Vyšší odborná škola a Střední průmyslová škola dopravní, Praha 1, Masná 18
Masná 18, 110 00 Praha 1

OBOR VZDĚLÁNÍ

26-41-M/01 Elektrotechnika

ZAMĚŘENÍ

Inteligentní dopravní systémy

MATURITNÍ PRÁCE

Model parkovacího výtahu

TŘÍDA: **E4** ŠKOLNÍ ROK: **2024/2025** **Jáchym Šťastný**

Shrnutí

Práce se zaměřuje na konstrukci a principy malého modelu páternosterového výtahu pro parkování vozidel. Cílem bylo navrhnout a vyrobit (pomocí 3D tisku a elektronických součástek) plně funkční demonstraci tohoto mechanismu. V teoretické části je vysvětleno fungování páternosterových systémů. Dále je popsáno navrhování a modelování klíčových dílů ve 3D softwaru.

V praktické části jsem provedl návrh kompletního modelu výtahu, a to od rámu a uchycení ložisek až po systém řetězu, kterým se platformy pohybují. Součástí práce byla instalace motoru, potřebných senzorů (například Hallových sond) a programování řídicí logiky. Následně jsem funkční model sestavil a otestoval, při čemž se ukázala nestabilita konstrukce, kvůli vytisknutému řetězu.

Obsah

Seznam obrázků

Úvod

Toto téma jsem si zvolil, protože s 3D tiskem, modelováním v CAD softwaru a prací s elektronikou dělám již několik let a baví mě to. Chtěl jsem si vybrat něco, kde můžu využít znalosti, které jsem už naučil a modelování autovýtahu mi přišel jako zajímavý nápad.

V této maturitní práci bych chtěl teoreticky představit princip tzv. páternosterového výtahu určeného pro parkování aut, tedy systém, kde se platformy neustále točí v nepřetržité smyčce nahoru a dolů. Současně se v praktické části zaměřím na vytvoření modelu tohoto výtahu, kde si čtenář může lépe představit, jak takový mechanismus funguje.

Hlavním cílem je ukázat, že i zdánlivě složitý systém se dá rozebrat na menší části a postavit v měřítku, které je zvládnutelné v rámci školního projektu. Pustil jsem se do samotného navrhování modelu, jeho tisku, sestavení a následného oživení. Vycházel jsem přitom z internetových zdrojů a také z vlastních praktických zkušeností.

Po přečtení této práce by si každý měl udělat jasnější představu o tom, jak vlastně princip páternosteru funguje a co všechno je potřeba k tomu, aby podobný systém pracoval plynule a spolehlivě.

1 Historie

1.1 První vertikální parkovací systémy

Na počátku 20. století, když množství aut výrazně stoupalo, tak to vytvořilo jednu velkou otázku: Kde všechny ty auta budou parkovat? Parkování na ulicích zabíralo strašně moc místa a aut bylo čím dál tím víc. Přišel tak nápad, že když nemůžeme rozšiřovat parkoviště do stran, tak to bude muset být do výšky. První vertikální parkovací systémy byly založeny na principu páternosteru. Ten byl vytvořen firmou Westinghouse Corporation v roce 1923 a jejich první výtah byl v roce 1932 postaven v Chicagu.

Obrázek 1 Historický autovýtah

2 Popis principu funkce

2.1 Co je a jak funguje páternoster

Páternoster je druh výtahu, který funguje na principu otáčejících se kabin ve dvou řadách, kde jedna z nich jede nahoru a druhá dolu. Řady jsou nahoře a dole propojeny koly, které otáčejí kabiny do druhého směru. Původně byly páternostery určeny pro lidi a výtah se proto točil nepřetržitě dokola takovou rychlostí, aby lidi stíhaly nastupovat a vystupovat. To byla výhoda, protože lidi nemuseli čekat na výtah, tak jak to bylo a je u tradičních systémů, jelikož byly kabiny v páternosteru postaveny hned za sebou a nepřetržitě se točily, tak bylo vždy volné místo. Páternostery byly také využívány jako parkoviště pro auta. Ta fungovala na stejném principu jako pro lidi jenom byly větší a s tím rozdílem, že výtah zastavoval a nejel nepřetržitě, aby mohla auta najet a vyjet.

3 Návrh vlastního výtahu

3.1 Představa, jak bude výtah vypadat

Chtěl jsem, aby výtah měl 4 platformy pro auta, uměl automaticky rozpoznat jaká platforma je jaká a automaticky uměl zastavit na přesném místě. Taky jsem chtěl, aby byla všechna elektronika schovaná pod základnou výtahu. Výtah bude možno ovládat pomocí čtyř tlačítek (každé tlačítko pro jednu platformu), infračerveného dálkového ovladače a taky přes mobil pomocí aplikace BLYNK.

3.2 Použité součástky a co budou dělat

Jako mikroprocesor pro řízení výtahu jsem chtěl použít Arduino UNO, které jsem stihl zničit dřív, než jsem vůbec začal, takže nakonec použiji ESP32, což se ukázalo jako lepší možnost, protože má víc dostupných pinů, programuje se stejně jako Arduino, může se připojit na Wi-Fi, je obecně rychlejší a menší.

Pro automatické zastavovaní chci použít RFID čtečku a Hallovy sondy. Na každé platformě bude ze spodní strany umístěna RFID karta a ze strany magnet. Na základně výtahu bude umístěna RFID čtečka, která rozpozná, o jakou platformu se jedná. Budou tam taky Hallovy sondy a když platforma přijede dolů, tak magnet aktivuje první sondu, která motor zpomalí a po aktivování druhé sondy se motor zastaví a tím se dosáhne toho, aby platformy zastavovaly vždy na stejném místě.

Tlačítka, infračervené dálkové ovládání a aplikace BLYNK budou na ovládání výtahu.

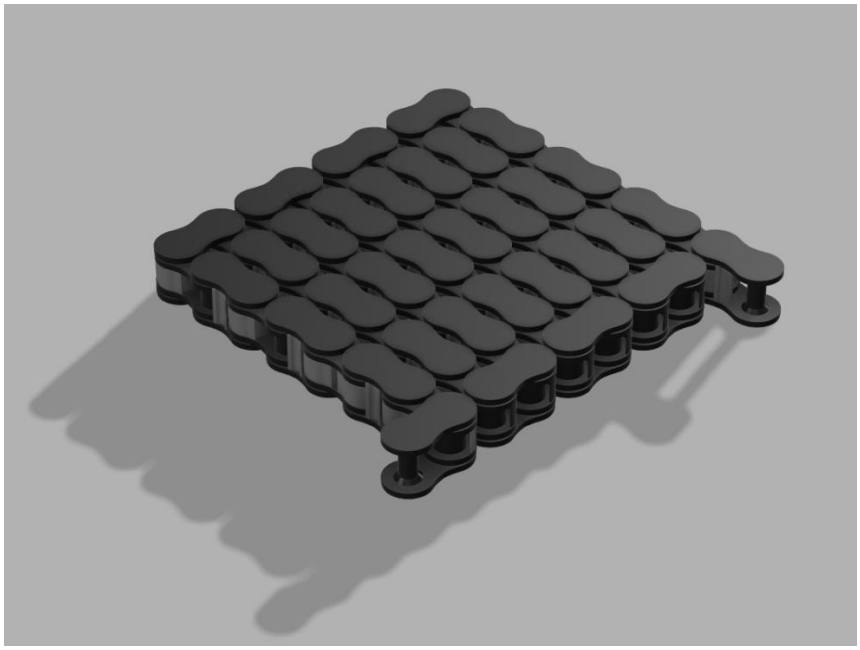
Vše bude zapojeno na nepájivém poli pro jednodušší zapojování a případnou diagnostiku problémů.

4 Modelování plastových částí

Program, který jsem si zvolil na modelovaní je Fusion 360, který používám již několik let a pracuje se mi v něm dobře.

