Electronic Mathematical Brailler

Final Report

Team Members:

Tarane Subramaniam

Marta Masramon Munoz

James Marshall

Helena Ferreira Pinto

Hye Lyoung Chang

Martina del Viscovo

Maciej Adam Zajaczkowski

Callan Egan

Petya Tomova

Supervisor: Dr Warren Macdonald

15 June 2020

Word Count: 3935

ABSTRACT

Blind and visually impaired individuals require specialized technologies that use tactile elements instead of visual cues as a tool for learning. There is currently a range of devices on the market that facilitate literacy through refreshable displays. They use the Braille system where a combination of raised dots, felt with the fingertips, represents each character. However, none supports complex mathematics. This results in a lack of numeracy that hinders the ability of blind and visually impaired people to thrive in STEM-related fields. Additionally, the vast majority of existing electronic Braillers are very expensive, bulky and tend to offer only single line displays.

The aim of this project is therefore to create a low cost, math-focused electronic Brailler. We first investigate different methods of producing tactile information using materials that are readily available. We base our choices on the user requirements provided by the Kilimanjaro Blind Trust (KBT) and find that there is a significant design trade-off between cost and speed at which a braille cell is refreshed.

We then build a prototype that integrates two different technologies to have a more extensive functionality. The first is a series of refreshable braille lines, each consisting of a set of 3D-printed octagons that rotate around a shaft, to represent mathematical equations. The second is a flip-dot grid (made by customizing an old bus display) that allows the representation of abstract shapes and graphs. We also include a keyboard to enable user-device interaction and an SD card port to read files and save workings. We use Arduino software for all input-output communication and create a database to quickly transition between storable text and interpretable commands.

This project does not end in a finished product, for the prototype requires scaling down and user testing to perfect the design, but it demonstrates the possibility of creating an accessible device that can increase numeracy among blind and visually impaired individuals.

ACKNOWLEDGEMENTS

Firstly, we would like to thank the Kilimanjaro Blind Trust for entrusting us with a project that can have a real positive impact on the blind and visually impaired community, as well as Jaime Aguilera for putting us in contact with them. We are also very grateful for the guidance provided by our supervisors, Dr. Warren MacDonald and Dr. Ian Radcliffe. We would like to thank Paschal Egan, Niraj Kanabar and Ji Young Yoon for their help in the development of the prototype's design. Finally, we would like to extend our gratitude to Joseph Clarke School; they showed us their classrooms and helped us understand the biggest obstacles blind students face when using the current Braille technology.

CONTENTS

1. INTRODUCTION

More than a quarter of the world's population - 2.2 billion people - are visually impaired, of whom 39 million are blind. A tactile reading and writing system where characters are represented by patterns of raised dots, called Braille, has provided such individuals with an effective vehicle for literacy. The scarcity of tools for numeracy, however, is still one of the major obstacles in their pursuit of careers in science, technology, engineering or mathematics.

1.1. AIMS

The main objective of this project is to develop an electronic mathematical Brailler that can aid blind and visually impaired learners with their educational requirements in STEM fields. This is a collaborative project with Kilimanjaro Blind Trust Africa, KBTA, a charitable organization that aims to facilitate quality education for children with vision impairment by providing them with access to Braille tools.

1.2. BACKGROUND RESEARCH

1.2.1. EXISTING TECHNOLOGY

There are currently various types of high-performing refreshable braille displays on the market. However, the vast majority of them are very expensive - making them inaccessible to many. Moreover, they are bulky despite displaying only a single line of characters. These features could certainly be identified as impediments to the effective usage of the existing machinery in classroom or work environment.

1.2.2. SCHOOL VISIT

A visit to Joseph Clarke School, a centre for pupils with visual impairment, enabled the identification of other possible areas of improvement based on real user experience. The feedback mainly emphasized the inefficiency of having a single line display and the need to be constantly switching between calculator and braille typewriter when solving mathematical equations.

1.2.3. INTERVIEW WITH KILIMANJARO BLIND TRUST

A video call was also held with the KBT representatives. It helped us specify the device requirements, as well as identify which features and functionalities were necessary to alleviate the shortcomings of the recently launched braille device, the Orbit Reader 20 [1]. Even though the Orbit Reader solved some of the previous limitations (i.e. they are compact, portable, and cheaper than the original braille systems), they are still costly, innumerate, and display a single line.

2. REQUIREMENTS DEFINITION

2.1. FUNCTIONALITY AND PERFORMANCE

1. Input - the Brailler should include 6 input keys (one for each of the dots in the braille cell), a space bar and an erase button.
2. Output - the Brailler display should be refreshable and able to show up to 10 lines.
3. The Brailler should allow an SD card entry so that input files in .txt format can be read and saved, which should further enable the conversion of braille notation to text for educators who lack knowledge of Nemeth Code.
4. Three different modes should be available: reading, writing and calculator.
5. Mathematical operations as required by the A-level mathematics syllabus should be computable.

2.2. DIMENSIONS AND COST

6. The Brailler should be slightly bigger than the Orbit reader 20, with suitable dimensions of 25 x 25 x5-8 cm and weigh less than 1 kg in order to achieve portability.
7. The cost should not exceed £200.

2.3. SAFETY AND SECURITY

8. Every electrical component must be secured and insulated to prevent any possible shock hazards on the users.
9. The Brailler must not have sharp edges.

2.4. MAINTENANCE

10. The device should have a lifetime of at least 5 years.
11. The device should also come with a rechargeable battery with sufficient battery life.

2.5. ERGONOMICS

12. Brailler should produce little noise when user is typing.

3. FINAL DESIGN

3.1. OVERVIEW

Our final design consists of two parts, the graph display (see component on the left in Figure 1) and the braille display (see component on the right). The keyboard, which takes inputs and controls the graph and braille display, is housed within the braille display box. The current solution was decided to be the best solution to the problem due to reasons stated in Appendix H.

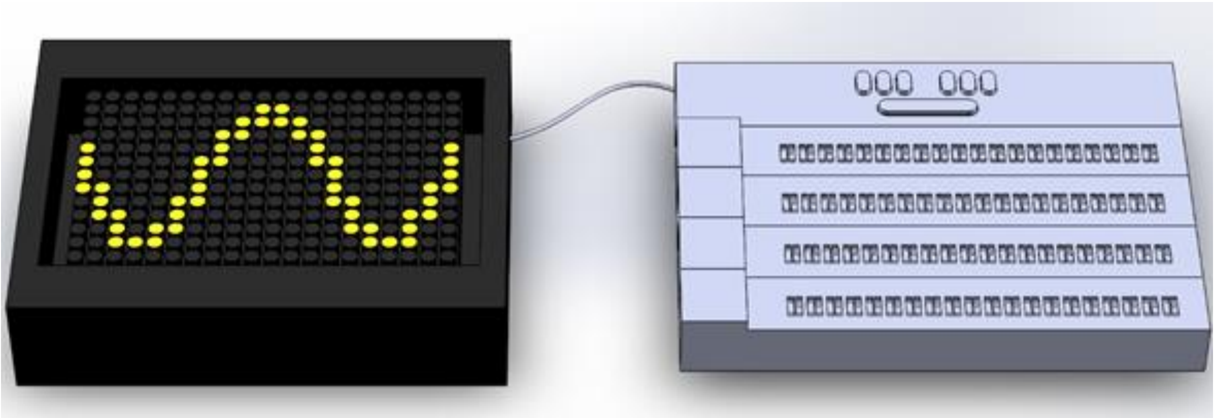


FIGURE 1. FINAL DESIGN OF THE BRAILLER

3.2. BRAILLE DISPLAY

The final design for our braille display is based on three constituent parts – the individual disks, the mechanism for rotating the disks and the housing which holds them in place.

3.2.1. ROTATING DISKS

The display of our Brailler is made up of octagonal disks. Each disk has a different combination of 3 braille dots (either raised or flat surface) on each of its 8 faces. Two of these discs combined can therefore form a braille cell, which represents a letter or number (see Figures 2 and 3). The desired combination of dots will be displayed on the top faces of the discs, the only faces that a user can feel. The dimensions of the disks have been chosen carefully to ensure that the size and spacing of the dots meet the standards set by the UK Association for Accessible Formats (UKAAF) [2].

Figures 2 and 3 show internal shape of the disks. Their centre has a gear-like cut-out to allow interaction with the other parts of the mechanism.

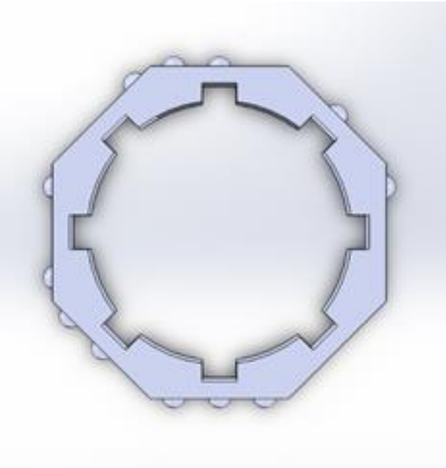


FIGURE 2. DISK WITH ALL POSSIBLE DOT COMBINATIONS

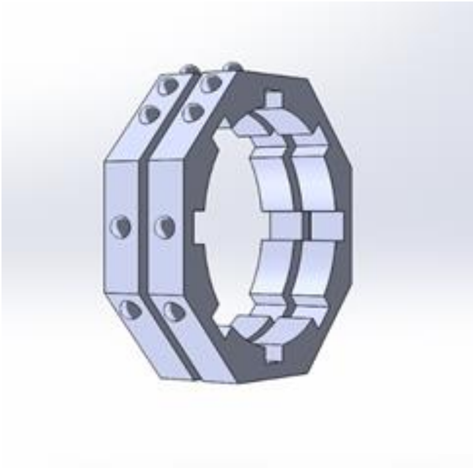


FIGURE 3. TWO DISKS MAKING A CELL

3.2.2. ROTATING MECHANISM

The rotating mechanism is further divided into three parts – the centrepiece, the leadscrew and the locking mechanism, shown in Figure 6.

