České vysoké učení technické v Praze

Fakulta elektrotechnická

Katedra počítačů

XY Plotter

Bakalářská práce

2024

Autor: Patrik Novák

Vedoucí práce: Doc. Ing. Stanislav Vítek, Ph.D.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Novák** Jméno: **Patrik** Osobní číslo: **492400**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

Specializace: **Technologie internetu věcí**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

XY plotter

Název bakalářské práce anglicky:

XY plotter

Pokyny pro vypracování:

Cílem bakalářské práce je návrh a implementace XY plotteru, tj. konstrukce, řídící jednotky a firmware při použití běžně dostupných komponent a technologie 3D tisku. Součástí návrhu je výběr vhodné kinematiky a platformy pro implementaci řídící jednotky. Zařízení otestujte v reálném provozu a diskutujte případné nedostatky.

Seznam doporučené literatury:

[1] Ilan E. Moyer, „CoreXY | Cartesian Motion Platform". [Online]. Dostupné z: http://corexy.com/
[2] J. E. Bresenham, „Algorithm for computer control of a digital plotter", IBM Systems Journal, roč. 4, s. 25–30, 1965
[3] Jiří Velebil, Abstraktní a konkrétní lineární algebra. 2023. [Online]. Dostupné z: https://math.fel.cvut.cz/en/people/velebil/akla.html

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Stanislav Vítek, Ph.D. katedra radioelektroniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **15.02.2024** Termín odevzdání bakalářské práce: **24.05.2024**

Platnost zadání bakalářské práce: **21.09.2025**

doc. Ing. Stanislav Vítek, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 23.05.2024

.....

Poděkování

Chtěl bych poděkovat mému vedoucímu Doc. Ing. Stanislav Vítkovi, Ph.D. za poskytnutí prostorů laboratoře, kde mohl projekt vznikat a také za cenné poznámky a rady k projektu, které mne směřovaly tím správným směrem.

Abstrakt

Bakalářská práce se zabývá kompletním návhem XY Plotteru, který umí kreslit tužkou, perem nebo fixou. Návrh představí originální kombinaci konstrukce, kinematiky a řídícího mikrokontroléru. Jsou popsány základní principy řízení XY Plotteru a jejich implementace ve vlastním firmwaru. Plotter je podroben benchmarku, který ověří základní parametry rozlišení a přesnosti.

Klíčová slova: plotter, Raspberry Pi Pico, CoreXY

Abstract

Bachelor thesis deals with the complete design of XY Plotter, which can draw with pencil, pen or marker. The design will present an original combination of design, kinematics and control microcontroller. The basic principles of XY Plotter control and their implementation in custom firmware are described. The Plotter is benchmarked to verify basic resolution and accuracy parameters.

Keywords: plotter, Raspberry Pi Pico, CoreXY

Obsah

1 Úvod

XY Plotter je typ mechanického robota, který využívá dvě osy, X a Y, k přesnému pohybu nástroje po dvourozměrné ploše. Díky tomu je schopen kreslit různé grafické tvary, technické výkresy, nebo dokonce psát texty. Na rozdíl od tiskárny je Plotter schopný kreslit i na jiná média než je obyčejný papír, například na dřevěné desky, stěny nebo na sklo. Navíc daný povrch nemusí být úplně rovný, protože pružinka generující přítlak popisovače dokáže nerovnosti kompenzovat. Nástrojem takového Plotteru je typicky obyčejná propiska, tužka nebo fixa. Výběr nástroje však není omezen pouze těmito možnostmi. Na Plotter lze snadno umístit například laser pro vyřezávání nebo gravírování, kamera pro skenování objektů, anebo pokud je konstrukce Plotteru dostatečně pevná, lze na něj umístit i frézku.

V současné době existuje na internetu mnoho návodů a projektů, podle kterých si lze plotter postavit v domácích podmínkách. Většina z těchto projektů je založena na platformě Arduino UNO [3] v kombinaci s firmwarem GRBL [4] a používá se kinematika, kdy se každá osa řídí odděleně a má dedikovaný motor.

Cílem této práce je navrhnout plotter na jiné platformě a s jinou kinematikou, který bude open-source a dostupný tak, aby si ho mohl v domácích podmínkách sestavit kdokoliv.

V kapitole 2 jsou podrobněji rozebrány požadavky, se kterými je plotter designován. Kapitola 3 se věnuje současnému stavu na poli plotterů a porovnává jejich vlastnosti. V kapitole 4 je věnována pozornost návrhu konstrukce, kinematiky a mechanismu pro uchycení popisovače. Výběr hardwaru potřebného pro řízení Plotteru a návrh řídící desky je popsán v kapitolách 5 a 6. Kapitola 7 pojednává o návrhu vlastního firmware a rozebírá použité návrhové vzory. Jak lze Plotter ovládat a jak připravit data pro Plotter je ukázáno v kapitole 8. V kapitole 9 je proveden a vyhodnocen benchmark Plotteru, který se zaměřuje na přesnost a rozlišení. Kapitola 10 obsahuje seznam komponent a přibližnou kalkulaci ceny projektu.

2 Požadavky na plotter

Požadavek 1: Rozhraní pro řízení

Plotter bude řízen pomocí standardizovaných příkazů G. [5]

Požadavek 2: Konstrukce

Konstrukce Plotteru bude kompaktní a bude tvořena ramenem uchyceném z jedné strany.

Požadavek 3: Komponenty konstrukce

Komponenty, ze kterých se Plotter skládá, budou běžně dostupné nebo vyrobeny pomocí technologie 3D tisku.

Požadavek 4: Kinematika

Plotter bude implementovat kinematiku typu CoreXY.

Požadavek 5: Kompatibilita s nástroji třetích stran

Plotter bude možno používat s již dostupnými generátory G-Code.

Požadavek 6: Mikrokontrolér

Jako MCU bude použito RPi Pico nebo jiná MCU založená na čipu RP2040.

Požadavek 7: Rozměry kreslící plochy

Kreslící plocha plotteru bude rozměru alespoň A4.

Požadavek 8: Přesnost

Stroj bude kreslit s přesností do jednoho milimetru.

Požadavek 9: Jednoduchost složení

Plotter bude designován tak, aby jeho konstrukce byla jednoduchá na složení i údržbu.

3 Současný stav

Na internetu mnoho návodů a projektů, podle kterých si lze plotter postavit v domácích podmínkách. Většina z těchto projektů je založena na platformě Arduino UNO [3] v kombinaci s firmwarem GRBL [4] a používá se kinematika, kdy se každá osa řídí odděleně a má dedikovaný motor. Existují i hotové komerční produkty, ale jejich nevýhodou je, že nejsou open-source.

Plotter s popisovačem je možné využít k různým účelům, například k převodu digitálních obrázků na plátno, k psaní personalizovaných dopisů či podepisování.

Možnosti Plotteru však nekončí pouze u překreslování a psaní. Zařízení implementuje obecný pohyb v rovině XY a místo uchycení pro popisovač je možně přimontovat i jiné věci a rozšířit tak jeho využití. Na Plotter lze snadno umístit například laser pro vyřezávání nebo gravírování, kamera pro skenování objektů, anebo pokud je konstrukce Plotteru dostatečně pevná, lze na něj umístit i frézku.

3.1 GRBL

GRBL je open-source firmware pro řízení CNC strojů určený pro použití s mikrokontrolérem Arduino UNO. Je napsán v jazyce C a implementuje asynchronní řízení krokových motorů i s lineární akcelerací pro plynulý pohyb. Firmware v sobě také implementje interpret standardního G-Kódu. [4]

Protože GRBL obsahuje spoustu optimalizací, implementuje velké množství funkčností a zaměřuje se obecně na CNC stroje, ne pouze na plottery, nejsou na první pohled zřejmé základní princypy ovládání takového Plotteru. To je jeden z důvodů proč bylo rozhodnuto o vytvoření vlastního firmware.

3.2 Arduino UNO

Arduino UNO je populární mikrokontrolér používaný pro různé elektronické projekty a prototypování. Je součástí open-source platformy Arduino, která je navržena pro jednoduché použití jak pro začátečníky. [3]

3.3 Simple Engraver

Simple Engraver[1] je projekt laserové gravírovačky založené na platformě Arduino v kombinaci s firmwarem GRBL. Jeho konstrukce je velmi rigidní a je určena pro upevnění na jednom místě. Pohyb v jednotlivých osách je řízen separátně. Komponenty, ze kterých je projekt poskládán jsou buď volně dostupné, anebo vytištěny na 3D tiskárně. Projekt je dostupný jako open-source.

3.4 Makeblock XY Plotter

Makeblock XY Plotter[2] je dostupný jako kit pro sestavení. Rám konstrukce má tvar do čtverce a pohyb v jednotlivých osách je řízen odděleně dedikovaným krokovým motorem. Komponenty konstrukce jsou vyrobeny na míru a nejsou dostupné jinde než v kitu. Firmware je založen na

[1] https://github.com/Adamslab/SimpleEngraver
[2] https://www.hwkitchen.cz/xy-plotter-robot-kit/

platformě Arduino a je dostupný jako open-source projekt. Tento plotter dokáže popsat plochu o velikosti 310 mm × 390 mm a jeho cena se pohybuje kolem 8000 Kč.