4.1 Stažené součástky

4.1.1 Řetěz

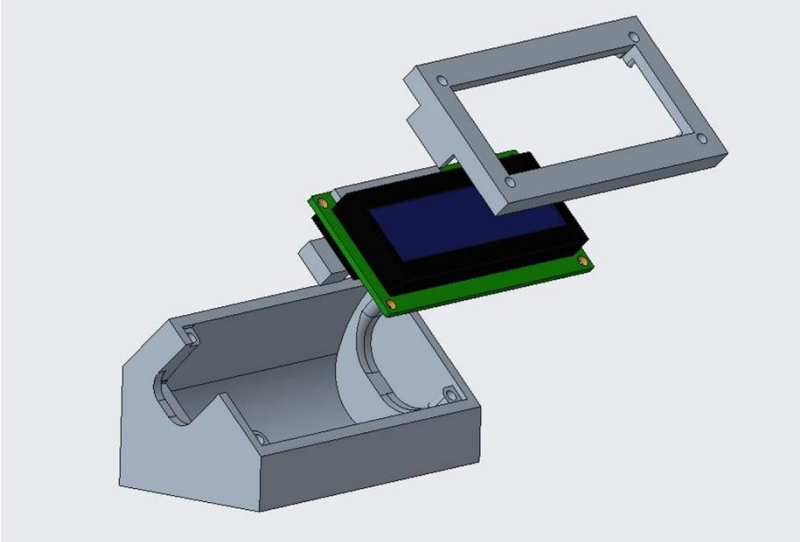


Obrázek 2 Řetěz

Řetěz jsem našel na internetu již velice dobře vymodelovaný, a proto mi přišlo zbytečné se trápit s modelováním. Jelikož byl řetěz větší, než jsem chtěl a potřeboval, tak jsem model zmenšil na 70 %. Řetěz se tiskne tak, jak je na obrázku a spojuje se pomocí vytisknuté spojky, kde stačí odlomit oba články na konci řetězu a spojit ho. Je tam ale hodně částí, které se tisknou ve vzduchu, a proto tam jsou podpory, které se musí odstranit článek po článku. Jelikož jich je tam hodně, tak to většinou zabralo přibližně hodinu. Po odstranění podpor se musí řetěz namazat, takže jsem do injekční stříkačky dal klasický řepkový olej a do každého článku řetězu jsem trošku kápl. Poté se musel řetěz trošku rozhýbat, takže jsem do vrtačky dal ozubené kolo, na které jsem řetěz dal a přibližně deset minut jsem s ním točil, aby se nikde nic nesekalo a vše běželo tak jak má.

Výtah má dohromady tři řetězy. Jeden spojuje samotný výtah s motorem a další dva jsou na spojení os.

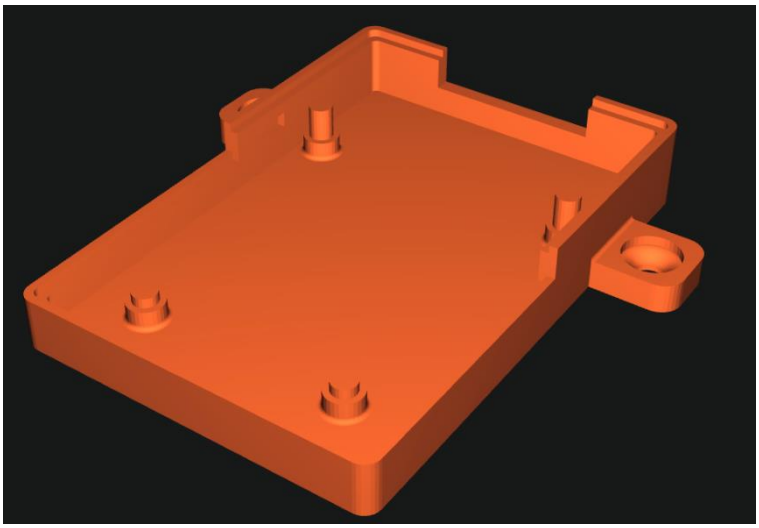
4.1.2 Držák na LCD displej



Obrázek 3 Držák na displej

Chtěl jsem, aby displej nebyl jen tak položený na základně, takže jsem našel na internetu tento model, který byl přesně pro můj displej a nemusel jsem na něm nic upravovat.

4.1.3 Držák na RFID čtečku



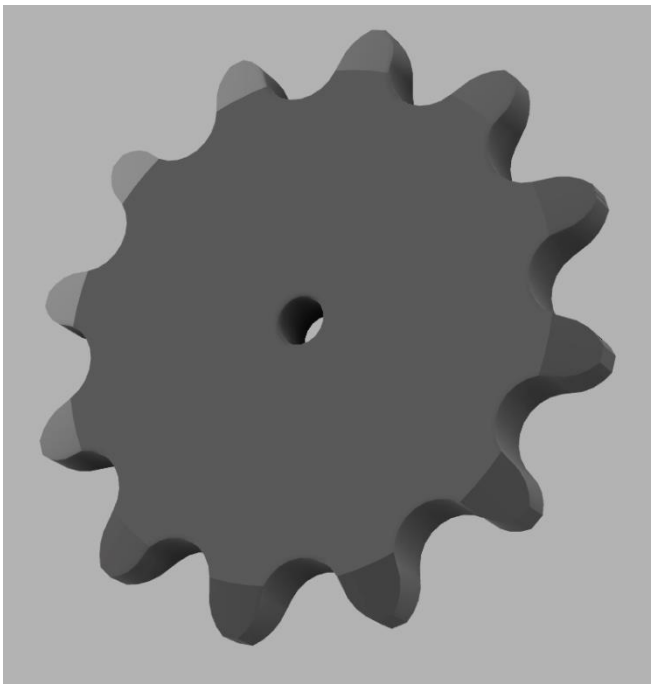
Obrázek 4 Držák na RFID čtečku

U tohoto modelu nebylo potřeba nic upravovat, jelikož ale mám na RFID čtečce připájený rezistor, tak je do výtisku pomocí páječky udělal díru, aby čtečka seděla rovně.

4.2 Stažené, ale upravené součástky

4.2.1 Ozubené kolečka na osy a na motor

Ozubené kolečko jsem stáhl od stejného autora, který vytvořil i řetěz. Byl to soubor STEP, který se na rozdíl od STL dá snadno upravovat, takže jsem ho bez potíží importoval do Fusion 360. Nejprve jsem kolečko zmenšil na 70 procent původní velikosti. Potom jsem otvor uprostřed z původních 4,5 mm zúžil na 3 mm aby do něj šla dřevěná osa jednoduše zatlačit.



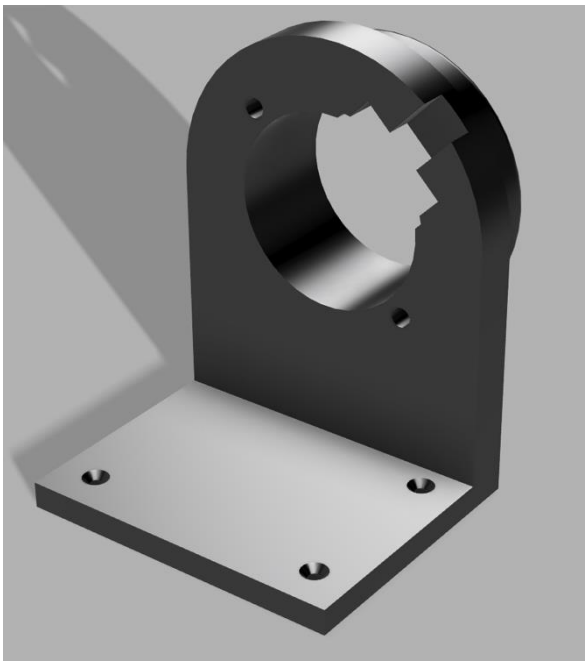
Obrázek 5 Ozubené kolečko

Ozubené kolo, které bude přímo na motoru, jsem upravoval podobným způsobem. Zmenšil jsem ho a vytvořil otvor ve tvaru hřídele motoru. Protože 3D tiskárna obvykle vytiskne díl o něco menší, než je navržený, nechal jsem kolem osy 0,3 mm vůli, abych měl jistotu, že se kolo bez problémů na hřídel vejde.