The centrepiece (see Figures 4 and 5) interacts with the rotating octagons. Its teeth are slightly smaller than the cut-out inside the disks, for a snug fit inside, allowing it to engage with and turn the octagonal disk. The gap in the middle of the centrepiece houses the leadscrew nut, allowing it to fit onto the leadscrew. The outermost holes accommodate the rods that drive the locking mechanism.

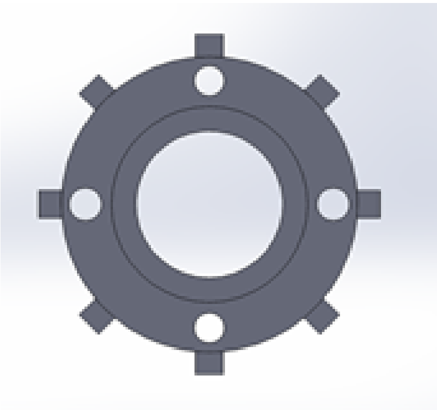


FIGURE 4. CENTREPIECE (TOP VIEW)

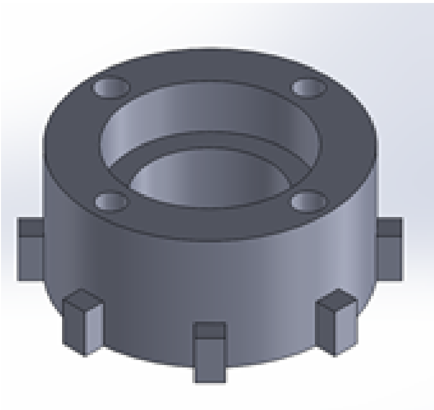


FIGURE 5. CENTREPIECE (BIRDS EYE VIEW)

The leadscrew rotates along with the stepper motor (NEMA 17 Bipolar stepper motor) , causing different actions depending on the state of the locking mechanism. The locking mechanism has two states – when the rods are locked onto the connector, the octagonal disk will rotate; when they are not, the centrepiece will move linearly along the shaft.

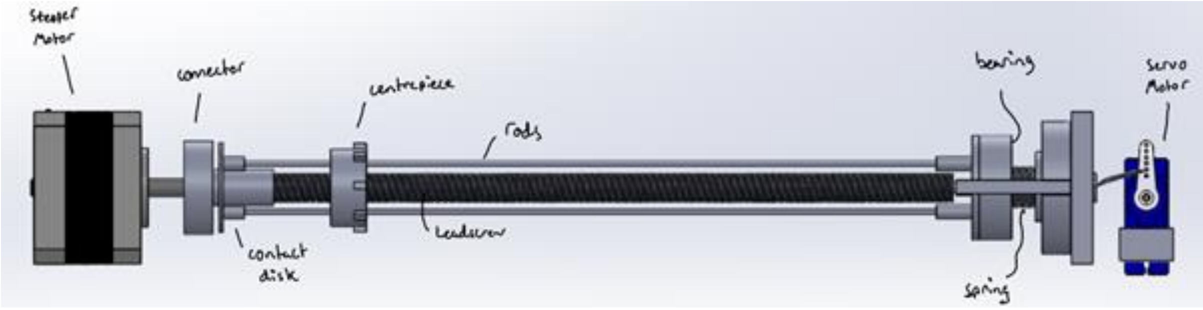


FIGURE 6. ROTATING MECHANISM

In more detail, when we want the centrepiece, and therefore the octagonal disk, to rotate - the contact disk will press against the connector. This is shown on the left of Figure 6. In this case, the rotation of the connector due to the movement of the stepper motor will cause the rods to circle around the leadscrew. This will in turn cause the centrepiece and the disk to rotate, changing the face that is being displayed. The bearing at the other end of the rods ensures the rotation is smooth.

On the other hand, when the contact disk is not touching the connector, the rods will not rotate. In this case, the centrepiece does not turn, and so rotation of the stepper motor and leadscrew will lead to linear movement – the centrepiece will be displaced along the shaft.

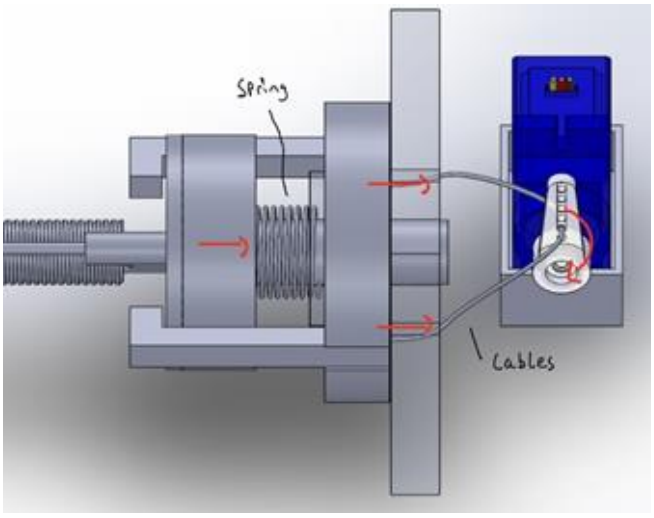


FIGURE 7. CONTROL OF LOCKING MECHANISM

The setting of the locking mechanism, i.e. whether the contact disk is touching the connector or not, is controlled by the system shown in Figure 7. In its standard setting, the spring pushes the contact disk and the connector together. To change the configuration of the mechanism to allow linear movement, the servo motor (SG90 micro servo 9G) pulls back the rods, compressing the spring and separating the contact disk from the connector.

We use an Arduino to control these changes, as well as to choose the degrees at which the disk is rotated. To send the commands to the stepper motor we use a motor controller, while the servo motor is connected directly to the Arduino's output pins.

3.2.3. HOUSING

The disks are only supported by the centrepiece and leadscrew when they are being individually addressed. Therefore, the disks require a structure to hold them in place before and after each rotation. This housing must accommodate the rotation of the disks, taking into account the uneven outer surface of the octagons. It must also be sturdy enough to resist the downward pressure force that users will exert when feeling the cells with their fingertips.

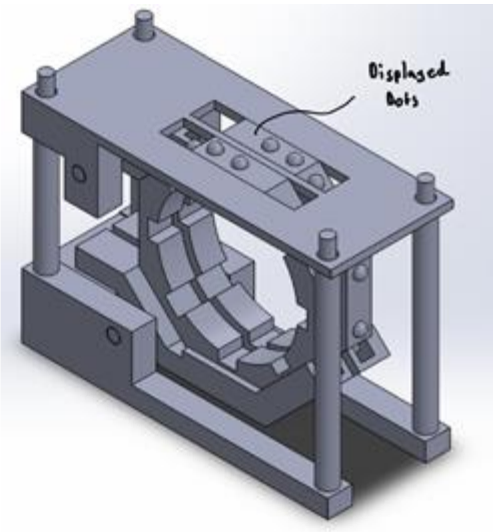


FIGURE 8. HOUSING DESIGN (BIRDS EYE VIEW)

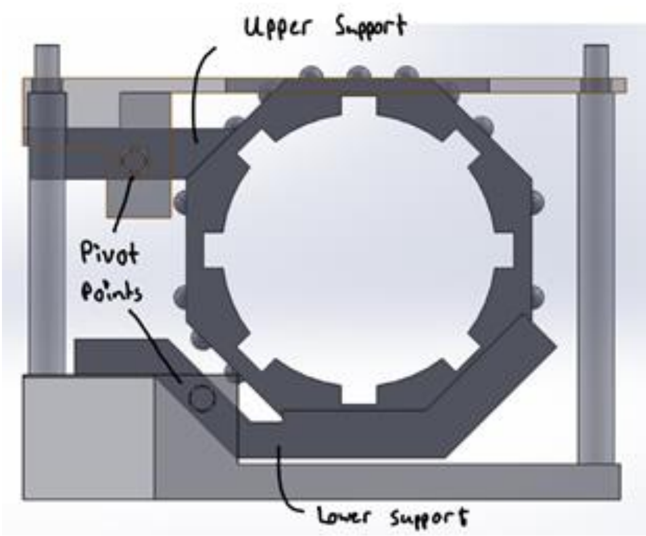


FIGURE 9. HOUSING DESIGN (SIDE VIEW)

The housing consists of two parts, one on the bottom and one on the top. Both of the supports are mobile, for they can rotate around a pivot point, and are held in place by an elastic band. The standard position for the disks, i.e. when in rest, is shown in Figures 8 and 9. The elastic bands do not exert a force strong enough to inhibit rotation, but they are able to hold the disks in place once rotation has happened. As for the outermost casing, the cut-outs on the top section allow the chosen dot combinations to be displayed. They also add an additional layer of support, further reducing any unwanted movements of the disks. Figure 10 shows the full assembly of all the components in a single line of the Braille display.

