Final projet

MEMBERS:

BENBOUALI MOHAMED AMINE

BERRICHI RIAD

BELHADJ AYMEN

GENERAL INTRODUCTION:

LOUIS BRAILLE, the inventor of the braille language, Louis Braille was a young blind kid who lived between year 1809 and 1852 passing away to tuberculosis, this impeccable young man developed the first ever universal reading and right language for blind people at the age of 15 years old and kept developing it until he published it at 20 years old, he also codified a music braille system. Louis Braille despite his lack of vision went out to become the first blind apprentice teacher at the New School for the Blind in Paris, France. There, he taught algebra and music.

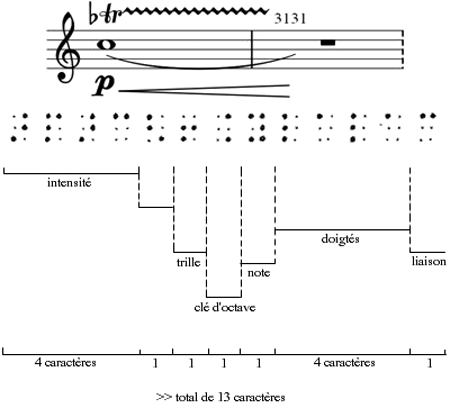


Figure 1: One of Braille’s musical systems

Braille is a reading and writing language that blind people can use to receive visual information by the form of touch, passing your fingertips over precisely threaded dots on a book you could read without the need of vision. In early days braille was printed on books using a braille writer (similar to a typewriter) or by using a pointed stylus to punch dots through paper using an instrument called a Braille slate.



Figure 2: Braille slates



Figure 3: Braille writer

CHAPTER 1:

GENERALITIES & DEFINITIONS:

G-code: a term wellknown in the 3D printing industry which is used to refer to any method used to convert digital content into machine coordinates which are then used to print something in 2D or 3D, this method is machine-specific and must be tailored for each size and purpose of machine, g-code is generally code programmed in a programming language chosen by the programmer himself and isn’t language specific.

CAD: a wellknow term in 3D printing and is used to refer to any program that is used to design objects in 3D, CAD itself is a 3D design program too.

Stepper Motor: a DC motor that is widely used for printing purpose due to its accuracy, it works by spinning at exact amounts of degrees every time and the fact that it spins the exact same angle every time makes it the perfect component to print something accurately.

Servo Motor: it is similar to a Stepper Motor but this one uses a potentiometer to spins to exact angles specified by code, it isn’t as accurate as a Stepper Motor but is cheaper.

Linear Shafts: an iron rod used as axis for the motor to keep its motion absolutely accurate, it is vastly used in 3D printers.

Timing Belt: a belt with teeth on it, it helps a motor convert its spinning motion into an axis motion, this axis relies on Linear Shaft to say in its path.

IDE: and program that lets you code any other program within, IDE stands for Integrated Development Environment (examples: Processing 3, Pycharm, Atom.io, CodeBlocks, Dev++).

Processing 3: a programming IDE used to make games and is used a lot in machinery.

Poking Head/Drawing Head: the mechanism that directly threads the paper to print braille, it is also used to draw 2D objects and is used by 3D printers to apply the hot plastic that forms the objects.

Numeric Indicator: a symbol in the braille language that is added before every string of numbers to indicate that what’s coming next are numbers, it is necessary as numbers and letters in braille have the same letters.

OUR WORK:

We noticed (with the help of our supervisor) that braille requires people who are not blind to be printed by Braille slates (Figure 2) or braille writers (Figure 3) and this is a major problem as this means blind people must be dependent on non-blind people to product books, printings, and overall messages.

This was partially solved by phones and audio playings which are great ways for blind people to receive information but there are two issues with this method:

1. Since they can’t see, putting on headphones at all times makes them vulnerable as they relly on their hearing sense for everything and it has to be interchangeable used for either listening to content or listening to the real world, an example of that is the true story that happened with Byron David Smith who is a blind man who lived alone and was being robbed by 3 people, he rellied on his hearing sense to find and shoot the robbers, this story was later featured in the movie “Don’t Breathe”.
2. Audio playings are hard to come by for everything the blind person likes, most books are now available in audio format on sites such as Audible by Amazon (is not free) but when it comes to any pdf found online you will not find the audio version of every single one, such as research papers.



Figure 4: A picture from the movie “Don’t Breathe” of the actor who casted “Byron David Smith” from the real story

PURPOSE & PROBLEM:

The purpose of our work is to offer a new way of printing braille, a way that doesn’t require people who could see to setup for a blind person and to offer blind people a way to become independent in printing their own braille at the click of a button.

