Středoškolská technika 2023

Setkání a prezentace prací středoškolských studentů na ČVUT

Prodejní automat

David Tran

SPŠ a VOŠ Písek, Karla Čapka 402, 397 11 Písek

Anotace

Cílem této práce je vytvořit funkční prodejní automat pomocí platformy Arduino. Tato práce zahrnuje popis programů, použitých součástek a jejich zapojení. Programování se provádí v programu Arduino IDE, který umožňuje psát a nahrávat programy do Arduina. Tento program bude obsahovat zobrazování informací na displeji, akceptor mincí, pohyb servomotoru a tlačítka sloužící pro výběr produktů.

Klíčová slova: Arduino UNO, prodejní automat, schéma zapojení, kód

Annotation

The aim of this final project is to create a functional vending machine using Arduino platform. This work includes a description of the programs, components used and their wiring. The programming is done in the Arduino IDE, which allows you to write and upload programs to the Arduino. This program will contain information on display, the coin acceptor, the movement of the servo motor and the buttons to select products.

Keywords: Arduino UNO, vending machine, circuit diagram, code

Obsah

1.Teoretický úvod

1.1 Prodejní automaty

Prodejní automaty jsou zařízení, která nám slouží k prodeji zboží nebo služeb bez potřeby obsluhy. Tyhle stroje jsou obvykle umístěny na veřejných místech, jako jsou nádraží, škola, letiště, obchodní centra a další. V posledních letech se výrazně rozšířilo i využití prodejních automatů v různých průmyslových odvětvích.

Prodejní automaty jsou typicky vybaveny senzory, které automatu pomáhají zajišťovat, že je zboží v automatu stále k dispozici a že platba za něj byla správně provedena. Tyhle senzory jsou obvykle propojeny s počítačovými systémy, které zajišťují řízení a administraci automatů.

V této maturitní práci bych se chtěl zaměřit na teoretický úvod do problematiky prodejních automatů, včetně uživatelského ovládání, zásobování zboží, technických možností a dalších faktorů, které ovlivňují úspěšnost a efektivitu prodejních automatů. Budeme se také zabývat budoucností prodejních automatů a jejich využitím v různých odvětvích.

1.2 Uživatelské ovládání v prodejních automatech

Uživatelské ovládání v prodejních automatech je klíčovým prvkem, který zajišťuje, že zákazníci si mohou jednoduše a pohodlně vybrat a koupit zboží bez nutnosti interakce s obsluhou. Správně navržené uživatelské rozhraní může také pomoci zvýšit prodeje a zlepšit zákaznickou spokojenost.

V dnešní době jsou prodejní automaty vybaveny různými typy uživatelských rozhraní, včetně dotykových obrazovek, klávesnic, tlačítek a dalších. Nejmodernější automaty umožňují také bezkontaktní platby pomocí mobilních telefonů nebo platebních karet s čipem.Kromě toho jsou uživatelská rozhraní často doplněna vizuálními prvky, jako jsou obrazovky, které zobrazují informace o zboží, ceny a další informace o automatové službě. V některých případech jsou tyto informace doplněny zvukovými efekty a hlasovými pokyny.Vzhledem k tomu, že prodejní automaty jsou umísťovány na místech s vysokým provozem lidí, jako jsou nádraží a letiště, je důležité zajistit, aby uživatelské rozhraní bylo jednoduché a intuitivní. Zákazníci musí být schopni rychle a

snadno vybrat zboží a provést platbu, aniž by se museli snažit pochopit složité navigační prvky.

1.3 Zásoba zboží v prodejních automatech

Zásoba zboží je jednou z klíčových součástí fungování prodejního automatu. Opatření dostatečného množství zboží je důležité pro uspokojení poptávky produktů. Existuje několik způsobů, jak opatřit dostatečnou zásobu zboží v prodejním automatu.

Jedním z možných řešení je manuální doplňování zásoby zboží. Tento způsob spočívá v pravidelném manuálním doplňování zboží do prodejního automatu provozovatelem. Tento postup vyžaduje neustálé sledování stavu zásoby zboží a pravidelné doplňování zboží.

Dalším možným řešením je automatické doplňování zboží. Tento postup je založen na instalaci senzorů, které sledují stav zásoby zboží v automatu. Když se zásoba zboží sníží pod určitou úroveň, senzory automaticky vyšlou signál na doplňovací jednotku, která doplní chybějící zboží. Tento způsob umožňuje provozovateli ušetřit čas a náklady na pravidelné manuální doplňování zásoby zboží.

Existují také prodejní automaty, které mají v sobě integrovanou chladící jednotku, což umožňuje prodej potravin, nápojů a dalších chladících výrobků. V takových automatech musí být zajištěna správná teplota a regulace vlhkosti, aby bylo zboží skladováno v optimálních podmínkách a bylo kvalitní pro zákazníka.

1.4 Technické možnosti prodejních automatů

Platební systémy – Prodejní automaty dnes často přijímají karty, mince, bankovky nebo dokonce mobilní platby pomocí aplikace.

Dotykové obrazovky – Mnoho automatů používá dotykové obrazovky pro snadnou a intuitivní navigaci uživatele.

Senzory – Senzory mohou být použity k detekci předmětů v automatech, jako jsou například produkty, mince nebo bankovky.

Telemetrie – Telemetrie umožňuje automatům posílat data o svém provozu a výkonu do centrálního systému, což může pomoci s vylepšením provozní efektivity.

Softwarová aktualizace – Díky připojení k internetu mohou být automatové systémy aktualizovány softwarově a lze tak přidávat nové funkce a vylepšení.

Komunikační technologie – Některé prodejní automaty používají bezdrátové technologie pro přenos dat a umožňují tak i vzdálené monitorování a zprávu.Vývoj výrobků – Výrobci automatů neustále vylepšují své produkty, aby poskytovali lepší a efektivnější služby. Například mohou nabízet více zdravých potravin, nebo integrovat šikovné funkce, jako je vytisknutí účtenky.

1.5 Prodejní automaty a jejich využití

Prodejní automaty jsou stroje, které prodávají zboží nebo služby bez nutnosti obsluhy člověkem. Tyto automaty se staly běžnou součástí moderního života a jsou využívány v mnoha různých oblastech.

Potraviny a nápoje – Prodejní automaty jsou velmi běžné v prodeji nápojů a svačin v oblastech s vysokým provozem, jako jsou například nádraží, letiště, univerzity a kancelářské budovy.

Zdravotní péče – V některých zemích se používají prodejní automaty pro výdej léků na předpis nebo prodej různých zdravotnických potřeb.

Knižní a hudební výrobky – Automaty nám také mohou sloužit k prodeji knih, časopisů, CD a DVD na mnoha místech, jako jsou například letiště, nádraží, univerzity a nákupní centra.

Výrobky pro zvířata – jsou využívány k prodeji potravin pro zvířata, jako jsou například krmiva pro psy a kočky, ale také k prodeji drobných předmětů pro domácí mazlíčky, jako jsou hračky nebo obojky.