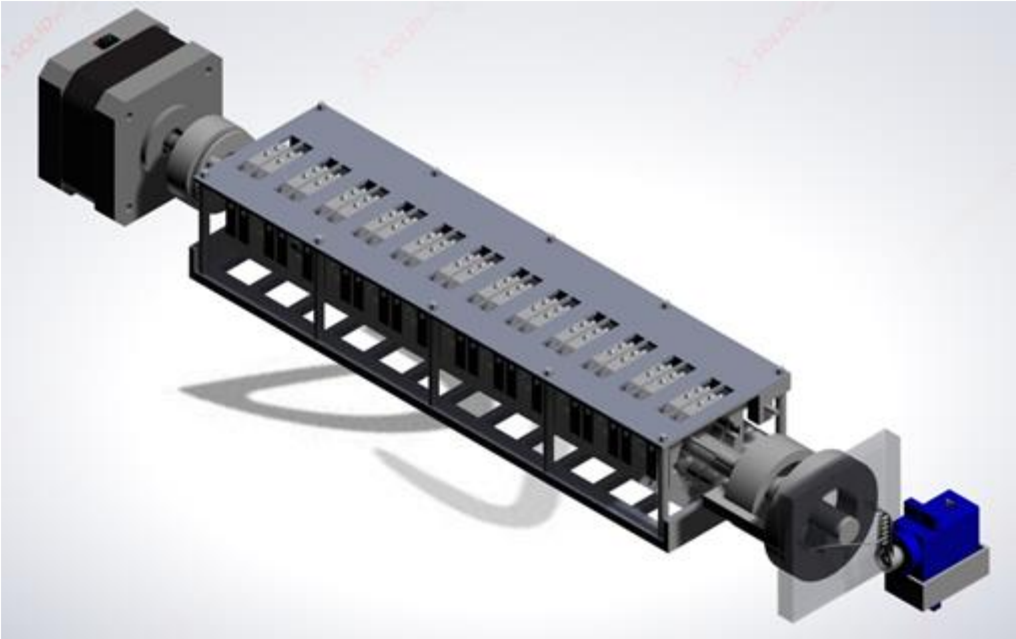


FIGURE 10. ONE LINE DISPLAY

3.2.4. INPUTTING A CHARACTER

The main input to the Braille display, as in similar products available on the market [1],[3],[4], is made up of six keys, each one representing a different dot of a Braille cell. Our current design requires the user to press the desired combination of keys, simultaneously with the *Set* and *Display* buttons (see Figure 12) to save a character into an array. After calculating the necessary rotation, the Arduino will send a command to the stepper motor, sequentially rotating the two component disks that make up a cell. The final position of each disk is stored for future reference. The device also displays the character on a 6-LED display, which allows quick feedback for sighted people[1]. This process can be repeated indefinitely to change and overwrite the character if a mistake has been made. When the user feels content with the current character, pressing the *Next* button (see Figure 13) will make the device focus on the following cell.

If time had permitted, this process could have been optimised by eliminating the *Set* and *Display* buttons. This could be done by adding a timer that starts once any of the input keys are pressed and recording which keys are engaged in this small interval. The input would then be automatically displayed. Another useful addition would be a *Previous* button, which would allow the user to go over the equation and fix past errors.

[1] This has been designed with a classroom setting in mind. The student can write and feel their progress while the teacher visually tracks the procedure.

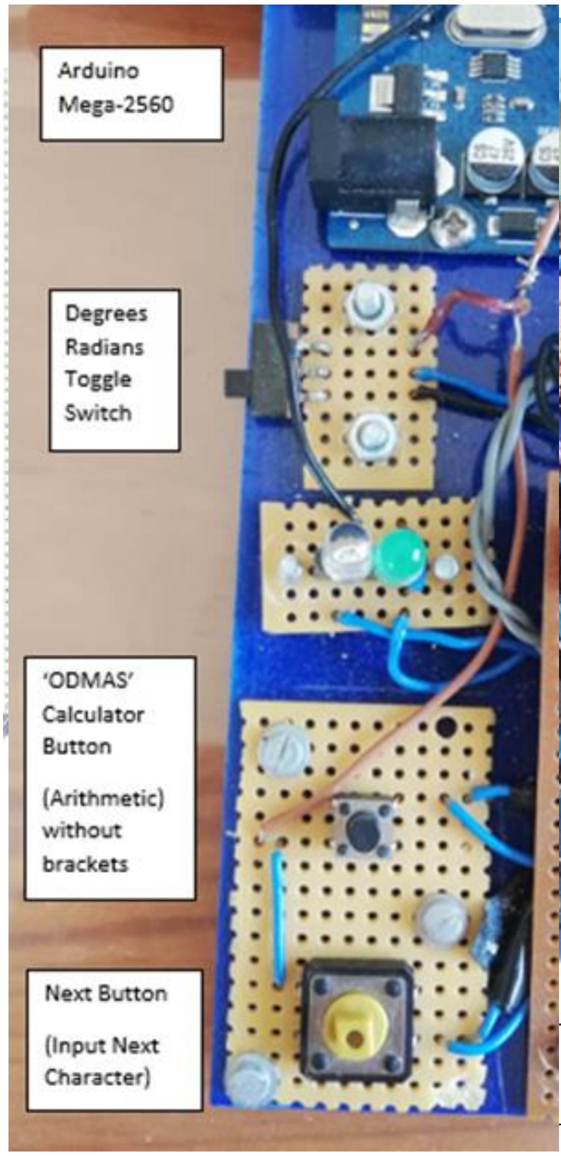
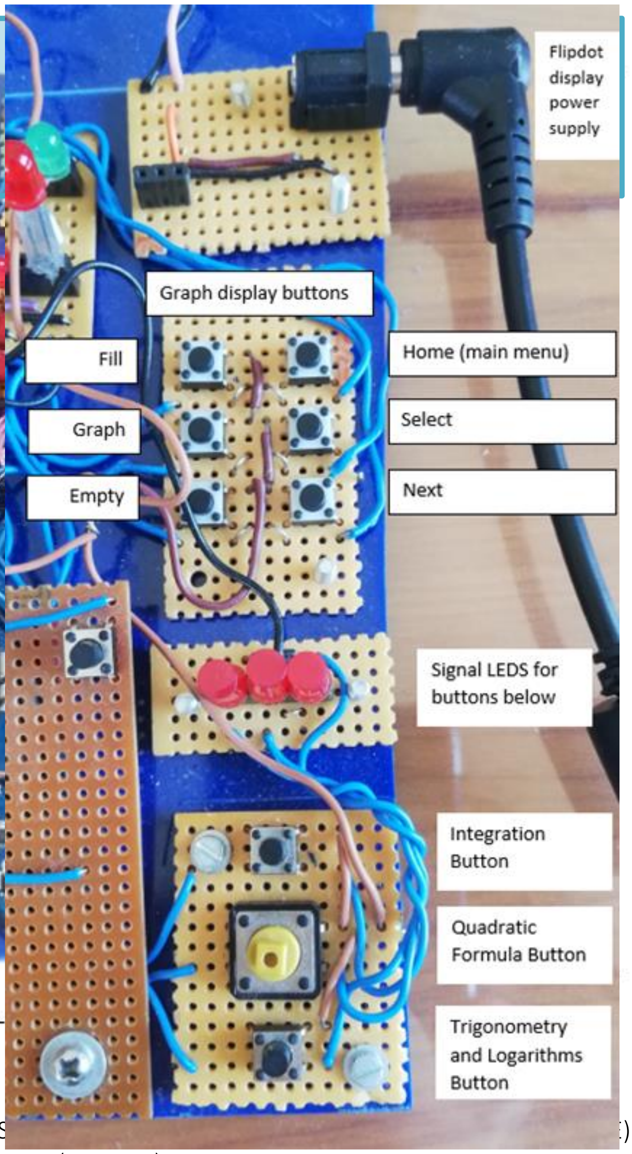


FIGURE 13. BUTTONS FOR CALCULATOR (LEFT SIDE)



19cm

FIGURE 14. BUTTONS FOR GRAPH (RIGHT SIDE)

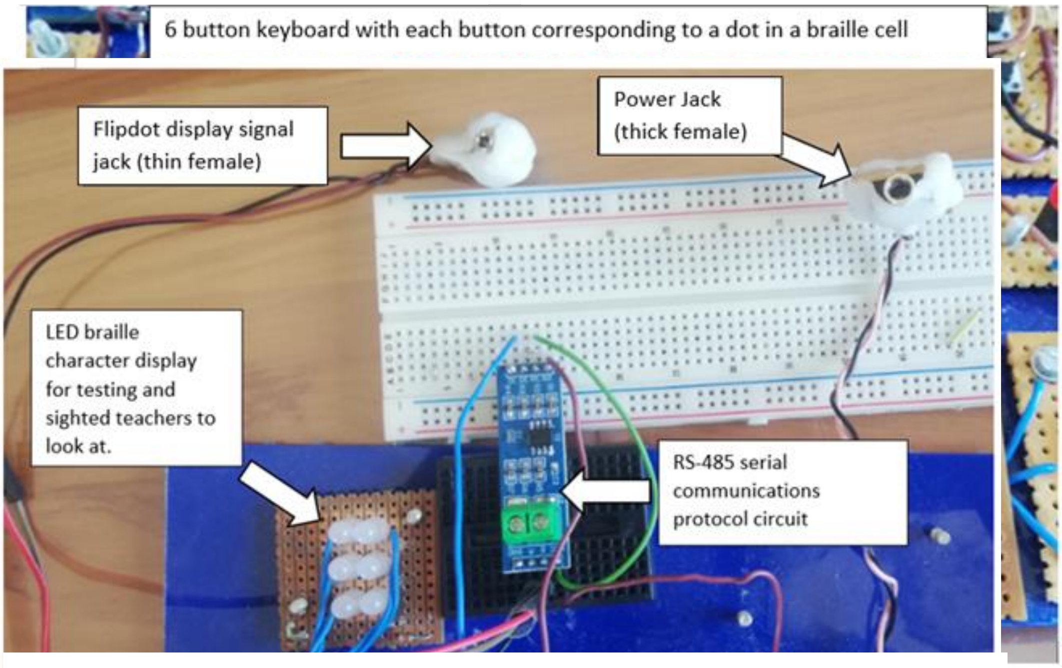


FIGURE 15. COMMUNICATION UNIT AND LED DISPLAY

3.2.5. CALCULATOR MODE

Our device has both read-write and calculator functionalities. To use the latter, the user has to press one of the *Functions* buttons, shown in Figures 13 and 14. This will take in the full equation that has been inputted, calculate the result and display it on the Brailler[2] (see Appendix E – Function Buttons). The input is also stored as a .txt file in the SD card attached to the Arduino. This allows the user to keep track of their working, since a new input will reset the display, allowing a new equation to be typed.