HOW TO PROCEED TO SOLVE THE PROBLEM:

So to solve this problem of blind people having to depend on non-blind people we decided it would be a suitable idea to offer a type of printing machine, NOT a braille writer like in figure 3 which still requires a person to type on in order to produce braille-written paper, but a sort of automatic braille writer where the machine itself types on the paper.

You may ask how will this machine print actual braille on a paper which involves threading it, how will it know which content you want to be printed? How can a blind person operate this machine and is it easier than just using audio? All these questions we asked ourselves and came to solutions for each one, starting from the next chapter.

CHAPTER 2:

OUR SOLUTION:

What is the machine’s main usage case? Everything, we want this machine to allow blind people to print out pdfs not into regular text on a paper but into braille text printed paper, this machine needs to be simple in use, the person could tell his phone via vocal commands to send the pdf into the machine and it would start printing it on the spot, friends of this person could send pdf file from across the world into the machine and it would print them for the visually impaired person directly allowing this person to get more involved in the world than ever before.

During the rise of 3D printers we engineers had to jump in and learn this technology as it is directly related to our domains, and on that note we will be using a lot of 3D printing technology is our solution, 3D printers are known to print absolutely anything you design on 3D design programs such as CAD and Blender and so that technology is exactly what we need to make the machine, we will be using 3D printer specific hardware parts and software technologies and methods to produce our result, of cours it vastly varies from a 3D printer since we will be threading paper instead of building 3D objects but the basics are there, such as moving the drawing head according to coordinates given by the computer.

This machine will require multiple things to come together to work which is what we love, we bring our own software, our own hardware and 3D printing technology together to build this machine, its skeleton will be 3D printed by the help of our school’s scientific club Wameedh, we obtain the machinery from electronics stores and we program our own software to interact with the machine.

NOTE ABOUT SOFTWARE & HARDWARE:

Since our machine involves both software and hardware coming together to achieve this result, we must have a section where both are developed hand in hand but we will first give both separate sections where we explains both the hardware parts and software code that don’t involve each other, after that we will have a software & hardware part where both are being zipped together.

HARDWARE:

A printing machine that could draw and poke a paper of size A4 requires a very strong and consistent structure along with motors, timing belts, linear shafts and specific structural parts or else it will not move precisely.

1. Motors:

For this structure we have to use stepper motors, a strong model of stepper motors is the Nema17 motor.

Why do we use stepper motors?

A stepper motor as opposed to regular DC motors, can spin with what we call steps, if say we were to use a DC motor to move the head to the left we would have to power it for a period of time and stop hoping it will stop at the location we wanted which is super imprecise, but if we were to use a stepper motor we can tell it to spin exactly 1 degree and it will do that every time and precisely.

Why Nema17?

Nema 17 stepper motors are well known to be used in 3D printers, they are strong, precise and cheap stepper motors for this kind of application.

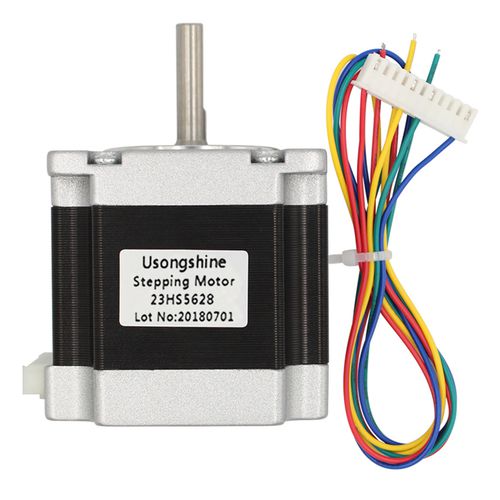


Figure 5: a Nema 17 stepper motor

To allow the poking head to move up and down we need a different type of motors called a servo motor, this motor will be directly above the paper operating our threader operation by commands from the computer in the form of two angles, an up angle and a down angle that can be varied until we achieve the amount of threading we need for a usable brailler reading paper. (figure 6)



figure 6: SG90 micro small 9g Sevo Motor 1.5KG

1. Timing Belts:

For our motors to move a structure left and right by just spinning, we will need timing belts and for that we’ll need a 2 meter long timing belt with relatively tiny teeth along with their matching pulleys.



figure 7: a 2 meter long timing belt with its 16 teeth spinning heads

1. Linear Shafts:

The main element that will hold the structure above the paper is called a linear shaft, it’s basically just iron bars with smooth surfaces, they are also the main element in 3D printers, our linear shafts are 8mm thick and 40cm long.



figure 8: linear shafts of length 400mm and diameter 8mm

1. Structural parts:

We want a structure similator to this one down below but on a larger scale, the benefits of this structure is that both motors can be situated on one axis only but still be able to steer the head anywhere, this is thanks to the unique timing belt usage we’ll show in the next section.