Oblečení a módní doplňky – Některé prodejní automaty nabízejí módní doplňky, jako jsou například dámské punčochy nebo kravaty, nebo dokonce oblečení jako trička, džíny a další.

1.6 Budoucnost prodejních automatů

Budoucnost prodejních automatů je velmi perspektivní, protože se stále zlepšují a vyvíjejí nové technologie, které mohou být využity pro jejich vylepšení a rozšíření využití. Níže uvádím několik trendů, které se očekávají v budoucnosti prodejních automatů:

Bezdotykové platby – Stále více lidí preferuje bezkontaktní platby pomocí mobilních telefonů nebo platebních karet. V budoucnosti mohou být prodejní automaty vybaveny technologií pro bezdotykové platby, aby se zvýšila bezpečnost a pohodlnost při platbě.

IoT - Internet věcí (IoT) může být využit pro vylepšení a zefektivnění prodejních automatů. Například by mohly být vybaveny senzory, které by mohly automaticky sledovat a zaznamenávat počet prodaných produktů, stav zásob nebo kvalitu produktů.

Interaktivita – Budoucnost prodejních automatů může být zaměřena na vytvoření interaktivních zážitků pro lidi, například pomocí dotykových obrazovek, virtuální reality nebo rozšířené reality.

1.7 Bezpečnost prodejních automatů

Bezpečnost prodejních automatů je velmi důležitá. Protože se jedná o zařízení, která jsou umístěna veřejně přístupných prostorách, mohou být náchylná k vandalismu, krádeži a dalším zločinům. Proto jsou moderní prodejní automaty vybaveny bezpečnostními prvky, jako jsou:

Kamery: Mnoho moderních prodejních automatů je vybaveno kamerami, které monitorují okolí stroje a zaznamenávají případné krádeže nebo vandalismus.

Alarmy: Některé prodejní automaty mají zabudované alarmy, které se aktivují, když je stroj napaden nebo narušen.

Bezpečnostní zámky: Mnoho prodejních automatů má speciální zámky, které chrání zásoby v automatu před krádeží.

1.8 Výhody a nevýhody prodejních automatů

Prodejní automaty mají několik výhod, jako jsou například:

24hodinová dostupnost: Prodejní automaty umožňují zákazníkům nakupovat kdykoliv.

Nízké náklady na provoz: Prodejní automaty nepotřebují obsluhu, což znamená, že náklady na provoz jsou nižší než u tradičních prodejních míst.

Efektivita: Prodejní automaty umožňují rychlý a efektivní nákup zboží, bez nutnosti čekání ve frontách a s minimálním množstvím interakce s jinými lidmi.

Široká škála produktů: Prodejní automaty mohou nabízet širokou škálu produktů od jídla a nápojů až po hygienické potřeby a elektroniku.

Flexibilita: Prodejní automaty mohou být umístěny na mnoha místech, včetně parkovišť, letišť, nádraží a jiných veřejných místech, což zvyšuje jejich dostupnost pro zákazníky.

Nicméně existují také určité nevýhody spojené s prodejními automaty:

Omezená interakce se zákazníkem: Prodejní automaty nemohou poskytnout stejnou úroveň služeb jako lidská obsluha, což může být problém pro zákazníky s dotazy nebo specifickými požadavky.

Technické problémy: Pokud dojde k technickým problémům s prodejním automatem, může být jeho oprava složitá a nákladná.

Omezená nabídka: V některých prodejních automatech může být omezená nabídka zboží, což může být pro zákazníky nevýhodou.

2. Autorské řešení

Při tvorbě konstrukce prodejního automatu jsem musel vzít v úvahu mnoho faktorů, včetně bezpečnosti, funkčnosti a designu. Snažil jsem se vytvořit konstrukci, která by byla co nejvíce odolná, aby nedošlo k poškození a zároveň by umožňovala snadnou obsluhu a údržbu. Zvolené materiály a konstrukční prvky jsem pečlivě vybral.

Kromě samotné konstrukce jsem musel také zvážit zapojování součástek v automatu, aby nedocházelo k nedostatku místa a aby se použitým součástkám nic nestalo. Software prodejního automatu, který by splňoval všechny požadavky, které byly dané. Musel jsem zvolit vhodné zařízení pro detekci mincí, tlačítka, displej jakož i vhodné servomotory pro vydávání zboží. Dále jsem musel vytvořit software pro řízení prodejního automatu, který by umožňoval snadnou obsluhu pro zákazníky a zároveň by zajišťoval bezpečné a spolehlivé fungování.

Během celého procesu jsem se musel vypořádat s mnoha výzvami a problémy, jako jsou nepřesnosti při tvoření návrhu konstrukce, snadnou kabeláž, software, a také výběr vhodných materiálů a součástek, který automat bude obsahovat.

2.1 Konstrukce

Základ konstrukce prodejního automatu jsem zpracoval na webové stránce Maker Case. Nejprve jsem si navrhl rozměry boxu dle svých požadavků, které se vztahovaly k požadované velikosti a funkcionalitě automatu. Následně jsem musel návrh upravit v programech Inkscape a AutoCAD, abych mohl přidávat další prvky jako otvory pro akceptor mincí, displej, servomotory, tlačítka a vice podlaží pro vydání zboží a zároveň vytvořit prostor pro zapojení. Všechny tyto prvky jsem umístil tak, aby byly snadno dostupné a aby se dobře sladily s celkovým designem automatu.

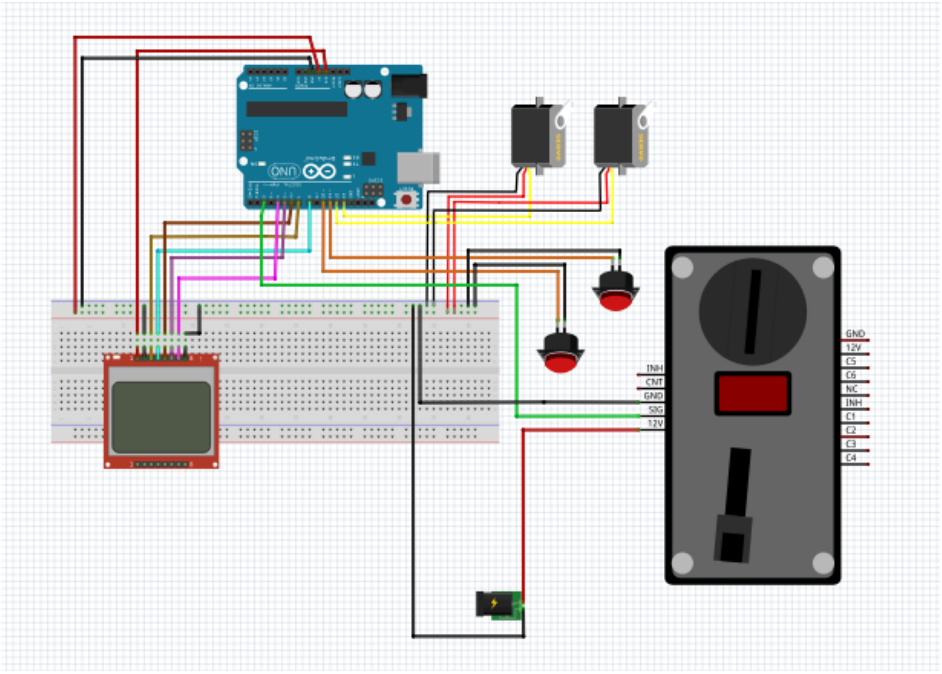
Vzhledem k tomu, že prodejní automat bude napájen tak jsem přidal také menší dírky, která jsou umístěna na vhodném místě pro usnadnění kabeláže.