3.5 AxiDraw

AxiDraw [6] je komerčně dostupný plotter, který se dá koupit plně sestavený. Používá kinematiku CoreXY v implementaci, kdy je použit jeden řemen a rameno plotteru se vysouvá i dozadu. Důsledkem toho je, že se plotter nemůže umístit na místa, kde není dostatek prostoru na tento zadní výsuv. Tento plotter používá vlastní proprietární firmware a elektroniku. Cena za verzi, která zvládne popsat plochu o velikosti formátu A4, se pohybuje kolem 14 000 Kč.

4 Konsktrukce

Konstrukce je inspirována 3D tiskárnami Prusa [7] a Voron [8] a skládá se z kombinace běžně dostupných dílů a 3D tištených dílů. Díky tomu jsou minimalizovány náklady na sestavení a také je zajištěna dostupnost.

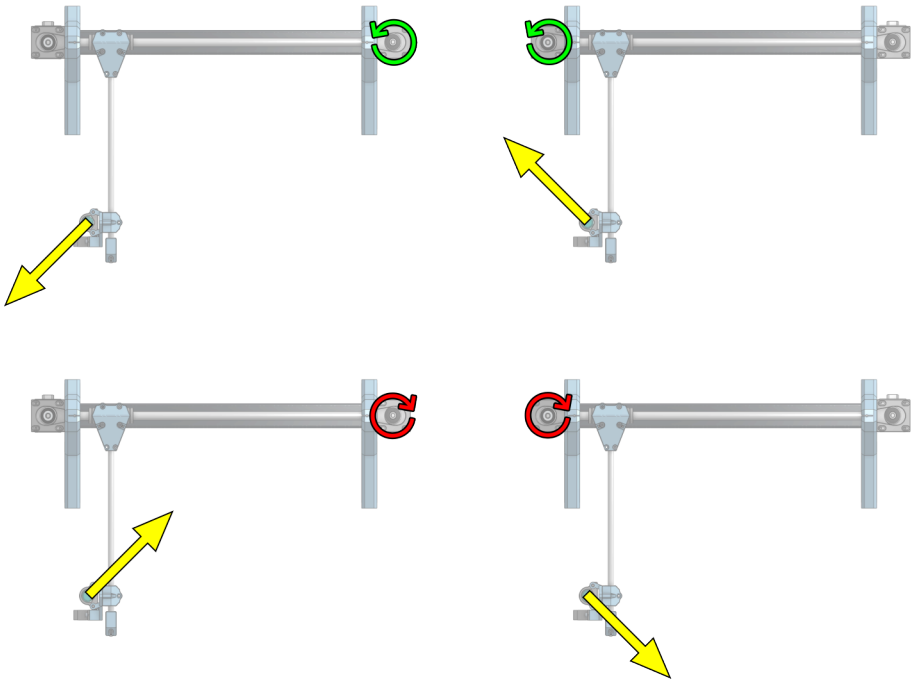
Rám konstrukce je tvořen hliníkovým profilem 30x30, který celou konstrukci vystužuje a umožňuje přimontování Plotteru například k držáku. Na jeho koncích jsou 3D tištěné díly, které slouží jako nožičky a zajišťují stabilitu. Z boku jsou navíc umístěny držáky pro krokové motoru a dovnitř těchto bočních dílů jsou zapuštěny hlazené tyče o průměru 10 mm, tvořící osu X. Na ose X se nachází vozík tvořený dalším 3D tištěným dílem, ve kterém je upevněno rameno osy Y. Rameno osy Y je stejně jako osa X tvořena hlazenými tyčemi, ale o menším průměru 8 mm. Na ose Z se nachází vozík, kde je umístěn držák popisovače se servem pro přízved, a také jsou v něm uchyceny všechny konce řemenů.

Byl použit řemen typu GT2 o průměru 6 mm a pro jeho vedení byly použity kladky tvořené ze 2 ložisek F623ZZ.

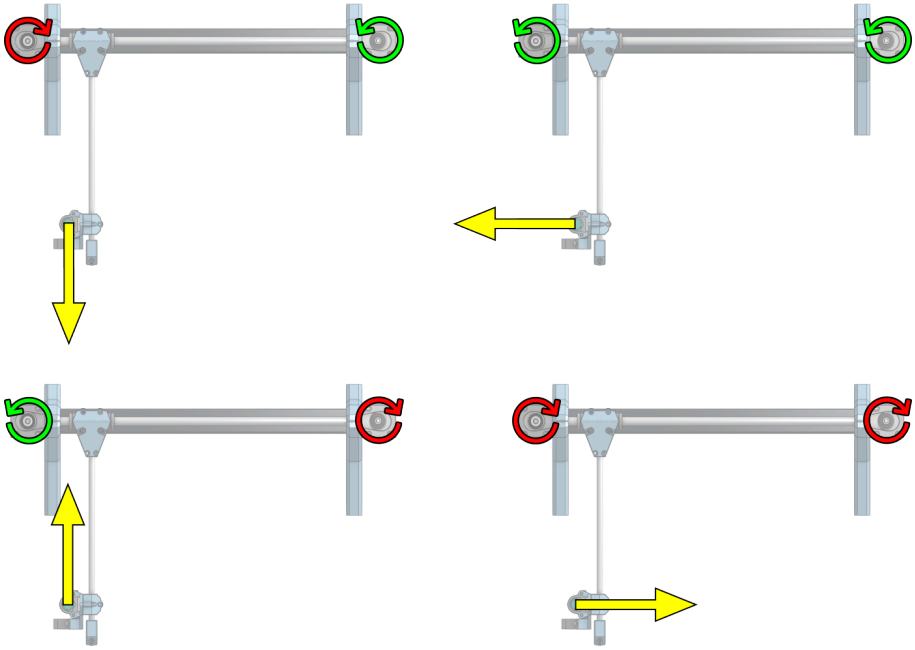
4.1 Kinematika

Plotter využívá kinematiku typu CoreXY [9]. V této kinematice je pro pohyb v jedné ose třeba otáčet oběma motory současně. Další vlastností je upevnění obou motorů staticky, tj. že ani jeden motor není umístěn na pohyblivé ose. Tyto vlastnosti umožňují symetrický a kompaktní design. Díky menší pohybující se hmotě je možné motory otáčet s vyšší akcelerací bez ztráty kroků.

Nevýhoda kinematiky CoreXY spočívá především ve větší složitosti implementace. Souřadnicový systém je pootočený o násobek 45°, a proto je třeba při zpracovávání G-Kódů provádět transformaci souřadnicového systému. Celková délka řemenu je oproti klasické kinematice zhruba dvojnásobná, jejich vedení je složitější, a je potřeba jejich preciznější seřízení, neboť rozdílné napnutí řemenů způsobí deformaci trajektorie pohybů. Na obrázku 1 a obrázku 2 jsou znázorněny pohyby v kinematice CoreXY.



Obrázek 1: Směr posunu při pohybu jednoho motoru

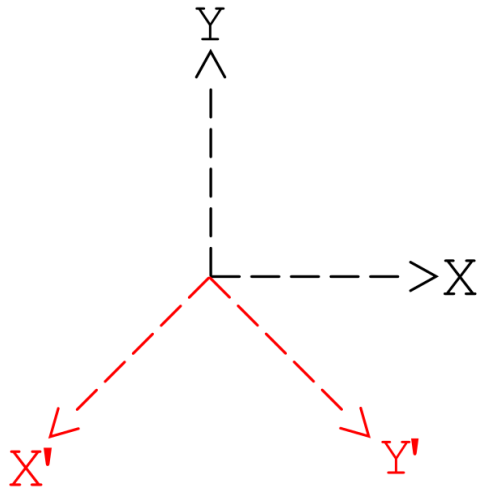


Obrázek 2: Směr posunu při pohybu 2 motorů zároveň

Byly použity 2 řemeny, které jsou vedeny přímo nad sebou ve dvou úrovních. První úroveň je napojena na jeden motor a druhá úroveň na motor druhý. Všechny konce řemenů jsou uchyceny ve vozíků na kterém je přimontovaný i držák popisovače. Každý řemen je veden podél obou os. Vzájemná pozice uchycení konců řemenů musí být do kříže, tj. když konce jednoho řemenu jsou vpravo nahoře, tak konce druhého řemenu musí být na vozíku připevněny vlevo dole.

4.1.1 Transformace souřadnic

Díky vlastnostem popsaným výše nelze jednotlivé souřadnice ze vstupu přímo přepočítávat na kroky motoru, ale je potřeba předtím provést jejich transformaci do souřadnicového systému, který odpovídá dané kinematice.



Obrázek 3: Vzájemná poloha souřadnicových systémů

Souřadnicový systém z obrázku 3 tvořený osami XY odpovídá souřadnicím, které dostaneme ze vstupu. Souřadnicový systém X'Y' vnitřně používa Plotter a je třeba souřadnice ze vstupu do něj převést.