Když jsem kolečka vytisknul a zkoušel jsem je spolu s řetězem, tak se řetěz občas zasekával, takže jsem kolečka zmenšil na 67 procent původní velikosti, a to vyřešilo ten problém.

4.2.2 Držák na motor

Držák na motor jsem stáhnul a dělal jsem na něm jen malé úpravy. Jelikož by se ozubené kolečko společně s řetězem na držák nevešlo. Tak jsem jenom posunul díru na motor, tak aby měly součástky více místa. Potom už jsem jenom vymazal původní díry na šroubky a vytvořil jsem si vlastní.



Obrázek 6 Držák na motor

4.3 Vymodelované součástky

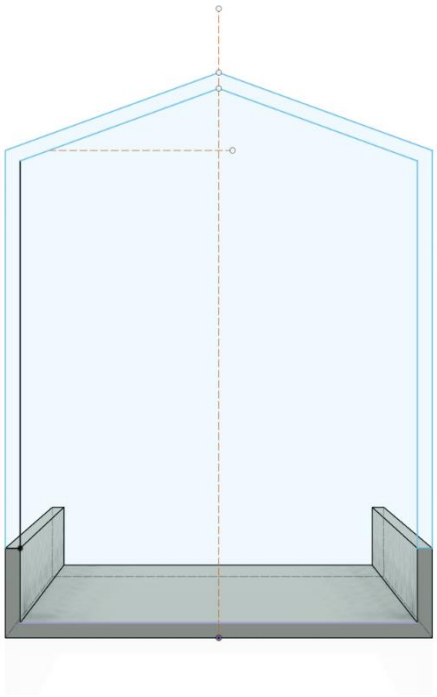
4.3.1 Platforma

Když jsem vymýšlel, jak bude vypadat výtah, tak jsem chtěl, aby byl co nejmenší, protože modely se budou rychleji tisknou a bude se lépe sestavoval. Proto jsem chtěl, aby byla platforma co nejmenší, takže jsem si vzal malý model auta, který jsem našel v pokoji a změřil jsem, jak je velký a podle toho jsem modeloval platformu.

Nejdřív jsem udělal nákres jednoduchého obdélníku ve správných rozměrech, který jsem následně extrudoval a přidal dvě stěny stejným způsobem. Stěny i základ platformy jsou jenom 1,5mm tenké, protože výtah používá vytisknutý řetěz a poměrně tenké dřevěné osy a čím méně bude platforma vážit, tím bude jednodušší pro výtah platformy zvednout.

Obrázek 7 Základna platformy

Potom jsem přidal rám, na kterém bude platforma zavěšena. Udělal jsem jednoduchý nákres, který jsem extrudoval na obou stranách platformy.



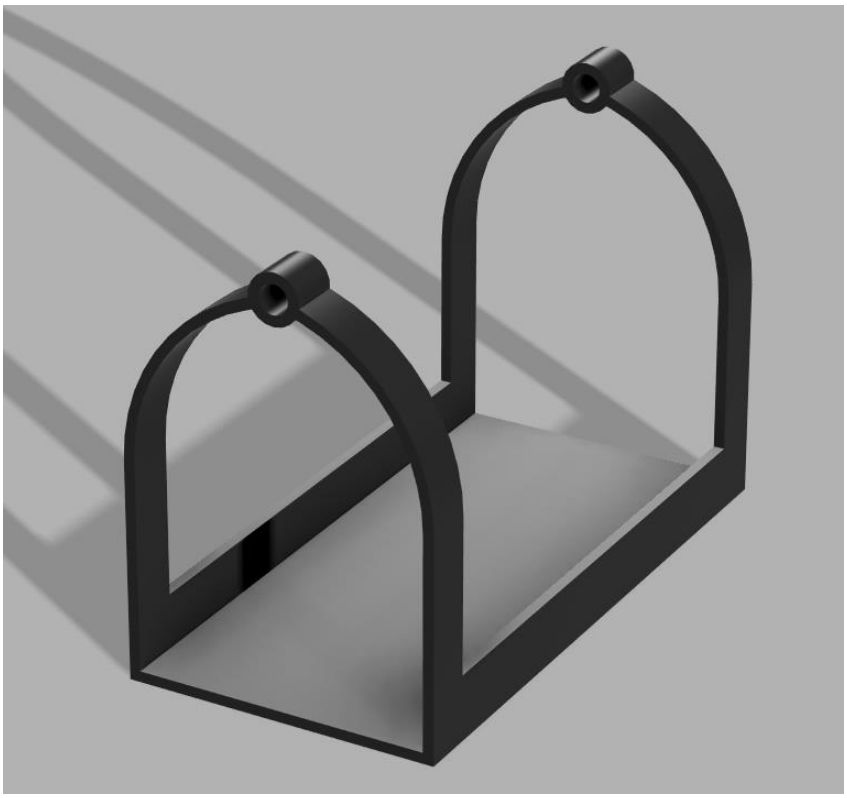
Obrázek 8 Nákres rámu platformy

Následně jsem pomocí funkce „Fillet“ zaoblil stěny rámu a přidal otvor na osu. Zaoblení je dobré, jak pro estetiku modelu, tak pro usnadnění práce 3D tiskárně. Když totiž 3D tiskárna tiskne část, kde jsou nějaké rohy, tak musí trošku přibrzdit a případně zastavit, a to může

zvýšit dobu tisku a někdy i kvalitu tisku. Když ale tiskne geometrii, ve které nejsou žádné rohy nebo prudké křivky, tak tiskárna nezastavuje ani nezpomaluje a kvalita a rychlost tisknu se výrazně zvýší.

Ke konci jsem přidal ještě zadní stěnu, aby model nevypadl a pomocí již zmiňované funkce „Fillet“ jsem zaoblil hranu na jednodušší najíždění.

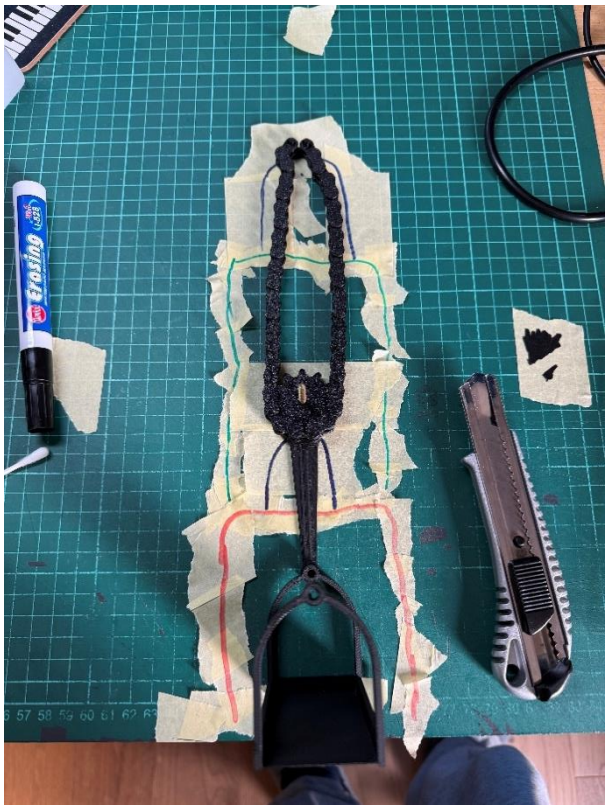
Platformu jsem tiskl vertikálně hlavně proto, aby byla co nejpevnější. U 3D tisku (metoda FDM) se plast pokládá ve vrstvách, které se lepí jedna na druhou. Když model stojí svisle, vrstvy jsou položené přesně ve směru síly, která působí shora dolů, takže je celá platforma odolnější proti prasknutí. Pokud bych se ji pokusil tisknout vodorovně, síly by tlačily přímo přes spoje vrstev a hrozilo by, že se platforma zlomí v místech, kde se vrstvy setkávají. Další důvod, proč jsem se rozhodl pro vertikální tisk, je ten, že při původním vodorovném tisku se rám nejprve tiskl po částech, a teprve až ke konci se sloupky spojily. To mělo za následek, že se sloupky při tisku mírně viklaly, a vrstvy se proto nespojovaly tak pevně, jak by měly, což vedlo k horší kvalitě výsledného dílu.