Konečný návrh jsem nechal vyřezat od specializované firmy, která použila 4 mm překližku což mi návrh trošku pokazil a nešlo si také nevšimnout nepřesností, které nastaly při měření parametru jednotlivých součástek, kde jsem nepřesně určil, ale nakonec jsem si poradil i se spatnými parametry, aby vytvořila pevnou a stabilní konstrukci prodejního automatu. Převážně jsem využil k držení jednotlivých překližek konstrukce tavnou pistoli a části které nezavazely ke kabeláži jsem zalepil tavným lepidlem. Maximální rozměry konstrukce jsou 26 cm výšky a 23,5 cm šířky, což zaručuje, že automat bude dostatečně kompaktní, aby se dal umístit na většinu míst, ale zároveň dostatečně velký, aby bylo možné v něm skladovat a prodávat různé menší předměty nebo sladké tyčinky.

2 Přední automat

1 Konstrukce automatu

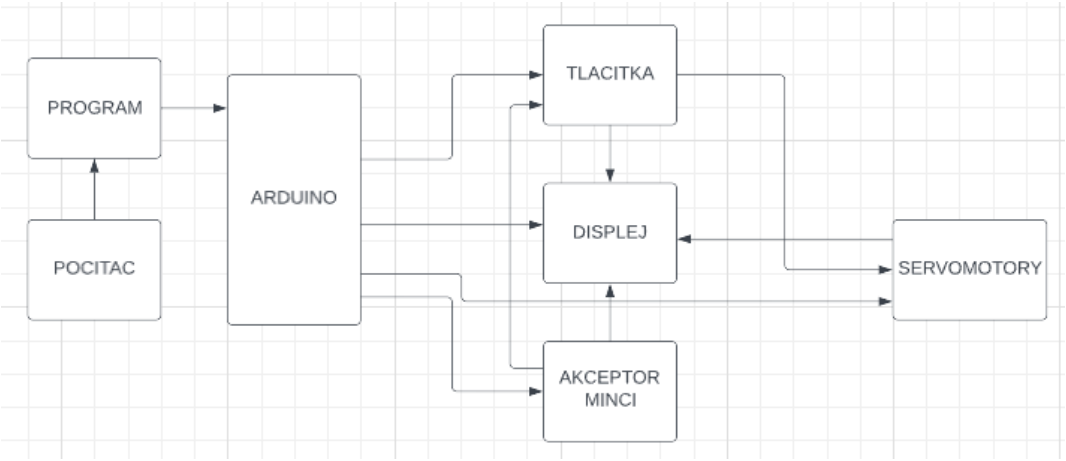
2.3 Schéma prodejního automatu ve Fritzingu



3 Schéma prodejního automatu ve Fritzingu

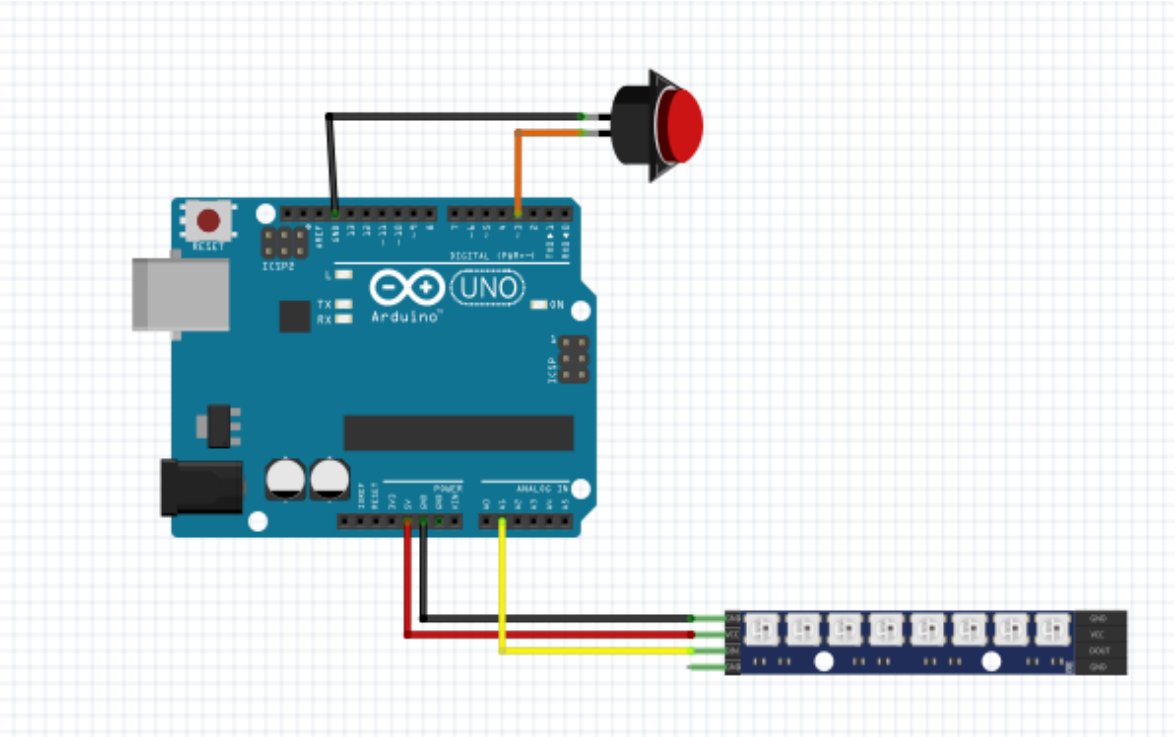
V zapojení se nám nachází mikrokontroler Arduino UNO, který nám řídí funkčnost jednotlivých periferií, LCD displej Nokia 5110, tlačítka, servomotory, akceptor mincí a konektor pro napájení akceptoru mincí.