Pro převod ze systému XY do systému X'Y' použijeme lineární transformaci užitím matice rotace [10]:

$$\begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix}$$

Samotná rotace souřadnicového systému však nestačí, protože jednotky v daných souřadnicových systémech jsou jinak velké. Je to vlastnost kinematiky CoreXY. Když se pohne jedním motorem o jeden krok, tak se vozík pohne o vzdálenost odpovídající jednomu kroku vynásobenou $\frac{1}{\sqrt{2}}$. Abychom tuto skutečnost kompenzovali použijeme lineární transformaci, která je reprezentována následující maticí:

$$\begin{pmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix}$$

Výslednou transformaci dostaneme složením dvou zmíněných transformací. Z obrázku 3 vyplývá, že budeme rotovat o 225°.

$$\begin{pmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{pmatrix} \begin{pmatrix} \cos(225°) & -\sin(225°) \\ \sin(225°) & \cos(225°) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix}$$

Po úpravách dostaneme výsledý vztah:

$$\begin{pmatrix} x - y \\ x + y \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix}$$

4.2 Zdvih popisovače

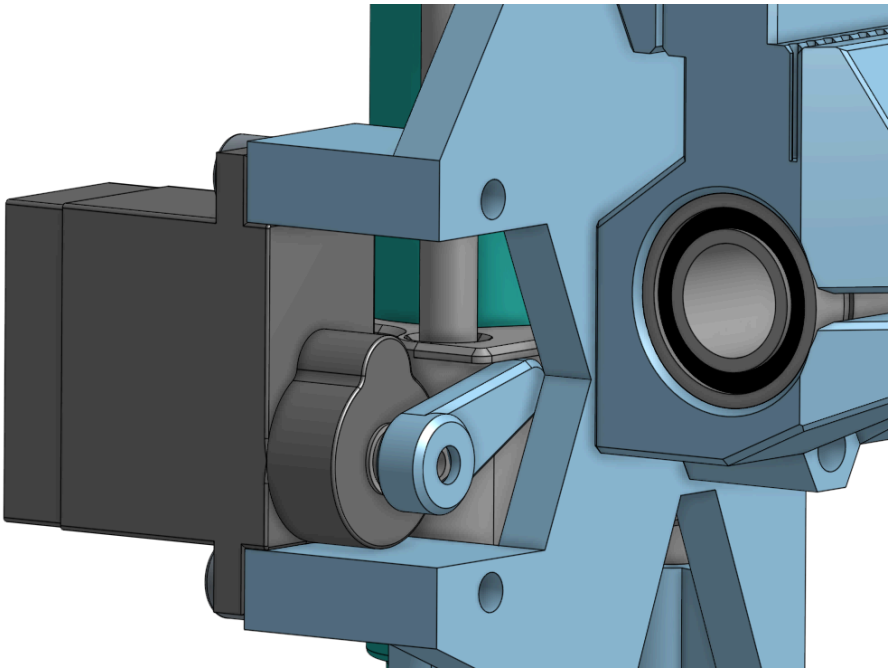
Zdvih popisovače je řešen pomocí jednoduchého mechanismu. Držák popisovače je umístěn na dvou ocelových tyčích tak, aby se s ním dalo pohybovat ve vertikální ose. Na jedné z tyčí je pružinka, která zajišťuje přítlak popisovače k podložce. Zdvih popisovače je potom zajištěn pomocí serva, které držák s popisovačem pouze přizvedává. Mechanismus zdvihu je vidět na obrázku 4.

4.3 Teoretická přesnost Plotteru

Přesnost plotteru záleží na několika faktorech, zejména na průměru řemenice a počtu kroků na otáčku krokového motoru. Dalšími faktory, které ovlivňují přesnost jsou preciznost konstrukce, vůle v ložiskách a pnutí řemenu. Tyto faktory však v rámci teoreticky dosažitelné přesnosti započítávat nebudeme. Vzorec pro výpočet vzdálenosti s o kterou se pohne řemen za jeden krok vypadá následovně:

$$s = \pi d \frac{\alpha}{360} m$$

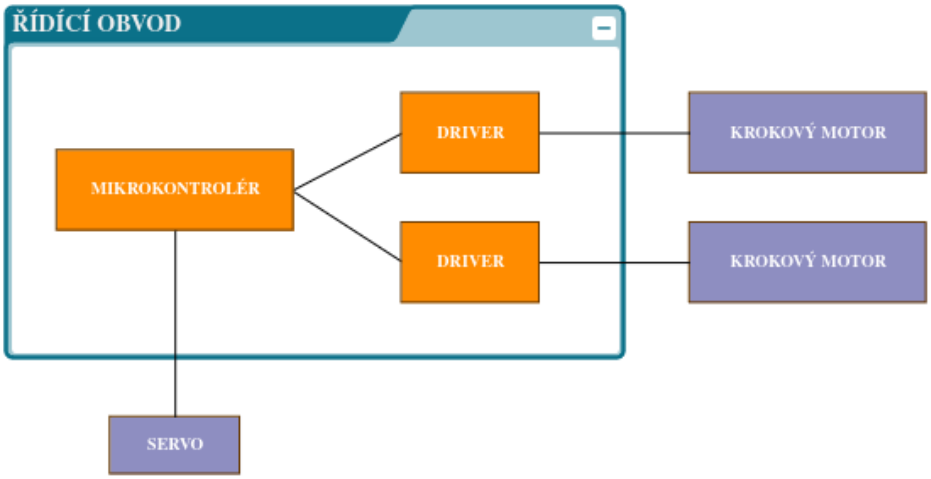
α je úhel, o který se pootočí hřídel motoru při jednom kroku ve stupních, m je mikrokrokování. Ze vztahů pro transformaci souřadnic vyplývá, že vzdálenost s přesně odpovídá vzdálenosti, o kterou se pohne vozík.



Obrázek 4: Detail mechanismu pro zdvih popisovače

5 Hardware

Pohyb Plotteru v rovině je typicky zajištěn pomocí krokových motorů a pro zdvih popisovače se nejčastěji používá servo. Tyto komponenty jsou řízeny mikrokontrolérem. Pro řízení krokových motorů je navíc potřeba driver. Servo může být řízeno přímo z mikrokontroléru, ale jeho napájení musí být oddělené. Na obrázku 5 je znázorněno blokové schéma Plotteru.



Obrázek 5: Blokové schéma XY Plotteru

5.1 Mikrokontrolér

Mikrokontrolér Raspberry Pi Pico byl vybrán z několika klíčových důvodů. Především je známý svou skvělou dokumentací, která výrazně usnadňuje vývoj a integraci jednotlivých komponent. Díky možnosti „drag and drop“ programování je nahrávání kódu jednoduché a uživatelsky přívětivé, což šetří čas a zjednodušuje celý vývojový proces. Integrované USB-UART rozhraní umožňuje snadnou komunikaci s počítačem a dalšími zařízeními. Nízká cena a dobrá dostupnost Raspberry Pi Pico z něj činí ekonomicky výhodnou volbu.

5.2 Krokové motory

Pro Plotter byly vybrány standardní krokové motory Nema17[3] se statickým momentem 0.4 Nm a rozlišením 200 kroků na otáčku. Jejich předností je zejména dobrá dostupnost a nízká cena.

5.3 Drivery krokových motorů

5.3.1 A4988

Driver **A4988**[4] je základním, a s cenou kolem 60 Kč, nejlevnějším driverem pro ovládání krokových motorů s rozhraním rozhraním STEP/DIR. Umožňuje mikrokrokování až v rozlišení 1/16, což je pro účely Plotteru dostačující. Nevýhodou tohoto driveru je velká hlučnost a přehřívání.

[3] https://www.laskakit.cz/krokovy-motor-nema-17-17hs4401-0-4nm/
[4] https://www.laskakit.cz/a4988-driver-pro-krokove-motory/

5.3.2 DRV8825

Driver **DRV8825**[5] je vylepšenou verzí driveru **A4988**. S cenou kolem 80 Kč se řadí k těm nejlevnějším. Na rozdíl od **A4988** umožňuje jemnější mikrokrokování a to až 1/32. Ikdyž nepotřebujeme větší přesnost, tak jemnější mikrokrokování zajistí plynulejší a tišší pohyb motorů. Bohužel i přes vylepšení oproti předchozímu dirveru je driver **DRV8825** při používání stále velmi hlučný a přehřívá se.

5.3.3 TMC2209

Driver **BTT TMC2209**[6] umožňuje ovládání nejen přes rozhraní STEP/DIR, ale i přes rozhraní UART. Rozlišení mikrokrokování se dá nastavit až na 1/64 pomocí externích pinů MS1 a MS2 a až na 1/256 pomocí rozhraní UART. Je sice v porovnání dražší (cca 140 Kč), ale obsahuje více funkcionalit, z nichž hlavní jsou StealthChop, StallGuard a CoolStep. StealthChop zajišťuje velmi tichý chod motorů. StallGuard je funkce, která dokáže detekovat přetížení motoru, například v důsledku toho, když vozík narazí na konec vedení. Tato funkce umožňuje implementovat *Sensorless Homing*, jak název napovídá, bez koncových spínačů. Funkce CoolStep automaticky nastavuje optimální proud pro řízení motoru. Využitím informací ze StallGuard nastavuje proud na nejnižší požadovaný a tím šetří energii a zajišťuje nižší teplotu komponent. [11]

Tabulka 1: Nastavení mikrokrokování TMC2209 pomocí pinů MS1 a MS2

MS1	MS2	Počet mikrokroků
GND	GND	8
GND	VCC_IO	64
VCC_IO	GND	32
VCC_IO	VCC_IO	16

Pro Plotter byl vybrán driver **TMC2209** díky tichému chodu a lepším termálním vlastnostem. Ikdyž všechny funkcionality nejsou ve firmwaru implementovány, poskytují prostor pro budoucí rozšíření bez nutnosti upgradovat hardware.