There are currently four different buttons that determine the type of calculation that the program will compute:

- *Simple arithmetic* (see Figure 13, 'ODMAS' calculator button)**:** this function takes in integers and is able to process a maximum of 5 operators. It follows the default order of operations (note that the 'modulo/remainder' operator has priority above 'power')[3], but does not recognise the use of brackets.
- *Quadratic Formula* (see figure 14)*:* this function uses the quadratic formula to calculate the roots of a quadratic polynomial in the form of $ax^2 + bx + c$. It internally determines the discriminant and can calculate both complex and real roots to a precision of three decimal places.
- *Trigonometry, Square Roots and Logarithms* (see figure 14): this function carries out basic trigonometric operations as well as square roots and logarithms. It also includes a toggle switch that allows the user to switch between radians and degrees. It can take in inputs of up to 3 decimal places.
- *Integration* (see figure 14)*:* this function evaluates the definite integral of a single power of x (e.g. $3x^2$) using the trapezium rule with 50 trapeziums[4]. The function can hold a total of 20 characters and best approximates narrow ranges and small values.

This is not an exhaustive list of possible calculations the user might input. Further improvements could include the use of decimal inputs and brackets in *Simple arithmetic* calculations, a larger range of computable integrals and the inclusion of derivative

---

[2] It is important to note that all calculations assume the omission of the Braille number sign (⠼). The program treats all operands as algebraic expressions.

[3] To compute the operation the program finds the 'heaviest' operator and then, in a recursive manner, puts the calculated result back into a now shortened equation until only one number remains, the answer.

[4] Following the formula for the trapezium rule: $\int_{x0}^{xn} f(x)dx = \frac{1}{2}h[(y0 + yn) + 2(y1 + y2 + yn - 1)]$ the number of trapeziums chosen is a compromise between accuracy and computing power required.

calculations. Another useful addition would be to integrate all the buttons into a single input, i.e. to create a program that can discern the different types of equations and consequentially carry out the appropriate computations.

3.3. GRAPH DISPLAY

The second part of our device consists of a graph display. Due to the restricted amount of time we had to develop our prototype, we adapted an old bus display - a 20 x14 flip-dot grid. To make it suitable for our project, we added tactile material to one side of each flip-dot. This way, we can program the device to display a range of graphs and shapes (the line is traced by a combination of dots flipped to the tactile side, while the empty space around the chosen shape is represented by the smooth side of the flip-dots). This is shown in Figure 16, where the yellow sides can be seen to have a red tactile dot.

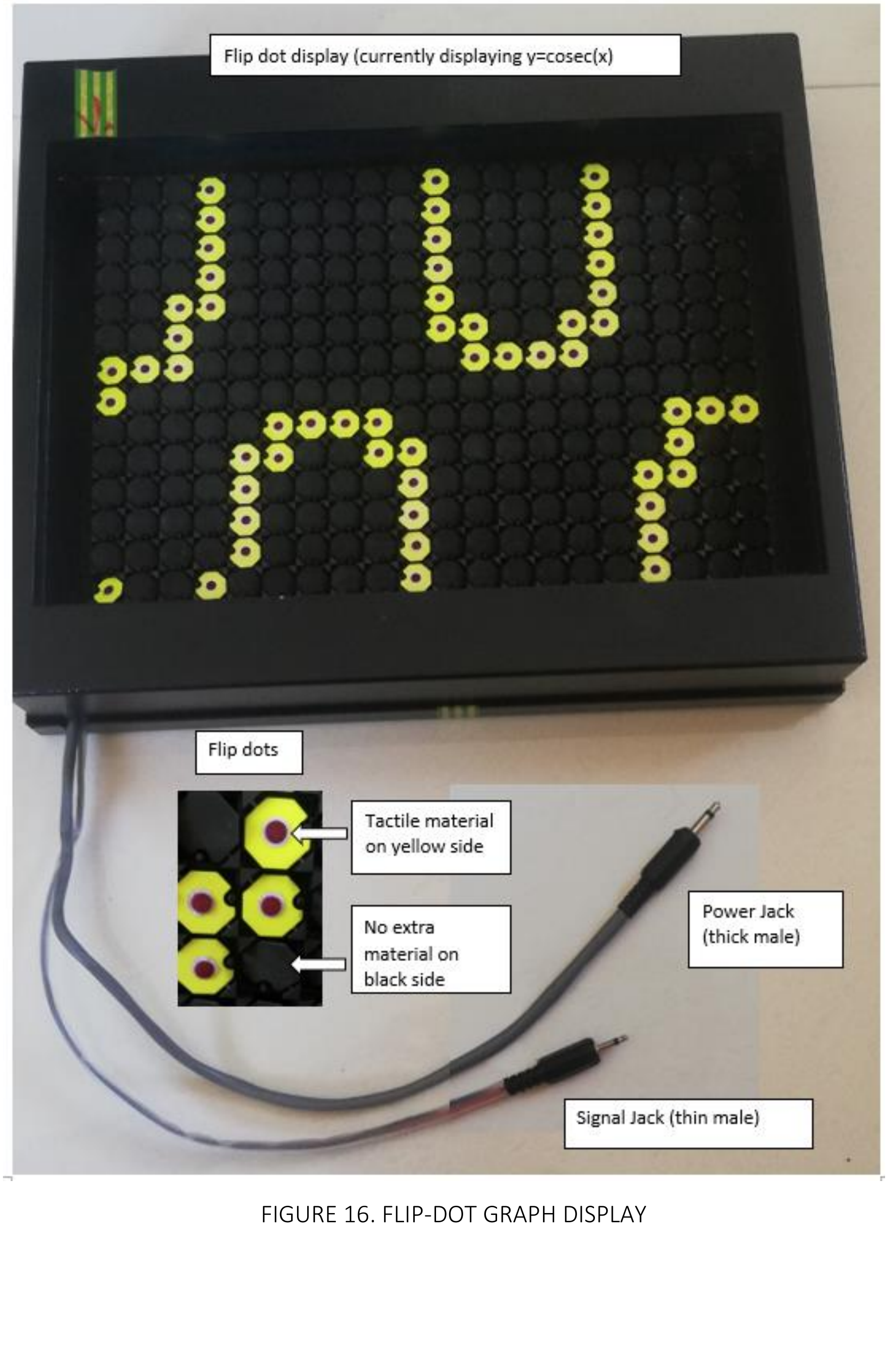


FIGURE 16. FLIP-DOT GRAPH DISPLAY

3.3.1. CONTROLLER - DISPLAY COMMUNICATION

The flip dot display uses a RS485 serial communication method [5] and device specific protocol [6]. To send information to the display, the signal from the Arduino board has to pass through a RS485 Arduino shield (Figure 15). The shield converts a unipolar digital signal (TTL), compatible with the Arduino) into a bipolar RS485 signal (compatible with the display), and vice versa.

Full description of the circuitry and the connections can be found in Appendix J. A short description and further reading on the communication protocol used by the display can be found in Appendix K. [5]

3.3.2. CHOOSING A GRAPH

There are six buttons that control the flip-dot grid. These are represented in Figures 14 and 17. The *Fill* and *Empty* buttons fully flip all the dots on the display to one state (*Fill* – tactile and *Empty* - smooth). The user can use these functions to clear displayed graphs and test for dysfunctional flip-dots.

To choose a graph to be displayed, the user must press the *Graph* button. This will initiate the menu selection process, where the *Next* and *Select* buttons allow the user to choose a given option. Current options are shown in Table 1. Each possible category and the subsequent options are communicated to the user using audio recordings. [6] Final selection will then display the chosen graph.

When the user wants to choose a new option, pressing the *Home* button will restart the selection process. It is important to note that the device automatically retains the last selection made, so the user should always press the *Home* button directly after *Graph*.

The graphing function currently incapacitates the Brailler display capabilities, i.e. you cannot use the Brailler display when the graph display is being used. This is not an important shortcoming since the user will probably concentrate on either one or the other. Nevertheless, future improvements should allow for independence of the displays, since using both might be useful in some cases (such as solving graph-related equations).

---

[5] It should be noted that the flip-dot display used imposed a specific communication protocol. The use of a different display in future designs would require new code and circuitry

[6] The current prototype does not have this feature integrated due to lack of time. Nevertheless, the menu buttons' functionality was trialed using an LCD display.