Obrázek 9 Platforma

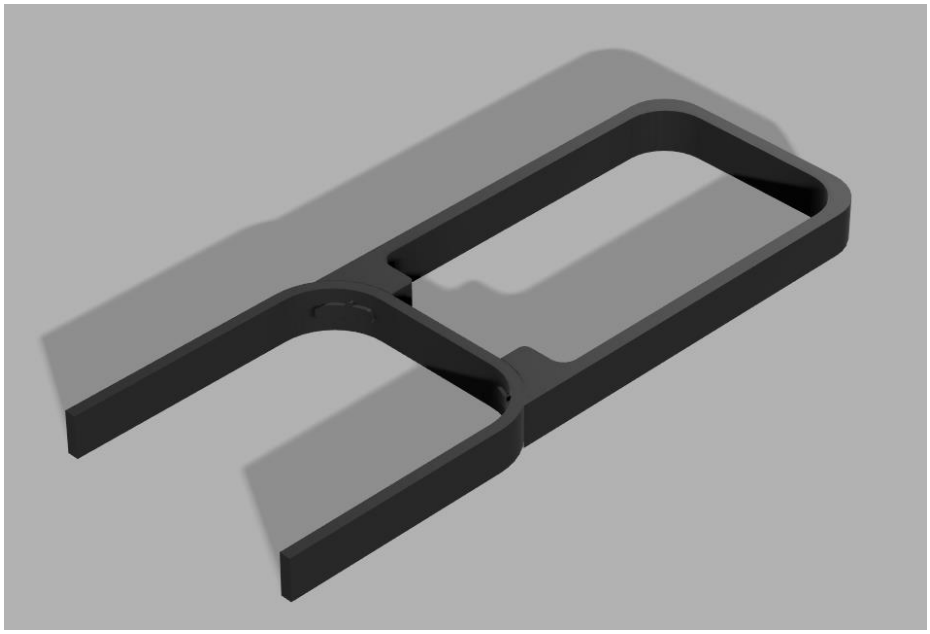
4.3.2 Hlavní rám výtahu

Když jsem měl vymodelovanou a vytisknutou platformu, tak jsem si na stůl nalepil papírovou pásku, abych zjistil, jak velký rám musí být. Nejdřív jsem si přibližně vyznačil, kde bude platforma umístěna, aby se nestalo to, že by platforma narazila do základny. Taky jsem použil řetěz abych zjistil, kde budou přibližně ložiska na ozubené kolečka.



Obrázek 10 Návrh rámu platformy

Model by byl nakonec moc velký na to, aby se dal vytisknout najednou, takže jsem ho rozdělil na dvě části, které jsou spojené pouze pomoci vytisknuté drážky. Navrhl jsem jednoduchý nákres podle rozměrů, které jsem zjistil z hrubého nákresu na stůl a ten jsem extrudoval a přidal drážky.



Obrázek 11 Rám výtahu

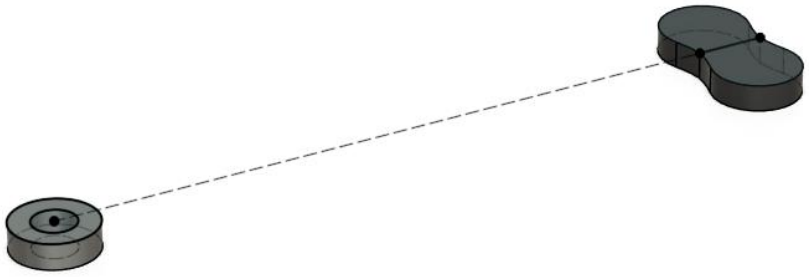
4.3.3 Osa pro uchycení platformy na řetěz

Na uchycení platform k řetězu jsem vymodeloval tuto součástku, která se přilepí na řetěz a na druhý konec se připevní platforma pomocí dřevěné tyčky.

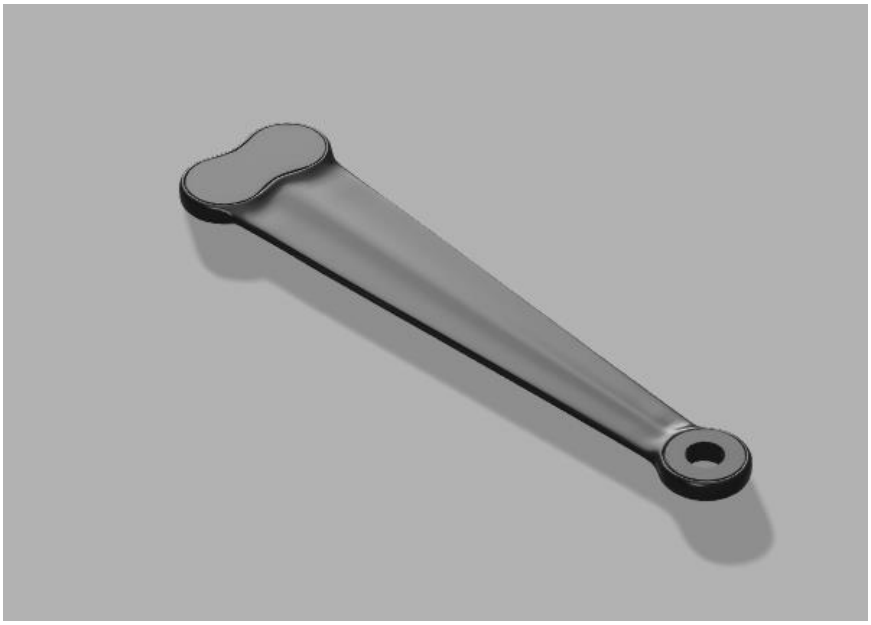
Nejdřív jsem si do programu importoval jeden článek z řetězu, abych mohl obkreslit jeho tvar, potom jsem si vypočítal, jaká vzdálenost bude potřeba mezi částí, která se připevňuje na platformu a mezi částí která se připevňuje na řetěz, tak aby platforma do ničeho nenarazila a měla dostatek místa při otáčení.

Obrázek 12 Nákres osy na řetěz

Následně jsem extrudoval plochy, které jsem spojil pomocí funkce „Automated Modeling“ která dokáže automaticky spojit dva modely tak, aby byla součástka co nejpevnější, ale aby bylo zároveň použito, co nejméně materiálu.



