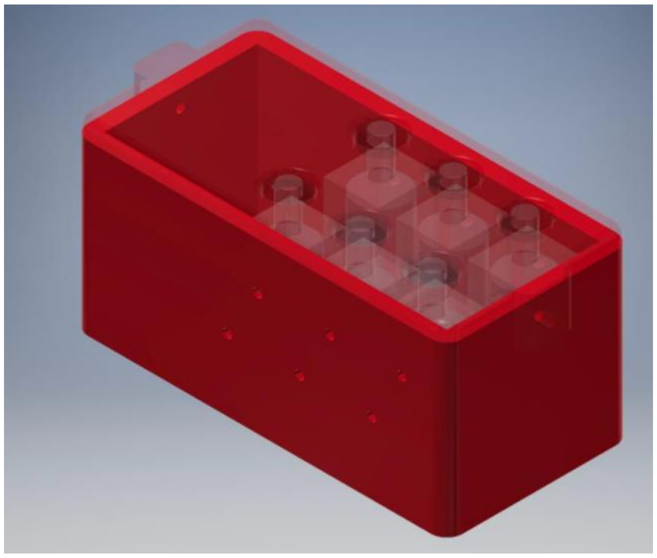

Figure 2. Bottom of Braille Block

Below, the full assembly of the braille block can be seen. This block was designed to be small and simple to keep manufacturing cost low, but large enough to be used by a small child.

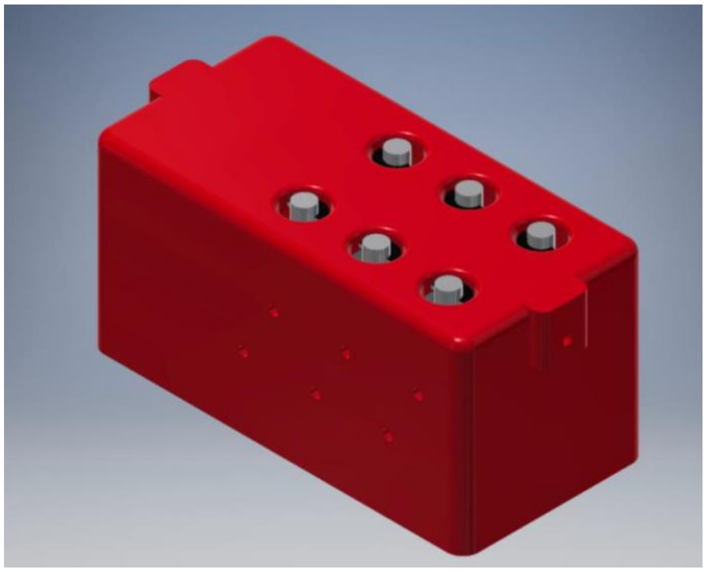

Figure 3. Full Assembly of Braille Block

Electrical Construction

Table one lists the components and prices used to create the Braille Block Learning System. These components were chosen with consideration of the size, dimensions and cost of the System.

Table 1: Parts List

Part	Cost (Individual)	Cost (total)
6 x 5 V Small Solenoid	$4.95	$29.70
1 x Arduino Uno	$29.99	$29.99
1 x 12 V 1 A DC Power Supply	$4.99	$4.99
1 x Micro Breadboard	$4.00	$4.00
6 x TIP120 Transistor	$0.85	$5.10
6 x 1N4001 Diode	$0.19	$1.14
1 X Voice Recognition Module	$29.99	$29.99

1 X TTL to USB	$5.50	$5.50
Total		$110.41

The circuit displayed in fig. 4 is powered by a 12-volt 1 Amp power supply. The Arduino Uno was chosen because a simple micro controller is needed to program the braille block. The purpose of the transistors and the diodes are to help protect the board from current when a pulse is sent from the Arduino to the solenoid. Each solenoid needs a transistor and diode combination to control current and prevent it from flowing backward [3].