2.4 Blokové schéma prodejního automatu



4 Blokové schéma prodejního automatu

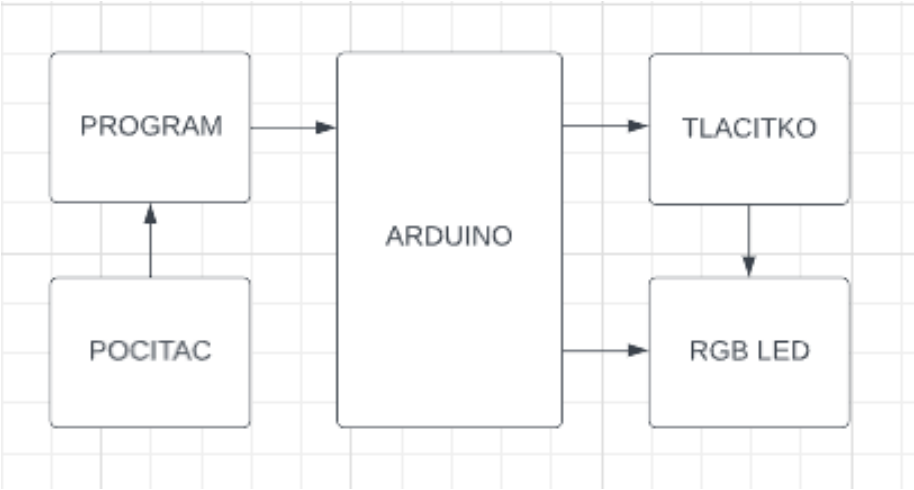
2.5 Schéma osvětlení ve Fritzingu



5 Schéma osvětlení ve Fritzingu

V zapojení se nám zas objevuje mikrokontroler Arduino Uno, tlačítko a RGB LED modul 8x NeoPixel WS2812.

2.6 Blokové schéma osvětlení



6 Blokové schéma osvětlení

3.Program

Při programování prodejního automatu jsem použil knihovny SPI.h, Adafruit_GFX.h a Adafruit_PCD8544.h. V souboru jsem nastavil propojovací piny displeje Adafruit_PCD8544 a definoval potřebné proměnné. Na začátku jsem nastavil pin pro akceptor mincí a inicializoval proměnné korun a credits na nulu. Dále jsem přidal potřebné proměnné pro servomotory a tlačítka, a definoval je jako vstupy pro tlačítka s pull-up rezistory. Poté jsem inicializoval sériovou komunikaci, přidal přerušení pro akceptor mincí a inicializoval displej s definovanými piny. Na začátku programu se zobrazí úvodní stránka na displeji s názvem PRODEJNI AUTOMAT a s pozdravem VAS VITA. Poté se zobrazí stránka pro vložení peněz s cenou produktu o hodnotě 20 Kč. Akceptor mincí jsem nastavil aby, přijímal pouze 5,10,20 Kč. Po vložení správných mincí, kde hodnota dosahuje 20 Kč, zobrazí se stránka s možností výběru. Pak se rozhoduje mezi dvěma varianty a po stisknutí tlačítka A nebo B se jeden ze servomotoru podle výběru začne točit a pošle našim směrem předmět, který byl vybrán. Poté se zase zobrazí stránka pro vložení peněz s cenou produktu a cyklus prodejního automatu se opakuje. Pro lepší pochopení mého programu jsem celý program popsal a ukazuji postupy, jaké jsem vykonával při programování. (viz kapitola 3.1–3.3.3)

3.1 Displej

Pro pracování s displejem jsem musel využít vhodnou knihovnu z Arduina, konkrétně jsem si vybral knihovny pro LCD displej 5110. Tyhle knihovny jsem si musel prvně stáhnout a poté jsem je vložil do svého projektu pomocí příkazu #include. Dále jsem využil dostupné možnosti této knihovny k vytvoření různých stránek na displeji.

3.1.1 Knihovny displeje

Knihovna SPI.h tahle knihovna mi poskytla funkci pro komunikaci se zařízeními, které využívají protokol SPI (Serial Peripheral Interface).

Knihovna Adafruit_GFX.h je grafická knihovna pro displeje, která mi poskytla možnosti grafických pomůcek (např. kreslení čar, kružnic, obdélníků atd.).

Knihovna Adafruit_PCD8544.h slouží jako ovladač pro řadič LCD PCD8544, který se používá v monochromatických displejích LCD Nokia 5110/3310. Tato knihovna mi poskytla inicializaci a ovládání displeje.

```
#include <SPI.h>
#include <Adafruit_GFX.h>
#include <Adafruit_PCD8544.h>
```

7 Knihovny displeje

3.1.2 Definice proměnných displeje

Aby mi mohl program pro displej fungoval tak jsem si definoval určité proměnné. Definoval jsem pět konstant, které mi určují číslo pinu na desce pro jednotlivé funkce, které jsem potřeboval pro komunikaci s displejem. Konkrétně:

Konstanta RST s hodnotou 8 určuje pin na desce, který slouží pro resetování displeje.

Konstanta CE s hodnotou 7 určuje pin na desce, který slouží pro výběr čipu displeje.

Konstanta DC s hodnotou 6 určuje pin na desce, který slouží pro ovládání datové a řídící komunikace s displejem.

Konstanta DIN s hodnotou 5 určuje pin na desce, který slouží pro přenos dat do displeje.

Konstanta CLK s hodnotou 4 určuje pin na desce, který slouží pro synchronizaci datového přenosu.

```
// nastavení propojovacích pinů
#define RST 8
#define CE 7
#define DC 6
#define DIN 5
#define CLK 4
```

8 Definice proměnných displeje

3.1.3 Inicializace displeje

Následně v kódu jsem nastavoval objekt display z knihovny Adafruit_PCD8544, která mi umožňuje komunikaci s LCD displejem. Tento objekt je vytvořen s parametry, které specifikují, jaký nebo který pin na desce slouží pro jednotlivé funkce komunikace s displejem

```
// inicializace LCD displeje z knihovny
Adafruit_PCD8544 display = Adafruit_PCD8544(4, 5, 6, 7,8);
```

9 Inicializace displeje

3.1.4 Stránky displeje

Úvodní stránka

Nejprve jsem pomocí příkazů *display. setTextSize (1)* a *display. setTextColor* (BLACK) nastavil velikost písma na 1 a barvu textu na černou. Následně se po dobu dvou sekund *delay (2000 ms)* která, slouží jako krátká pauza nic neděje. Poté *display.drawRect(0, 0, 84, 48, BLACK)* na displeji nakreslí černý obdélník o velikosti 84x48 pixelů. Opět následuje pauza po dobu dvou sekund. Další příkazy *display.setCursor(20,10)*, *display.println("PRODEJNI")*, *display.setCursor(23,20)*, *display.println("AUTOMAT")* a

display.setCursor(20,30), *display.println("VAS VITA")* nastavují pozici kurzoru a vypisují na displej text "PRODEJNI", "AUTOMAT" a "VAS VITA" na předem určených pozicích. *Display.display()* nám všechny změny promítne na displej a následuje další dvousekundová pauza. Úvodní stránka se nám zobrazí pouze jednou na začátku pří zapnutí automatu, a poté se už neobjeví.

