eyeLearn

School of Engineering and Applied Sciences
Harvard University

Cambridge MA.

Description: An interactive device for seeing individuals to learn Braille through the use of visual, auditory, and tactile signals transmitted via seven segment displays, LEDs, speakers, and solenoids.

By: Andrew Jiang, David Xu, Nabib Ahmed, and Sharanya Pulapura

Class: ES50
Instructor: Gu-Yeon Wei & Christopher Lombardo

December 2015

ABSTRACT

Visually impaired persons use the Braille system to navigate their daily lives. Often, the difficulty of memorizing tactile patterns presents a significant barrier to learning Braille, especially for seeing persons. eyeLearn provides an alternative, more engaging way to learn Braille for seeing persons. Users input English text, which is then displayed on eyeLearn's interactive board in four different formats. One section consists of fourteen characters of the English text scrolling across the board - displayed in seven-segment displays: below the scrolling English text is an LED display of the Braille characters corresponding to the English text. In addition, six solenoids arranged in a three by two array allow users to feel the Braille character corresponding to the most recently displayed English letter. A speaker allows users to hear a distinct sound corresponding to each character. Together, the aforementioned visual, auditory, and tactile interfaces provide a variety of different ways for each individual user to optimize his or her learning of the Braille language.

INTRODUCTION

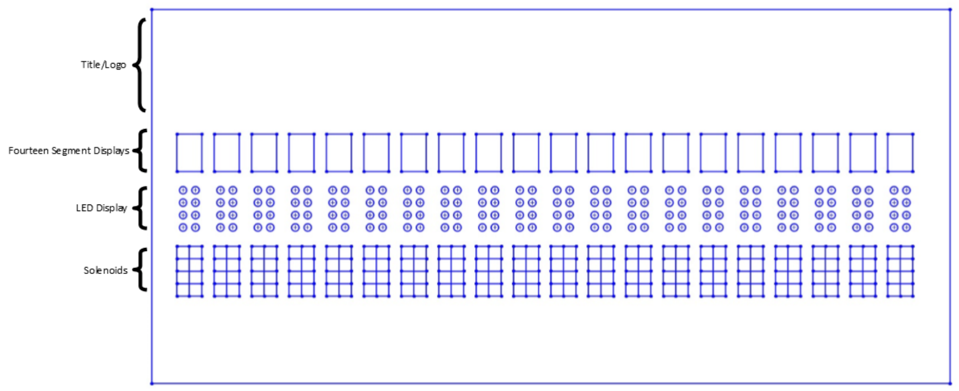
Over the past two years, Nabib worked with a startup company called Vista Wearables, which focused on creating assistive technology for the visually impaired. In his work, understanding the lives of the visually impaired was integral for success. These prior experiences inspired him with the idea of creating an educational device for teaching Braille that would allow seeing people a gateway into the lives of the visually impaired.

We chose the idea primarily because it would have an impact beyond ES50 and the lab. The project combined our shared interests in service, education, and engineering, while still being relevant to issues in the world at large. The overall goal was to create an engaging, interactive device to assist seeing people wishing to learn Braille. Users would input text through Arduino code, which would then be transformed into a Braille display.

Our approach in tackling the project relied on implementing seven-segment displays and LEDs for visual displays of English and Braille, speakers for auditory cues corresponding to each character, and solenoids for tactile representation of Braille characters, while using shift registers to shift letters along the subunits in each component of the project.