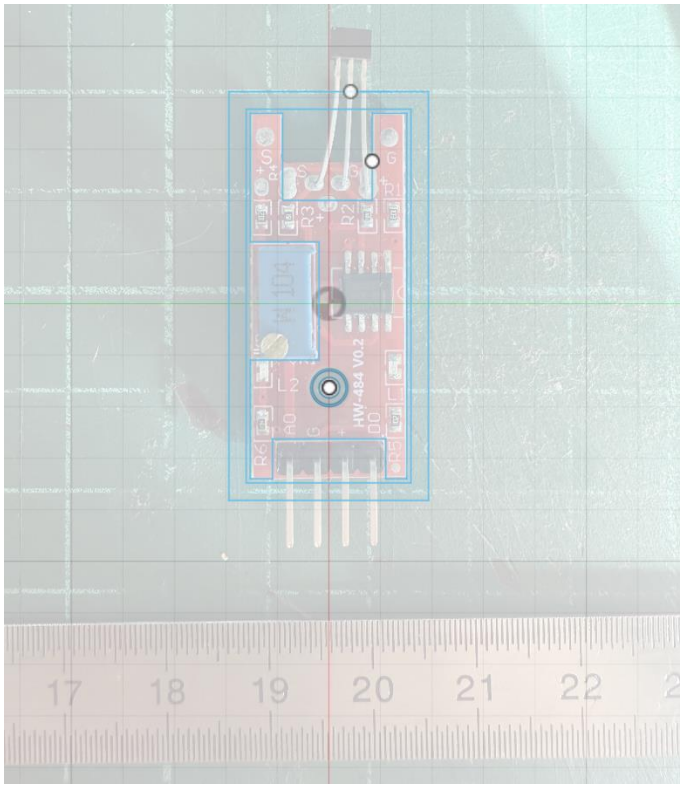
Obrázek 13 Modelování osy na řetěz



Obrázek 14 Osa na řetěz

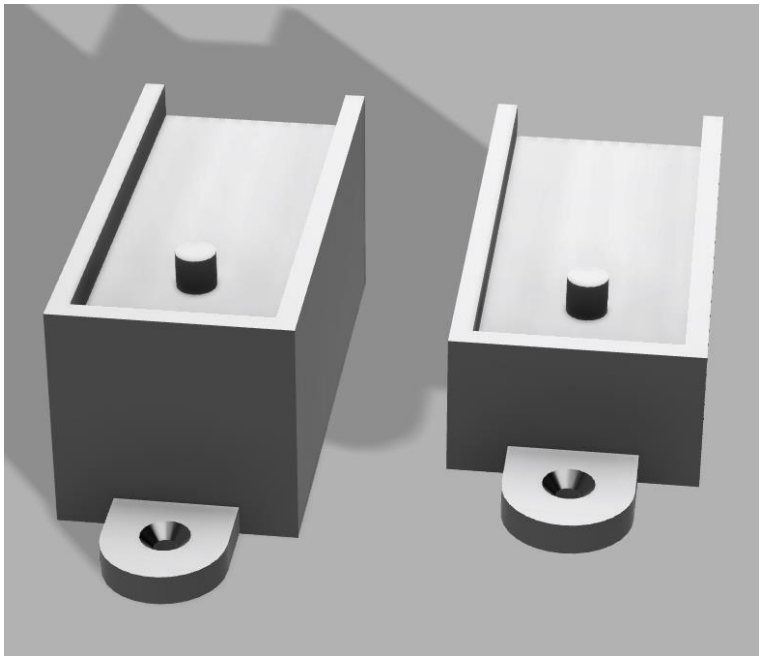
4.3.4 Držák pro Hallovy sondy

Nejdřív jsem si vyfotil samotnou Hallovu sondu společně s pravítkem. Fotku jsem následně importoval do Fusion 360 kde jsem fotku upravil tak, aby byla v měřítku 1:1 pomocí již zmíněného pravítka. Dále jsem obkreslil Hallovu sondu, přidal 0,3 mm toleranci a přidal otvor na šroubek, který slouží k upevnění Hallovy sondy k držáku.



Obrázek 15 Nákres držáku Hallovy sondy

Následně jsem si změřil, kde budou přibližně magnety. Jedna poloha byla, když je platforma skoro dole pro Hallovu sondu, která slouží pro zpomalení motoru a druhá poloha je, když je platforma úplně dole pro Hallovu sondu, která slouží k úplnému zastavení motoru. O tyto hodnoty jsem extrudoval dva totožné výkresy a potom už jsem jenom přidal díru na šroubek pro uchycení k základně.



Obrázek 16 Držák Hallovy sond

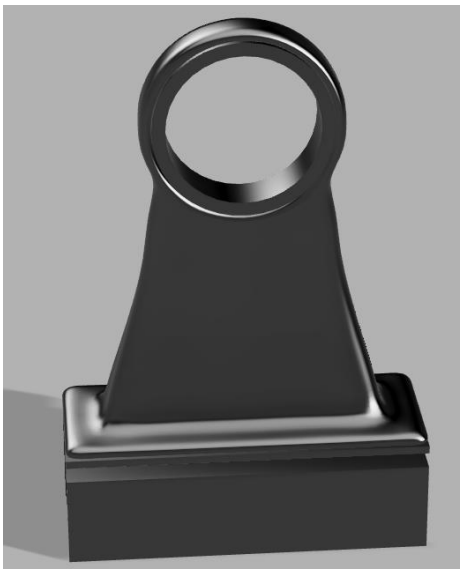
4.3.5 Držák na ložiska

Jelikož jsem si nebyl jistý, jak velký rozestup mezi ložisky přesně zvolit, aby byl řetěz správně napnutý, nechtěl jsem držáky ložisek napevno spojovat s rámem hned od začátku. Místo toho jsem je navrhl tak, aby se daly přilepit na rám až ve chvíli, kdy ověřím, že jsou všechny rozměry v pořádku.

Nejdřív jsem vymodeloval jednoduchý díl ve tvaru „U“, který se umístí na spodní část rámu a slouží jako základna pro připevnění ložiska. Následně jsem vytvořil kroužek s vnitřním průměrem stejným jako vnější průměr ložiska, aby se v něm dalo ložisko pevně usadit.

Obrázek 17 Modelování držáku na ložiska

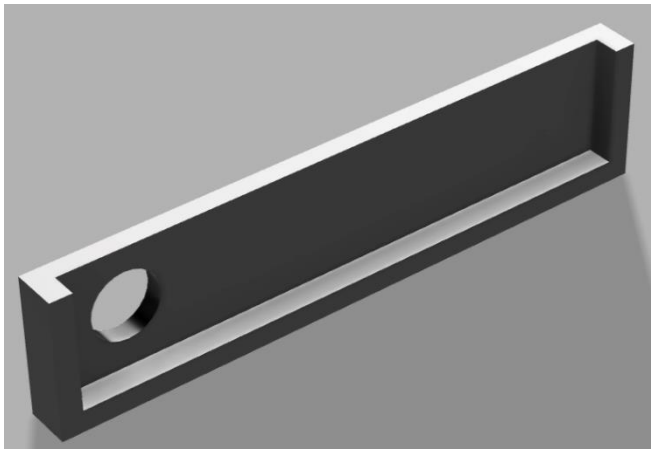
Modely jsem následně spojil pomocí již zmiňované funkce „Automated Modeling“ v programu Fusion 360, což mi ušetřilo spoustu času. Původně měl tento model navržené malé háčky, které měly sloužit k tomu, aby nebylo nutné nic lepit a díly by se daly jednoduše nacvaknout. Při zkoušení jsem ale zjistil, že háčky nejsou dostatečně pevné a mohou se snadno ulomit nebo uvolnit. Proto jsem se rozhodl tuto část odstranit a místo toho držák jednoduše přilepit, což se ukázalo jako mnohem spolehlivější řešení.



Obrázek 18 Držák na ložiska

4.3.6 Nástroj pro přesné nalepení magnetů

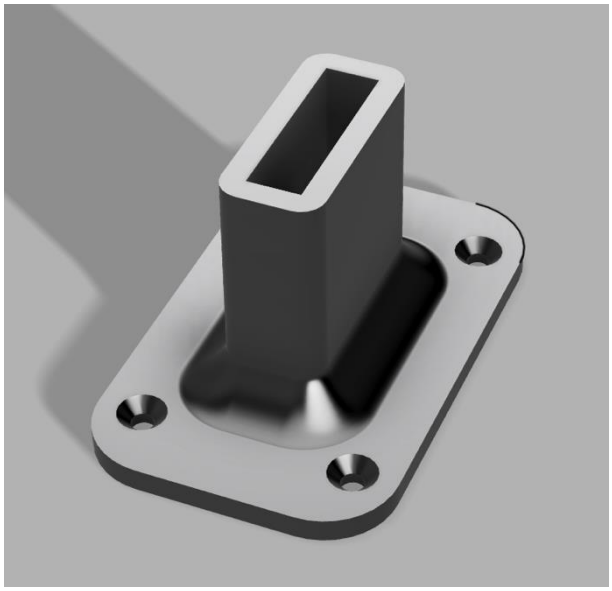
Chtěl jsem, aby magnety byly na platformách nalepeny vždy na stejném místě. Proto jsem vymodeloval tuto součástku. Udělal jsem výkres na zadní straně platformy, do kterého jsem udělal díru, kde jsem chtěl, aby magnet byl a extrudoval jsem stěny o 1,7 mm.



Obrázek 19 Nástroj pro přesné nalepení magnetů

4.3.7 Nožičky pro připevnění výtahu k desce

Na vymodelování nožiček jsem použil rozměry spodní strany rámu. Udělal jsem podle toho výkres a ten jsem vyextrudoval. Na spodní část modelu jsem přidal plochu, na kterou jsem udělal otvory pro šroubky a pomocí již zmiňované funkce „fillet“ jsem zaoblil hrany.