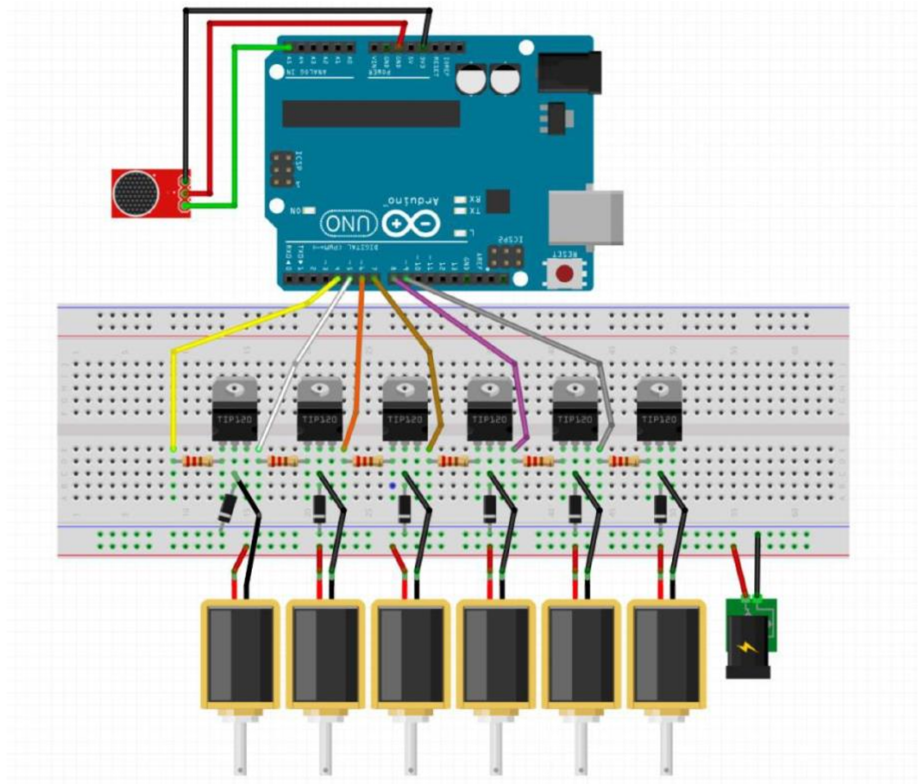

Figure 4. Braille Block Circuit

The necessary operating voltage of the solenoid is 5V. The top of each solenoid acts as a single raised dot in the braille cell matrix. The pattern displayed by the raised solenoids indicates the letter spoken. The solenoids will only activate when the microphone hears a known command.

Six solenoid sub-circuits are required for this braille block to function. Each consists of a TIP120 transistor, a diode, a resistor, and a solenoid. Common ground is shared between the emitters, the power supply, and the Arduino Uno. The resistor connects the base to an output pin on the Arduino while the diode is in parallel from the collector to the solenoid. This solenoid sub-circuit can be seen in Figure 5 [3].

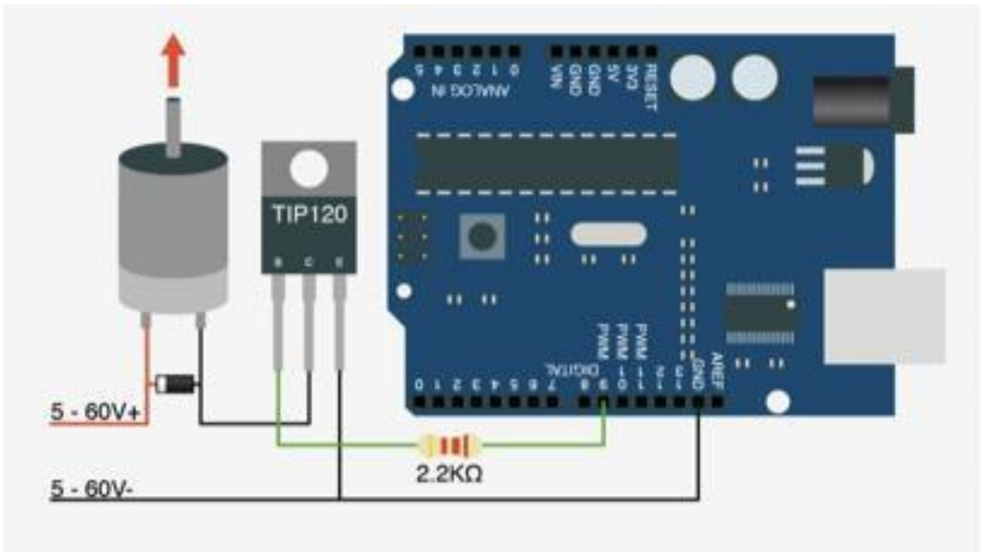

Figure 5. Solenoid Sub-circuit

The voice recognition module is used to process voice commands for the Braille Block. The TTL to USB connects the Voice Recognition Module to the MAC computer and records the inputs. Once the inputs are recorded the voice recognition module is integrated into the main Braille Block Circuit.