10 Úvodní stránka displeje

Hlavní stránka

Tento kód mi nastavil displej, aby mi zobrazil nápis "Vložte částku" a částku 20 Kč s okrajem. Prvně se vymaže celý obsah displeje pomocí funkce *display.clearDisplay().*Změní se velikost textu na 1 a barva textu na černou pomocí *display.setTextSize(1)* a *display.setTextColor(BLACK).* Následně se na displeji vykreslí obdélník se souřadnicemi 0, 0 pro horní levý roh a 84, 48 pro dolní pravý roh, opět černou barvou, za pomocí funkce *display.drawRect(0, 0, 84, 48, BLACK).*

Po dvou sekundách funkce *delay(2000),* aby se kupující stihl na displej podívat. Nám *display.setCursor(4,5)* nastaví pozice kurzoru na souřadnice 4,5 (řádek 4 a sloupec 5) a funkce display.println("Vložte částku") se na displej zobrazí text "Vložte částku".

Následně se pracuje s velikosti textu na 2 *display.setTextSize(2)* a vykreslí se obdélník s větším textem částky 20 Kč pomocí *display.drawRect(8, 15, 70, 25, BLACK).* Pozice je daná *display.setCursor(15,20)* a text se bude zobrazen funkci *display.println("20 KC")* a *display.display()* zobrazuje všechny změny na displeji.

```
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(BLACK);
display.drawRect(0, 0, 84, 48, BLACK);
display.setCursor(4,5);
display.println("VLOZTE CASTKU");
display.setTextSize(2);
display.drawRect(8, 15, 70, 25, BLACK);
display.setCursor(15,20);
display.println("20 KC");
display.display();
```

11 Hlavní stránka displeje

Výběrová stránka

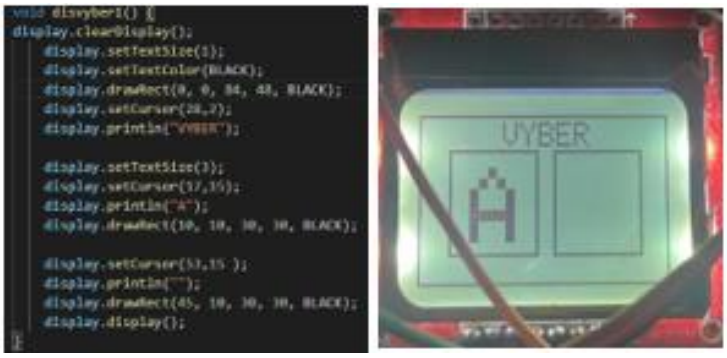
Je stránka displeje, na které jsou nastaveny dvě volby označené písmeny "A" a "B". Na obou volbách je černý rámeček a jsou umístěny uprostřed stránky. Nahoře je nápis "VYBER". Princip celkově je velmi podobný předchozímu kódu, s výjimkou toho, že tentokrát jde o výběr ze dvou možností.

```
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(BLACK);
display.drawRect(0, 0, 84, 48, BLACK);
display.setCursor(20,2);
display.println("VYBER");

display.setTextSize(3);
display.setCursor(17,15);
display.println("A");
display.drawRect(10, 10, 30, 30, BLACK);

display.setCursor(53,15);
display.println("B");
display.drawRect(45, 10, 30, 30, BLACK);
display.display();
```

12 Výběrová stránka displeje

První výběrová stránka

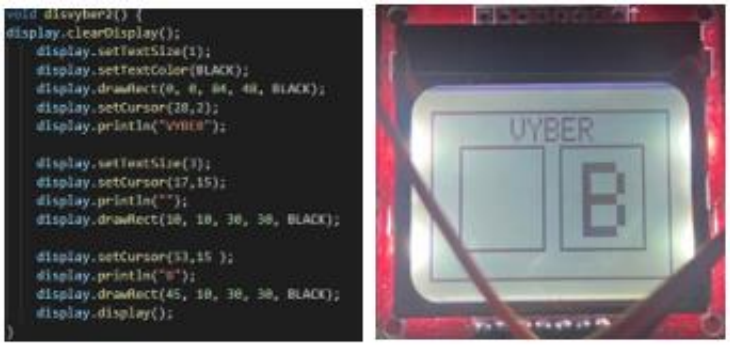
Tahle stránka mi nastavuje displej, aby zobrazoval text a tvary. Nejprve se vymaže obrazovka a nastaví se velikost písma, barva textu a rámu. Poté se nastavuje pozice kurzoru a vypíše se text "VYBER". Dále se nastaví velikost písma na 3 a vypíšou se tvary "A" a " " s rámy. Nakonec se displej aktualizuje a zobrazí se na něm to, co bylo nastaveno.



13 První výběrová stránka

Druhá výběrová stránka

Tahle stránka je totožná stránce výběrová1. Rozdíl se pouze nachází ve zobrazených písmech, kde se zde zobrazí písmo B místo písmena A.



14 Druhá výběrová stránka

3.1.5 Funkce stránek

Každá stránka v programu je aktivní a přístupná, avšak některé z nich se zobrazují pouze jednou, jako například úvodní stránka, která se nastavuje v funkci void setup a nemění se. Ostatní stránky jsou umístěny v funkci void loop, například hlavní stránka, která informuje uživatele o vkládání peněz. Jakmile je hodnota vkládaných peněz dosažena, displej přepne na výběrovou stránku, která nabízí 2 možnosti výběru pomocí 2 tlačítek. Po výběru se zobrazí stránka výběrová 1 nebo výběrová 2, která ukazuje vybranou možnost. Nakonec se ale na displeji opět objeví hlavní stránka, kde cyklus čeká na další vklad peněz pro opětovné spuštění procesu.

3.2 Servomotory a tlačítka

3.2.1 Nastavení konstant

Prvně jsem si nastavil konstanty, které následně budu využívat v programu pro účel řízení servomotoru pomocí tlačítka. Hodnoty konstant jsou pevně stanovené a nelze je měnit během běhu programu, což umožňuje jednodušší a stabilnější programování.

```
////////////////// Servomotory a Tlačítka//////////////////
const int clockwise = 2475;
const int counterclockwise = 1300;
const int ServoA = 9;
const int ServoB = 12;
const int ButtonA = 11;
const int ButtonB = 10;
```

15 Nastavení konstant tlačítek a serva

3.2.2 Nastavení void setup

Jelikož piny Arduina jsou vstupně výstupní, je potřeba je nastavit na stav, který zrovna potřebujeme. Celkově tedy tento kód mi nastavuje dva vstupní piny na Arduinu. Jako tlačítka, která jsou připojena s interními pull-up rezistory pro stabilitu signálu.

```
  pinMode(ButtonA, INPUT_PULLUP);
  pinMode(ButtonB, INPUT_PULLUP);
```