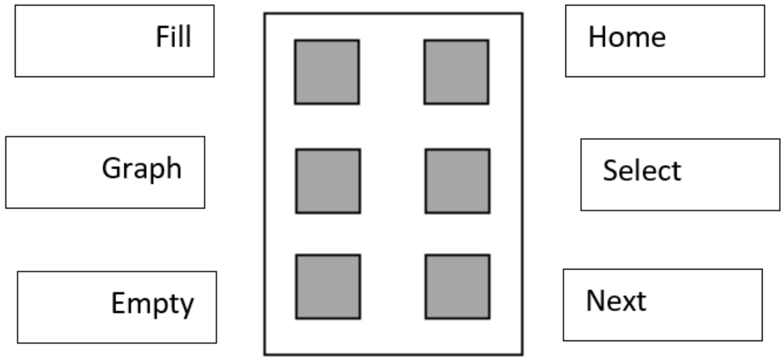


FIGURE 17. SELECTION BUTTONS REPRESENTATION

Graph Category	**Graph Options**
Sinusoidal	$sin(x)\ , cos(x), tan(x), sec(x), sin(2x)$
Polynomial	$x^2, -x^2, x^2 + x, x^2 + 3,\ x^3$
Linear	$x, -x, 2x, 4x, x + 3$
Conics	circle, ellipse, parabola, hyperbola, test_conic
Miscellaneous	$ln(x),\ e^x,\ \frac{1}{x}, \sqrt{x},\ \lvert x \rvert$

TABLE 1. GRAPHS DISPLAY MENU

```
. . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . .
. . X X X . . . . . . . . . . X X X . .
. . X . X X . . . . . . . . X X . X . .
. X X . . X . . . . . . . . X . . X X .
. X . . . X X . . . . . . X X . . . X .
X X . . . . X . . . . . . X . . . . X X
X . . . . . X . . . . . X X . . . . . X
X . . . . . X X . . . . X . . . . . . X
X . . . . . . X . . . . X . . . . . . X
. . . . . . . X X . . X X . . . . . . .
. . . . . . . . X X X X . . . . . . . .
. . . . . . . . . X X . . . . . . . . .
. . . . . . . . . . . . . . . . . . . .
```

FIGURE 18. GRAPH EXAMPLE - SINUSOIDAL

```
. . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . .
. . . . . . . . X X X X X . . . . . . .
. . . . . . X X X . . . X X X . . . . .
. . . . . . X . . . . . . . X . . . . .
. . . . . X X . . . . . . . X X . . . .
. . . . . X . . . . . . . . . X . . . .
. . . . . X . . . . . . . . . X . . . .
. . . . . X . . . . . . . . . X . . . .
. . . . . X X . . . . . . . X X . . . .
. . . . . . X . . . . . . . X . . . . .
. . . . . . X X X . . . X X X . . . . .
. . . . . . . . X X X X X . . . . . . .
X . . . . . . . . . . . . . . . . . . .
```

FIGURE 19. GRAPH EXAMPLE - CIRCLE

4. DISCUSSION

4.1. DESIGN EVALUATION

Our aim for this project was to create an electronic Brailler which enabled visually impaired children, otherwise unable to learn complex mathematics, to have access to STEM related disciplines. We wanted to make our device affordable and accessible, balancing functionality and precision with cost and weight. Our final design consists of two separate displays that allow for extensive functionalities. It meets many, but not all, of the user requirements.

An important specification was to keep the cost of the device under £200 to make it more accessible than other Braillers. The prototype cost considerably more than the set price (see the Bill of Materials in Appendix C), but we believe that bulk production would significantly reduce this number. Firstly, we bought an old bus display and its controller, both adding up to £180. Making our own flip-dot grid would eliminate this cost and allow for a larger amount of customisation of the display[7]. It is also important to note that many components were 3D printed. This makes the parts easily replicable and low-cost, although future manufacturing would require a more precise 3D printer to ensure correct functioning and uniformity of all Braillers.

3D printing also creates lightweight components, an important characteristic that contributes to the portability of the device. Nevertheless, other components such as the stepper motor and the graph display are very heavy and their dimensions are too large to be easy to carry. The future use of a custom-made flip-dot display would allow us to choose the ideal dimensions, balancing the portability and precision of the display.[8] The finalised Brailler would also require the use of a smaller stepper motor, which would significantly decrease both the size and the weight of the device. Another aspect that decreases portability is the need for a mains power supply. This could be easily solved by adding a battery, at the cost of an increased weight of the product.

It is also worth pointing out that our design allows for both rotation and linear motion with the use of a single stepper motor. We chose to use a stepper motor to obtain a high amount of precision, but the use of two motors, one for rotation and one for linear motion, would have added considerably to the price, weight and size of our Brailler. To eliminate the need of this second motor we came up with a locking mechanism, described previously (see

[7] As previously mentioned, we did not attempt to build our own flip-dot grid due to the time constraint. Using already available parts allowed us to prove the concept.
[8] A larger display increases the resolution but decreases the portability.

section 3.1.2). This requires the addition of a servo motor, but it is not a large inconvenience, for it is much smaller, lighter and cheaper than the stepper motor.

Another requirement set by KBT and reinforced by the feedback received from students at Joseph Clarke School was to have more than a single line display, since this is the main limiting factor for students trying to learn math. It is important to note that there are multi-line Braillers on the market [7] but the price increases sharply as you increase the number of lines. We wanted our design to provide more space to allow for complex calculations, but we wanted to avoid making the device too expensive or bulky. We settled on a four-line display as this provides much more working space than students currently have while keeping the price of our product lower than other one-line Braillers.

The user requirements also stated that our Brailler should be able to compute A-level mathematics [8]. The device can currently perform functions that range from simple arithmetic to logarithms and integration. This does not yet exhaust the list of functionalities needed, but it does reach the level of difficulty required, which proves that the Brailler could fulfil the requirements in the near future. Further improvements could include decimal and negative inputs for all functions, complex number manipulation, algebraic manipulation, and the ability to process integrands that are more complex.

As for the graph display, it can currently display trigonometric, logarithmic and conic graphs, reaching GCSE standards [9]. To reach the full desired capabilities, we would need to add a more complex program that can read and display any function that the user inputs. Other improvements could include scrolling and zooming functionalities, displaying the axes or finding minimums/maximums. [9]

Another specification was the input format. Our Brailler adheres to the widely used six key input, with a space bar that represents an empty cell. SD card reading and writing is not included in the current design. We wrote the appropriate code for Arduino (see Appendix I) and tested the functionality, but we did not have enough input pins in the processor we were using. Future improvements would therefore have to include a different Arduino board, a combination of Arduinos or a different hardware altogether.[10]

As we have already mentioned, the time constraint on this project impeded us from building a complete prototype. It also meant that we had to settle for a much simpler device than we would have liked. There is a wide range of functionalities that we aim to include in the future, which are outlined in Appendix G.

---

[9] All these new functionalities would require a larger precision given by the custom-made display mentioned previously.

[10] Using a more powerful processor would also increase computational speed

4.2. TEAM DYNAMICS

In the duration of this project all members showed interest and commitment, contributing to a good team dynamic and pleasant work environment. Weekly meetings always had full attendance, so they allowed us to analyse each person’s progress, discuss their implications with respect to the final design and plan our next steps. All tasks were first evenly divided and then distributed, taking into consideration each member’s strengths and interests. This methodology had an especially positive impact on extensive assignments, for example when writing the script for our presentation, since it broke down a difficult and time-consuming task into manageable chunks.

One aspect that we would have liked to approach differently is the project timeline planning. We feel that our Gantt chart was created too late and we did not follow it in a strict manner. We were very tight on time when developing the prototype, so had we done the planning sooner and adhered to all proposed deadlines, we might have been able to obtain a more complete product. In conclusion, we are highly satisfied with the Brailler that we have designed, and we believe that the hard work we have put in and the cooperation between team members is reflected in this project.

REFERENCES

[1] RNIB. (2018). *Introducing the Orbit Reader 20.* [online] Available at: <https://www.rnib.org.uk/Orbit-Reader-20> [Accessed 14th June 2020].

[2] 2017. *Standard Dimensions for the UK Braille Cell.* [ebook] UK Association for Accessible Formats, p.2-4. Available at: <https://www.ukaaf.org/wp-content/uploads/BrailleStandardDimensionsFinal.pdf> [Accessed 14th June 2020].

[3] Perkins Solutions. (n.d.). *Perkins Brailler.* [online] Available at: <https://brailler.perkins.org/pages/perkins-brailler> [Accessed 14th June 2020].[11]

[4] Sight and Sound Technology. (2010-2017). *Smart Beetle.* [online] Available at: <http://www.sightandsound.co.uk/smart-beetle.html> [Accessed 14th June 2020].[12]

[5] 2008. *Interface Circuits for TIA/EIA-485 (RS-485).* [ebook] Texas Instruments Incorporated. Available at: <https://www.ti.com/lit/an/slla036d/slla036d.pdf?HQS=slla036-aaj&ts=1592159430190&ref_url=https%253A%252F%252Fwww.google.com%252F> [Accessed 14th June 2020].

[6] 1994. *RS-485 Display Communication Protocol*. [ebook] Hanover Displays Ltd. Available at: <https://www.dropbox.com/s/11cyeker6kh2yvw/proto_20131210142804.pdf?dl=0> [Accessed 14th June 2020].

[7] Brauner, D. (2018). *Bristol Braille Canute: Multi-Line Refreshable Braille*. [online] Perkinselearning.org. Available at: <https://www.perkinselearning.org/technology/posts/bristol-braille-canute-multi-line-refreshable-braille> [Accessed 14th June 2020].

