

Creating a Complete Custom Keyboard from Scratch.

A Documentation of my first Split Keyboard: pengo-keyboard

Also look how you want to word things and use page breaks.

Do you want to use I or we the entire time, also which time (past/present) as that should
be consistent throughout the entire document

Elias Glauert

25 September 2024

Table of Contents

INTRODUCTION	3
GOALS AND SPECIFICATIONS FOR THE PROJECT	4
IMPORTANT NOTICE FOR USING THIS AS A GUIDE	4
WHAT ARE MY GOALS AND SPECIFICATIONS?	4
<i>List of all the Goals I have for this project.....</i>	4
DESIGNING THE KEYBOARD LAYOUT BASED ON MY HANDS AND THE COMPONENTS.....	6
GETTING THE LAYOUT RIGHT AND PERSONALIZED.....	6
ERGOGEN – THE SOFTWARE USED TO CREATE THE LAYOUT.....	6
<i>A little Introduction to Ergogen</i>	6
<i>How to describe the Layout to look like we want it to</i>	7
CHOOSING THE MICROCONTROLLER	9
WHAT IS A MICROCONTROLLER?	9
WHICH OPTIONS ARE THERE, AND WHICH DID I CHOOSE?	9
BATTERIES – WHICH FIT MY BUILD?.....	9
SPECIFICATIONS	9
MY PICK FOR THE PROJECT	9
THE PCB PLATE AND THE ROUTING ON IT	10
WHAT IS A PCB PLATE?.....	10
WHICH SWITCH SHOULD I USE?.....	12
THE KEYCAPS THAT I WILL BE USING FOR MY KEYBOARD	13
SOLDERING THE ELECTRONICS ONTO THE PCB.....	14
THE CASE – A CONTAINER FOR THE COMPONENTS.....	15
ASSEMBLY OF THE PCB, KEYS, CASE AND THE REST.....	16
FIRMWARE – MAKING THE KEYBOARD DO WHAT IT SHOULD	17
CREATING THE KEYMAP FOR MY WINDOWS COMPUTERS	17
<i>Short Notice</i>	17
<i>How the most important keys were placed</i>	17
<i>What did I put in the different layers?.....</i>	18
CREATING THE KEYMAP FOR MY MacBook.....	19
<i>Short Notice</i>	19
LOADING THE KEYMAPS ONTO THE CONTROLLER	19
CONNECTING THE KEYBOARD TO THE COMPUTER.....	21
WHERE CAN I BUY ALL THE PARTS AND WHAT WILL IT COST ME?.....	21
IMPORTANT NOTICE	21
PARTS LIST	21
3D PRINTING AND SOLDERING	22
CREDITS	22
REFERENCES FOR THE WORK.....	23

Introduction

This paper is about my journey of creating a custom-made keyboard from complete scratch. It all started after I watched a video¹ made by Christian Selig, a video where he himself went about creating one of these custom boards. He is the person that I got inspired from, so do feel free to watch his video and interact with his page. It gives you a great overview of the scope this project has, as it can take a few weeks to complete.

I think a lot of people that stumble across a paper like this have had contact with mechanical or non-stock keyboards, maybe even thought something along the lines of „There has to be something else out there that fits me better“. Keyboards are expensive, especially the high-end or custom ones. The price is something that should be made clear early, so if you do not want to invest a few hundred euros or dollars you might not want to follow along the path that I and so many others took.

If you are already informed about the topic of keyboards you might know about what kind of switches, keycaps and mods to customize keyboards exist out there. Some might even know about Corne or 40% Keyboards, which are a topic that I will not cover in detail. I would say that I felt inspired by the Corne variations, just like Cristian Selig was, but for me the forced layout of the keys all stock Corne boards have seemed a bit off for my hands, so I went about making a board from complete scratch.

The paper you are reading is written to document my journey and mark down the findings I made along the way, as a lot of work goes into researching, designing and building, especially for a first timer like me in this kind of project field. You can read it like a guide, for which the content section should be a good overview on the general structure of the paper and jumping to the parts you find most important and skipping those you already know are quite certainly alright. You can do what you want with this text: share it, build upon it, that does not matter to me, if you give me credit, like I did for Selig, as most of what you will read is built upon his research, and are trying to spread the knowledge about next generation keyboards.

If you need a certain source that I used when creating this paper, you can look at references at the end. They include all the material referenced in the footnotes.

¹ You can his YouTube video here: <https://www.youtube.com/watch?v=7UXsD7nSfDY> (Last accessed 22 September 2024), a blog about his project is linked in the description (Under the tag “Parts List”).

Goals and Specifications for the Project

Important Notice for using this as a Guide

The goal of creating a board from scratch is to have a personalized and enjoyable experience when using it, which explains why I am explicitly stating that these exact choices work best for me and do not have to reflect your choices. Each person has different expectations, wishes and most importantly hands and fingers, so please do not just follow and copy my exact steps in the design sections if you wish to also create a keyboard from scratch and are reading this as research. Really try to question my way of doing things and try to create your own personal board, maybe even improve my ideas, as they are not perfect in any way.

What are my Goals and Specifications?

I wanted to create a keyboard that makes typing as easy and simple as possible, as the keyboard of the Mac that I started learning typing with 10 fingers on felt weird and unoptimized. My right hand moved too much when using the backspace, messing up my positioning, my thumbs were just sitting there, doing nothing compared to all my other fingers and some fingers still tried pressing the keys that other fingers were supposed to press. All these things were issues I tried to fix with this project.