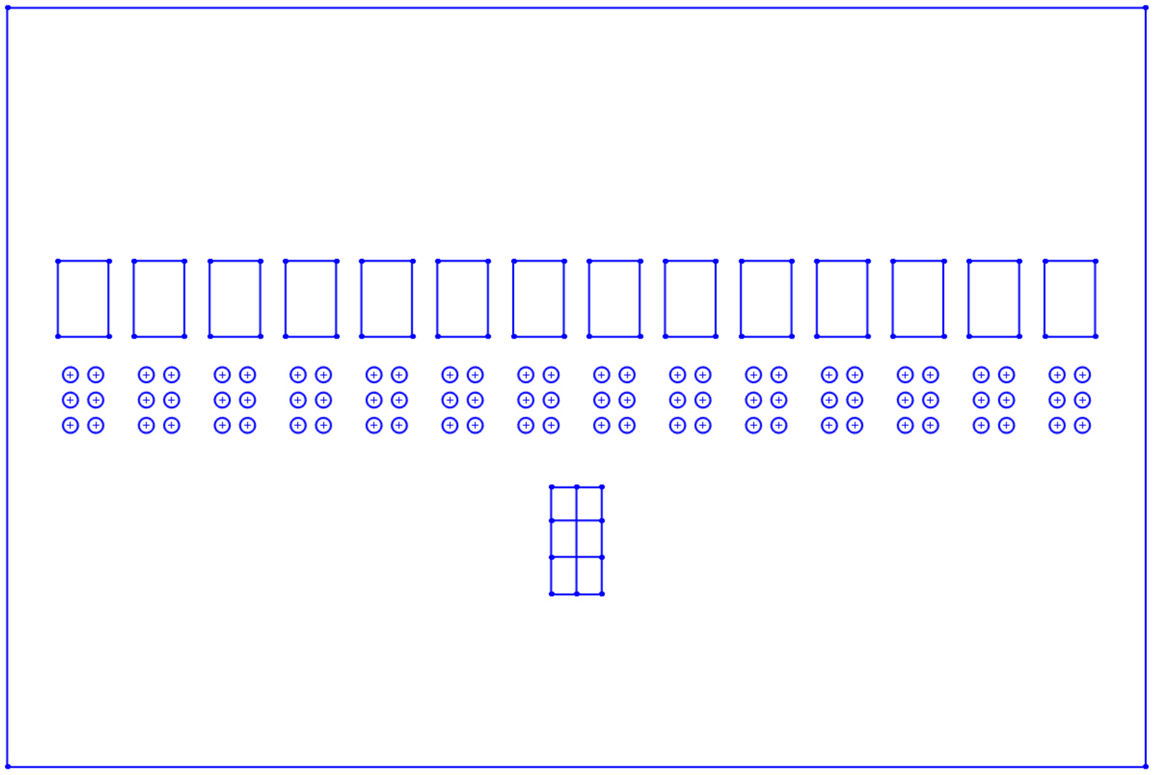
DESIGN

We originally designed a device that would display twenty characters at a time in three formats: English characters, visual Braille, and tactile Braille, while simultaneously vocalizing each character as it moved through the shift registers. We initially intended to use new Braille (which has eight dots and is useful in communicating in specialized fields like science) throughout our project. We envisioned that users would connect the device to their computer and input the displayed text via Arduino code. Our original design required twenty fourteen segment displays for displaying the text in English, 160 red LEDs for displaying the text in visual Braille, 160 solenoids for displaying the text in tactile Braille, and one speaker to output the vocalization of each character on the board. To allow the text to scroll across the displays, we planned to use 80 eight-bit shift registers: 40 for the 14-segment displays, 20 for the LEDs, and 20 for each array of 6 solenoids. We also planned to attach the speakers underneath the board. Our original design sketch looked as follows:



We decided on this design as we believed it would be the most intuitive for users and it would provide an effective learning environment that would seamlessly integrate Braille and English texts.

However, due to insufficient equipment and funding, we modified our original design to fit the constraints we faced. Solenoids were not available in the lab, so we had to order them, with each solenoid costing \$5. Since we were confined to a \$75 budget, we reduced the number of tactile Braille characters to just one. Moreover, we incorrectly assumed that the lab would have fourteen segment displays, so we decided to use the ES50 lab's 15 available seven-segment displays instead of ordering new fourteen segment displays. Since we burned out 1 seven-segment display in our experimentation, our final board displayed only fourteen characters. Lastly, we switched from new to old Braille, as we reasoned that old Braille is more common and easier to learn than new Braille, and users would most likely not need the extra features of new Braille. This also reduced the number of LED pins that we needed to use. The design then looked like:



The top row has fourteen seven-segment displays, the second row has six LEDs per seven segment display for a total of 84 LED pins, and the last row has one tactile Braille character composed of six solenoids. The speaker is stored to the side of the board and wired to the rest of the circuit. This design requires one shift register per LED array, one per seven segment display, and only one for the tactile Braille character, resulting in a total of 29 shift registers.