[8] 2020. *Syllabus Cambridge International AS & A Level Mathematics 9709*. [ebook] University of Cambridge Local Examinations Syndicate, p.7. Available at: <https://www.cambridgeinternational.org/Images/415060-2020-2022-syllabus.pdf> [Accessed 14th June 2020].

[9] 2020. *Syllabus Cambridge IGCSE Mathematics 0580.* [ebook] University of Cambridge Local Examinations Syndicate, p.14-17. Available at: <https://www.cambridgeinternational.org/Images/414416-2020-2022-syllabus.pdf> [Accessed 14th June 2020].

[11] Braillers have had (at least) 6 keys since their inception.
[12] It should be noted that many Braillers have more keys for added functionality.

APPENDIX A – PROJECT MANAGEMENT

In the early stage of the project, important roles were assigned:

- A project manager to organize, overview, delegate and be a point of contact for official communications
- A manufacturing manager to organize the manufacture process and interacting with technicians
- A procurement manager to purchase items
- A secretary to keep record of progress and monitor how deadlines are approached and met.

Independently of the roles, every member conducted an introductory research, both on blindness as an impairment and on electronic Brailler technologies. This helped us become familiar with the relevant topics related to our project and to define the key features that the device must have.

The team has relied heavily on user feedback: four members visited a school for the visually impaired and talked to students and teachers to define the most relevant user requirements.

After conducting some more research and creating a plan of action (the Gantt Chart outlining our planning is shown in Figure 20), we decided to create a device that was composed of two separate parts: a graph display and a Braille display with dual functionality (read/write and calculator). Therefore, we decided that the most efficient way to move forward was to split the group into two subgroups, each one specializing in a part of the device.

Further along the process, both groups overlapped. We had to make sure that a single processor was able to control both displays and focus our attention on the areas of the prototype that turned out to be the most complex.

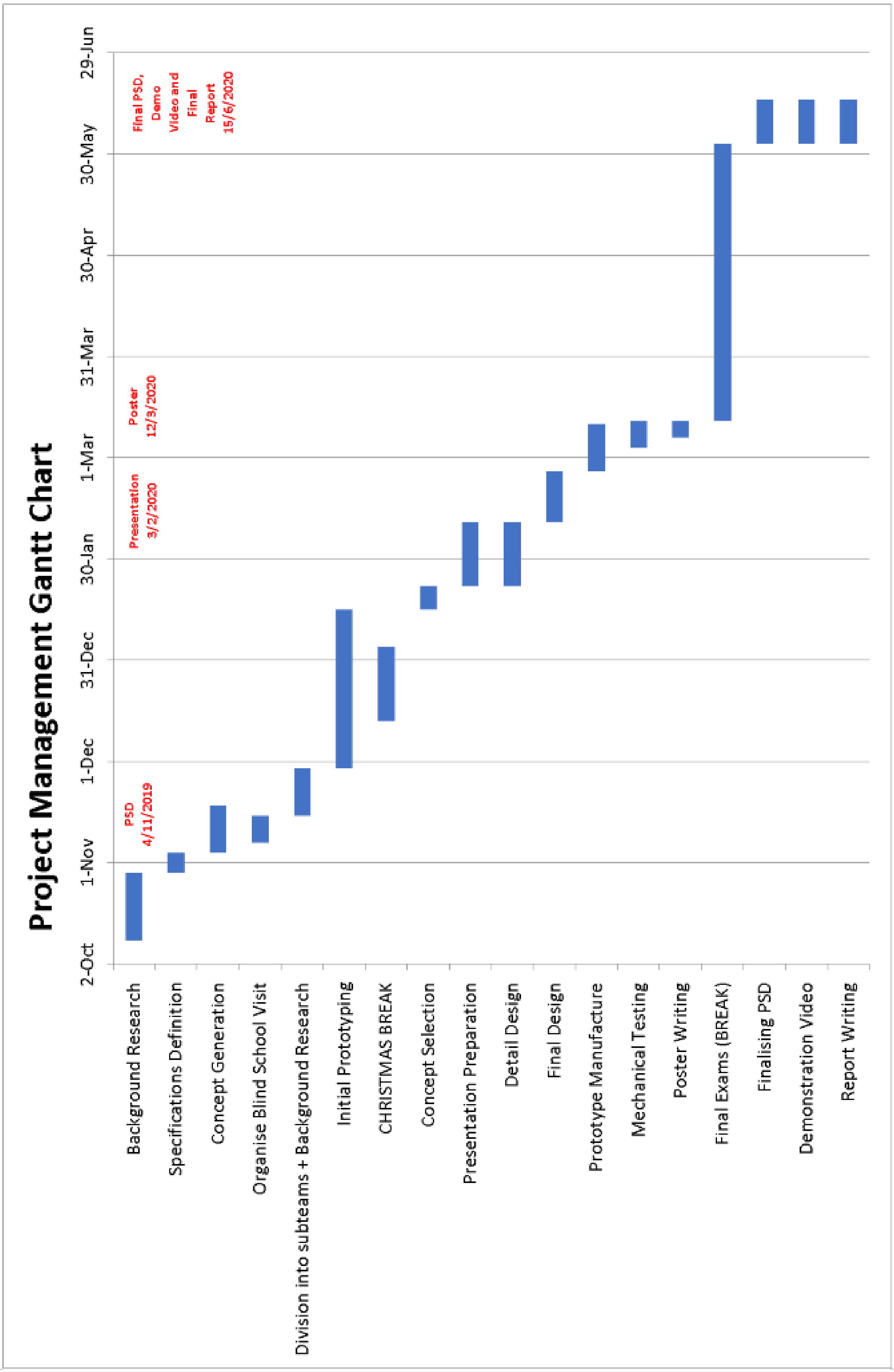


FIGURE 20. GANTT CHART

APPENDIX B – RISK MANAGEMENT

CRITICAL RISK PRIORITY NUMBER

During risk analysis each risk or failure is analysed and rated with respect to its severity (S), probability of occurrence (O), and detection rate (D). The rating for each of the three aspects ranges from 1 (low security risk/failure, low probability of occurrence, high detection probability) to 10 (severe injuries or death, high probability of occurrence, no/low probability for detection). The product out of these three ratings is called Risk Priority Number (RPN). In the cases where the RPN is greater than a critical threshold preventing measures (reasonable and justifiable) are required in order to lower this number and obtain a final RPN lower or equal to the critical threshold. We set our threshold to 75.

FACTORS OF THE RISK PRIORITY NUMBER (RPN)

Find below a recommendation how to rate occurrence, severity, and detection. The "Risk Priority Number before" is a mathematical product of the numerical Severity- (S), Occurrence- (O), and Detection-Ratings (D) obtained before applying any preventing measures to reduce the likelihood for dangerous incidents, thus: RPN before = (S1) x (O1) x (D1). This "RPN before" should be set to prioritize items that require additional quality planning or action.

The "RPN after" is a mathematical product of the numerical Severity- (S), Occurrence- (O), and Detection-Ratings (D) obtained after applying the preventing measures to reduce the likelihood for dangerous incidents, i.e. RPN after = (S2) x (O2) x (D2). The "RPN after" has to be equal or below the predefined threshold in order to guarantee safe use of the part/element/device.

Preventing measures are mechanisms that prevent the cause of the failure mode from occurring or that detect the failure and stop the application before an incident can happen. It could also reduce the severity by e.g. designing softer and rounder edges. Preventing measures could include specific inspection, testing or quality assurance procedures; selection of other components or materials; de-rating; limiting environmental stresses or operating ranges; redesign of the item to avoid the failure mode; monitoring mechanisms; performing preventative maintenance; or inclusion of back-up systems or redundancy.

S – Severity

Rating S	Criteria: Severity of effect	Consequence	Treatment
10	Death	-	-
9	Quadriplegia	Life-long medical care necessary / coma / permanent damage	Hospital stay
8	Amputations, paraplegia, blindness, deafness, traumatic brain injury (severe), fourth-degree burns	Life-long medical care necessary / coma / permanent damage	Hospital stay
7	Complex fractures, open fracture, inner injuries, traumatic brain injury (severe), third-degree burns	Permanent damage possible	Hospital stay
6	Gash, fractures, torn muscles, articular cartilage injury, traumatic brain injury (moderate), second-degree burns	Permanent damage possible	Hospital stay
5	Gash, fractures, torn muscles, articular cartilage injury, traumatic brain injury (mild), second-degree burns	Reversible injury	Hospital stay or ambulant treatment
4	Severe cuts, severe scratches, severe contusions, strains, first-degree burns	Reversible injury	Ambulant treatment or self-treatment
3	Minor cuts, minor scratches, minor contusions, stiff muscles, tension, blisters, excoriations, sickness, first-degree burns	Discomfort during application up to three days after application	Self-treatment
2	Slight sickness, pressure marks	Discomfort	-
1	No harm	-	-

O – Occurrence

Rating O	Criteria: Probability of occurrence
10	Occurs or may occur very likely during every use of the session
9	Occurs or may occur likely during every use of the session
8	Occurs in 1 of 5 sessions (less than once a day)
7	Occurs in 1 of 10 sessions (less than once a day)
6	Occurs in 1 of 50 sessions (less than once half a month)
5	Occurs in 1 of 100 sessions (less than once a month)
4	Occurs in 1 of 500 sessions (less than once half a year)
3	Occurs in 1 of 1000 sessions (less than once per year)
2	Occurrence very unlikely
1	Occurrence nearly impossible