List of all the Goals I have for this project

- 1) The board fits my hands, so I don't have to move them when typing
 - a) Ortholinear layout: Normal keyboards have their rows moved a little bit, a leftover from the times typewriters were the norm, as back then each individual key was connected to the main machine via type bars, that needed to be spaced apart, as otherwise the mechanics would break. It is easier for fingers to move up and down than side to side, hence I want to use an ortholinear layout, where all the keys are in a straight line beneath each other.²
 - b) Column staggered: Adjusted key column positions that are based on the different lengths of the fingers that are assigned to them to minimize strain and travel length. An example would be the middle fingers keys, which are higher up than the index finger ones.
 - c) Less keys, as I would rather use different layers³. Thumbs have more functionality when adding this, as they are used to switch between layers. Hands

² A visual comparrision can be found here: <https://tech-fairy.com/staggered-vs-ortholinear-keyboard-what-are-the-differences/> (Last accessed 22 September 2024).

³ A layer describes the outcome of a keypress. The base layer gives back an “c” when the C-Key is pressed, but when pressing shift the layer is moved to instead give back a “C”.

don't move this way optimally, as you just switch the layer up or down and press the key same keys as before, only with a different outcome.

- 2) Layout can be toggled between Windows and MacOS
 - a) Even though I originally only wanted to make a Windows keyboard, as I use MacOS machines for work and I want to use the board for working, I need a flexible keymap.
- 3) The board is split into two parts.
 - a) This gives me a more ergonomic experience, as I can adjust the position of both sides to fit my current need, being able to move them around on my desk.
- 4) Wireless connection (Bluetooth)
 - a) I like the aesthetic of this, and it does not bind me to having both side close to each other or the computer they are connected to.
 - b) The cable I would have to use otherwise will also not be in the way of other things on my desk.
- 5) No backlight, like all my keyboards have had until now, as it slumps battery time and should not be needed.
 - a) You should know where all the keys are without looking at all times.
 - b) If you make the keyboard wired, you can include backlighting as battery life is of no concern for you.
- 6) Silent switches, as being able to not annoy others around you is a plus in my eyes.
- 7) Overall, the design should be simple, sleek and clean and fit my setup/room style-and color-wise

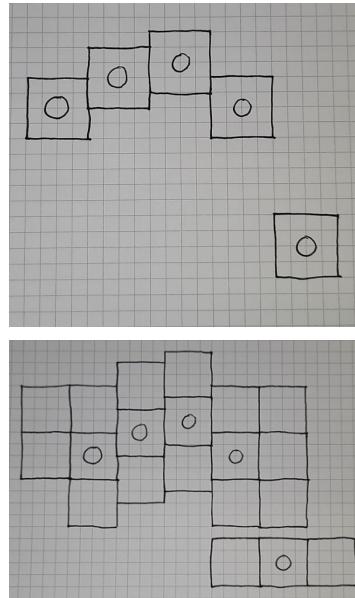
Designing the Keyboard Layout based on my Hands and the Components

Getting the Layout right and personalized

To implement points number 1 and 3 from the goals I first started by putting my fingers on a 5mm checkered sheet of paper in the way wanted to place my fingers on the keyboard and drew a 2×2 square around them, as that is pretty much the real size of a key⁴. These 5 positions are the base for all the keys in the home row.

From here you can add the movement range of each of your fingers, but that is not really that necessary, except the reasoning on why I did not include an outer bottom key for the pinkie finger. On the same Note I added the rest of the keys inspired by Corne Layouts.

This entire process is done not only for the left-hand side, also known as the center side⁵, but also for the right-hand side, which is called the peripheral side⁶. These names are widely spread in the split keyboard space; hence I will be using them.



Ergogen – The Software used to create the Layout

A little Introduction to Ergogen

To get the first sketch that was drawn from paper to a format that can be worked with in the rest of the process I used Ergogen⁷, a free open-source software that helps with generating files used for custom keyboard building. Ergogen works by you describing what the keyboard will look like, and it does the rest for you. You can do a lot more than just creating the layout in Ergogen, but that will be covered in the later points in the paper when the topic is more relevant. This is also not a full tutorial, as I would not call myself an expert, please read the documentation that is provided on the website or watch some tutorials on YouTube if what I am doing does not match your scope.

⁴ The real size of a keycap is around 18mm * 18mm and with the padding around the keycap you have a size of 19.05mm * 19.05mm. This is pretty much the same as 20mm * 20mm that a 2×2 square has.

⁵ The centre side is what sends the signals received by itself and the other side to the computer, which is why it consumes more energy and has to always be connected if you want to use the keyboard.

⁶ The peripheral side does not have to be used all the time, as it only sends its inputs to the centre side, making it use less energy.

⁷ Ergogen can be found under <https://ergogen.xyz>, but I like <https://ergogen.ceoloide.com/> more, as it gives you a better user interface. Later on we will need it locally, but more on that then.