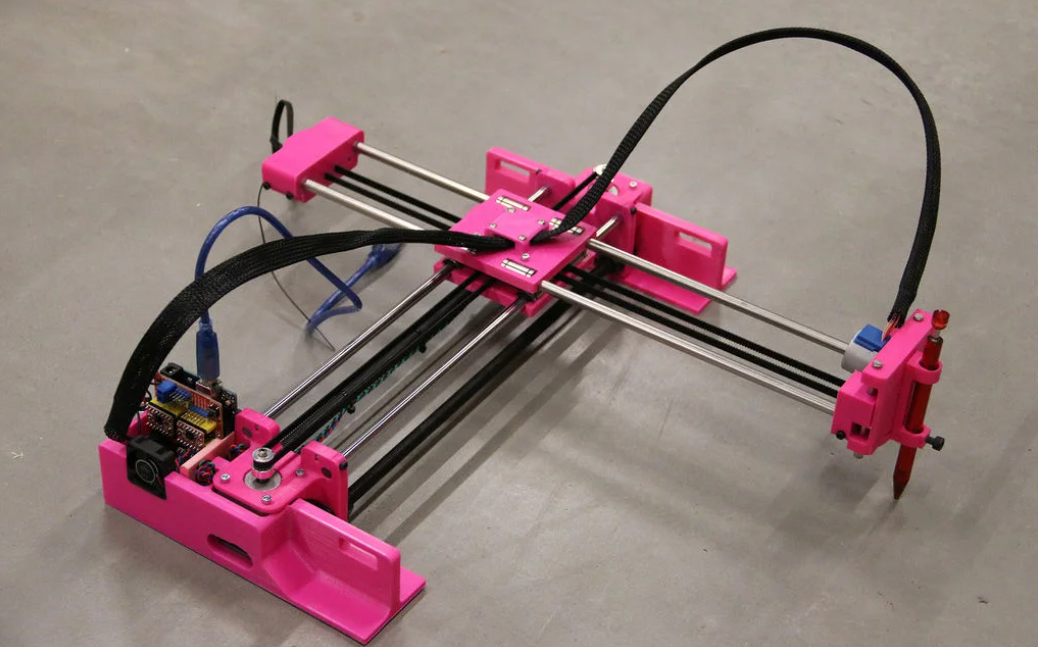
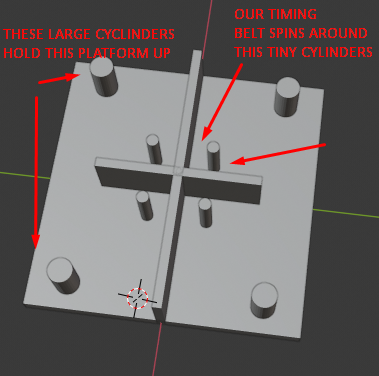
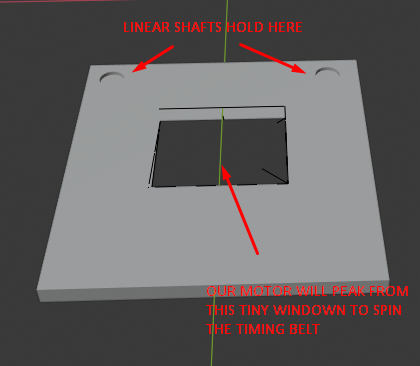


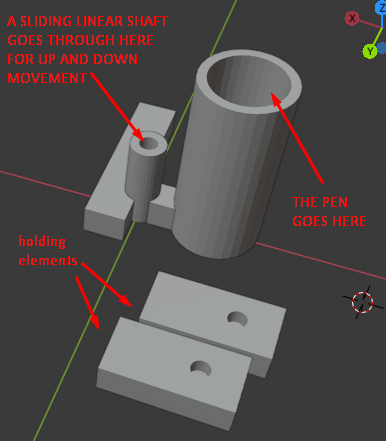
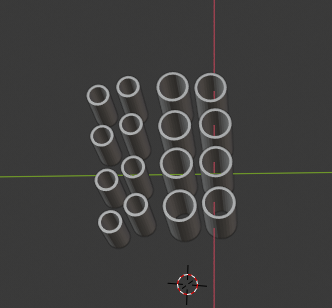
Figure 9: an exmple of a regular drawing machine design that drawswith a pen

To hold all of these elements in place we need specifically cut pieces of structure, In this case we will be 3D printing these pieces, below are all of the pieces we designed in the 3D modeling program called Blender.