5.4 Servo

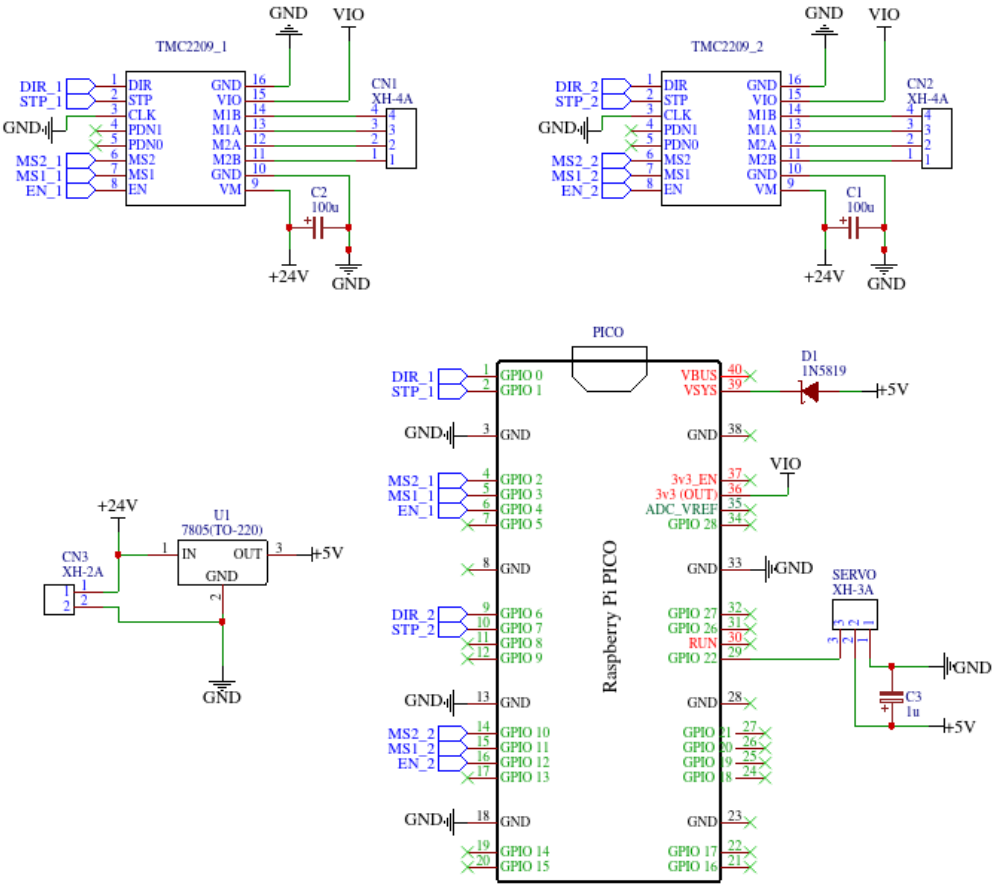
Pro obsluhu zdvihu popisovače bylo vybráno servo MG90S[7] s kovovými převody. Jedná se o dostupné servo s nízkou cenou, které je spolehlivější než servo SG90 s plastovými převody.

[5] https://www.laskakit.cz/drv8825-driver-pro-krokove-motory/
[6] https://www.hotend.cz/p/bigtreetech-tmc2209-driver-v13-uart
[7] https://www.laskakit.cz/mini-servo-mg90s-s-kovovymi-prevody/

6 Řídící obvod

Řídící obvod se skládá z mikrokontroléru Raspberry Pi Pico, dvou driverů krokových motorů BTT TMC2209 a lineárního regulátoru napětí LM7805. Lineární regulátor napětí je použit pro napájení mikrokontroléru a serva. Pro napájení serva však není vhodný, protože má nízkou účinost a při větších odběrech proudu se přehřívá. Místo lineárního regulátoru je doporučeno použít *step down* měnič, který dosahuje vyšší účinosti.

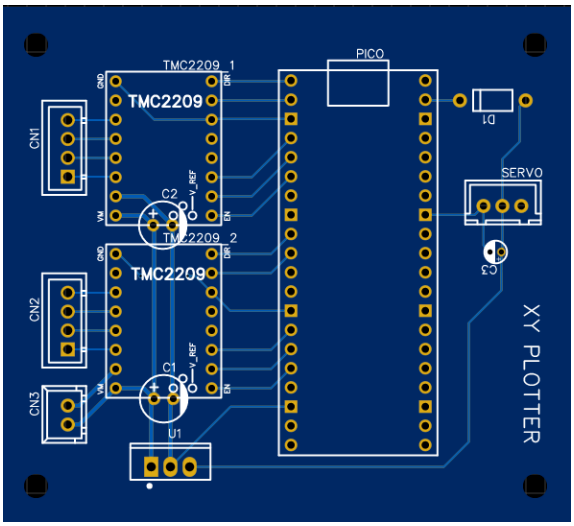
Na k napájecímu vstupu VSYS byla přidána Schottkyho dioda, která zabraňuje zpětným proudům při napájení z USB a VSYS současně tak, jak je popsáno v datasheetu. [12]



Obrázek 6: Schéma zapojení

6.1 Deska plošných spojů

Pro Plotter byla navržena jednoduchá deska plošných spojů, která slouží k propojení všech modulů dohromady. Výhodou průmyslově vyrobené desky spojů oproti ručně pájené desce je jednoznačně spolehlivost spojů zároveň je jednodušší na osazení, protože jsou na ní vytištěny obrysy a popisky jednotlivých komponent. Deska byla objednána z JLCPCB[8] v objemu 5 kusů za cenu, včetně dopravného, cca 100 Kč.



Obrázek 7: Design desky plošných spojů

[8] https://jlcpcb.com/

7 Software

7.1 TMC2209 HAL

Pohyb popisovače v rovině XY zajišťují 2 krokové motory NEMA17 společně s jejich drivery TMC 2209. Pro tyto drivery byl ve firmwaru vytvořen jednoduchý HAL ve formě třídy `TMC2209`. HAL implementuje metody, pomocí kterých je možné nastavit mikrokrokování, nastavit směr rotace, aktivovat či deaktivovat driver a pootočit motorem o jeden krok.

Mikrokrokování se nastavuje pomocí pinů `MS1` a `MS2` na driveru. Jednotlivé kombinace nastavení a k nim odpovídající mikrokrokování je vidět v obrázku 1.

Defaultní hodnota, která je ve firmwaru nastavena a která bude použita v dalších výpočtech je 64 mikrokroků.

7.2 Ovládání posunu popisovače

Posun popisovače je realizován sekvencí kroků jednoho a druhého motoru. Přesná sekvence se generuje ve 2 krocích.

1. Převod milimetrů na kroky
2. Aplikování transformace z kapitoly 4.1.1
3. Generování posloupnosti kroků z počátečního a koncového bodu

7.2.1 Převod milimetrů na kroky

Pro převod vzdálenosti na počet kroků krokového motoru je třeba znát počet kroků na jednu otáčku a nastavení mikrokrokování driveru.

Pro převod vzdálenosti v milimetrech na počet kroků využijeme vztah:

$$s = \pi d \frac{\alpha}{360} \frac{1}{m}$$

kde s je vzdálenost o kterou se pohne řemen po otočení hřídele o jeden krok, α je úhel o který se pootočí hřídel motoru při jednom kroku ve stupních, m je mikrokrokování. Po dosazení parametrů odpovídajím komponentům Plotteru ($d = 10$ mm, $\alpha = 1.8°, m = 64$) spočteme, že jeden krok odpovídá posunu o 0.00245 mm. Převod milimetrů na kroky se poté provede celočíselným vydělením hodnoty ze vstupu hodnotou, kterou jsme právě spočetli.

7.2.2 Chyba při převodu milimetrů na kroky

Převod se provádí ze spojitého prostoru, do prostoru, který spojitý není. To znamená, že rozdílné hodnoty v prvním prostoru se mohou zobrazit jako jedna hodnota v prostoru druhém. Například hodnoty 0.002 mm a 0.003 mm se promítnou jako vzdálenost 1 kroku. Je to chyba, která vzniká zaokrouhlováním. Tato chyba může vznikat v obou osách zároveň a tudíž její maximum nastane v případě, že se trefíme doprostřed čtverce, jehož vrcholy jsou tvořeny souřadnicemi sousedních kroků. Maximální chybu můžeme spočítat jako polovinu úhlopříčky tohoto čtverce.

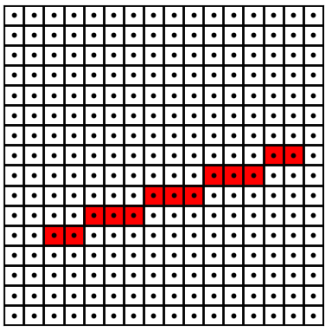
$$\text{err} = \frac{s}{\sqrt{2}}$$

Pro $s = 0.00245$ mm je maximální chyba převodu 0.00173 mm, což je hodnota, která je pro účely Plotteru v podstatě zanedbatelná.