How to describe the Layout to look like we want it to

Like mentioned above this is only about getting the key positions right. For this we start with defining the rows and columns we will have in the final product, as Ergogen is based on a matrix system. I defined my columns as these: outer, pinky, ring, middle, index and inner and my rows as these: bottom, home and top.

```
points:  
  zones:  
    matrix:  
      columns:  
        outer:  
        pinky:  
        ring:  
        middle:  
        index:  
        inner:  
      rows:  
        bottom:  
        home:  
        top:
```

The points are pretty much just the individual keys, the zones are the just a collection of key-grids, of which matrix is one example. Another zone we need to define is the thumb zone with the three thumb keys. All the numbers to define or shift positions around are in millimeters, as Ergogen is based on the metric system.

```
thumbfan:  
  anchor: # defines the point for the zone, from where we start our positions  
  ref: matrix_inner_bottom # this sets the base reference for the anchor to  
  the bottom row of the inner column in the matrix zone  
  shift: [-24, -20] # moves this previously defined reference to where we  
  want along the [x-axis, y-axis], shift cannot be used for specific keys  
  columns:  
    near:  
    home:  
    far:  
  rows:  
    thumb:
```

Now this itself would only generate the keys without any column stagger, which is not what I want with this keyboard. The rotations for the thumbfan are also not included right now, which makes us use the attributes stagger, splay, origin and spread.

```
outer: # in matrix
  rows.bottom.skip: true # this is for not having a third key in the outer
  column

  middle: # in matrix
    key: # use this for the key attributes
      stagger: 4 # moves the column up or down
    home: # in thumbfan
      key:
        spread: 21.25 # how far apart all the columns should be from the
        previous one, or rather the distance between the points of the keys
        splay: -12 # rotates the key based on the origin point
      origin: [-11.75, -9] # used to move key base positions
```

This is not the full file to complete the description of the layout, rather just some things so you can try to implement things yourself, as that will have a greater learning effect than just blindly following instructions. The complete files can be found in my GitHub Repository⁸.

⁸ The repository can be found here: <https://github.com/lulton/pengo-keyboard>

Choosing the Microcontroller

What is a microcontroller?

A microcontroller can be explained as a tiny computer inside the keyboard. Just like your brain tells your body what to do, the controller tells the computer how to react to your key presses. It handles all the important tasks your keyboard needs to function. When you press a key, the microcontroller detects this action and translates it into a signal. This signal is then sent to your computer, which understands it as the letter or function you intended.

Which options are there, and which did I choose?

There are a lot of different controllers that all work for different price ranges and use-cases, but for me the nice!nano⁹ works best, as I need the Bluetooth capabilities like mentioned in point 4 in the goals, but you could also consider another popular one, the Pro Micro. A list of different microcontrollers based on which use case you have can be found in an article by Golem published in 2021¹⁰. It might be a good start for you, if the other controllers do not suit your ideas.

Batteries – Which fit my Build?

Specifications

This is linked to the microcontroller you have chosen, as all of them have different needs and requirements. If your microcontroller does not support wireless usage, then you will not need a battery and can skip this part. The general goal is to find a compact battery with a huge capacity, so I don't have to charge the keyboard that often.

The nice!nano needs a 3.7 V lithium-ion battery with at least a 100 mAh capacity. It is also about the size the battery takes up, as I am sure you can find a lot of rechargeable batteries online that fit this description.

My Pick for the Project

I am okay with the battery taking up more space and changing the tilt of the PCB, which is why I chose the PS3 Controller Battery, as it, with a size of 36mm * 58mm * 6mm, still stores 1800 mAh. It is a bit big for my project, which will force me to adjust my case size a little bit to fit it in. The bigger the capacity of your battery is the longer you will have to charge it as well, that much should be clear. The tilt of this pick creates a lightly more

⁹ You can find more information here: <https://nicekeyboards.com/nice-nano> (Last accessed 22 September 2024)

¹⁰ Article Link: <https://golem.hu/guide/controllers/> (Last accessed 22 September 2024)

ergonomic experience, with the inner part of either side being a bit higher than the outer part.

The ZMK Power Profiler¹¹ is a tool to give you a rough estimate on how long the center and peripheral sides will last for until you must charge them again. If I plug in my data, the tool calculates that I need to charge my center side around every 6 months and the peripheral side around every year.

The PCB Plate and the Routing on it

SUBHEADINGS?

What is a PCB Plate?

A PCB plate is where the controller, the battery, the switches and the other small components are all connected with each other. It is very simple in theory, so you could hand wire your keyboard, but from any source I read from, that had created a board with this method said something along the lines of „never again“.

And the second video, no clue right now what is mentioned:

https://www.youtube.com/watch?v=M_VuXVErD6E

Its an in depth look as far as I know, but that is also a bit outdated as it seems

Install Ergogen locally so you can add all the footprints that you need, just copy them from either Cristian or myself

- I already talked about points, so you only need to talk about the ‘outlines’ and ‘pcbs’
- Change the outlines (edge cuts) to look like you want
 - They need to surround all the keys in a fluid way, so that it is one single pcb plate later on, talk about how you achieved that
 - Keys outline (?) (left and right) and fulls (left and right)
 - For later on for the case you can create the cutouts (if you want to)
- Add a footprint for the microcontroller, power switch, reset button, diodes, mounting holes, battery connection

¹¹ The ZMK Power Profiler can be found under this link: <https://zmk.dev/power-profiler> (Last accessed on 22 September 2024)

- How is this done exactly so electrical signals do not interfere with each other (look at the kicad files from Christian how his things are placed and how he talked about it in the code) – basically how the diodes are used and how to connect everything with them
- Use the same community made ones, for which you compile locally, but that is something you should mention early on how to do
- Download the footprints and move them to the footprints folder
- Compile that stuff in a folder you created (move into ‘ergogen’ folder (type cd and then move the folder into the command line if you use iTerm I think) and then just run ‘ergogen .’, which creates all the correct folder/updates them and the files in them)
- Which pin on the controller is connected to what