Side panel (each motor will have this platform) Center panel (takes care of head directions)

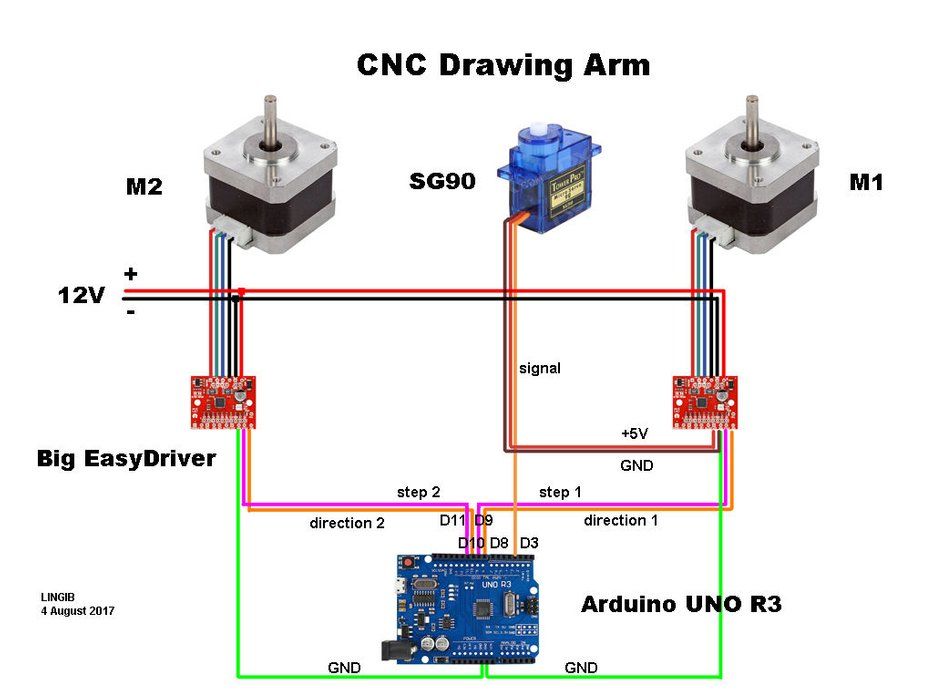


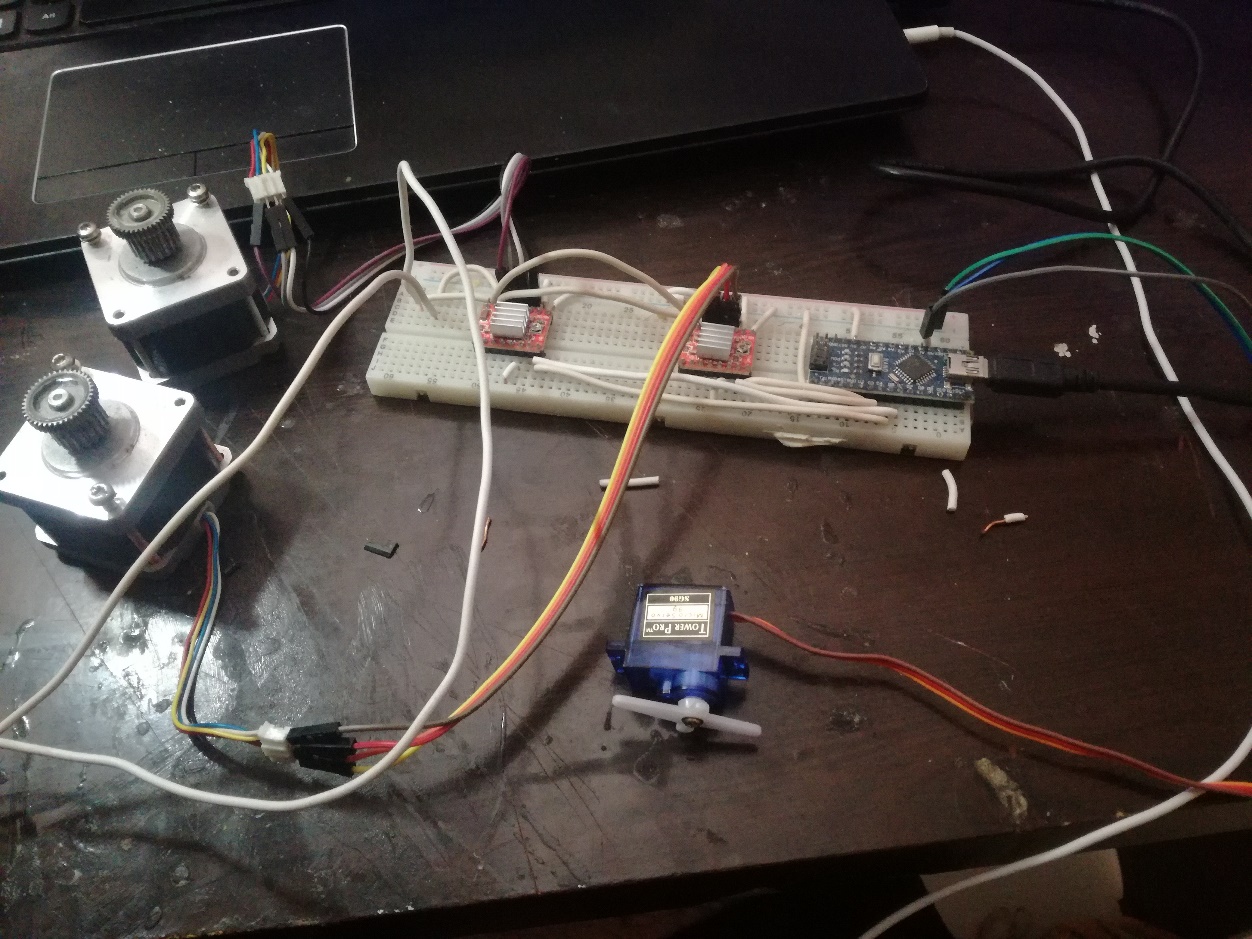
Drawing mechanism (along with a spring) Spinning bolts (allow timing belt to spin interruptionless)