Essentially, budgetary restrictions and a lack of equipment led us to modify and scale down our original design to create a final design. Similar to our original design, the final design involves shift registers shifting inputted text through visual, auditory, and tactile components.

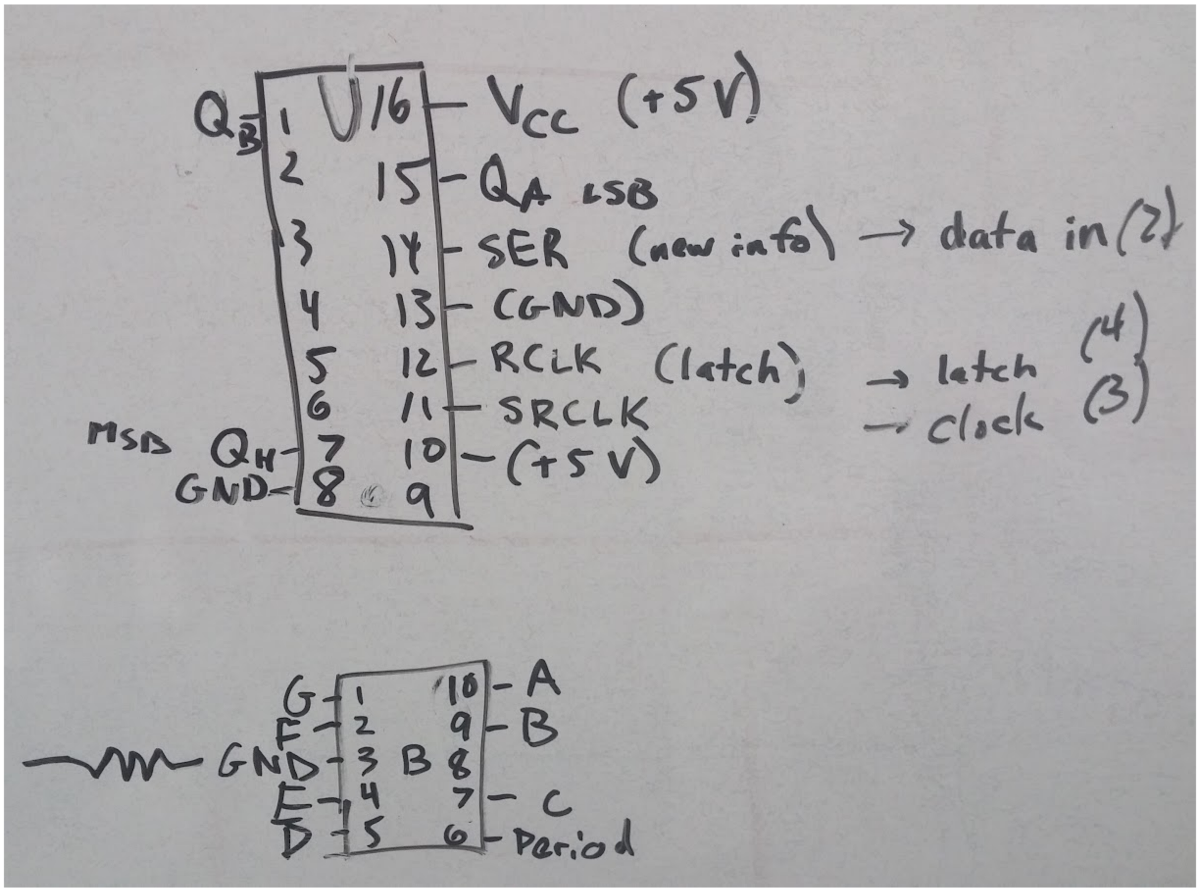
PARTS LIST

Part Name	Part Number	Link	Quantity	Price
5V Solenoid (small)	ROB-11015 ROHS	https://www.sparkfun.com/products/11015	10	Unit Price: $4.95 Total: $49.50
*Red LED	COM-09590	https://www.sparkfun.com/products/9590	84	Unit Price: $0.35 Total: $29.40
*Seven-Segment Common Cathode Display	LTS-4301JR	http://www.digikey.com/product-detail/en/LTS-4301JR/160-1533-5-ND/408206	14	Unit Price: $1.45 Total: $20.30
*Jumper Wire	PRT-11026 ROHS	https://www.sparkfun.com/products/11026	~1500	Unit Price: $0.17 Total: $255.00
*Breadboard	BB400	http://www.amazon.com/BB400-Solderless-Plug-BreadBoard-tie-points/dp/B0040Z1ERO	24	Unit Price: $5.50 Total: $132.00
*Acrylic Board, Clear, 12" x 12" x ¼"	S1-12x12-.25	http://www.amazon.com/Source-Premium-Acrylic-PlexiGlass-S1-12x12-25/dp/B00844SOSE	1	Unit Price: $30.00 Total: $30.00
*Arduino Uno	DEV-11021 ROHS	https://www.sparkfun.com/products/11021	1	Unit Price: $24.95 Total: $24.95
*Transistor	IRLZ14	http://www.digikey.com/product-detail/en/IRLZ14/IRLZ14-ND/51140	6	Unit Price: $0.85 Total: $5.10
*Diode	1N4001	http://www.digikey.com/product-detail/en/1N4001/1N4001FSCT-ND/1532742	6	Unit Price: $0.17 Total: $1.02
*Shift Register	SN74HC595	https://www.sparkfun.com/products/13699	29	Unit Price: $0.95 Total: $27.55
*Speaker			1	