Software Design

To program the braille block, the Arduino software was used along with an Arduino Uno. The code is located in the appendix for further reference. The braille block will have several commands requiring the use of a microcontroller.

To program the inputs of the alphabet, a protocol called CoolTerm for Mac was used. This program only requires the recording of each letter within the program. Once recording is complete, the code is implemented into the Arduino software using its library. The code of the recording is needed to link with the desired solenoid actuation. For example, when “R” is stated by the user, solenoids 1, 3, 4, and 5 are actuated displaying the desired braille cell, as shown in Figure 7. This process can be repeated over and over until the user disconnects the Braille Block.

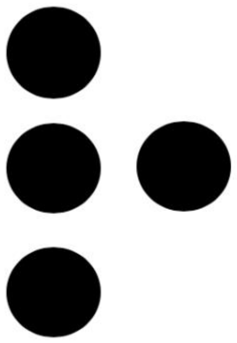

Figure 6. R Braille Cell

Conclusion

The Braille Block Learning System serves as the first step in a full learning experience. This Braille Block will make learning braille exciting and simple for all ages and levels. This device is an excellent way to assist children in the learning process at an affordable price, making it available to families and schools.

For the future, this device will be able to connect to multiple blocks. This will allow users to make words using the Braille Block Learning System. This will allow the learning process to continue as the user's knowledge increases.