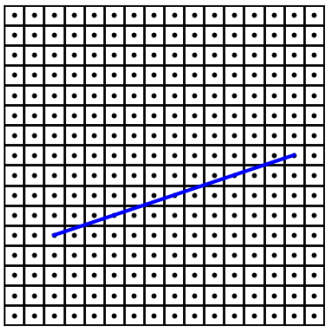
7.2.3 Bresenhamův algoritmus

Vykreslení rovné úsečky se v XY plotterech běžně řeší pomocí Bresenhamova algoritmu [13] pro rasterizaci úseček. Tento algoritmus má původní využití v rastrové grafice, kde je obrázek tvořen pixely uspořádanými do mřížky. Když chceme nakreslit rovnou úsečku mezi dvěma body, která je navíc mírně nakloněná, nelze to udělat tak, že z počátečního bodu budeme kreslit přímo ve směru k bodu koncovému. V mřížce lze jít pouze 8 směry. Bresenhamův algoritmus v každém kroku vybere nejvhodnější směr, kterým kreslit tak, že po skončení se zabarvené pixely jeví jako rovná úsečka.

Přesně stejný postup lze použít při kreslení plotterem. Místo mřížky tvořené jednotlivými pixely zde máme mřížku tvořenou jednotlivými kroky krokových motorů. Bresenhamův algoritmus potom rozhoduje o tom, který motor má udělat krok.



Obrázek 8: Šikmý posun v mřížce



Obrázek 9: Šikmý posun spojitě

Bresenhamův algoritmus je ve firmaru Plotteru implementován třídou `Line`, která implementuje návrhový vzor iterátor [14]. Průchodem iterátoru dostaneme postupně všechny souřadnice jak je vidět na obrázku 8.

7.3 Synchronní ovládání krokových motorů

Pro správnou funkčnost Plotteru je třeba ovládat oba motory současně. Implementace Bresenhamova algoritmu nám tuto práci výrazně zjednodušuje, protože posloupnost souřadnic, kterou iterátor generuje se ve složkách liší vždy maximálně o 1. Díky tomu stačí iterovat přes všechny souřadnice a provést krok pouze pokud se aktuální souřadnice z motoru neshoduje se souřadnicí z iterátoru.

```
for ( const auto& point : line ) {
      if (point.x != current.x) {
          motorX.step();
      }
      if (point.y != current.y) {
          motorY.step();
      }
      current = point;
      sleep_us(delay);
}
```

Ukázka kódu 1: Synchronní pohyb motorů

7.4 Ovládání zdvihu popisovače

Vertikální pohyb popisovače je realizován pomocí serva MG90S. Parametry tohoto serva jsou:

- Provozní Napětí:4.8-6.0V
- Úhel natočení: 180°
- Střída: 0.5 až 2.5 ms
- Frekvence: 50 Hz

Pro správné a efektivní nastavení ovládání serva je třeba nejprve vypočítat příslušné parametry. K tomu je třeba znát taktovací frekvenci RPi Pico (125 MHz) a rozlišení PWM na RPi Pico (16 bit = 65537).

Nejprve vypočítáme ideální frekvenci PWM, při které by se využil celý 16 bitový rozsah. Tuto hodnotu použijeme jako horní mez pro výpočet dělícího poměru.

$$\text{clk\_h} = 50\ \text{Hz} * 65537 = 3276850\ \text{Hz}$$

Dělící poměr poté dostaneme jako:

$$\text{div} = \frac{125\ \text{MHz}}{3276850} = 38.146\ \sim 40$$

Výsledný dělící poměr je třeba zaokrouhlit na nejbližší vyšší celé číslo takové, aby výsledek dělení taktovací frekvence a dělícího poměru vyšel jako celé číslo. Ze získaného dělícího poměru dále vypočítáme novou taktovací frekvenci po dělení a parametr wrap pro periferii PWM.

$$\text{clk\_new} = \frac{125\ \text{MHz}}{40} = 3125\ \text{kHz}$$

$$\text{wrap} = \frac{3125\ \text{kHz}}{50\ \text{Hz}} = 62499$$

Z vypočítaných parametrů se pak snadno dopočítají úrovně překlopení pro příslušné úhly natočení. [15]

Tabulka 2: Vztah střídy a polohy serva

Střída [ms]	Úhel [deg]	Úroveň překlopení
0.5	-90	1562
1	-45	3125
1.5	0	4167
2	45	6250
2.5	90	7812

Nastavení PWM pro servo v kódu probíhá v konstruktoru třídy `MG90S`:

```
MG90S::MG90S(uint pin)
    : pin(pin),
      pwmSlice(pwm_gpio_to_slice_num(pin)),
      pwmChannel(pwm_gpio_to_channel(pin))
{
    gpio_set_function(pin, GPIO_FUNC_PWM);

    // assuming that pico runs at 125 000 000 Hz
    pwm_set_clkdiv(this->pwmSlice, MG90S_PWM_CLK_DIV);
    pwm_set_wrap(this->pwmSlice, MG90S_PWM_WRAP);
}
```

Ukázka kódu 2: Konstruktor třídy MG90S

7.5 Zpracování příkazů

7.5.1 Návrhový vzor Visitor [1]

Návrhový vzor Visitor umožňuje oddělit implementaci algoritmů mimo objekt, se kterým pracují.

Ve firmwaru pro Plotter tento návrhový vzor umožňuje implementaci akcí pro jednotlivé G-Kódy přímo ve třídě `Plotter`. Dalším příjemným benefitem je snížení couplingu. Kdyby se někdy v budoucnu měl parser G-Kódu použít s jiným projektem než XY Plotterem, nebude potřeba v parseru dělat žádné změny, pouze postačí, když daný projekt bude implementovat potřebné G-Kódy.

```
class Plotter
{
public:
    ....
    execute(G0 *g0)
    {
      move(g0->x, g0->y);
    }

    execute(G1 *g1)
    {
      ...
    }
    ...
}

class G0
{
public:
    double x, y;
    ...
    execute(Plotter *plotter)
    {
      plotter->execute(this);
    }
}
```

Ukázka kódu 3: Užití návrhového vzoru Visitor

Ukázka kódu 3 znázorňuje konkrétní připad užití návrhového vzoru visitor ve firmwaru pro Plotter. Díky tomuto návrhovému vzoru je možné ve třídách implementujících konkrétní G-Kódy mít pouze atributy pro daný G-Kód bez žádných dalších metod.

7.5.2 Návrhový vzor CRTP [2]

Návrhový vzor CRTP slouží k dosažení „compile-time“ polymorfismu. Jedná se například o případ, kdy rodičovská třída poskytuje interface, jehož implementace je stejná pro všechny odvozené třídy a zároveň se v těle metody volá přetížená funkce, která má jako vstupní parametr pointer na konkrétní instanci odvozené třídy.

Kvůli druhému zmíněnému požadavku není možné použít jednoduchou dědičnost, protože v tom případě by vstupním parametrem dané funkce nebyl pointer na instanci odvozené třídy, ale pointer na třídu rodičovskou.

Požadované funkcionality je dosaženo použitím šablonové třídy, jejímž šablonovým parametrem je třída odvozená. V požadované metodě poté nevoláme funkci s parametrem `this`, ale díky šablonovému parametru můžeme provést `static_cast` na konkrétní odvozený typ, a zajistit volání správné verze přetížené funkce.

```
template<class ConcreteGCode>
class GCode : public AbstractGCode
{
public:
    void execute(Plotter *machine)
    {
        machine->execute(static_cast<ConcreteGCode*>(this));
    }
}

class G0 : public GCode<G0>
{
  ...
}
```

Ukázka kódu 4: Užití návrhového vzoru CRTP

Ukázka kódu 4 reprezentuje konkrétní implementaci návrhového vzoru CRTP ve firmwaru pro Plotter. Šablonová třída `GCode` obsahuje metodu `execute`, jejíž implementace je totožná pro všechny odvozené třídy. Díky užití tohoto návrhového vzoru je minimalizována duplikace kódu a případné budoucí zavádění nových odvozených G-Kódu je zjednodušeno.