D – Detection

Rating D	Criteria: Likelihood of detection by design control
10	No chance of detection
9	Very remote chance of detection
8	Remote chance of detection
7	Very low chance of detection by indirect methods (hardware or software)
6	Low chance of detection by indirect methods (hardware or software)
5	Moderate chance of detection by indirect methods (hardware or software)
4	High chance of detection by indirect methods (hardware or software)
3	High chance of detection by direct or indirect methods (hardware/software)
2	Direct and indirect detection: Hardware or software
1	Direct detection: Hardware or safe software (category 4, performance level e)

RISK ANALYSIS

Assembly	Failure & Effect	S1	O1	D1	RPN before	Preventing measures	S2	O2	D2	RPN after
Internal circuitry of the device	If not perfectly insulating, in case of problems in the internal circuitry, user get electric shock.	7	2	8	**112**	Using a perfectly insulating material for the casing.	4	2	8	**64**
Sharp edges of casing	It could harm the users if sharp in some sides.	4	3	4	**48**	Making the casing externally safe with rounded sides and no sharpness at all.	-	-	-	**-**
Casing	If it accidentally opened, it could harm the users as some internal parts are sharp and the visually impaired could have additional difficulties in identifying were the sharp components are.	4	3	2	**24**	Making the casing as well-welded as possible.	4	1	2	**8**
Unsafe materials	An interference in the circuitry could spark a small fire, which would have serious consequences if the casing were made of flammable material.	6	2	4	**48**	Using a flame-retardant material for the casing.	-	-	-	**-**
Bulk of the device	Device mishandling (e.g. device falls onto the ground) could accidentally cause injury to user.	3	5	4	**60**	Adding adhesive material to the bottom of the device and using non-slip material for the casing.	3	3	4	**36**

Cables	Loose cables causing trip or snag hazards.	2	2	4	**16**	Cables all secured to framework with ties.	-	-	-	-
Small parts	There are many parts which are of a size where they pose a choking hazard if ingested.	10	1	10	**100**	Ensuring a sighted individual is always present when user is using the device.	10	1	2	20
Heated Parts	If the device malfunctions it may reach harmfully high temperatures.	3	2	10	**60**	Have an audio warning message to invite user to shut down at 50 ºC, at some point shutting down automatically at 60 ºC.	-	-	-	-
Rotating elements	Long hair could get trapped in the rotating octagon mechanism.	3	4	3	**36**	Issue audio warning with device and 3D print part so decrease size of gap between octagon disk and casing.	-	-	-	-

TABLE 2. RISK ANALYSIS TABLE

APPENDIX C – ETHICS

The mathematical Brailler meets the moral criteria necessary to be introduced in a class environment, as well as to be launched on the market. In terms of the realization process, there are no issues of plagiarism. All technologies used have been referenced and the device itself is a product of the team.

For what concerns the intellectual property and the confidentiality of the Brailler, users are able to connect SD cards to their devices. The data contained in the SD cards can be read by the Brailler, but it is not shared to any external device. None of the programs that have been coded allow unauthorized usage of the device, since there is no dependence on external sources. Further improvements could include the use of Bluetooth, but this would still be a safe functionality, for connections would require authorization from the user.

The Brailler can be used to calculate results of mathematical operations and equations, note the students' workings when solving problems autonomously or read and write data from an SD card. It is therefore comparable to a calculator and an exercise book, a tool that aids graphical visualization and written calculations. This makes it suitable to a class environment, both when students can use a calculator and when they are required to show the working out of exercises.

The two functions of the Braille display (read/write and calculator) are currently controlled by a switch, which cannot be locked on one mode. This implies that this device could not be used in an exam setting where the use calculators is prohibited. However, further improvements could include the addition of a Bluetooth module to the Brailler display. A complementary mobile app on the teacher's phone could then connect to the Brailler and track which mode is being used[13]. This would make it possible for visually impaired students that use this device to be tested in equal conditions to the rest of the class, guaranteeing non-discrimination.

As for environmental concerns, the Brailler is made up of mainly metal components, but there is also a significant number of plastic pieces (all the 3D printed components and the flip-dots of the graph display). This could compromise the requirements for an environmentally friendly device, but the issue could be easily solved by replacing the plastic pieces with components made from recyclable material.

---

[13] This functionality was tested and proven to be feasible. We created a simple app that connected to our Arduino and managed to send information between the devices. We did not include this in our final design because, due to the time constraint, we only managed to include a very basic functionality

APPENDIX D – BILL OF MATERIALS

The list below depicts the cost of each purchased item (in GBP). It is important to note that the Hanover DERIC controller and the Hanover data cabling were used solely for testing and so did not add to the manufacturing cost of the prototype. Furthermore, the aluminium components had to be bought in bulk although only one round and one square bar were used.

DESCRIPTION	PRICE (GBP)
Servo Motor SG90	2
300mm Effective Travel Linear Slider with Nema23 Stepper Motor CNC Ball Screw Linear Slider	111.67
3D Printer 100-600mm 8mm Threaded Rod Lead Screw Anti-Backlash Nut Kit	4.87
Aluminium round bars (5) and square bars (5)	36.48
Cemobile 3PCS Solderless Breadboard Prototype PCB Board MB-102	7.59
Qumox 16GB SDHC Class 10 Memory Card (2)	7.00
Kwmobile SD Card Reader Module for Arduino (2)	5.99
PLA 3D printing filament	5
Flexo Compressible Spring C2-2737-2M	1.05
Hanover data cable cabling-controller to sign-LED flip dot bus destination	14.99
Hanover R014C 20x14 Flip Dot Bus Coach Destination Display sign home office 24V	124.99
Hanover DERIC destination display Controller LED flip dot bus coach sign RS232	54.99

TABLE 3. BILL OF MATERIALS

APPENDIX E – FUNCTION BUTTONS

SIMPLE ARITHMETIC

Braille	Operation	Symbol
⠤	Subtraction	-
⠬	Addition	+
⠡	Multiplication	*
⠌	Division	/
⠘	Power	^
⠴	Modulo (remainder)	%

	Input	Output	Special output characters
Negative numbers	No	Yes	⠤ (negative marker)
Decimal numbers	No	Yes	⠅ (decimal point)
Multi-digit numbers	Yes	Yes	

Example calculations		Answer	
2 + 2	⠃⠬⠃	4	⠙
32 + 7^3	⠉⠃⠬⠛⠘⠉	375	⠉⠛⠑
3 - 2	⠉⠤⠃	-1	⠤⠁

3 / 2		1.5	

Quadratic Formula

Input format:	$ax^2 + bx + c$	a • b • c • [14]
Output format:	X =, X=	

	Input	Output	Special output characters
Negatives	Yes	Yes	negative
Decimals	No	Yes	decimal
Multi-digit Numbers	Yes	Yes	
Complex Numbers	No	Yes	Imaginary unit

Example calculations		Answer	
$x^2 + x - 6$		x = 2 x = -3	
$x^2 + x - 1$		x = 0.618 x = -1.618	

[14] Input requires 3 left sided separator points •

$x^2 + 1$	⠁⠂⠚⠂⠁⠂	x = +-0.618i	⠭⠿⠬⠤⠁⠹
$x^2 + x + 7$	⠁⠂⠁⠂⠛⠂	x =-0.5 +-2.598i	⠭⠿⠚⠨⠑⠬⠤⠃ ⠨⠑⠊⠓⠹

TRIGONOMETRY, SQUARE ROOTS AND LOGARITHMS

Functions	
sin	⠎⠊⠝⠐
cos	⠉⠕⠎⠐
tan	⠞⠁⠝⠐
asin (arcsin)	⠁⠎⠊⠝⠐
acos (arcos)	⠁⠉⠕⠎⠐
atan (arctan)	⠁⠞⠁⠝⠐
sqrt (square root)	⠎⠟⠗⠞⠐
log (In natural logarithm)	⠇⠕⠛⠐

The function requires a right marker dot ⠐ at the end of each declaration for the device to identify the number that follows

	Input	Output
Negatives	Yes	Yes
Decimals	Yes	Yes
Multi-digit Numbers	Yes	Yes

Example calculations		Answer	
log 2		0.62	
sqrt 2		1.414	
sin 30 (deg)		0.5	
sin 0.524 (rad)		~0.5	
asin 0.5 (deg)		30	
asin 0.5 (rad)		0.524	

INTEGRATION

Input format:	∫f(x)dx Limits: a - b	( ) a b f(x)
Output format:	Single Real Number	

	Input	Output
Negatives	No	Yes
Decimals	No	Yes
Multi-digit Numbers	Yes	Yes

Special output characters				
( ) Integral Sign (Optional)	Lower Limit	Upper Limit	Integrand Start (Right marker dot)	Integrand End (Left marker dot)

Example calculations		Answer	
$\int x dx$ Limits: 2 - 1		1.5	
$\int x^3 dx$ Limits: 2 - 1		3.75	
$\int 3x^2 dx$ Limits: 10 - 8		448	
$\int 1/x dx$ Limits: 2 - 1		0.693	

APPENDIX F – EVALUATION PLAN

An evaluation plan was made to verify whether the following features, in decreasing order of importance, had been met: safety, functionality, completeness, resistance and efficiency. Due to timing constraints, not all these aspects have been evaluated.

Safety is the top priority and a detailed risk analysis has been conducted to maximise it. See Appendix B – Risk Assessment.