CIRCUITRY:

The circuit that brings all of this together is very simple and logically ties all of the hardware together.



After acquiring all of the materials including an arduino board we were able to wire all of the materials together to have the exact circuit, but without the 3D models shown previously

Due to the inability to obtain 3D printed pieces we resorted to building it fully in wood by cutting it and nailing it together, this causes the machine to be inaccurate in its movement but nonetheless is still a remarkable prototype with what we are able to obtain, so after a long time of sawing and nailing we got the build below.

SOFTWARE:

We need two types of software:

* First, there is no publicly available code that takes in a regular pdf file and converts it into machine coordinates for Braille printing.
* Second, all of this hardware is useless without proper code that steers it, thankfully 3D printers have existed for a long time now and there exists a special method that allows any printing/drawing machine to work on any plane without reinventing the wheel and it’s called G-code, this format allows us to convert any information into machine coordinates.

Starting with the first type of software,

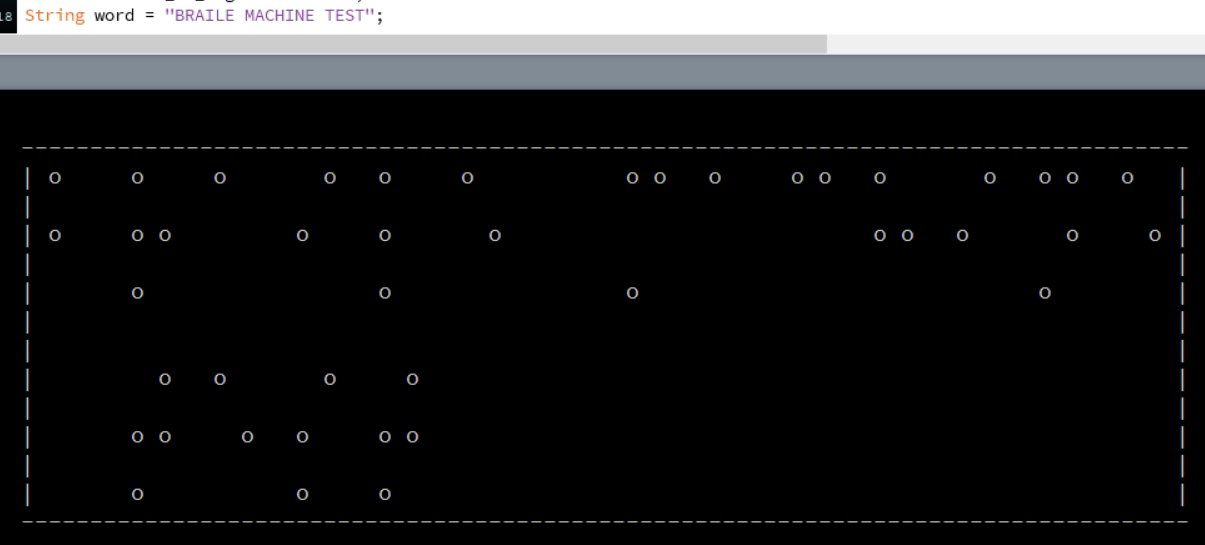
Text To Braille Language Conversion

We’ll be writing this code on a program called Processing 3, it was originally made to code games but it’s also easy in dealing with machines.