16 Nastavení void setup

3.2.3 Cyklus programu

```
//////////////////////Pokud tlacitko bude stisknuty//////////////
 if ((digitalRead(ButtonA) == LOW) ) {
 disvyber1();

//////////////////POHYB SERVOMOTORU/////////////////////////////
 for(int i=0; i<70; i++)  // SERVO
 {
 digitalWrite(ServoA,HIGH);
 delayMicroseconds(clockwise);
 digitalWrite(ServoA,LOW);
 credits = 0;
 delay(18.5); // 18.5ms
```

```
//////////////////////Pokud 2 tlacitko bude stisknuty//////////////
  if ((digitalRead(ButtonB) == LOW) ) {
   disvyber2();

//////////////////POHYB SERVOMOTORU/////////////////////////////
   for(int i=0; i<70; i++)  // SERVO
   {
   digitalWrite(ServoB,HIGH);
   delayMicroseconds(clockwise);
   digitalWrite(ServoB,LOW);
   credits = 0;
   delay(18.5); // 18.5ms
```

17 Cyklus programu

Tento program mi slouží pro řízení pohybu dvou servomotorů, které jsou využity jako výdejní zařízení. Servomotory jsou označeny jako ServoA a ServoB a jsou připojeny k pinům 13 a 12 na Arduinu.

Pokud je stisknuto tlačítko A, je zavolána funkce disvyber1(), která zobrazí zprávu (viz obrázek 13), a poté je servomotor připojený k pinu 13 (ServoA) rozpohybován nastavením pinu na HIGH, zpožděním po určitou dobu *(delayMicroseconds(clockwise))* a dále nastavením pinu zpět na LOW. Tento proces se opakuje 70krát ve smyčce for. Poté se zavolá funkce *dispenize()*, která zobrazí jinou zprávu. (viz obrazek 11).

Pokud je stisknuto tlačítko B, proběhne podobný proces, ale se servomotorem připojeným na pin 12 *(ServoB)*. Nejprve se zavolá funkce disvyber2(), aby se zobrazila zpráva (viz obrazek 14) pak se servomotor rozpohybuje nastavením pinu na HIGH, zpožděním po určitou dobu *(delayMicroseconds(clockwise))* a poté nastavením pinu zpět na LOW. Tento proces se opakuje 70krát ve smyčce for. Nakonec je zavolána funkce *dispenize()*, která zobrazí jinou zprávu(viz obrázek 11),.

3.3 Akceptor mincí

Jednotlivé mince jsem si pomocí manuálu nastavil na určité impulzy. Akceptor mincí přijme pouze 5,10 a 20 kč. To znamená, že když mincovník přijme minci v hodnotě 5 Kč, vyšle jeden impulz, když přijme minci v hodnotě 10 Kč, vyšle dva impulzy, a když přijme minci v hodnotě 20 Kč, vyšle čtyři impulzy.

3.3.1 Nastavení konstant

Jedná se pouze o deklaraci několika proměnných, které jsem použil v programu pro akceptor mincí.

Konstanta "coinpin" určuje číslo pinu, na kterém jsem připojil akceptor mincí k Arduino desce. Konstanta "celek" udává cílový počet peněz, které potřebuje získat pomocí mincovníku.

Proměnná "korun" jsem využil k uchování počtu mincí, které byly přijaty akceptorem mincí, a je deklarována jako "volatile". To znamená, že hodnota této proměnné může být kdykoliv změněna, i když k tomu nedojde v samotném programu.

```
//////////////////// AKCEPTORMINCI//////////////////////
const int coinpin = 2;
const int celek = 20;
// Variables
volatile int korun = 0;
int credits = 0;
```

18 Nastavení konstant akceptor mincí

3.3.2 Inicializace programu

Serial.begin(9600) byla použita pro inicializaci sériové komunikace s rychlostí přenosu dat 9600 bodů za sekundu. Tuto funkci jsem použil pro nastavení sériového portu pro přenos dat mezi Arduinem a počítačem.

Pomocí funkce *attachInterrupt(digitalPinToInterrupt(coinpin), coinInterrupt, RISING)* jsem nastavil přerušení pro reakci na signál z mincovníku. Použil jsem funkci *digitalPinToInterrupt* k převodu čísla pinu na číslo přerušení pro daný pin coinpin. Toto číslo jsem pak použil ve funkci *attachInterrupt* k navázání přerušení a specifikování, že přerušení má být vyvoláno na stoupající hraně signálu (RISING).

Toto nastavení znamená, že při každém detekování mince na mincovníku se vykoná funkce *coinInterrupt*, která bude dále zpracovávat příchozí data z mincovníku.

```
//////////////////AKCEPTORMINCI//////////////////
  Serial.begin(9600);
  attachInterrupt(digitalPinToInterrupt(coinpin), coinInterrupt, RISING);
```

```
// Interrupt
void coinInterrupt(){
  // pokazde kdyz pulz je poslan z akceptoru minci, přidá se 5 korun
  korun = korun + 5;
}
```

19 Inicializace akceptoru mincí

3.3.3 Podmínka

Podmínka říká, že pokud byl přijat dostatečný počet mincí, aby se dosáhlo cílové hodnoty, provede se následující:

Proměnná credits se zvýší o hodnotu 1. Tato proměnná uchovává celkový počet kreditů, tedy kolikrát se podařilo získat požadovanou cílovou hodnotu mincí.

Proměnná korun se sníží o hodnotu celek. celek je proměnná, která obsahuje hodnotu mince, kterou chceme akceptoru přijímat. Tímto krokem se "odečtou" přijaté mince, které už byly použity k dosažení cílové hodnoty.

Celkově tato podmínka implementuje mechanismus pro ukládání a zpracování mincí přijatých mincovníkem, a to tak, aby se vždy získala cílová hodnota mincí. Pokud mincovník přijme dostatečný počet mincí, program to pozná, zvýší počet kreditů a odečte přijaté mince od aktuálního stavu mincí.

```
if (korun >= celek) {
  credits = credits + 1;
  korun = korun - celek;
}
```

20 Podmínka

```
if (credits > 0) {
  ////////////////////////////DISPLAJ ZOBRAZUJICI VYBER MOZNOSTI////////////////////////
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(BLACK);
  display.drawRect(0, 0, 84, 48, BLACK);
  display.setCursor(28,2);
  display.println("VYBER");

  display.setTextSize(3);
  display.setCursor(17,15);
  display.println("A");
  display.drawRect(10, 10, 30, 30, BLACK);

  display.setCursor(53,15);
  display.println("B");
  display.drawRect(45, 10, 30, 30, BLACK);
  display.display();
```

21 Podmínka if

Tento kód mi zobrazuje výběrové možnosti na displeji, pokud uživatel má na účtu alespoň víc než 0 kreditu neboli pokud je credits> 0 (větší než nula) tak proběhne na displeji kód.