7.5.3 Abstrakce G-Kódu

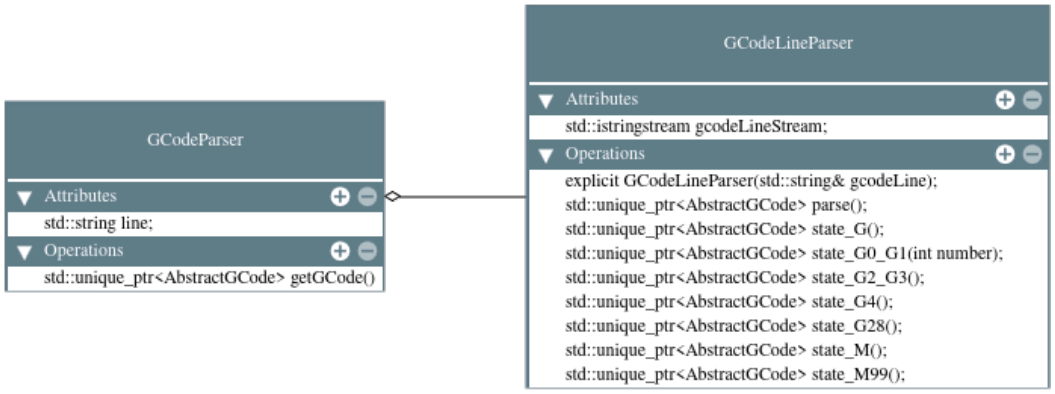
G-Kódy mohou být různých typů. Aby bylo možné načítat různé typy do jednoho bufferu, byla vytvořena abstraktní třída `AbstractGcode` s pure virtual metodami, která definuje interface, které musí implementovat všechny odvozené třídy.

```
class AbstractGCode
{
public:
    virtual void execute(Plotter *machine) = 0;
};

...

Plotter plotter(...);
std::vector<std::unique_ptr<AbstractGcode>> buffer;

while (true) {
  buffer.insert(buffer.begin(), gcodeParser.getGCode());
  buffer.back()->execute(&plotter);
  buffer.pop_back();
}
```

Ukázka kódu 5: Načítání a interpretování různých G-Kódů do bufferu

7.5.4 Parser G-Kódu

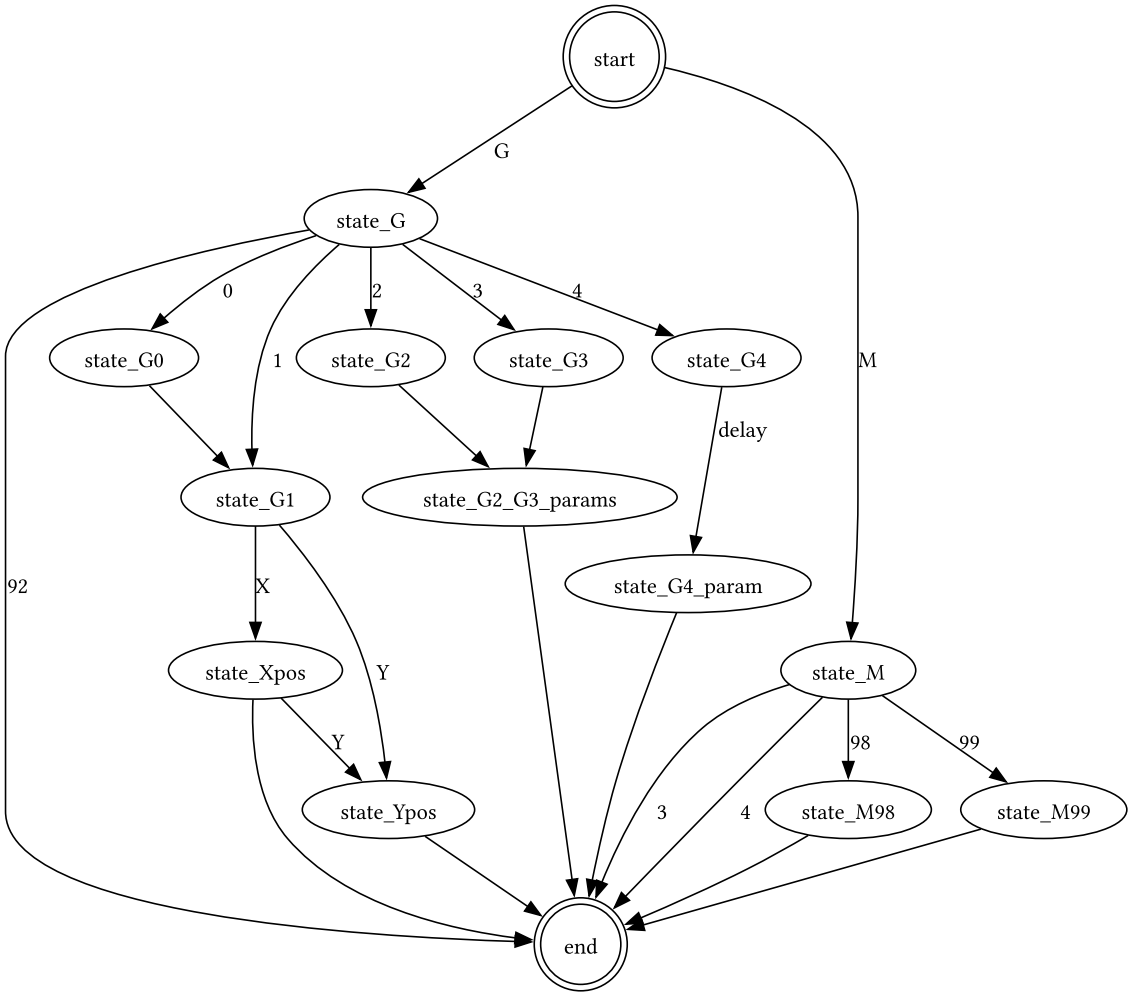
Frimware Plotteru obsahuje modul s vlastní implementací parseru G-Kódu. Parser se skládá ze 2 vrstev: třídy `GCodeParser` a třídy `GCodeLineParser`.



Obrázek 10: Třídy parseru

Třída `GCodeParser` načítá vstup ze sériové linky po jednotlivých řádcích. Řádky jsou nejprve načteny do stringu pomocí funkce `getline()`. Načtené řádky poté předává třídě `GCodeLineParser`.

Třída `GCodeLineParser` si nejprve převede string se vstupní řádkou na stringstream pro snažší parsování vstupu. Tato třída implementuje stavový automat. Průchodem jednotlivými stavy se postupně parsují typ, číslo a jednotlivé parametry G-Kódu. Pokud je G-Kód ze vstupu validní a parsování se podaří, tak se vytvoří instance konkrétního G-Kódu a na výstupu se předá smart pointer na danou instanci. V opačném případě, když se parsování nezdaří, předá se na výstupu nullptr.



Obrázek 11: Stavový automat G-Code parseru

7.6 Implementované G-Kódy

7.6.1 G0 & G1: Pohyb

- **G0**: rychlý pohyb
- **G1**: lineární pohyb

Příkaz G0 a G1 je interpretován stejně.

Parametry

- **Xnnn** - koncová pozice pro pohyb osy X
- **Ynnn** - koncová pozice pro pohyb osy X

Příklad

```
G0 X12      ; pohyb na pozici 12mm na ose X
G0 X24 Y10  ; pohyb na pozici 24mm na ose X a na pozici 10mm
na ose Y
```

7.6.2 G4: Čekání

Pozastavení Plotteru na určitý čas.

Parametry

- **Pnnn** - doba čekání v milisekundách
- **Snnn** - doba čekání v sekundách

Příklad

```
G4 P200  ; pozastav Plotter na 200ms
```

7.6.3 G92: Reset pozice

Resetování pozice Plotteru. Nastavení aktuální pozice jako počátek.

Příklad

```
G92  ; nastav aktuální pozici jako počátek
```

7.6.4 M3: Zdvih popisovače

Nastaví popisovač do horní pozice.

Příklad

```
M3  ; zdvih popisovače
```

7.6.5 M4: Přitlačení popisovače

Nastaví popisovač do spodní pozice.

Příklad

```
M4  ; přitlačení popisovače
```

7.6.6 M98: Nastavení rychlosti

Nastaví časový interval mezi jednotlivými kroky krokového motoru.

Parametry

- **nnn** - časový interval v mikrosekundách

Příklad

```
M98 200  ; nastav interval mezi kroky na 200 us
```

7.6.7 M99: Ruční zdvih popisovače

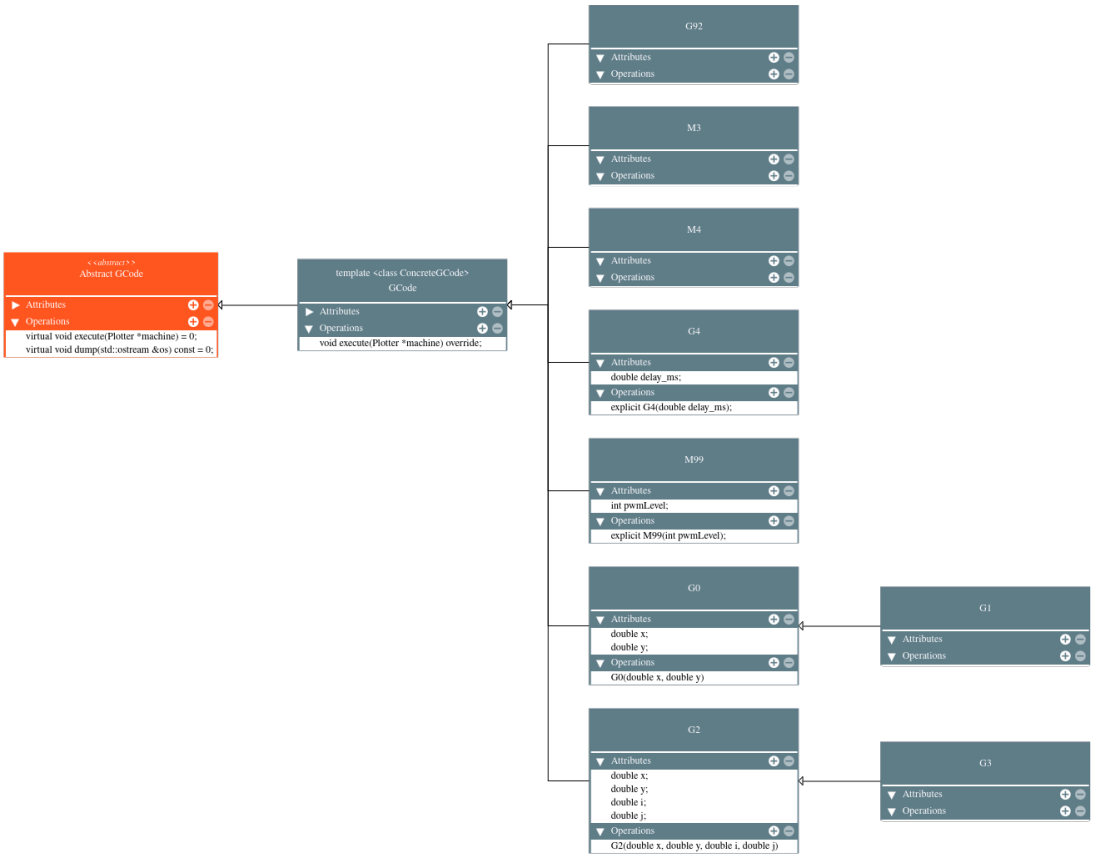
Zdvihni popisovačo do polohy odpovídající úrovni PWM z parametru.