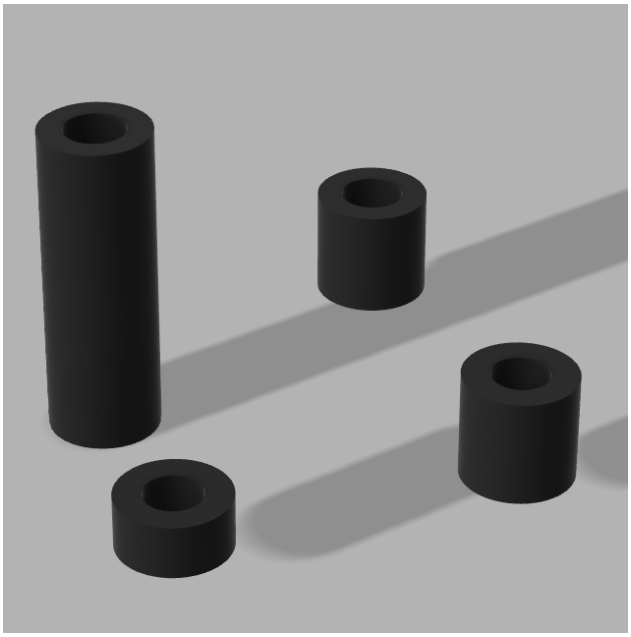


Obrázek 20 Nožička pro připevnění rámu

4.3.8 Ostatní malé součástky

Pro připevnění součástek na osy stačilo jen vymodelovat jednoduché podložky, které patří mezi platformu a držákem, který je umístěný na řetězu a zarážky, které slouží k tomu, aby jednotlivé modely nevypadly z osy.

Pro podložky jsem udělal vnitřní průměr 3,5 mm, tedy o trošku větší, než průměr dřevěné osy a pro zarážky jsem udělal vnitřní průměr 3 mm, kde stačí aby se na osu silou natlačily.



Obrázek 21 Podložky a zarážky

5 Popis zdrojového kódu

5.1 Základní definice a připojení knihoven

```
#define BLYNK_TEMPLATE_ID "TMPL4yGaaa-X6"
#define BLYNK_TEMPLATE_NAME "lift"
#define BLYNK_AUTH_TOKEN "[redacted]"
char auth[] = BLYNK_AUTH_TOKEN;

#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <IRremote.hpp>
#include <AccelStepper.h>
#include <Wire.h>
#include <hd44780.h>
#include <hd44780ioClass/hd44780_I2Cexp.h>
#include <SPI.h>
#include <MFRC522.h>
```

Obrázek 22 Zdrojový kód – základní definice a připojení knihoven

- BLYNK_TEMPLATE_ID, BLYNK_TEMPLATE_NAME, BLYNK_AUTH_TOKEN: Jedná se o údaje pro cloudovou službu Blynk. Díky nim se zařízení připojí k Blynku a může odesílat nebo přijímat data (např. jakou platformu zvolil uživatel).
- Knihovny:
  - WiFi.h: Umožňuje připojení k Wi-Fi síti na ESP32.
  - BlynkSimpleEsp32.h: Poskytuje funkce pro komunikaci s platformou Blynk.
  - IRremote.hpp: Zajišťuje příjem signálů z IR dálkového ovládání.
  - AccelStepper.h: Knihovna pro snadné řízení krokových motorů.
  - Wire.h: Pro I2C komunikaci (využívá se u LCD).
  - hd44780.h a hd44780_I2Cexp.h: Ovládání LCD displeje přes I2C. – Opravdu potřebuje ten display všechny tři knihovny?
  - SPI.h a MFRC522.h: Pro komunikaci s RFID čtečkou (MFRC522).

5.2 Přihlašovací údaje pro Wi-Fi a nastavení RFID

```
16  // WiFi údaje
17  const char* ssid = "[redacted]";
18  const char* password = "[redacted]";
19
20  // RFID čtečka
21  #define RST_PIN 33
22  #define SS_PIN 13
23  MFRC522 rfid(SS_PIN, RST_PIN);
```

Obrázek 23 Zdrojový kód – přihlašovací údaje pro Wi-Fi a nastavení RFID

- ssid, password: Název a heslo Wi-Fi sítě, ke které se ESP32 připojuje.
- RST_PIN, SS_PIN: Definuje, kam jsou zapojeny piny RFID čtečky na ESP32

5.3 Definice pinů, tlačítek a senzorů

```
25  // Piny
26  #define IR_PIN 15
27  IRrecv irrecv(IR_PIN);
28  IRData results;
29
30  const int buttonPins[] = {4, 16, 17, 5}; // tlačítka pro platformy 1–4
31
32  // Hallovy sondy (aktivní = HIGH)
33  #define HALL_SLOW_PIN 34 // První sonda – zpomalení
34  #define HALL_STOP_PIN 35 // Druhá sonda – zastavení
```

Obrázek 24 Zdrojový kód – definice pinů, tlačítek a senzorů

- IR_PIN: Pin, na kterém ESP32 přijímá infračervené signály dálkového ovládání.
- irrecv: IRrecv zpracovává přijaté kódy z dálkového ovládání.
- buttonPins: Pole obsahující piny, na kterých jsou připojena čtyři tlačítka. Každé odpovídá jedné platformě.
- HALL_SLOW_PIN a HALL_STOP_PIN: Piny, kde se čtou signály z Hallových sond. Tyto sondy reagují na magnet: jedna slouží ke zpomalení (SLOW), druhá k zastavení (STOP).

5.4 Nastavení motoru, bzučáku a displeje

```
36  // Motor
37  #define IN1 26
38  #define IN2 27
39  #define IN3 14
40  #define IN4 12
41  AccelStepper stepper(AccelStepper::HALF4WIRE, IN1, IN3, IN2, IN4);
42
43  // Buzzer
44  #define BUZZER_PIN 32
45
46  // LCD
47  hd44780_I2Cexp lcd; // I2C LCD
```

Obrázek 25 Zdrojový kód – nastavení motoru, bzučáku a displeje

- IN1, IN2, IN3, IN4: Piny, které ovládají cívky krokového motoru.
- AccelStepper stepper: Objekt pro řízení krokového motoru v režimu HALF4WIRE (poloviční krokování).
- BUZZER_PIN: Pin s bzučákem.
- lcd: Ovládání 16×2 LCD displeje po I2C sběrnici.

5.5 Definice platforem a proměnných

```
49  // Platformy a jejich UID
50  String platformUIDs[4] = {
51    "30C30722", // P1
52    "4384EF2F", // P2
53    "531F462F", // P3
54    "41B5DE1B"  // P4
55  };
56
57  // Stavová proměnná
58  int currentPlatform = 1;     // Aktuálně detekovaná platforma (dle RFID)
59  int targetPlatform = 1;      // Uživatelem zvolená cílová platforma
60  int motorState = 0;          // 0=zastaven,1=rychlá jízda,2=pomalá jízda
61  bool correctPlatformFound = false; // indikace nalezení správné platformy
62
63  // IR kódy tlačítek
64  unsigned long knownIRCodes[4] = {0xC, 0x18, 0x5E, 0x8};
```

Obrázek 26 Zdrojový kód – definice platforem a proměnných

- platformUIDs: Uchovává RFID kódy pro čtyři různé platformy. Když čtečka přečte nějaký kód, porovná ho s těmito hodnotami, aby poznala, o kterou platformu jde.

- currentPlatform: Číslo platformy, která je právě přečtená (RFID).
- targetPlatform: Číslo platformy, kam chce uživatel nechat výtah dojet.
- motorState: Stav motoru (0=zastaven, 1=rychlý chod, 2=pomalý chod).
- correctPlatformFound: Proměnná, zda už čtečka našla správnou platformu.
- knownIRCodes: Obsahuje infračervené kódy čtyř tlačítek na dálkovém ovladači, z nichž každé odpovídá jedné platformě.