STILL DO THE NETS WHERE ALL THE KEYS ARE ROUTED TO, AS THAT IS SCUFFED RIGHT NOW DUDE

- In the PCB section tell the microcontroller which key is which
 - In Christians video at timestamp 7:26
 - How does this work with the diodes and the row/cols
 - Rows cols: connect all the signals from one row/col together and then route them to the hole you assigned them to
- Create the pcb files for both sides and also do the wiring for both
- Export the PCB files for both the center and peripheral sides
 - File: plot
 - Generate drill files (define a folder for where they go inside your files)
 - Click plot again to do everything
 - Compress folder to a .zip file
 - Upload zip to a pcb manufacturer

With all the components positions defined all that must be done is to route the electrical signals along, so they do not interfere with one another. For this I used KiCad, also a free software that tells us how many connections are unrouted and where they

need to go, just like a big connect the points game, which you must click through to get your final PCB files.

(TRY AND INCLUDE VISUALIZATION FOR THIS STEP)

With the finalized version of the PCB we can go to a supplier of your choice (I would recommend [JLCPCB](#) or [aisler](#), those gave me the best quotes, but that can differ for you) and click through the quoting. These were the choices I took:

- Does Christian talk about this in his video?
- He only said to pick lead free and to upload the zip file, nothing more
- Oh and the minimum order size of 5 per pcb (so 10 in total, 5 left, 5 right)

Which Switch should I use?

Key Switches decide a lot of the feel of the keyboard, so I needed to think about this aspect a good bit. I will solder on hotswap sockets to allow the changing of switches, so if I feel like I chose the wrong ones I can just replace the switches I have installed. As I wanted to make my project silent or at least as silent as possible (point 6 of the goals), I chose tactile switches that make as little noise as possible. This can also be influenced by modding the keyboard with PE Foam or Silicone plates beneath the switches. Some people might also like a low-profile switch in comparison to the normal profile, which gives it more of a laptop look and feel. In each category you have high- and low-quality options, where I would personally rather spend a bit more, just to be sure to avoid key chatter¹².

My final pick were the Gazzew Boba U4 Silents¹³, which are tactile, have a normal profile and are designed to be quiet¹⁴. If you want other inspirations in a similar direction, maybe because you did not like my pick, you can look at the following options:

- Akko Penguin Silent
- Zeal PC Zilents V2 62/65g
- Glorious Gateron Brown
- MX RGB Ergo Clear
- TTC Venus 45g Linear

¹² Key Chatter is the problem of one key press giving out multiple actions, so one C-Key press writing a “cc” instead of a “c”.

¹³ The Gazzew Boba U4 Silents can be found under this link: <https://thocstock.com/switches/gazzew-boba-u4-silents> (Last Accessed 22 September 2024)

¹⁴ If you want the non-silent version look for the Gazzew Boba U4T Switches.

- Everglide Aqua King V3

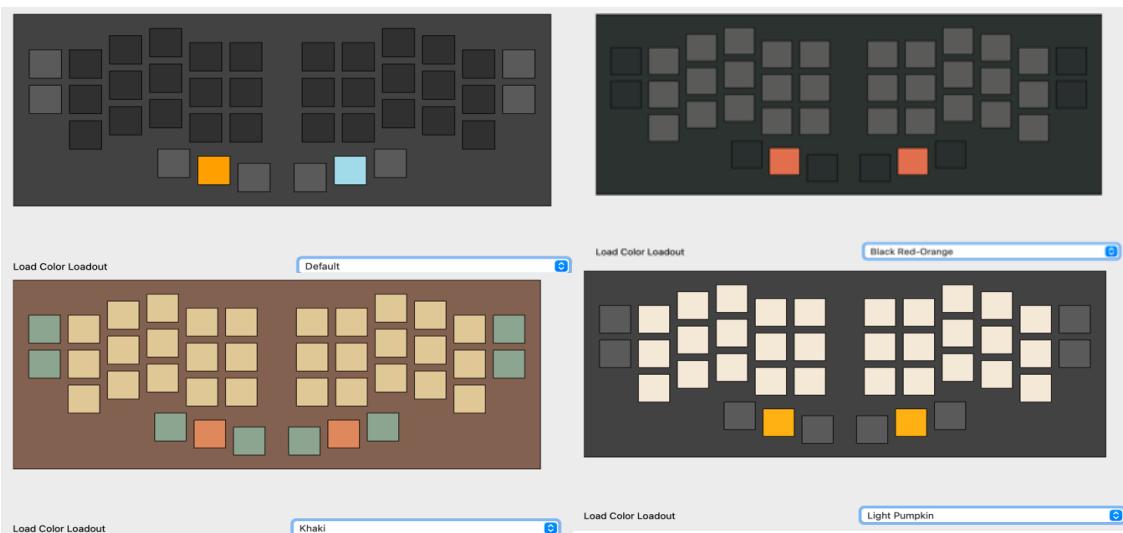
All the links to where you can buy these can be found in the references.

Do you want to lube the switches? This could do something for the feel, but is not really necessary, I think.

The Keycaps that I will be using for my Keyboard

Keycaps play a huge role in how your keyboard looks. They are exchangeable so you do not need to worry about only getting one set of them and having to stick with them, in the end it is about design choices and money. My parameters are that they are normal profile ones, just like my switches, they have the MX/Cherry profile¹⁵ as my switches have the same and that they are made from PBT Plastic, this them feel nicer to the touch. They will also be blank, because as mentioned in point 5 in my goals you should not need to look at your board.