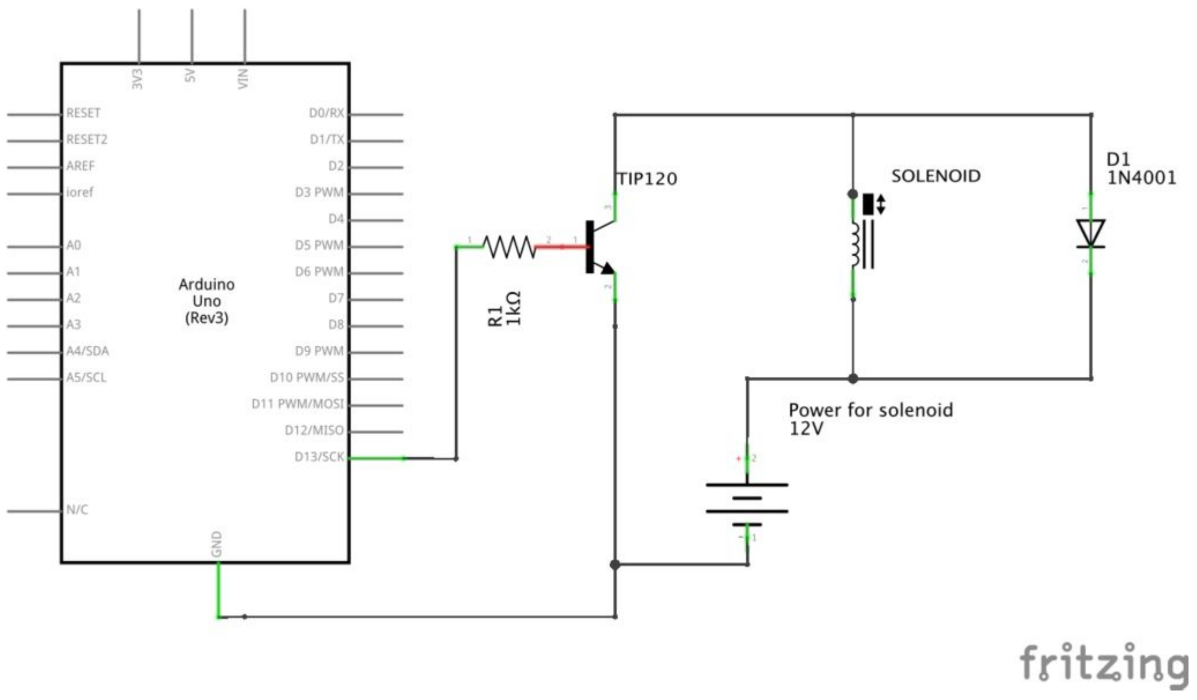
*found in the lab

PROJECT IMPLEMENTATION

We divided the implementation of our project into four main parts: the seven-segment displays, the LEDs, the solenoids, and the speaker. Before scaling up, we made smaller versions of each element: we connected three seven-segment displays, three sets of LEDs, and two solenoids to the main shift register. The circuitry for the LEDs and seven-segment displays were adapted and modified from labs we had completed earlier in ES50. The following image depicts how we wired the shift registers to the displays.



We developed the circuitry for the solenoids by modifying the circuit below, which we found online, to include an extra resistor from the transistor gate to the Arduino ground. We added the extra resistor because we used a MOSFET instead of the TIP120 transistor shown in the circuit diagram below.



Then we wrote code to shift the Arduino input into each component and tested our smaller versions. Up until this point, we did not have much to debug, since the smaller versions all worked as we expected they would. (A video of the small scale version of our project can be found at https://www.youtube.com/watch?v=kTJt7OKv5oM.)

Our next task was to scale up the project. We first tackled the solenoids. Once we finished wiring all six solenoids, we noticed that three were working as anticipated and the other three were not. After quickly noticing that some battery packs were switched off, we had five solenoids that worked and one that did not work, despite having the same wiring and circuitry as the other five. Using the oscilloscope, we realized that in all the functional solenoids, we had voltage across both resistors and the diode, but in the dysfunctional solenoid, all of the voltage went to the resistors and none went to the diode. We replaced every part - including the wires and breadboard - and rewired the circuit several times but to no avail. Finally, heeding Cameron's humorous advice that the dysfunctional circuit might just be "cursed", we built another solenoid circuit on a separate breadboard that ended up working. We never determined what was wrong with the other circuit, but proceeded to the rest of the project after spending several hours trying to fix it.