We coded a basic converter for starters to get ahold of braille and how to use it, good thing about braill is that both letters and numbers have the same braille characters! So A has the same braille as number 1, B has the same braille as number 2 and so on! How we distinguish between them is the letter that comes before, it lets the reader decide if the next characters are numbers or letters.



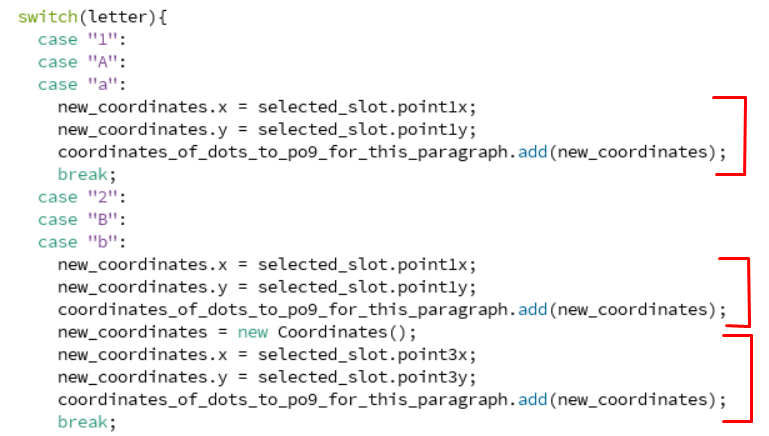
Here’s a simple conversion example:



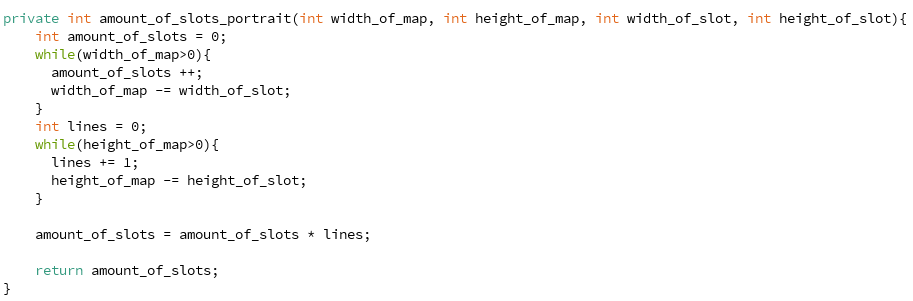
For starters, we need 6 dots for every characters in our braille table, but each letter fills specific dots of the 6 available slots, we can use a simple switch case statement to decide which letter we have, for example the letter A and the number 1 are designated by one dot on the top left, the letter B and number 2 are designated by two dots.

(in the code below, we are taking the X and Y coordinates of each dot and addingthem to our printing machine’s list of dots that it needs to thread)



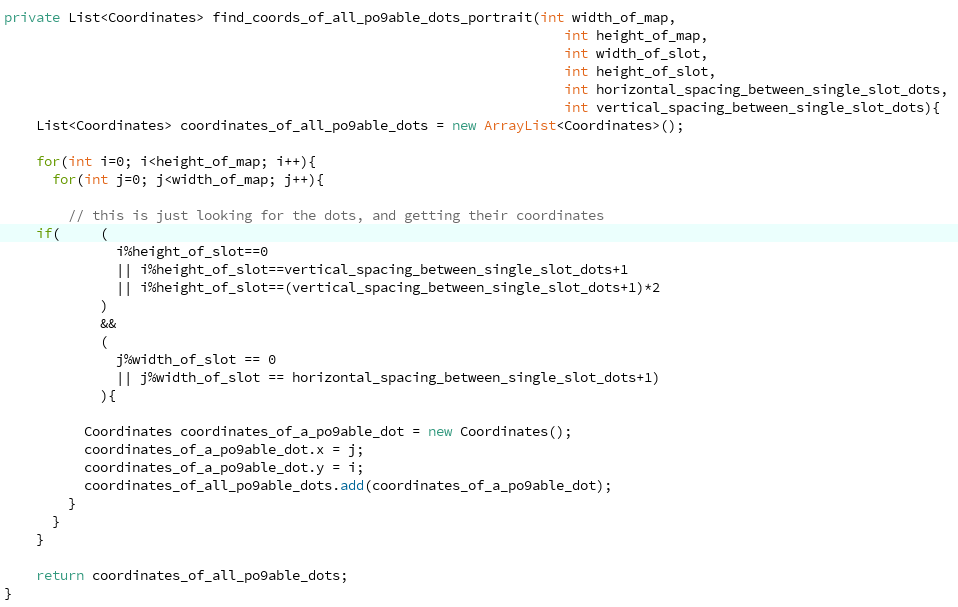


One of the most important features we added was the size of the braille language on the real paper and it can be done with a simple function

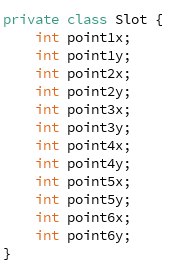


You could give this function the total width and height, and the width and height of each letter and it would count how many letters it could fit in it and returns amount\_of\_slots, this variable is essential in the printing process as it allows the code to quickly place any sentence into the required slots and instantly send it to the machine.