For me a huge part is also color design, as everything should look good together, like mentioned in point 7 in my goals. I got my inspiration on Pinterest, where I pinned all the designs I could consider in my vision, which gave me a good idea on what I would go for. I created a little program in Python with the help of ChatGPT that would let me display



and change colors based on my layout¹⁶. My favorite start combinations based on the pins on Pinterest were these:

¹⁵ A medium deep cross sized hole in the center of the backside.

¹⁶ You can find the programm in my GitHub repository under the name 'layout_color_tester.py'.

There were a few more and these are also not the first draft, just the ones I liked personally. I would like to just order some custom keycaps that fit my color scheme, but that seems to be out of question as I did not find any retailers that did exactly that. Most just sold sets or artisan keycaps, which are not what I am looking for, but I ended up picking one set from amazon that is similar the ‘Light Pumpkin’ color scheme, so that was decided on after a bit of discouraged googling. There are a lot of websites out there that sell keycaps so please do your own research into which set suits you best.

Soldering the electronics onto the PCB

SUBHEADINGS?

WHAT SAUTERING THINGS DO I NEED TO GET? WRITE RESULTS INTO THE SHOPING PART

- Lead free sauter
- Proper ventilation so you definitely do not inhale that shit

- What we will sauter:
 - o Diodes
 - Since they are tiny put some sauter on the position
 - Add the diode
 - Sauter the other side
 - o Hotswap sockets
 - I think these are also on the backside, but I am not 100% sure
 - Look at around 15 minutes to check or something
 - Nvm at 16.29 he showed that they are on the back side
 - o Controller sockets
 - These are on the backside
 - After sautered drop in the millmax pins that connect the controller to the hotswap sockets (that does not make sense fr) (press them in till they click tho)
 - Sauter them all to the controller that you have added
 - o Controller
 - Already done
 - o Reset button
 - Easy I guess
 - o Battery plug
 - Easy I guess
 - o Battery on/off
 - Easy I guess

- Key switches (?!)
 - He was capping
- Battery
 - And againnnn

The Case – A Container for the Components

SUBHEADINGS?

HELLO PLEASE THINK ABOUT THE BATTERY BEING A BIT BIGGER

USE ERGOGENS CASE PART, THAT CAN HELP YOU KICKSTART YOUR JOURNEY I THINK

- ➔ Cases section, which gives you a jscad file
- ➔ For this we want a key cutout outline I think

Also Christian uses fusion 360 for his shit, look into that stuff (you could also use that site where you designed your room, forgot the name tih)

I want to create a case for my board, some like the look of exposed PCB plates and wires and whatnot, but as I want a clean and simple look like mentioned in point 7 in the goals, I want to hide all of that. The case itself is 3D printed, so you can decide yourself if you want to print it yourself or if you want to use an online printing company to do so. This can be decided as many people told me that self-printed is often of lower quality, hence I went with a company to print one for me.

The case consists of 2 parts: the plate/outside and the bottom, which is screwed to the other part. The plate hides the PCB, as it sits between said PCB and the Keycaps and is usually separated from the outside, but due to myself not wanting to separate the parts in the print I did not, it does not really matter that much.

(THE PLATE SWITCH HOLE IS 14mm * 14mm AND THE DISTANCE BETWEEN SWITCH HOLES IS 5.05mm)

The start of creating a case consists of taking your key positions

Try to measure all your components and where they will be in the case, but only once you have created the PCB and the electronics on there.

WHERE CAN I 3D PRINT THE THINGS, WHAT DO I TELL IN THE DOCS
-> DID I ALREADY WRITE ABOUT IT?

THE COLOR IS BLACK OR DARK GREY

Also think about trying to add click on key protection (like a lid) and many maybes try to add a feature where you can click together both backsides of the halves so you can take it with you more easily

Assembly of the PCB, Keys, Case and the rest

SUBHEADINGS?

Create the case when you all done the things before this step, as you can now get the measurements and shit more easily.

What is with these goofy things: <https://www.caseking.de/glorious-mx-o-ring-key-dampeners-soft-long-key-travel-40a-thin/GAZU-720.html>

- ➔ Where are they included and do you need them with your switches?
- ➔ Maybe ask the chat man himself what he thinks of this, as I am sure the research will take a good bit

10x M2x3 screws, and 5x M2x4 standoffs for assembling each side of the case (so 20 screws and 10 standoffs in total) (I'd just grab a small kit on Amazon)

- This is for screwing the top and bottom bits of the case together
- Also the pcb and top I think, which is why there are holes in the pcb

(Optional) 6mm (0.24") diameter x 2mm (0.08") height rubber feet for the bottom of the case

- He added them to hide the screw holes or something, which could be an idea, but you will have to look yourself

He added the switches to the case first, what a goofy ah, but putting them on the pcb and that into the case should be fine too, like idk bro what do I know its fucking 18.47 and I have been giving myself cancer for the last 10 hours (with schnitzel break in between)

Firmware – Making the Keyboard do what it should

Creating the Keymap for my Windows Computers

Short Notice

This part covers the general design process that I went through to find the ultimate and best fitting keymap for my needs, but still only talks about the requirements windows keyboards have. The Mac part is further down, which completes point 2 of the goals)