Fortunately, scaling up the LEDs and seven-segment displays and adding in the speaker went far more smoothly. (A video of the scaled up LEDs can be found at https://www.youtube.com/watch?v=jzXhhWpEzcM. A video of the scaled up seven-segment displays can be found at https://www.youtube.com/watch?v=EQHamKYtlfY.)

However, when it came time to mount our project on the acrylic board, we needed to extend the wires attached to the LEDs and seven-segment displays. Our first idea to solve this problem was to solder extra wire to all the parts, but we soon realized that this was inefficient and that our problem was easily solved by using male-female wires instead. The downside of this solution was that we ended up using hundreds of wires. Further exacerbating this problem was our decision to mount our project high above the breadboards, which forced us to use three or four wires to connect each part to the display board. Ultimately, these decisions collectively led us to use over a thousand wires in the project. Since we used so many shift registers for our project, and shift registers generally require many wires, we perhaps used many more wires than were necessary. If we had more time to work on this project, we would have reorganized the breadboards beneath the acrylic board to effectively minimize the number of wires we used. We would have also used manufactured grids of multiple LEDs rather than individual LEDs to implement the visual Braille displays, which in turn would have also reduced the number of wires that we used.

Here is a video of the finished product:
https://www.youtube.com/watch?v=_Az465dNBgc.

TEAM MANAGEMENT

We started the project by collectively tackling the challenges that we encountered in the beginning. As an entire group, we worked on wiring the 7 segment displays and the first few shift registers. After developing an initial intuition of the project as a whole, we tackled the LEDs in a similar manner, with Nabib and David working on the theoretical aspects of the circuitry and Andrew and Sharanya working on connecting the circuits together.