Parametry

- **nnn** - úroveň PWM

Příklad

```
M99 3000  ; nastav PWM úroveň pro servo na 3000
```



Obrázek 12: Třídy G-Code

8 Použití

Plotter dokáže překreslit téměř jakýkoliv obrázek. Nejvhodnější ke kreslení jsou však vektorové obrázky, které se dají jednoduše převést na G-Kód. Existují však i techniky pro kreslení rastrových obrázků.

8.1 Vektorové obrázky

Na internetu existuje spousta zdrojů, kde se dají zdarma stáhnou vektorové obrázky pro následné překreslení. Nejlepších výsledků Plotter dosahuje s použitím takzvaných „Line Art“ obrázků, dostupných například na webu FREEPIK[9].

Pro tvorbu vlastní vektorové grafiky existuje spousta různých nástrojů. Volně dostupným zástupcem je například program Inkscape[10]

Hotový soubor s křivkami je poté třeba převést na G-Kód. Na to opět existuje spousta různých nástrojů. Dají se použít například programy pro generování G-Kódu pro laserové řezačky. Doporučeným programem je Pythonní modul **vpype**[11]. Tento modul navíc obsahuje spoustu užitečných funkcí pro práci s plottery, například funkci řazení jednotlivých čar tak, aby se při kreslení minimalizovaly doby přejezdů.

Obrázek 13: Překreslený Line Art obrázek

8.2 Rastrové obrázky

Kreslení rastrových obrázků je možné 2 způsoby. Tím prvním je převod obrázků do vektorového formátu typicky pomocí funkce **trace bitmap** v programech pro vektorovou grafiku. Poté se postupuje stejně jako v kapitole 8.1.

[9] https://www.freepik.com/free-photos-vectors/line-art-svg
[10] https://inkscape.org/
[11] https://github.com/abey79/vpype

Druhou možností je převést rastrový obrázek přímo na G-Kód tak, že plotter nebude kreslit obrysy, ale bude obrázek stínovat. Výsledný obrazec je potom tvořen pouze rovnoběžkami nebo pouze soustřednými kružnicemi, které mají v různých částech obrazce jinou hustotu a tím je dosaženo efektu stínování. Tuto funkcionalitu umožňuje například plugin pro **vpype** s názvem **hatched**[12]. Na obrázku 14 je vidět rastrový obrázek překreslený právě touto technikou.

Obrázek 14: Překreslený rastrový obrázek

8.3 Posílání instrukcí do Plotteru

Pro komunikaci PC a Plotteru byl vytvořen Python skript *plot.py*. Skript načte soubor s instrukcemi pro plotter a poté jednotlivé instrukce posílá po sériové lince Plotteru. Po zpracování pošle Plotter zpátky signál **OK**, který značí, že se může poslat další instrukce.

Skript se spouští z příkazové řádky a bere 2 parametry:

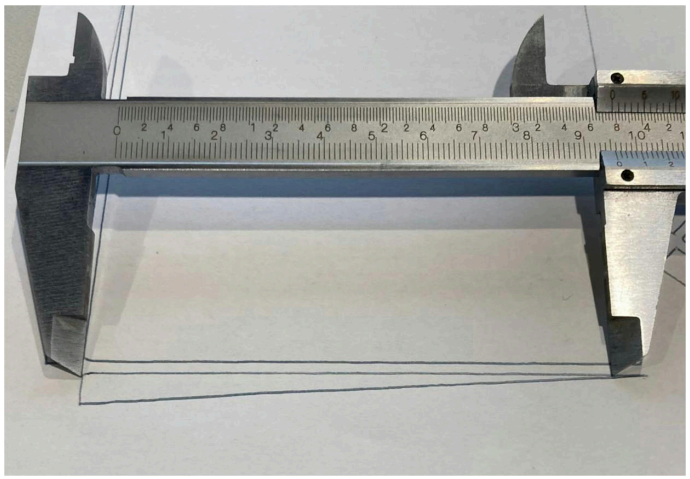
1. Sériový port, na kterém je plotter připojen.
2. Cesta k souboru s instrukcemi.

Užitím přepínače `-l` skript vypíše dostupné sériové porty.

[12]https://github.com/plottertools/hatched

9 Benchmark

Pro ověření funkčnosti a za účelem zjištění přesnosti a rozlišení Plotteru byl vytvořen jednoduchý benchmark. Benchmark se skládá z rozměrové zkoušky a ze zkoušky rozlišení. Rozměrová zkouška se skládá ze dvou čtverců o straně **100mm**, přičemž jeden je pootočený o 45°. Zde budeme porovnávat délky stran čtverců se zadanou hodnotou. Měření bude v pořadí horní, levá spodní, pravá. Zkouška rozlišení je prováděna na pravoúhlém trojúhelníku o známých rozměrech odvěsen **(100 a 5 mm)**. V benchmarku budou celkem 4 trojúhelníky, které se budou lišit svou orientací. Pozorovanou hodnotou bude nejmenší vzdálenost, kdy čáry do sebe ještě nesplývají. Tuto vzdálenost nebudeme měřit přímo, ale dopočítáme za využití vlastností podobných trojúhelníků.



Obrázek 15: Nepřímé měření rozlišení

Sestavená rovnice na základě podobnosti trojúhelníku vypadá následovně:

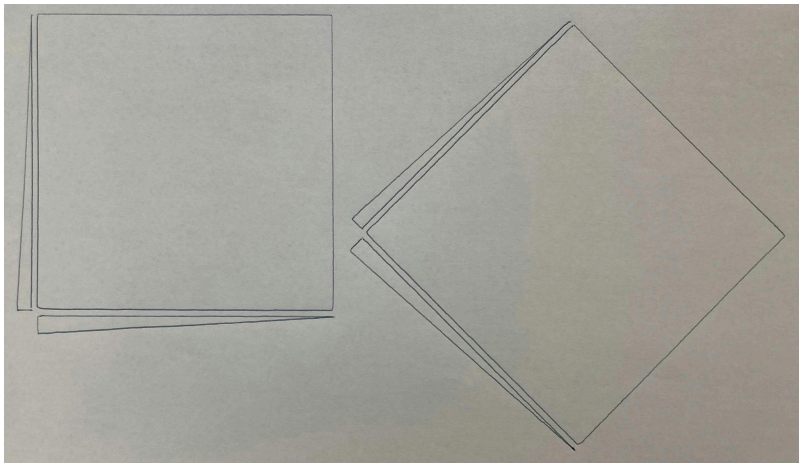
$$\frac{x}{5} = \frac{100 - s}{100}$$

x je rozlišení, s je naměřená vzdálenost způsobem jak je znázorněno na obrázku. Po úpravách dostaneme tuto rovnici:

$$x = 5 - \frac{s}{20}$$

9.1.1 Test 1

První test proběhl s kuličkovým perem o šíři stopy 0.5 mm.



Obrázek 16: Test 1

Tabulka 3: Výsledky rozměrové zkoušky testu 1

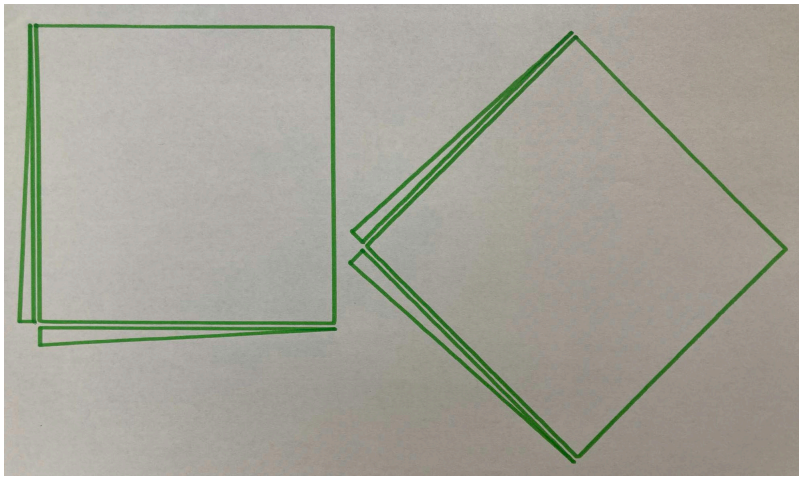
Strana	Délka [mm]	Rozdíl [mm]
1	101.8	1.8
2	101.3	1.3
3	101.9	1.9
4	101.2	1.2
5	102.4	2.4
6	101.3	1.3
7	102.5	2.5
8	101.2	1.2

Tabulka 4: Výsledky zkoušky rozlišení testu 1

s [mm]	x [mm]
91	0.45
92	0.4
87	0.65
93	0.35

9.1.2 Test 2

Druhý test proběhl s fixem o šíři stopy 1.0 mm.