How the most important keys were placed

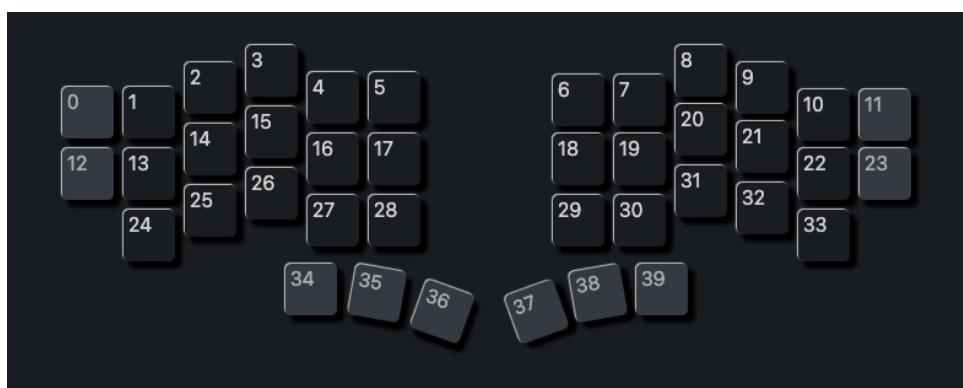
SUBHEADINGS?

ALSO THINK ABOUT MENTIONING THINKING ABOUT THIS PART WHEN DESIGNING THE LAYOUT, AS THAT IS SOMETHING PRETTY IMPORTANT

Where can I mention this: Use this website: <https://www.keymapper.dev/>

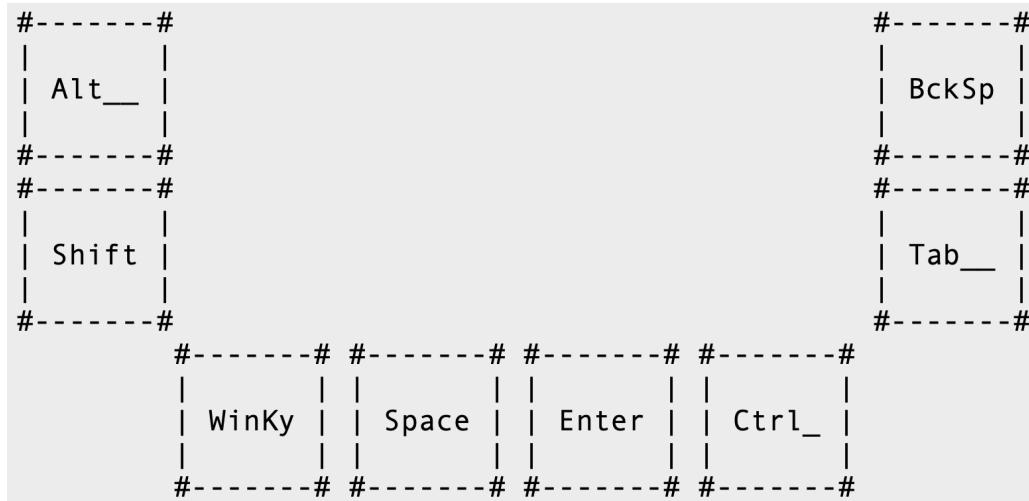
What about this: <https://github.com/JanLunge/pog> (docs: <https://pog.heaper.de/>)

When creating a keymap there are a lot of things you need to take into consideration, such as finger travel or even the functionality of the different layers. The main problem that I encountered whilst doing this task was with the removed third outer pinkie key, I only had 10 outer/thumb keys that would stay, no matter the layer, as you would need them for shortcuts and such, whilst I needed to incorporate 11 keys: Shift, Control, Windows, Alt, Raise Layer, Lower Layer, Space, Enter, Tab, Backspace and Escape. One key needed to be moved to another layer where it would be accessible and, in the end, I decided to put Escape in another group on the board. The position of the remaining 10 keys was the next hurdle as they had to be sorted into these spots:



The thumb home keys (35 and 38 in this visualization) are reserved for the layer switching, as that was what most people used them for and that seemed to make the most sense when you need to move layers a lot.

I again wrote a simple python program with the help of old reliable ChatGPT to help with finding the perfect combination¹⁷, which would again be helpful when it comes to the Mac part. There were a few rules set in place so that I would not have to press two keys with one finger for the most important windows shortcuts and the keys were only in positions that I wanted them to be in. After scouring the last remaining 16 layouts this is what I ended up with:



Here I didn't show the layer shifting keys (35 and 38), as those would not change anyways.

What did I put in the different layers?

SUBHEADINGS?

Like mentioned before we have 3 different layers to put all our different keys. The base, lower und raise layers. When we remove all the keys, we have put in regardless of the layer we are left with 30 free slots per layer.

The base layer, the one that is normally active I have put the letter of the alphabet, the escape key and the most used symbols: comma, colon and single quote. I went with a lightly modified QWERTY layout, so that when I do not have my board I won't be so thrown off. When everything is put together it look like this:

CHANGE THE LAYOUT TO BE QWERTY INSPIRED, BUT OPTIMIZED FOR CODING AND WRITING IN ENGLISH (AND GERMAN)

(INCLUDE THE NEW BASE LAYER (put the escape key where the ‘ is, the ‘ where the / is (/ is killed), the p in the ‘ slot and the esc in the p slot))

¹⁷ You can find the files in my GitHub repository, where you must, after downloading, put ‘perma_keys_lt_gen.py’, ‘layout_visualizer.py’ and ‘layouts_list.txt’ in a folder called ‘layout_generator’. After this you can define the rules after which all possible layouts are generated at the start of ‘perma_keys_lt_gen.py’ and show the newly generated layouts with the ‘layout_visualizer.py’ script.