After we completed the aforementioned tasks, Andrew worked on the code of the project while Sharanya and David developed the circuitry associated with the speakers and created and debugged the circuits associated with the solenoids.

In the final days of the project, Nabib created the project display board by designing a template on SolidWorks and laser cutting the design on an acrylic board. Andrew, David, and Sharanya worked on extending the remainder of the project and then finally connecting the circuitry to the visible display.

OUTLOOK AND POSSIBLE IMPROVEMENTS

As we described earlier, the main problem with our project was the sheer quantity of wires that we used. If we had more time, we could have reorganized the breadboards to significantly reduce the number of wires in the project. Furthermore, as we have mentioned, the initial design of our project was more complex than the final product due to a lack of resources and time. It would be easier to use eyeLearn to read tactile Braille if we implemented more than one solenoid array, but we were constrained by the high cost of the solenoids. We could have also made the device easier to use by making the tactile Braille character smaller so that the user would be able to touch it with a single finger, as is the norm for Braille characters.

ACKNOWLEDGEMENTS:

We would like to thank our project TF Joy Hui as well as the other ES50 TFs and professors who helped us throughout this project.

Special thanks to Greg Hewett, Cameron Akker, and Professor Lombardo for their patience and expertise as they helped us with everything from design to tedious debugging.

DISCLAIMER

We hereby grant permission to share our report, code that we wrote, and photos and videos of our project.

REFERENCES:

Circuit for Solenoids: http://www.instructables.com/id/Controlling-solenoids-with-arduino/
7 Segment Displays: http://datasheet.octopart.com/LTS-4301JR-Lite-On-datasheet-58425.pdf
Circuit for Speaker: http://www.instructables.com/id/Arduino-Basics-Making-Sound/
Solenoid Datasheet: https://cdn.sparkfun.com/datasheets/Robotics/ZHO-0420S-05A4.5%20SPECIFICATION.pdf

APPENDIX

All of our drawings and diagrams are either pictured above or are accessible via the links in the References section.