5.6 Funkce setup()

- Serial.begin(9600): Spouští sériovou komunikaci pro ladicí výpisy.
- pinMode u tlačítek, sond, buzzer: Nastavení pinů jako vstupy nebo výstupy.
- stepper.setMaxSpeed(2000), stepper.setAcceleration(500): Nastavení vlastností motoru (maximální rychlost a zrychlení).
- SPI.begin(…) a rfid.PCD_Init(): Spouští komunikaci po SPI sběrnici (SCK, MISO, MOSI, SS) a inicializuje RFID modul pro čtení karet.
- Wire.begin(21, 22) a lcd.begin(16, 2): Spuštění I2C linky a inicializace 16×2 LCD.
- WiFi.begin(…) a Blynk.begin(…): Připojení ESP32 k Wi-Fi a Blynk serveru.
- updateLCDandBlynk(): Načte a zobrazí počáteční hodnoty na LCD a v Blynk aplikaci.

5.7 Funkce loop()

- Blynk.run(): Stará se o zpracování událostí z Blynku.
- motorState == 0: Motor stojí, čeká na vstupy (tlačítka, IR).
    - Pokud některé tlačítko detekuje HIGH, nastaví se targetPlatform a motor se spustí (funkce startMotorFast()).
    - Pokud přijmeme IR kód, porovná se s knownIRCodes a podle toho se nastaví targetPlatform.
- motorState == 1 nebo 2: Motor jede (1=rychle, 2=pomalu).
    - RFID čtečka průběžně kontroluje, jestli nepřišel nový RFID tag. Pokud ano, získá z něj UID, porovná ho s platformUIDs a zjistí, jestli je to hledaná platforma (targetPlatform).
    - Pokud je to ta správná platforma, correctPlatformFound = true, a zapojují se do hry Hallovy sondy (HALL_SLOW_PIN, HALL_STOP_PIN). Ty mohou motor zpomalit (slowDownMotor()) a pak zastavit (stopMotor()).

- stepper.runSpeed() se musí volat neustále, aby knihovna AccelStepper mohla posouvat motor krok za krokem.

5.8 Práce s RFID

```
202  // Funkce pro získání UID jako hex string
203  String getUIDString() {
204    String uidStr = "";
205    for (byte i = 0; i < rfid.uid.size; i++) {
206      byte val = rfid.uid.uidByte[i];
207      if (val < 0x10) uidStr += "0";
208      uidStr += String(val, HEX);
209    }
210    uidStr.toUpperCase();
211    return uidStr;
212  }
213
214  // Najde platformu na základě UID
215  int getPlatformFromUID(String uid) {
216    if (uid == "30C30722") return 1;
217    if (uid == "4384EF2F") return 2;
218    if (uid == "531F462F") return 3;
219    if (uid == "41B5DE1B") return 4;
220    return -1;
221  }
```

Obrázek 27 Zdrojový kód – práce s RFID

- getUIDString(): Převádí bajty UID z RFID na čitelný řetězec v hexadecimálním tvaru, např. “30C30722”.
- getPlatformFromUID(): Určí, ke které platformě (1 až 4) daný UID patří. Pokud UID neodpovídá žádné z uložených, vrací -1.

5.9 Funkce pro řízení motoru

```
223  // Funkce pro rychlý start motoru
224  void startMotorFast() {
225    motorState = 1;
226    correctPlatformFound = false;
227    // Nastavíme rychlou rychlost, vyšší než předtím
228    // Vyzkoušejte 1500, 2000, 2500...
229    stepper.setSpeed(1000);
230  }
231
232  // Funkce pro zpomalení motoru
233  void slowDownMotor() {
234    motorState = 2;
235    stepper.setSpeed(300); // pomalejší otáčky vpřed (můžete zkusit 300–500)
236  }
237
238  // Funkce pro zastavení motoru
239  void stopMotor() {
240    motorState = 0;
241    stepper.setSpeed(0);
242
243    // Pípnutí krátce po zastavení
244    tone(BUZZER_PIN, 1500, 300); // 1500 Hz, 300 ms
245
246    // Nyní jsme na správné platformě
247    currentPlatform = targetPlatform;
248    updateLCDandBlynk();
249  }
```

Obrázek 28 Zdrojový kód – funkce pro řízení motoru

- startMotorFast(): Přepne stav motoru na 1 (rychlý chod), vynuluje correctPlatformFound a nastaví rychlost na 1000 kroků za sekundu.
- slowDownMotor(): Nastaví stav motoru na 2 (pomalý chod), rychlost 300 kroků/s.
- stopMotor(): Zastaví motor (motorState=0, rychlost=0), krátce pípne buzzerem a srovná currentPlatform = targetPlatform.

5.10 Funkce pro aktualizaci LCD a Blynk

```
251  // Aktualizace LCD a Blynk
252  void updateLCDandBlynk() {
253    lcd.clear();
254    lcd.setCursor(0,0);
255    lcd.print("Vybrana: ");
256    lcd.print(targetPlatform);
257    lcd.setCursor(0,1);
258    lcd.print("Aktualni: ");
259    lcd.print(currentPlatform);
260
261    // Blynk labely
262    Blynk.virtualWrite(V5, String("Vybraná platforma: ") + String(targetPlatform));
263    Blynk.virtualWrite(V6, String("Aktuální platforma: ") + String(currentPlatform));
264  }
```

Obrázek 29 Zdrojový kód – funkce pro aktualizaci LCD a Blynk

- Na LCD se zobrazí číslo vybrané a aktuální platformy. Stejná informace se pošle do aplikace Blynk.

5.11 Obsluha Blynk virtuálních pinů

```
266  // Blynk obsluha
267  BLYNK_WRITE(V1) {
268    targetPlatform = 1;
269    Serial.println("Blynk V1 - targetPlatform=1");
270    updateLCDandBlynk();
271    startMotorFast();
272  }
273  BLYNK_WRITE(V2) {
274    targetPlatform = 2;
275    Serial.println("Blynk V2 - targetPlatform=2");
276    updateLCDandBlynk();
277    startMotorFast();
278  }
279  BLYNK_WRITE(V3) {
280    targetPlatform = 3;
281    Serial.println("Blynk V3 - targetPlatform=3");
282    updateLCDandBlynk();
283    startMotorFast();
284  }
285  BLYNK_WRITE(V4) {
286    targetPlatform = 4;
287    Serial.println("Blynk V4 - targetPlatform=4");
288    updateLCDandBlynk();
289    startMotorFast();
290  }
```

Obrázek 30 Zdrojový kód – obsluha Blynk virtuálních pinů

- Na LCD se zobrazí číslo vybrané a aktuální platformy. Stejná informace se pošle do aplikace Blynk.

- Každý virtuální pin (V1 až V4) odpovídá jedné platformě. Když uživatel v Blynk aplikaci klikne na konkrétní tlačítko, nastaví se targetPlatform a motor se okamžitě rozjede (rychlý chod).

6 Vlastní realizace

6.1 Programování elektroniky

Na začátku jsem otestoval všechny součástky, jestli správně fungují. Začal jsem s motorem a displejem. Když se zmáčklo tlačítko 4, tak se motor rozjel a na displeji se ukázalo, že je výtah na platformě 4. Potom jsem přidal dálkové ovládání a integraci aplikace BLYNK. Když tohle všechno fungovalo a chtěl jsem otestovat RFID čtečku, tak se ukázalo, že jsem koupil špatné RFID karty, které pracují na frekvenci 125kHz a nefungovaly s mojí čtečkou, která podporuje karty MIFARE, které pracují na frekvenci 13,56 MHz. Po vyřešení problému s kartami jsem pomocí čtečky naskenoval UID karet, které jsem následně přiřadil k jednotlivým platformám 1-4. Dále jsem zapojil dvě Hallovy sondy a naprogramoval jsem je tak, aby když první senzor zaznamená magnet, tak aby motor zpomalil a když druhý senzor zaznamená magnet, tak aby motor úplně zastavil. Potom jsem začal spojovat vše dohromady.