Konkrétně se na displeji objeví text "VYBER" v horní části, následovaný volbami – A a B. (viz obrázek 12)

viz. Obrázek Výběrová stránka. Kód dále následuje na servomotory a tlačítka

4 Druhý program v Arduinu

V druhém programu se chci zaměřit na osvětlení, u kterého jsem bohužel musel použít druhé Arduino kvůli nefunkčnosti akceptoru mincí a displeje po připojení modulu 8x NeoPixel WS2812 RGB LED. Důvodem nejspíše bude velký odběr. Akceptor mincí nereagoval na impulzy a poskytoval mi nesmyslné hodnoty, zatímco brava písmen u displeje se výrazně snížila nebo displej vůbec nefungoval.

4.1 Knihovna pro RGB LED

```
#include <Adafruit_NeoPixel.h>
```

22 Knihovna RGB LED

Jako v předchozím programu jsem si stáhl z Arduina knihovnu Adafruit_Neopixel.h , která mi poskytla snadný způsob ovládání modulů NeoPixel WS2812 RGB LED a umožňuje mi měnit barvu a jas jednotlivých pixelů v modulu. Zařazením knihovny jsem získal přístup k funkcím knihovny.

4.2 Definice proměnných a konstant

```
#define PIN A1
#define NUM_LEDS 8
const int TlacitkoLED = 3;
Adafruit_NeoPixel rgb_leds(NUM_LEDS, PIN, NEO_GRB + NEO_KHZ800);

int ledState = 0; // 0 = vypnuto, 1 = bílá, 2 = modrá
```

23 Definice proměnných a konstant RGB LED

V kódu, který jsem uvedl, jsou na začátku programu definovány některé konstanty a proměnné.

Pomocí #define PIN A1 jsem si definoval konstantu s názvem PIN a nastavuji její hodnotu na A1 k definování pinu na desce Arduino, ke kterému je připojen modul LED NeoPixel WS2812.

#define NUM_LEDS 8 mi definuje další konstantu s názvem NUM_LEDS a nastavuje její hodnotu na 8, který mi určuje počet LED diod v modulu.

const int TlacitkoLED = 3; vytvoří proměnnou s názvem TlacitkoLED a nastaví její hodnotu na 3 k definování pinu na desce Arduino, ke kterému je připojeno tlačítko.

Adafruit_NeoPixel rgb_leds(NUM_LEDS, PIN, NEO_GRB + NEO_KHZ800); mi vytvoří instanci třídy Adafruit_NeoPixel s názvem rgb_leds a přijímá tři argumenty: NUM_LEDS, PIN a NEO_GRB + NEO_KHZ800. Tím se nastaví modul LED NeoPixel WS2812 se zadaným počtem LED diod(8), připojených k zadanému pinu (PIN A1) a používajících zadané pořadí barev LED (NEO_GRB) a rychlost přenosu dat (NEO_KHZ800).

int ledState = 0; deklaruje celočíselnou proměnnou s názvem ledState a inicializuje ji na hodnotu 0. Tuhle proměnnou jsem později v programu používal k sledování aktuálního stavu modulu LED. Pokud je její hodnota 0, je modul LED vypnutý. Pokud je její hodnota 1, LED modul zobrazuje bílou barvu. Pokud je její hodnota 2, modul LED zobrazuje modrou barvu.

4.3 Nastavení void setup

```
//////////////////VOID SETUP////////////////////////
void setup() {
  pinMode(TlacitkoLED, INPUT_PULLUP);
  rgb_leds.begin();
  delay(10);
}
```

24 Nastavení void setup RGB LED

pinMode(TlacitkoLED, INPUT_PULLUP); nastaví pin definovaný proměnnou TlacitkoLED jako vstupní pin s pull-up rezistorem. Tím se pin nastaví na čtení stavu tlačítka nebo přepínače, který je k němu připojen.

rgb_leds.begin(); inicializuje dříve vytvořený objekt rgb_leds. Tím se nastaví modul LED NeoPixel nebo WS2812 pro komunikaci s deskou Arduino.

delay(10); v programu přidává zpoždění 10 milisekund. Jedná se o krátkou prodlevu, která slouží k tomu, aby měl modul LED NeoPixel WS2812 čas na zapnutí a stabilizaci, než s ním začne program komunikovat.

4.4 Nastavení barev RGB LED

```
void setColorLED(uint32_t color) {
  for (int led_number = 0; led_number < NUM_LEDS; led_number++) {
    rgb_leds.setPixelColor(led_number, color);
  }
  rgb_leds.show();
}
```

25 Nastavení barev RGB LED

Vytvořil jsem si vlastní funkci *setColorLED()*, která slouží k nastavení barvy všech LED diod v modulu NeoPixel WS2812 LED na stejnou barvu, kterou zadáme parametrem color.

V definici funkce void *setColorLED(uint32_t color)* jsem určil, že tato funkce nevrací žádnou hodnotu (void) a přijímá jediný parametr color typu uint32_t, což je celé 32bitové číslo bez znaménka. Parametr color slouží k zadání požadované barvy pro modul LED.

Následně jsem vytvořil smyčku *for (int led_number = 0; led_number < NUM_LEDS; led_number++)*, která prochází přes všechny LED v modulu LED. Tuto smyčku jsem vytvořil pomocí *for* cyklu, který opakuje blok kódu uvnitř složených závorek určitý počet krát, který je určen dříve definovanou konstantou NUM_LEDS.

Pomocí funkce *rgb_leds.setPixelColor(led_number, color)*; jsem nastavil barvu LED na aktuálním indexu led_number na zadanou barvu.

Nakonec jsem využil funkce *rgb_leds.show();* k aktualizaci modulu LED tak, aby zobrazoval nové barvy nastavené funkcí *setPixelColor()*.

4.5 Nastavení void loop

```
//////////////////////////VOID LOOP NASTAVENI//////////////////////////////
void loop() {
  if ((digitalRead(TlacitkoLED) == LOW)) {
    if (ledState == 0) {
      // LED jsou vypnuté, nastaví se bílá barva
      setColorLED(rgb_leds.Color(255, 255, 255));
      ledState = 1;
    } else if (ledState == 1) {
      // LED jsou nastavené na bílou, nastaví se modrá barva
      setColorLED(rgb_leds.Color(0, 0, 255));
      ledState = 2;
    } else {
      // LED jsou nastavené na modrou, vypnou se
      setColorLED(rgb_leds.Color(0, 0, 0));
      ledState = 0;
    }
    delay(500); // oddálení pro snížení šumu
  }
}
```

26 Nastavení void loop RGB LED

Tento kód, který se nachází ve funkci loop(), je neustále opakován, jakmile nahraji program do Arduina. Tato funkce nevrací žádnou hodnotu (void).

V podmínce *if ((digitalRead(TlacitkoLED) == LOW))* testuji, zda je tlačítko pro změnu barvy stisknuto. Tlačítko je připojeno na pin č. 3, který jsem v setup() funkci nastavil jako INPUT_PULLUP. Pokud je tlačítko stisknuto, kód uvnitř podmínky se provede.