Arduino Code:

```
int latchPin = 2; //general for all three systems
int clockPin1 = 6; //displays
int dataPin1 = 5;
int clockPin2 = 9; //LEDs
int dataPin2 = 8;
int clockPin3 = 12; //solenoids
int dataPin3 = 11;
int soundPin = 13; //speaker

static int parts = 3; //how many sections the main breadboard is connected to

void setup() {
 //set pins to output to control shift register
 pinMode(latchPin, OUTPUT);
 pinMode(clockPin1, OUTPUT);
 pinMode(dataPin1, OUTPUT);
 pinMode(clockPin2, OUTPUT);
 pinMode(dataPin2, OUTPUT);
 pinMode(clockPin3, OUTPUT);
 pinMode(dataPin3, OUTPUT);
 pinMode(soundPin, OUTPUT);
}

void loop() {
 char input[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ  ";
 int leng = sizeof(input);
 int bits[parts*leng]; // create array that outputs signals
 double sound[leng];
 for (int i = 0; i < leng; i++) {
   bits[parts*i] = getDisplay(input[i]);
   bits[parts*i+1] = getBraille(input[i]);
   bits[parts*i+2] = getInverseBraille(input[i]);
   sound[i] = getSignal(input[i]);
 }

 for (int i = 0; i < leng; i++) {
   digitalWrite(latchPin, LOW);
   shiftOut(dataPin1, clockPin1, MSBFIRST, bits[parts*i]);
   shiftOut(dataPin2, clockPin2, MSBFIRST, bits[parts*i+1]);
   shiftOut(dataPin3, clockPin3, MSBFIRST, bits[parts*i+2]);
   digitalWrite(latchPin,HIGH);
   tone(soundPin, sound[i]);
   delay(2000);
 }


 //while(true); //stop the loop
}

int getDisplay(char letter) {
 switch (letter) {
   case 'A': return B11110110; break;
   case 'B': return B01111100; break;
   case 'C': return B10111000; break;
   case 'D': return B01011110; break;
   case 'E': return B11111000; break;
   case 'F': return B11110000; break;
   case 'G': return B10111100; break;
   case 'H': return B01110100; break;
   case 'I': return B00110000; break;
   case 'J': return B00011110; break;
   case 'K': return B01110110; break;
   case 'L': return B00111000; break;
   case 'M': return B10010101; break;
   case 'N': return B01010100; break;
   case 'O': return B10111110; break;
   case 'P': return B11110010; break;
   case 'Q': return B11100110; break;
   case 'R': return B10110010; break;
   case 'S': return B11101100; break;
   case 'T': return B01111000; break;
   case 'U': return B00111110; break;
   case 'V': return B00011101; break;
   case 'W': return B00101011; break;
   case 'X': return B01110111; break;
   case 'Y': return B01101110; break;
   case 'Z': return B11011010; break;
   case '.': return B00000001; break;
   default: return B00000000; break;
 }
}

int getBraille (char letter) {
 switch (letter) {
   case 'A': return B100000; break;
   case 'B': return B110000; break;
   case 'C': return B100100; break;
   case 'D': return B100110; break;
   case 'E': return B100010; break;
   case 'F': return B110100; break;
   case 'G': return B110110; break;
   case 'H': return B110010; break;
   case 'I': return B010100; break;
   case 'J': return B010110; break;
   case 'K': return B101000; break;
   case 'L': return B111000; break;
   case 'M': return B101100; break;
   case 'N': return B101110; break;
   case 'O': return B101010; break;
   case 'P': return B111100; break;
   case 'Q': return B111110; break;
   case 'R': return B111010; break;
   case 'S': return B011100; break;
   case 'T': return B011110; break;
   case 'U': return B101001; break;
   case 'V': return B111001; break;
   case 'W': return B010111; break;
   case 'X': return B101101; break;
   case 'Y': return B101111; break;
   case 'Z': return B101011; break;
   case '.': return B010011; break;
   default: return B000000; break;
 }
}

int getInverseBraille (char letter) {
switch (letter) {
   case 'A': return B011111; break;
   case 'B': return B001111; break;
   case 'C': return B011110; break;
   case 'D': return B011100; break;
   case 'E': return B011101; break;
   case 'F': return B001110; break;
   case 'G': return B001100; break;
   case 'L': return B000111; break;
   case 'H': return B001101; break;
   case 'I': return B101110; break;
   case 'J': return B101100; break;
   case 'K': return B010111; break;
   case 'M': return B010110; break;
   case 'N': return B010100; break;
   case 'O': return B010101; break;
   case 'P': return B000110; break;
   case 'Q': return B000100; break;
   case 'R': return B000101; break;
   case 'S': return B100110; break;
   case 'T': return B100100; break;
   case 'U': return B010011; break;
   case 'V': return B000011; break;
   case 'W': return B101000; break;
   case 'X': return B010010; break;
   case 'Y': return B010000; break;
   case 'Z': return B010001; break;
   case '.': return B101001; break;
   default: return B111111; break;
 }
}

double getSignal (char letter) {
 switch (letter) {
   case 'A': return 130.813; break;
   case 'B': return 138.591; break;
   case 'C': return 146.832; break;
   case 'D': return 155.563; break;
   case 'E': return 164.814; break;
   case 'F': return 174.614; break;
   case 'G': return 184.997; break;
   case 'L': return 195.998; break;
   case 'H': return 207.652; break;
   case 'I': return 220.000; break;
   case 'J': return 233.082; break;
   case 'K': return 246.924; break;
   case 'M': return 261.626; break;
   case 'N': return 277.183; break;
   case 'O': return 293.665; break;
   case 'P': return 311.127; break;
   case 'Q': return 329.628; break;
   case 'R': return 349.228; break;
   case 'S': return 369.994; break;
   case 'T': return 391.995; break;
   case 'U': return 415.305; break;
   case 'V': return 440.000; break;
   case 'W': return 466.164; break;
   case 'X': return 493.883; break;
   case 'Y': return 523.251; break;
   case 'Z': return 554.251; break;
   default: return 587.330; break;
 }
}
```