Appendix

```
#include <SoftwareSerial.h>
#include
"VoiceRecognitionV3.h" VR
myVR(2,3);
uint8_t
records[7];
uint8_t buf[64];
```

```
int solenoidPin1 =
9; int solenoidPin2
= 8; int
solenoidPin3 = 7;
int solenoidPin4 =
6; int solenoidPin5
= 5; int
solenoidPin6 = 4;

#define a (0)
#define b (1)
#define c (2)
#define d (3)
#define e (4)
#define f (5)
#define g (6)
#define h (7)
#define i (8)
#define j (9)
#define k (10)
#define l (11)
#define m (12)
#define n (13)
#define o (14)
#define p (15)
#define q (16)
#define r (17)
#define s (18)
#define t (19)
#define u (20)
#define v (21)
#define w (22)
```

```
  #define x (23)
  #define y (24)
  #define z (25)


  void printVR(uint8_t *buf)
  {
  Serial.println("VR Index\tGroup\tRecordNum\tSignature");


  Serial.print(buf[2], DEC);
  Serial.print("\t\t");

if(buf[0] == 0xFF){
  Serial.print("NONE");
  }
else if(buf[0]&0x80){ Serial.print("UG
  "); Serial.print(buf[0]&(~0x80),
  DEC);
  }
else{ Serial.print("SG
  ");
  Serial.print(buf[0], DEC);
  }
  Serial.print("\t");


  Serial.print(buf[1], DEC);
  Serial.print("\t\t");

  }
  void setup()
  {
  /** initialize */
  myVR.begin(9600);


  Serial.begin(115200);
  Serial.println("Braille");
  ;
  pinMode(solenoidPin1, OUTPUT);
  pinMode(solenoidPin2, OUTPUT);
  pinMode(solenoidPin3, OUTPUT);
  pinMode(solenoidPin4, OUTPUT);
  pinMode(solenoidPin5, OUTPUT);
  pinMode(solenoidPin6, OUTPUT);

if(myVR.clear() == 0){ Serial.println("Recognizer
  cleared.");
  }else{
  Serial.println("VoiceRecognitionModule not found."); Serial.println("Please
  check connection and restart Arduino.");
  }
  }
  void loop()
  {
  int ret;
  ret = myVR.recognize(buf, 50);
  if(ret>0){
switch(buf[1]){
  case a:
  digitalWrite(solenoidPin1,        LOW);
  digitalWrite(solenoidPin2,       HIGH);
  digitalWrite(solenoidPin3,       HIGH);
  digitalWrite(solenoidPin4,       HIGH);
  digitalWrite(solenoidPin5, HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  break;
  case b:
  digitalWrite(solenoidPin1,        LOW);
  digitalWrite(solenoidPin2,       HIGH);
  digitalWrite(solenoidPin3,        LOW);
  digitalWrite(solenoidPin4,       HIGH);
  digitalWrite(solenoidPin5,       HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  break;
  case c:
  digitalWrite(solenoidPin1,        LOW);
  digitalWrite(solenoidPin2,        LOW);
  digitalWrite(solenoidPin3,       HIGH);
  digitalWrite(solenoidPin4,       HIGH);
  digitalWrite(solenoidPin5,       HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  break;
  case d:
   digitalWrite(solenoidPin1,       LOW);
  digitalWrite(solenoidPin2,        LOW);
  digitalWrite(solenoidPin3,       HIGH);
  digitalWrite(solenoidPin4,        LOW);
  digitalWrite(solenoidPin5,       HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  case e:
  digitalWrite(solenoidPin1,        LOW);
  digitalWrite(solenoidPin2,       HIGH);
  digitalWrite(solenoidPin3,       HIGH);
  digitalWrite(solenoidPin4,        LOW);
  digitalWrite(solenoidPin5, HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  break;
  case f:
  digitalWrite(solenoidPin1,        LOW);
  digitalWrite(solenoidPin2,        LOW);
  digitalWrite(solenoidPin3,        LOW);
  digitalWrite(solenoidPin4,       HIGH);
  digitalWrite(solenoidPin5,       HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  break;
  case g:
  digitalWrite(solenoidPin1,        LOW);
  digitalWrite(solenoidPin2,        LOW);
  digitalWrite(solenoidPin3,        LOW);
  digitalWrite(solenoidPin4,        LOW);
  digitalWrite(solenoidPin5,       HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  break;
  case h:
   digitalWrite(solenoidPin1,       LOW);
  digitalWrite(solenoidPin2,       HIGH);
  digitalWrite(solenoidPin3,        LOW);
  digitalWrite(solenoidPin4,        LOW);
  digitalWrite(solenoidPin5,       HIGH);
  digitalWrite(solenoidPin6,       HIGH);
  case i:
  digitalWrite(solenoidPin1,       HIGH);
  digitalWrite(solenoidPin2,        LOW);
  digitalWrite(solenoidPin3,        LOW);
  digitalWrite(solenoidPin4,       HIGH);
  digitalWrite(solenoidPin5, HIGH);
  digitalWrite(solenoidPin6,      HIGH);
  break;
  case j:
  digitalWrite(solenoidPin1,      HIGH);
  digitalWrite(solenoidPin2,       LOW);
  digitalWrite(solenoidPin3,       LOW);
  digitalWrite(solenoidPin4,       LOW);
  digitalWrite(solenoidPin5,      HIGH);
  digitalWrite(solenoidPin6,      HIGH);
  break;
  case k:
  digitalWrite(solenoidPin1,       LOW);
  digitalWrite(solenoidPin2,      HIGH);
  digitalWrite(solenoidPin3,      HIGH);
  digitalWrite(solenoidPin4,      HIGH);
  digitalWrite(solenoidPin5,      HIGH);
  digitalWrite(solenoidPin6,      HIGH);