První problém nastal, když jsem chtěl spojit RFID čtečku a Hallovy sondy. Chtěl jsem, aby když uživatel vybere například platformu 3, tak aby magnety nezastavily platformu 1, protože zaznamenali magnet. Takže jsem se to snažil naprogramovat tak, aby když RFID čtečka zaznamená například platformu 1, tak aby Hallovy sondy nic nedělaly, a když zaznamenají platformu 3, kterou uživatel vybral, tak aby se sondy aktivovaly a platformu zastavily. To ze začátku nefungovalo a karty čtečka vůbec nečetla, ale nakonec se ukázalo, že jsem používal jen špatnou Arduino knihovnu.

Další problém byl, když jsem spojil všechny součástky dohromady a v tu chvíli nefungovalo vůbec nic. Po zmačknutí tlačítek se motor měl začít točit, to se ale nedělo. Po hodinách u počítače jsem přišel na to, že si program myslel, že tlačítka jsou stisknutá, i když nebyly a řešení bylo změnit logické úrovně tak, že když tlačítko je zmáčknuto, tak je stav HIGH a když puštěno, tak je stav LOW. Stejný problém byl i u Hallových sond, kde bylo řešení problému stejné jako u tlačítek.

Poslední a do teď nevyřešený problém je, že se motor točí strašně pomalu. Čtrnáct dní jsem seděl u počítače a snažil jsem se ho vyřešit. Nejdřív jsem si myslel, že je to softwarová chyba a že nějaká smyčka blokuje impulzy ke krokovému kódu, protože samostatně zapojený motor fungoval bez problémů. Začal jsem tedy promazávat kód, aby tam bylo co nejméně rušivých prvků, které by mohly motor brzdit. Po smazání skoro všech funkcí jako

bylo dálkové ovládání, integrace aplikace BLYNK a vymazání všech řádků, které vypisovaly funkce do sériového monitor se motor stále netočil. Snažil jsem se taky měnit v kódu parametru motoru jako rychlost, akceleraci a typ zapojení (FULL4WIRE a HALF4WIRE), to ale ničemu nepomohlo, naopak, když jeden z desítek programů nahrál, tak se z displeje začalo kouřit. Nějakým zázrakem se ale nic nestalo a displej fungoval dál. Když už jsem to chtěl vzdát, tak jsem se dotknul krystalu na RFID čtečce a motor se rozjel plnou rychlostí, jak jsem chtěl. Chování bylo ale velice divné a nestabilní. Někdy stačilo abych se dotknul jen jednoho pinu krystalu, někdy obou a někdy nefungovalo vůbec nic. Myslel jsem si tedy, že je chyba v krystalu, tak jsem ho přepájel, kdyby tam byl špatný spoj, to ale nepomohlo. Tak jsem koupil další RFID čtečku, ta ale dělala to samé. To už jsem byl zoufalý a přinesl jsem projekt do školy, jestli někdo nebude vědět, co s tím je. Tímto chci poděkovat panu učiteli, inženýru Lubomíru Harwotovi, který se mnou strávil několik hodin, kdy jsme se snažili přijít na problém, proč to nefunguje. Bohužel neúspěšně. Motor si dělal, co se mu zachtělo, někdy se začal točit, když se čtečka otočila vzhůru nohama a někdy zas když se deska fyzicky zmáčkla. Dospěli jsme k závěru, že jelikož mám všechny součástky propojeny na nepájivém poli a spojené dlouhými vodiči, které fungují jako anténa, tak obvod přijímá nějaké rušivé signály, které ovlivňují motor. Musel bych celý obvod udělat znova pomocí tištěných plošných spojů, a i tak by to nebyla garance úspěchu. Takže jsme připájeli ke krystalu odpor, který z nepochopitelných důvodů rozjede motor takovou rychlostí, jakou chci, když se ho dotýkám prstem.

6.2 Sestavování modelu

Nejdříve jsem začal s přilepením držáku ložisek na rám výtahu. Ještě jsem ověřil, že vzdálenost mezi ložisky je taková, aby nebyl řetěz napnutý moc, což by mohlo způsobit, že se dřevěné osy prohnou, nebo že by byl napnutý moc málo, kde by se řetěz mohl zasekávat nebo vypadnout z ozubených koleček. Potom jsem pomocí vteřinového lepidla nalepil držáky na rám a pomocí svorek jsem je dočasně uchytil, aby mělo lepidlo dost času ztuhnout.

Následně jsem přilepil osy na uchycení platform na řetěz dvousložkovým lepidlem, které je silnější než vteřinové lepidlo a chtěl jsem mít jistotu, že osy neupadnou. Opět jsem pomocí svorek přichytil osy na řetěz, protože lepidlo potřebuje několik hodin do úplného zaschnutí.

Obrázek 31 Slepování řetězu

Umístil jsem dřevěné osy s ozubenými kolečky do držáku a nasadil na ně řetěz. Na dřevěnou základnu jsem si nakreslil, kde bude rám umístěn, vyvrtal jsem díry a našrouboval jsem nožičky do který jsem rám usadil. Do os jsem nasadil samotné platformy společně s vytisknutými podložkami, které slouží k tomu, aby měla platforma a řetěz dost místa mezi sebou a nezasekávali se o sebe.

Potom už jsem jenom přišrouboval RFID čtečku, Hallovy sondy, displej, motor a nepájivé pole na desku, nasadil jsem řetěz, který je mezi motorem a samotným výtahem a už stačilo jen vyzkoušet a bylo hotovo. Jediný problém je, že musím z nějakého nepochopitelného důvodu držet rezistor, aby se motor rozjel plnou rychlostí, takže občas omylem zavadím o platformu.

Obrázek 32 Konstrukce výtahu

Závěr

Cílem mé práce bylo vyrobit a naprogramovat model autovýtahu. Většina věcí šlo podle plánu, ale jsou tam i věci, které se nepovedly a udělal bych je příště jinak. Pohyb celého výtahu není vůbec plynulý, a to způsobuje to, že platforma nezastaví vždy na stejném místě. To by se dalo nejspíš vyřešit tím, že bych použil normální kovový řetěz a ozubená kolečka, která by se nezasekávala a pohyb by byl mnohem plynulejší. Největší problém je ten, že se motor nerozjede bez toho, aniž bych musel držet rezistor napájený na RFID čtečce. Nejvíc mě mrzí na tom to, že nevím, kvůli čemu to je a nemůžu s tím nic udělat. Možná bych to příště nedělal na nepájivém poli, ale vyrobil bych si vlastní plošný spoj, který by redukoval množství rušivých signálů.

Obecně mě práce poměrně bavila až na zadrhnutí s motorem a naučil jsem se spoustu nových věcí a taky vím, co příště udělat jinak.

Seznam použité literatury

1. Držák na RFID čtečku. Online. 2024. Dostupné z: https://www.printables.com/model/1012641-rfid-rc522-module-case. [cit. 2025-02-11].
2. Držák na LCD displej. Online. 2021. Dostupné z: https://www.printables.com/model/97206-16x2-lcd-housing. [cit. 2025-02-11].
3. Držák na motor. Online. 2024. Dostupné z: https://www.printables.com/model/1040314-little-stepper-motor-28byj-48-holder. [cit. 2025-02-11].
4. Řetěz. Online. 2019. Dostupné z: https://www.thingiverse.com/thing:3480133. [cit. 2025-02-11].
5. What are the Reasons and History of Car Lifts? Online. 2020. Dostupné z: https://isfelevator.com/what-are-the-reasons-and-history-of-car-lifts/. [cit. 2025-02-11].
6. Automated parking system. Online. 2024. Dostupné z: https://en.wikipedia.org/wiki/Automated_parking_system. [cit. 2025-02-11].
7. Space Saving: Amazing Vintage Photographs of Vertical Parking Lots From Between the 1920s and 1950s. Online. 2019. Dostupné z: https://www.vintag.es/2019/05/vertical-parking-lots.html. [cit. 2025-02-11].
8. Vintage photographs of early vertical parking garages, 1920-1960. Online. 2024. Dostupné z: https://rarehistoricalphotos.com/vintage-photographs-of-early-vertical-parking-garages/. [cit. 2025-02-11].
9. Vintage photographs of early vertical parking garages, 1920-1960. Online. 2024. Dostupné z: https://thehistoryinsider.com/the-history-of-vertical-parking-garages-through-photos/. [cit. 2025-02-11].