Now that we got the amounts of slots available, we need to know the exact coordinates of all of those slots and this next function does just that, it takes in the spacing between each slot that is given by the user and the given height & width to calculate the exact coordinates of each dot it needs to thread, this function is what ensures each dot is properly placed relative to the others and letters are efficiently away from each other.



Of cours, we are storing all of these coordinates and so we need an architecture, a Slot default architecture, the Slot one contains the x and y for each of its 6 slots.

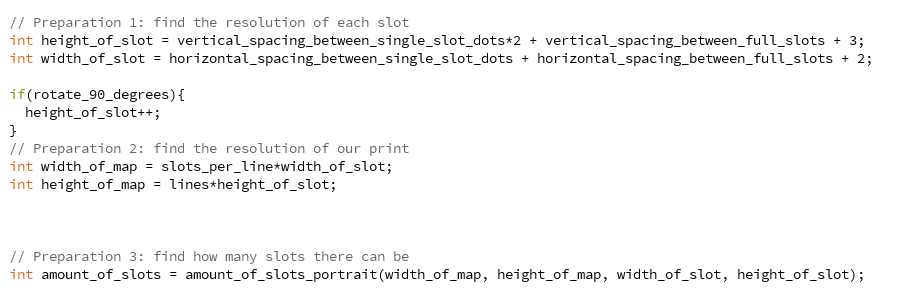


Now everything is organized and we can pass the sentences to our code and it would return a bunch of coordinates that our machine could use, we can plot that on the computer as an example for now.

This is how easy it is to use the function, we input a word/sentence and it would return a set of coordinates.

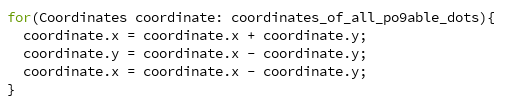


In this converter function we use all of the functions above to determine the necessary information to be able to convert words in three preparation steps



The user can also choose the rotation of the text allowing for horizontal and vertical braille prints on paper.

If the user had chosen vertical prints all we need to do is to switch between our x and y coordinates for each dot and that’s it!



The function above seems unnecessary but it is a cpu friendly way of exchanging values between two variables, we first add the two values and store them in the first variable and then we substract the results and store them in the second variable which eliminates one of the values and then we just substract that value from the total and get the second value, essentially this for loop above is just exchanging values between both variables x and y.

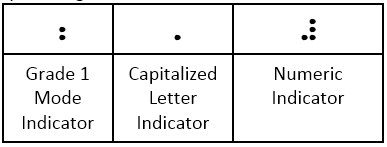
Now we use a simple print function to decorate the example on a console and we named all of that the print\_map\_portait, a similar function works for landscape aswell.



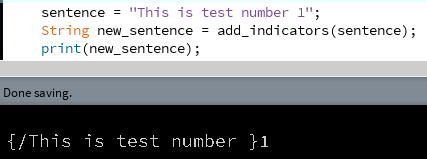
This gets us this test print!



As you could see above the console rotate\_90\_degrees Is set to false indicating this is a landscape regular print, and the sentence printed is THIS IS A TEST, if you compare this to the string you could notice it is missing the letter indicating symbols that are put between letters and numbers to distinguish between them so let’s add them! The indicators are of the shape:



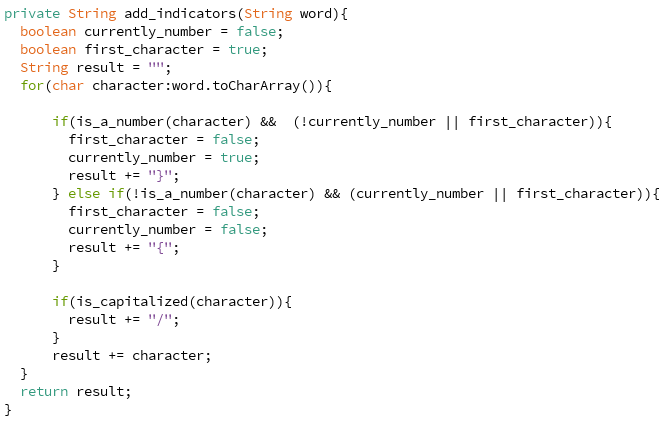
We can easily name them for example the symbol { is for Grade 1 Mode Indicator (which is the letter indicator),and the symbol } for the Numeric Indicator, and / for the Capitalized Letter Indicator and have them be added to this example “{/This is test number }1” of cours the braille reader won’t be reading these brackets in his paper, our code will convert these brackets into the desired indicators from the table above, now all we need to do is write a function that adds these indicators into our sentence.