The lower layer holds the numbers, Bluetooth functionality keys, the function keys (f-keys), and the arrow keys:

(INCLUDE THE LOWER LAYER)

The raise layer holds the last symbols and the keys that start/stop music and such:

(INCLUDE THE RAISE LAYER)

ARE ALL THE IMPORTANT KEYS ALREADY INCLUDED?

Creating the Keymap for my MacBook

Short Notice

This part is only about how I implemented the changes between the Windows and Mac Layouts and how I went about the switching between both profiles.

Asdf -> ZMK (?) implementation for this exact point.

Loading the Keymaps onto the Controller

SUBHEADINGS?

Asdf

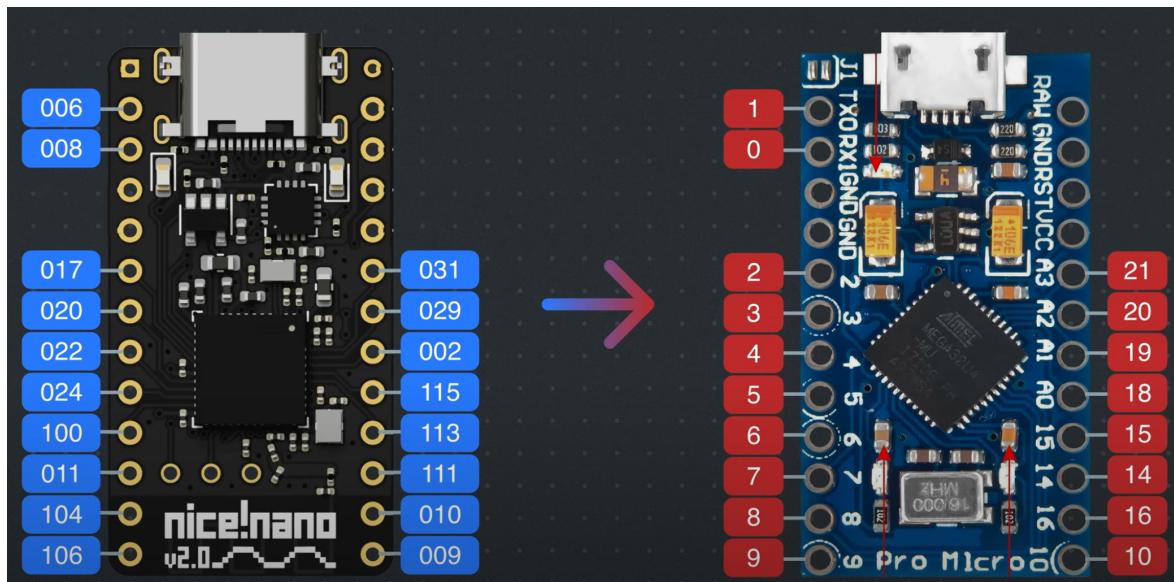
What is with ZMK? Look at Christians video again for this part as this looks kind of scuffed tbh.

This is a ChatGPT answer for how to switch between mac and windows: he chattet shit, trust me brother

You might want to ask him again on how this could be implemented with a start on the files and more information about controllers, diodes and how the keys are read in and stuff.

- Dependencies for ZMK: git (check), you also need a github account, create a repo like this ‘zmk-config-(keyboard name)’
 - o THIS DOES NOT NEED TO BE THE CASE AS YOU CAN ALSO JUST CREATE A FOLDER IN ANOTHER REPO AS FAR AS I KNOW
- Open terminal and copy the bash command in there
- Choose a keyboard template (Corne)
- Select the nice!nanoV2, as we are using that in the next step
- Say yes to copy in the stock keymap
- Then it asks for github infos (username, repo name) after which you hit enter to confirm
- Then y for if you want to continue

- This creates a folder for use from where we take the 2 corne files and rename them to what our name is
- Move the board folder into the config folder and then the 2 files we just renamed into boards and then shields
- Now we just create the files the zmk docs list based on wether the keyboard is split or not (read the docs... fucking hell man, im finished)
- A standout is the dtsi file where we are telling zmk that we are using split keyboards that are 6 columns by 4 rows and which pins in the controller we are using for which columns (ZMK USES PRO MICRO SO YOU HAVE TO TRANSLATE YOUR ERGOGEN ONES TO THE PRO MICRO EQUVELANTS)



- This should be good enough for the translation part muhehe
- Do the same for the rows (for split keyboards that is in the '(name)_left/right.overlay' files)
- Right should just be reverse of the left
- The other standout file is the .keymap file, where you can define all the layers and maps and shit
- Filling in the rest of the files through the docs should be pretty easy and then you are done
- Use git to push everything to github
 - o Move the 'zmk-config-...' folder into terminal
 - o Type in 'git add -A'
 - o Type in 'git commit -m "Funneh commit text hehe"'
 - o Type in 'git push'
- In the github in Actions you can see the firmware being build which should take a few minutes to complete

- Download it (under artifacts 'firmware.zip')
- Unzip it
- You have a file for the right and left side
- Make sure both halves are switched off and port the files one at a time
- Left side, double tap reset button, mounts to pc like a flash drive
- Drag and drop the left side, which will auto reject once complete
- Do the same for the right side
- Turn both keyboards on and tap the reset button on both at the same time

Connecting the Keyboard to the Computer

Asdf

Its legit just connecting it per Bluetooth now bruh

FIND A WAY TO SPLIT THIS PART ABOUT FIRMWARE IN A LOGICAL WAY

Where can I buy all the Parts and what will it cost me?