  break;
  case l:
  digitalWrite(solenoidPin1,       LOW);
  digitalWrite(solenoidPin2,      HIGH);
  digitalWrite(solenoidPin3,       LOW);
  digitalWrite(solenoidPin4,      HIGH);
  digitalWrite(solenoidPin5,       LOW);
  digitalWrite(solenoidPin6,      HIGH);
  break;
  case m:
  digitalWrite(solenoidPin1,       LOW);
  digitalWrite(solenoidPin2,       LOW);
  digitalWrite(solenoidPin3,      HIGH);
  digitalWrite(solenoidPin4, HIGH);
```

```
 digitalWrite(solenoidPin5,       LOW);
digitalWrite(solenoidPin6,       HIGH);
case n:
digitalWrite(solenoidPin1,        LOW);
digitalWrite(solenoidPin2,        LOW);
digitalWrite(solenoidPin3,       HIGH);
digitalWrite(solenoidPin4,        LOW);
digitalWrite(solenoidPin5,        LOW);
digitalWrite(solenoidPin6,       HIGH);
break;
case o:
digitalWrite(solenoidPin1,        LOW);
digitalWrite(solenoidPin2,       HIGH);
digitalWrite(solenoidPin3,       HIGH);
digitalWrite(solenoidPin4,        LOW);
digitalWrite(solenoidPin5,        LOW);
digitalWrite(solenoidPin6,       HIGH);
break;
case p:
digitalWrite(solenoidPin1,        LOW);
digitalWrite(solenoidPin2,        LOW);
digitalWrite(solenoidPin3,        LOW);
digitalWrite(solenoidPin4,       HIGH);
digitalWrite(solenoidPin5,        LOW);
digitalWrite(solenoidPin6,       HIGH);
break;
case q:
digitalWrite(solenoidPin1,       LOW);
digitalWrite(solenoidPin2,       LOW);
digitalWrite(solenoidPin3,       LOW);
digitalWrite(solenoidPin4, LOW);
digitalWrite(solenoidPin5,        LOW);
digitalWrite(solenoidPin6,       HIGH);
break;
case r:
 digitalWrite(solenoidPin1,       LOW);
digitalWrite(solenoidPin2,       HIGH);
digitalWrite(solenoidPin3,        LOW);
digitalWrite(solenoidPin4,        LOW);
digitalWrite(solenoidPin5,        LOW);
digitalWrite(solenoidPin6,       HIGH);
case s:
digitalWrite(solenoidPin1,       HIGH);
digitalWrite(solenoidPin2,        LOW);
digitalWrite(solenoidPin3,        LOW);
digitalWrite(solenoidPin4,       HIGH);
digitalWrite(solenoidPin5,        LOW);
digitalWrite(solenoidPin6,       HIGH);
break;
case t:
digitalWrite(solenoidPin1,       HIGH);
digitalWrite(solenoidPin2,        LOW);
digitalWrite(solenoidPin3,        LOW);
digitalWrite(solenoidPin4,        LOW);
digitalWrite(solenoidPin5,        LOW);
digitalWrite(solenoidPin6,       HIGH);
break;
case u:
digitalWrite(solenoidPin1,        LOW);
digitalWrite(solenoidPin2,       HIGH);
digitalWrite(solenoidPin3,       HIGH);
digitalWrite(solenoidPin4, HIGH);
```

```
 digitalWrite(solenoidPin5,       LOW);
 digitalWrite(solenoidPin6,       LOW);
 break;
 case v:
  digitalWrite(solenoidPin1,       LOW);
 digitalWrite(solenoidPin2,      HIGH);
 digitalWrite(solenoidPin3,       LOW);
 digitalWrite(solenoidPin4,      HIGH);
 digitalWrite(solenoidPin5,       LOW);
 digitalWrite(solenoidPin6,       LOW);
 case w:
digitalWrite(solenoidPin1,         LOW);
 digitalWrite(solenoidPin2,       LOW);
 digitalWrite(solenoidPin3,       LOW);
 digitalWrite(solenoidPin4,       LOW);
 digitalWrite(solenoidPin5,      HIGH);
 digitalWrite(solenoidPin6,       LOW);
 break;
 case x:
 digitalWrite(solenoidPin1,       LOW);
 digitalWrite(solenoidPin2,       LOW);
 digitalWrite(solenoidPin3,      HIGH);
 digitalWrite(solenoidPin4,      HIGH);
 digitalWrite(solenoidPin5,       LOW);
 digitalWrite(solenoidPin6,       LOW);
 break;
 case y:
 digitalWrite(solenoidPin1,       LOW);
 digitalWrite(solenoidPin2,       LOW);
 digitalWrite(solenoidPin3,      HIGH);
 digitalWrite(solenoidPin4, LOW);
digitalWrite(solenoidPin5,      LOW);
digitalWrite(solenoidPin6,      LOW);
break;
case z:
digitalWrite(solenoidPin1,       LOW);
digitalWrite(solenoidPin2,      HIGH);
digitalWrite(solenoidPin3,      HIGH);
digitalWrite(solenoidPin4,       LOW);
digitalWrite(solenoidPin5,       LOW);
digitalWrite(solenoidPin6,       LOW);
break;
default:
Serial.println("Record function undefined");
break;
}
printVR(buf);
}
}
```

REFERENCES

[1] “What Is Braille?” *Vision Rehabilitation Services for Older People Who Are Visually Impaired - American Foundation for the Blind*, *www.afb.org/info/living-with-vision-loss/braille/what-is-braille/123*.

[2] Blindness Statistics, *https://nfb.org/resources/blindness-statistics*

[3] Rabbani, S., & D'Arrigo, J. D., & He, S., & Elzawawy , A., & Rahemi, H. (2016, June), *MAKER: A Braille Clock* Paper presented at 2016 ASEE Annual Conference & Exposition, New Orleans, Louisiana. 10.18260/p.25593

[4] Annual Report 2017: Distribution of Eligible Students Based on the Federal Quota Census of January 4, 2016 (Fiscal Year 2017). *https://www.aph.org/federal-quota/distribution-of-students- 2017/*