And there it is, the add\_indicators() function will add these indicators according to our sentence’s:

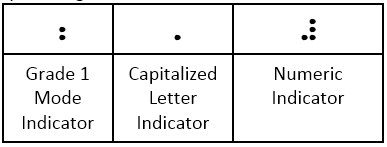
1. Capitalization
2. Numbers
3. Letters

And it simply works by looping over the entire sentence and checking if we have switched from letters to numbers and vice versa, it also checks for capitalization, we repeat,these symbols are not going to be read by anyone but our machine in order to convert them into the appropriate indicators in braille language.



Last but not least we add the symbols into our switch case to have them converted into their designated symbols:

This is essentially just adding the x and y axis of the dots in this table

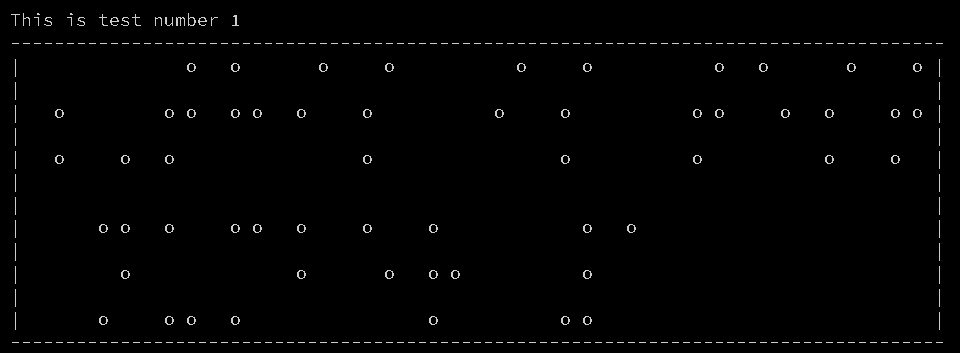


Letters Indicator: checks both dots number 4 and number 6.

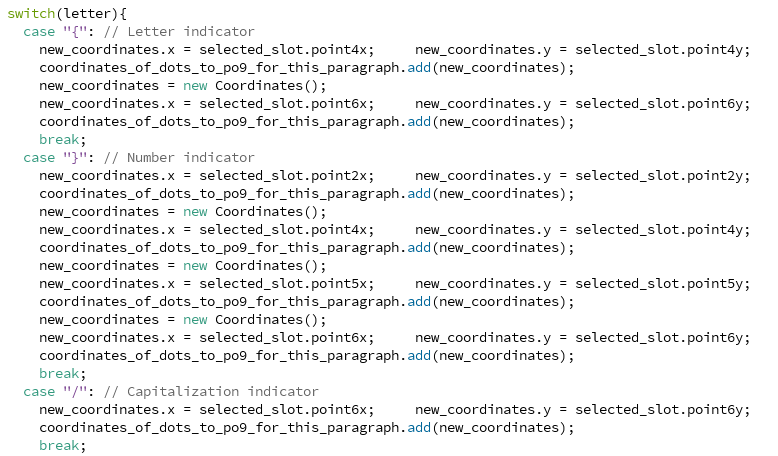
Number Indicator: checks the dots number 2, 4, 5 and 6.

Capitalization Indicator: only checks the dot number 6.

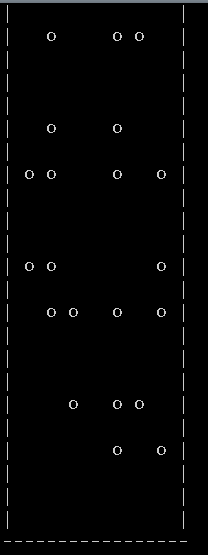
And now every sentence includes all the required indicators!



And that’s what the code below exactly does.



The 90 degree print is to tall to be displayed on the console but it works the same way



CHAPTER 4:

SOFTWARE & HARDWARE:

In this chapter we will be developing both software and hardware together to achieve the result.

Now that we have built the circuit, we are going to control it using Arduino IDE, this is a very simple piece of code written ina modified version of C made for Arduino and it will help us control our motors in order to program a proper braille machine.



If we determine the stepper motor’s STEPS variable to be 180, it will step 1 degree each time, but a universally used value is 200 for more accuracy yet less jittering, higher values are simply too hard to step without inaccuracy.



Now it is as easy as one line of code to step each of our stepper motors in any direction by simply entereing positive and negative values into the step() function for both stepper and stepper2.