Important Notice

As this is only about my project, I can't condone every single item I bought for you personally, you might use the same products, but dislike some of the I linked, so take everything with a pinch of salt. The prices can also vary for you, as this project was realized in late 2024, so things might be cheaper or more expensive or there might even be new products that can be used instead of the ones I used.

Parts List

Please consider that the amounts you need for every part of the board can vary compared to your one. You might need more than just 40 switches or only 1 microcontroller if you do not plan on making a split keyboard. This list only talks about consumables and not tools that you need to create this project.

Component	Name	Amount	Final Cost	Buying Link
Controller	nice!nano	2	51,88€	https://shorturl.at/bXQa3
PCB	-	2 + 8	20,00€	https://jlpcb.com
Key Switch	Gazzew Boba U4 Silents	40 + 0	23,80€	https://shorturl.at/ye4b8
Key Caps	YMDK DSA Profile Blank	40 + 21	20,73€	Link

Battery	PS3 Controller	2	35,98€	https://shorturl.at/nsxka
Case	-	1	20,00€	?
Machine Sockets and Pins	-	2	5,95€	https://shorturl.at/EGez0
Diodes	-	40	3,80€	Link
Battery Jack	-	2	0,95€	https://shorturl.at/O23kq
Reset Button	-	2	1,95€	https://shorturl.at/uzzaJ
Hotswap Sockets	-	40	7,80€	https://shorturl.at/cXdu8
Power Switch	-	2	1,95€	https://shorturl.at/iUYgP
Screws (For the Case)	?	?	0,00€	?
Total Cost	-	-	194,79 €	-

3D Printing and Soldering

Some might already have the tools that are needed for 3D printing and soldering or borrowed them from a friend/family member, but not everyone will have an idea what tools they need to have so I have put together a list of all the parts I used to fulfil the project:

- 3D printer or 3D printing company
- Soldering Iron
- Look at the sauntering section, some things were mentioned

Credits

I again want to suggest the video made by Christian Selig, as it was the thing that got the stone rolling for me. He has also linked his blog post in the description, so you can go and read that as well, if you want another viewpoint on this topic. There are also a wide

plethora of videos and articles about every topic I discussed in this paper so you can do your own research if you feel like I explained something so briefly.

Many thanks go to Tom, Louis and the others that helped me along the way with design choices and motivational support. Without them this project would have been tossed in the bin way before I would have ever really started with it.

I also want to thank you for reading or even just looking at this, as my goal was to inspire others to make their own creations and bring their ideas to real life. If any other questions arise you can always ask in the forums of the many software's and guides that were used, as the people are most likely happy to help you out.

Thank you for reading,

Elias Glauert

References for the Work

SORT ALL OF THESE ONCE YOU HAVE FINISHED THE PAPER

1. “I Built My Dream Keyboard from Absolute Scratch” by Cristian Selig under the link <https://www.youtube.com/watch?v=7UXsD7nSfDY> where you can also find his parts/files and socials in the description.
2. “Staggered VS Ortholinear keyboard, what are the differences” by Tech Fairy under the link <https://tech-fairy.com/staggered-vs-ortholinear-keyboard-what-are-the-differences/> where a nice visualization of staggered and ortholinear keyboards are to be found.
3. The website of nicekeyboards who make the nice!nano under the link <https://nicekeyboards.com/nice-nano> where you can find all the technical information on the nice!nano and where to buy the microcontroller.
4. “Common controllers for keyboard building 2021” by Golem under the link <https://golem.hu/guide/controllers/> which includes a list of controllers that can be used to build keyboards.
5. “ZMK Power Profiler” by ZMK Firmware under this link <https://zmk.dev/power-profiler> can show you how long your keyboard will last until the batteries need to be charged again.
6. The Gazzew Boba U4 Silent Switches can be found under this link: <https://thocstock.com/switches/gazzew-boba-u4-silents> (Last Accessed 22 September 2024)
7. These are the links to the different switch-options that I mentioned in that part of the paper:
 - a. Akko Penguin Silent: <https://geekboards.de/shop/akko-penguin-akko-penguin-silent-3239?variant=4414>
 - b. Zeal PC Zilents V2

- i. 62g Version: <https://geekboards.de/shop/zt0348-zeal-pc-zilents-v2-62g-413?variant=1197>
 - ii. 65g Version: <https://geekboards.de/shop/zt0349-zeal-pc-zilents-v2-65g-412?variant=1196>
 - c. Glorious Gateron Brown: <https://www.caseking.de/glorious-gateron-brown-switches-120-pieces/GAKC-052.html>
 - d. MX RGB Ergo Clear: <https://www.cherry.de/mx-rgb-ergo-clear-switch-kit>
 - e. TTC Venus 45g Linear: <https://mechanicalkeyboards.com/products/ttc-venus-45g-linear-pcb-mount-switch>
 - f. Everglide Aqua King V3:
<https://everglide.co/collections/switches/products/everglide-aqua-king-water-king-switches-v3?variant=42887000326356>
 - g. All these links were last accessed on 22 September 2024
8. Ergogen can be found under <https://ergogen.xyz>, but I like <https://ergogen.ceoloide.com/> more, as it gives you a better user interface.
9. The GitHub repository can be found here: <https://github.com/lulton/pengo-keyboard>, which contains all the files that were used and created in this project.