The next relevant feature is functionality. We aimed to prove tactile detectability of the pins with user feedback, and hoped to test the device in a real setting by asking visually impaired students at Joseph Clarke School to use our Brailler in one of their mathematics classes. We did not have time to carry out this testing, but we did test the detectability of the flip-dots in the laboratory. We asked non-visually impaired subjects to use the device and they obtained highly accurate results. This suggests that our Brailler would be easily readable to a more expert hand, like visually-impaired individuals who have learnt to distinguish Braille writing with their fingertips. Furthermore, electrical circuits and coded programs have been assessed with simulations in the lab and using black box testing via IDEs.

For completeness, we determined that the device should be able to compute a wide range of equations, read and write lengthy and complex mathematical workings from and into an SD card and display a large number of graphs and shapes. We were able to test the capabilities of both displays described in the main text, but these are not extensive enough to account for completeness. Being unable to include the SD read/write functionalities in the prototype also negatively impacted this feature. Future testing would require exhaustive trials that demonstrated adequate functionality of all computations.

The

resistance of the device could not be assessed. Planned evaluations included testing under extreme conditions: mechanical resistance tested under pressure and thermal resistance tested at high and low temperatures.

Finally, we did not manage to assess the efficiency of our Brailler. This evaluation would require time measurements for every possible action of the device: mathematical computations, rotation of orthogonal disks to set a Braille cell, graph displays, SD card-Brailler interactions, etc.

It is worth noticing that completeness, resistance and efficiency are features that should be evaluated once the device is built and fully functioning. Since we did not manage to create a finalised product, it was not possible to analyse them in this project.

APPENDIX G – FURTHER IMPROVEMENTS

There are a few drawbacks to our current design which we would like to address and improve on future models. In the Braille display, each disk is individually addressed and so the time needed to display a line of writing is significant. While it is possible to start reading the content while disks further down the line are being set, this is not idea. Stepper motors create a lot of vibrations and might disrupt the reading process. One way of minimising the display time would be to fine tune communication with the locking mechanism to make the process more efficient.

Another shortcoming can be found in the housing of the octagonal disks, for they are currently not very precise. In our prototype the support pieces have been 3D printed and so are prone to having small imperfections. Future improvements could include the use of manufactured pieces, a higher quality 3D printer or an improved design that allows for a larger error without compromising the functionality of the device.

As previously stated, our prototype could not be completed due to time constraints[15]. An aspect that we have not yet focused on is the casing of the device. The full prototype would have included a metal or plastic casing for the Braille display that housed all of its components. This would provide stability, increased safety and improved aesthetics. Following are a number of improvements that assume a fully functioning device.

The introduction of a more powerful microprocessor (with respect to the Arduino we are currently using) would allow the implementation of more complex mathematical computations. Examples include a wider range of integrations, the possibility of decimal and negative inputs for all functions and algebraic manipulation.

The Braille display could be modified to minimize its dimensions. We would have to use smaller components that maintain the same functionality as their larger counterparts. This is particularly relevant for the stepper motors. Additionally, other materials that are more user-friendly and durable could be used.

As for the graph display, possible future improvements include: maximizing graph resolution, introducing more types of graph, options to zoom in or out when needed, increasing graph accuracy or adaptable axis. All of these would be ideally implemented on a custom-made flip-dot display, which would give us more control on the dimensions, resolution, weight and cost of the device.

Finally, we would like to create an online database of exercises arranged according to difficulty and topic. Adding audio feedback to our device would make this functionality more user-friendly.

[15] Not only due to the limitations of a year-long project, but also due to not being able to access the laboratories as a result of the COVID-19 pandemic

APPENDIX H – INITIAL PROTOTYPES

Before deciding on the final design of the Brailler we came up with and tested a number of different mechanisms, discarding each one for different reasons. Our prototype has three interfaces – the keyboard, the Braille text display and the graph display. The keyboard design did not change from the first proposal, for we adhered to the generalised six key input.

In terms of the Braille display, the only significant modification we implemented was the shape of the disks, which were initially circular (see Figure 21). This first design did not have a clear separation between the eight possible three-dot combinations. Instead, each disk had a continuous surface of dots and “holes”, such that each dot/hole unit could be used to display three different combinations. The change from circular to octagonal allowed for a much clearer switch between displayed combinations. Another concern was the effect of a round surface in terms of tactile detectability, which we believe would have led to confusions when reading.

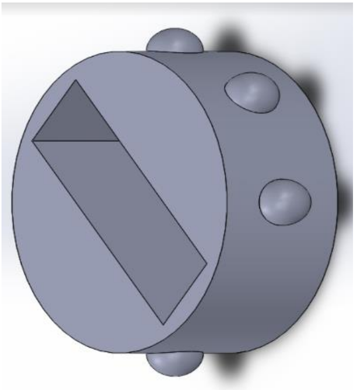


FIGURE 21. CIRCULAR DISK

The design of the graph display, on the other hand, underwent many modifications. The initial idea was to create an array of invertible silicone dots. A mechanical nozzle would raise the desired dots, while an open pressure valve would be used to reset the display.

We tried to create these silicone dots in various ways: silicone sealant mixed with liquid soap (see Figure 22), 2 parts silicone clay with PCL mould (see Figure 23), and 3-D printed “sandwich” mould filled with liquid silicone (see Figure 24).

FIGURE 22. SILICONE SEALANT MIXED WITH LIQUID SOAP

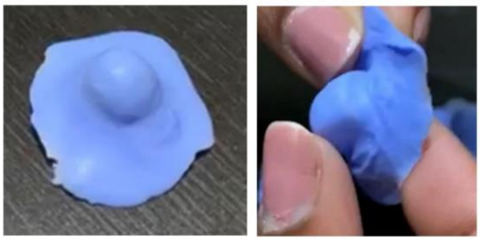


FIGURE 23. 2 PART SILICONE CLAY WITH PCL MOULD

FIGURE 24. LIQUID SILICONE WITH 3D PRINTED MOULD

The 3-D printed mould allowed us to produce silicone dots of standard dimensions, however, the liquid silicone was too flimsy and tore after few trials of exerting a minimum downward reaction force. The dots were also hard to detect with the fingertips. One of the main reasons for discarding this solution was the time constraint. Our major concerns included researching for other possible flexible material, finding the appropriate thickness of each dot (they have to be easy enough to flip but strong enough to stay in position when a small downward reaction force is exerted by the user), and determining the nozzle’s mechanism for flipping the dots. This led us to settle for the flip-dot mechanism, since it was much less time-consuming and had a lower risk of failure, and ultimately allowed us to obtain a functioning device that successfully displays a range of functions and shapes.

APPENDIX I – ARDUINO CODE

All the functionalities of our Brailler mentioned throughout the text have been compiled in a single file. Following is a link to the Arduino code:

https://imperiallondon.sharepoint.com/sites/EDP-MathsBraillergroup5-BM/Shared%20Documents/General/arduinocode.updated.txt

APPENDIX J – CONTROLLER-DISPLAY INTERFACE

The connections between the flip-dot display, the Arduino Mega board and the flip-dot display are shown in Table 4.

Arduino pin	RS485 Shield pin	Flip-dot display
RX3	R0	-
TX3	DI	-
5V	DE+RE	-
5V	VCC	-
GND	GND	-
	A+B	Signal Jack

TABLE 4. CONNECTIONS

Additionally, the serial communication at Serial3 should be set at a baud rate of 4800bps to match the flip-dot display protocol.

APPENDIX K – DISPLAY PROTOCOL SUMMARY

Note: The following information either follows from the display protocol specification document [6] or was experimentally derived and expands on the parts of the protocol that were implemented in this project.

Communication is one way – the controller sends information to the display. To start a transmission, the controller must send a start-of-transmission character – 0x02 (in hexadecimal), to the flip-dot display. This is followed by the specific command character - 0x31(HEX) denoting a ‘graphic’ message. The ‘graphic’ mode must be used in order to be able to address the individual dots on the display. The next character sent denotes the address shown on the rotary switch inside the PCB board of the display. However, you can also use 0x30 (HEX).

DISPLAY CONTENT

Each set of 8 dots on the flip-dot display is represented by one byte (8 bits). A “1” denotes a yellow/tactile dot and a “0” denotes a black/smooth dot.

Ex. *A 01010101 byte represents a*  *column on the display.*

Each byte (8 dots) has to be sent to the flip-dot display as two ASCII characters, each carrying the information for 4 dots.

Ex. *01011111 can be split into 0101 and 1111*

0101 in binary is 0x05 in hexadecimal, so the ASCII character ‘5’ is sent first (represented by 0x35 in HEX)

1111(BIN) is 0x0F(HEX). The ASCII character ‘F’ (0x46 in HEX) is sent second.

To address all the dots on the flip-dot display, 80 ASCII characters (40 bytes) have to be transmitted. The number of the bytes (in this case, 40) must also be represented as two ASCII characters and be sent preceding the content.

Ex. *40(DEC) = 00101000(BIN)*

00101000 is split into 0010 and 1000

0010(BIN) = 0x02(HEX)

1000(BIN) = 0x08(HEX)

⇨ *40(DEC) is sent as ‘2’ and ‘8’ in ASCII (0x32 and 0x38 in HEX)*

To end the transmission, the end-of-transmission character 0x03(HEX) is sent followed by the checksum of all preceding characters within the transmission (excluding the start-of-transmission character 0x02(HEX)). The checksum must also be represented as two ASCII characters.

Example transmission for a fully flipped (all dots are yellow/tactile) display:

Meaning	Hexadecimal representation of the signal
Start of transmission	0x02
Command ‘Graphic message’	0x01
Address	0x30
Resolution	0x32, 0x38
Body	0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46, 0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46
End of transmission	0x03
Checksum	0x35, 0x32

TABLE 5. EXAMPLE TRANSMISSION