Obrázek 17: Test 2

Tabulka 5: Výsledky rozměrové zkoušky testu 2

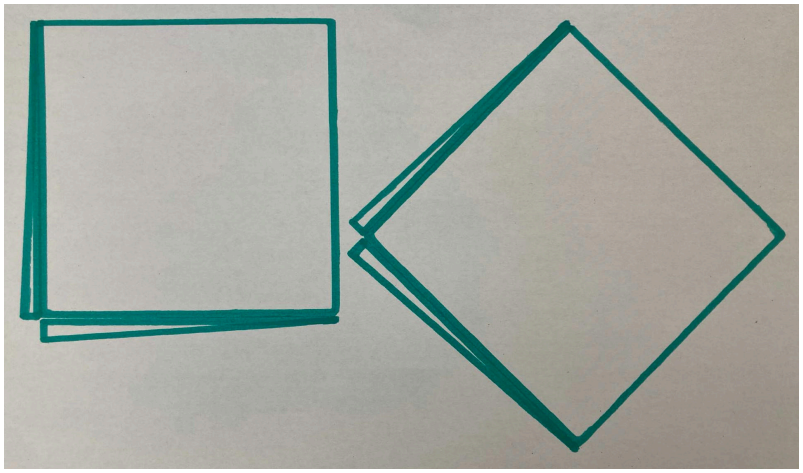
Strana	Délka [mm]	Rozdíl [mm]
1	101.7	1.7
2	101.2	1.2
3	101.7	1.7
4	101.4	1.4
5	102.2	2.2
6	100.9	0.9
7	102.3	2.3
8	101.0	1.0

Tabulka 6: Výsledky zkoušky rozlišení testu 2

s [mm]	x [mm]
68	1.6
80	1
69	1.55
79	1.05

9.1.3 Test 3

Třetí test proběhl s fixe o šíři stopy 2.5 mm.



Obrázek 18: Test 3

Tabulka 7: Výsledky rozměrové zkoušky testu 3

Strana	Délka [mm]	Rozdíl [mm]
1	101.2	1.2
2	100.9	0.9
3	101.2	1.2
4	100.8	0.8
5	102.0	2.0
6	100.4	0.4
7	102.0	2.0
8	100.7	0.7

Tabulka 8: Výsledky zkoušky rozlišení testu 3

s [mm]	x [mm]
35	3.25
60	2
52	2.4
60	2

9.1.4 Vyhodnocení

Z výsledků rozměrových testů je patrné, že Plotter není tak přesný jak se původně očekávalo. V průměru je odchylka od očekávané vzdálenosti 1.47 mm neboli 1.47 %.

V datech je možné pozorovat určitý opakující se vzor. Když si data rozdělíme na půl (4 a 4) a pozorujeme odchylky ob jednu řádku, tak jsou téměř shodné. Díky postupu měření, tyto odchylky vždy odpovídají protějším stranám čtverce. Tento vzor se opakuje ve všech testech. Znamená to, že Plotter je konzistentní, a odchylky bude možné jednoduše kompoenzovat ve firmware.

Rozdílné odchylky sousedních stran čtverců jsou pravděpodobně způsobeny rozdílným napnutím řemenů. Na celkovou odchylku může mít vliv více faktorů, především napnutí řemenů a vůle v konstrukci.

V testech rozlišení pozorujeme obdobný vzor, kdy se hodnoty ob jednu téměř shodují. Tyto rozdíly jsou opět pravděpodobně způsobeny rozdílným napětím řemenů. Za předpokladu, že jsou řemeny správně napnuté, tak rozlišení Plotteru odpovídá šíři stopy popisovače.

10 Celková cena projektu

Jedním z požadavků projektu bylo použití levných a dostupných komponent, tak aby cena byla co možná nejnižší. V tabulce 9 jsou zaznamenány všechny použité komponenty a jejich cena. V tabulce se nenachází drobné komponenty jako je kondenzátor, dioda nebo spojovací materiál.

Tabulka 9: Seznam komponent

Položka	**Počet**	**Cena [Kč]**
RPi Pico	1	119
PCB	1	20
Krokový motor Nema 17 0.4Nm	2	656
Servo MG90S	1	78
Driver BTT TMC2209	2	458
Zdroj	1	428
Step down měnič	1	32
Napájecí kabel C14	1	83
Konektor C14 s pojistkou	1	235
Hlazená tyč 8x300mm	2	96
Hlazená tyč 10x400mm	2	200
Hliníkový profil 30x30x400mm	1	140
Řemenice GT2	2	138
Řemen GT2 6mm 2m	2	800
Kuličkové ložisko F623ZZ	16	320
Ložisko LM8UU	2	68
Ložisko LM10UU	4	240
Filament		250
Celkem		**4361**

Závěr

Tato bakalářská práce představila kompletní návrh XY Plotteru, který je schopen kreslit tužkou, perem nebo fixou. Předložený návrh kombinoval originální konstrukci, kinematiku a řídící mikrokontrolér, čímž vzniklo unikátní zařízení. V práci byly popsány základní principy řízení XY Plotteru a jejich implementace.

Výsledky benchmarku ukázaly, že zařízení v současném stavu nesplňuje požadavek na přesnost, ale také se ukázalo, že tato nepřesnost je systematická a tudíž je možná její softwarová kompenzace. Všechny ostatní požadavky byly splněny.

Budoucí rozvoj a možná vylepšení

XY Plotter zkonstruovaný v rámci této bakalářské práce je stále pouze prototypem. Aby z něj mohl být plnohodnotný produkt, je třeba některé věci doladit. Jedná se například o vedení kabelu serva, který v současné verzi není ničím přidržováno a plandá. Stejně tak konce řemenů nejsou nijak úhledně shovány a vyčnívají.

Důležitou vlastností, kterou je potřeba doimplementovat je řízení krokových motorů s lineárním zrychlením. Tato vlastnost umožní navýšit rychlost kreslení při zachování přesnosti Plotteru.

Budoucí rozšíření Plotteru by mohlo obsahovat například ovládací prvky s displejem a rozhraní pro připojení flashdisku přímo na Plotteru. Díky tomu by bylo možné Plotter ovládat samostatně bez nutnosti připojení k počítači přes sériový port.

Reference

[1] „Visitor". Viděno: 20. květen 2024. [Online]. Dostupné z: https://refactoring.guru/design-patterns/visitor

[2] „Curiously Recurring Template Pattern". Viděno: 20. květen 2024. [Online]. Dostupné z: https://en.cppreference.com/w/cpp/language/crtp

[3] „Arduino Uno Rev3". Viděno: 22. květen 2024. [Online]. Dostupné z: https://store.arduino.cc/products/arduino-uno-rev3

[4] „Home · gnea/grbl Wiki". Viděno: 22. květen 2024. [Online]. Dostupné z: https://github.com/gnea/grbl/wiki

[5] „G-Code - RepRap". Viděno: 4. říjen 2024. [Online]. Dostupné z: https://reprap.org/wiki/G-code

[6] „AxiDraw Writing and Drawing Machines". Viděno: 20. květen 2024. [Online]. Dostupné z: https://axidraw.com/

[7] „3D printers | Original Prusa 3D printers directly from Josef Prusa". Viděno: 20. květen 2024. [Online]. Dostupné z: https://www.prusa3d.com/product/original-prusa-mk4-2/

[8] „VORON Design". Viděno: 20. květen 2024. [Online]. Dostupné z: https://vorondesign.com/

[9] Ilan E. Moyer, „CoreXY | Cartesian Motion Platform". Viděno: 20. květen 2024. [Online]. Dostupné z: http://corexy.com/

[10] Jiří Velebil, *Abstraktní a konkrétní lineární algebra*. 2023. [Online]. Dostupné z: https://math.fel.cvut.cz/en/people/velebil/akla.html

[11] TRINAMIC Motion Control GmbH & Co. KG, „TMC2209 Datasheet". 16. únor 2023. [Online]. Dostupné z: https://www.analog.com/media/en/technical-documentation/data-sheets/TMC2209_datasheet_rev1.09.pdf

[12] Raspberry Pi Ltd, *Raspberry Pi Pico Datasheet*. 2024. [Online]. Dostupné z: https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf

[13] J. E. Bresenham, „Algorithm for computer control of a digital plotter", *IBM Systems Journal*, roč. 4, s. 25–30, 1965.

[14] „Iterator". Viděno: 20. květen 2024. [Online]. Dostupné z: https://refactoring.guru/design-patterns/iterator

[15] Harry Fairhead, „The Pico In C: Basic PWM". Viděno: 20. květen 2024. [Online]. Dostupné z: https://www.i-programmer.info/programming/hardware/14849-the-pico-in-c-basic-pwm.html?start=1

Seznam obrázků

Seznam tabulek

Seznam ukázek kódů