Vnitřek podmínky se skládá z několika vnořených podmínek *if-else*. Pokud je ledState rovno 0, což znamená, že jsou LED vypnuté, nastavím pomocí funkce *setColorLED()* bílou barvu a ledState změním na 1. (viz obrázek 27)

Pokud je ledState rovno 1, znamená to, že jsou LED již bílé, a tak pomocí funkce *setColorLED()* nastavím modrou barvu a ledState změním na 2. (viz obrázek 28)

Pokud je ledState rovno 2, znamená to, že jsou LED již modré, a tak pomocí funkce *setColorLED()* vypnu (nastavím černou barvu) a ledState změním zpět na 0. (viz obrázek 29)

Funkce *delay(500)* mi vloží časové zpoždění o 500 milisekund mezi každou změnou barvy. To snižuje šum na signálech a zajišťuje, že uživatel bude mít dostatek času stisknout tlačítko pro změnu barvy LED.

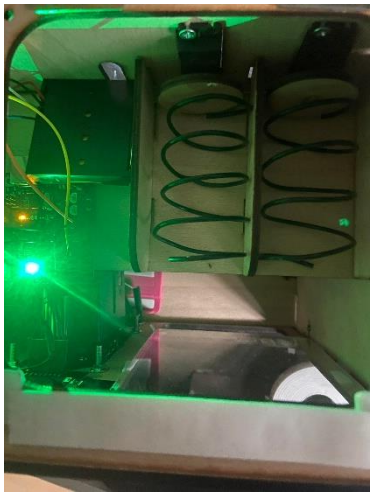
4.6 Ukázka funkčnosti

Při prvním stisknutí tlačítka se LED světlo NeoPixel WS2812 nastaví na bílou barvu, jak je specifikováno v programu. Z obrázku nelze vidět bíla barva kvůli barvě dřeva, ale také špatnému osvětlení.

27 RGB LED při prvním stistku tlačítka

Při druhém stisku tlačítka se LED světlo NeoPixel WS2812 LED nastaví na modrou barvu, jak je specifikováno v programu.

28 RGB LED při druhém stistku tlačítka



Při třetím stisku tlačítka se světlo NeoPixel WS2812 LED vypne.

29 RGB LED při třetim stistku tlačítka

Závěr

Cílem této maturitní práce bylo vytvořit prodejní automat, aby obsahoval vhodné součástky a také mikrokontroler. Prodejní automat musel splňovat přijímání mincí a na základě toho nabízet produkty. Při plnění úkolu jsem se setkal s mnoha nečekanými komplikacemi, které mi původně znemožňovaly dokončit práci včas. Nicméně jsem se nevzdal a trpělivě pracoval na nacházení problémů a řešení, což mi nakonec umožnilo úkol úspěšně splnit. V této práci jsem nejdříve rozebral teoretický úvod k problematice prodejního automatu. Poté jsem popsal své autorské řešení, kde jsem se věnoval postupu řešení, konstrukci modelu a ukázal schémata ve Fritzingu a blokové schéma, kde jsem naznačil použité periférie v mé práci. Zapojení periférií nebylo složité, ale přišlo k několika problémům, zejména s funkčností akceptoru mincí, který na začátku posílal nesmyslné hodnoty. Chyba byla nakonec nalezena ve špatných drátech a kabeláži celého projektu, kde jsem musel rozebrat celé zapojení a navrhnout lepší řešení a uspořádání místa v konstrukci. V další části jsem se věnoval popisu mého programu. Nejdůležitější částí pro mě bylo propojení všech součástek. Při programování jsem se zdržel u nastavování displeje, kde jsem nejprve měl problémy s komunikací, ale nakonec jsem našel vhodnou knihovnu s dostatečnými funkci. Propojení součástek nakonec fungovalo dobře a program nevykazoval žádné chyby. Propojení nakonec dokážu říct, že dopadlo dobře. Dále jsem k programu chtěl také přidat osvětlení k automatu využitím RGB LED a nastavovat různé barvy světla pro můj automat, ale při přidání mi nereagovalo polovina součástek tak jsem využil zbytek svého času a přidal zvlášť další Arduino, které mi sloužilo pouze pro ovládání samotného světla pomocí tlačítka. Napsání programu pro jedno rozsvícení nebylo tak obtížné, ale tlačítkem jsem chtěl nastavovat více barev než jednu a také i samostatné vypínání stejným tlačítkem světlo. Tím, že jsem se rozhodl přidat k prodejnímu automatu osvětlení pomocí RGB LED a programovat jeho ovládání, jsem si ztížil práci a musel věnovat zbytečně mnoho času programování. Tato přidaná práce vyžadovala další součástky a také další Arduino desku naštěstí na poslední chvílí se mi podařilo zprovoznit osvětlení podle mých představ a byl jsem tedy spokojen se svou finální prací. Celkově bych tedy shrnul, že cílem mé maturitní práce bylo vytvořit plně funkční prodejní automat, který bude schopen přijímat mince a nabízet odpovídající produkty na základě hodnoty mincí a výsledkem musím říct, že Jsem pyšný na svou práci a věřím, že má práce možná poslouží jako inspirace pro další studenty, kteří se v budoucnu rozhodnou věnovat podobnému projektu.

Seznam obrázků

Seznam použité literatury

[1] „Wikipedia,“ [Online]. Available: https://en.wikipedia.org/wiki/Vending_machine.]]

[2] „Svavending,“ [Online]. Available: https://www.svavending.com.au/useful-information/safety-issues-with-vending-machines/.

[3] „Blogs.vendify.in,“ [Online]. Available: https://blogs.vendify.in/advantages-disadvantages-of-vending-machine/.

[4] „Wevolver,“ [Online]. Available: https://www.wevolver.com/article/building-internet-connected-smart-vending-machines-with-powerful-single-board-computers.

[5] „ccv.eu,“ [Online]. Available: https://www.ccv.eu/en/2023/the-rise-of-smart-vending-machines/.

[6] „dtc-uk,“ [Online]. Available: https://www.dtc-uk.com/knowledge-centre/stock-control-vending-machines-more-than-just-a-gimmick/?cn-reloaded=1.

[7] V. S. ,. K. M. Eriyeti Murena, „researchgate,“ [Online]. Available: https://www.researchgate.net/publication/343730952_Design_of_a_Control_System_for_a_Vending_Machine.

Příloha I.

```
#include <SPI.h>
#include <Adafruit_GFX.h>
#include <Adafruit_PCD8544.h>
//////////////////// AKCEPTORMINCI//////////////////////
const int coinpin = 2;
const int celek = 20;
// Variables
volatile int korun = 0;
int credits = 0;
///////////////// Servomotory a Tlačítka//////////////////
const int clockwise = 2475;
const int counterclockwise = 1300;
const int ServoA = 9;
const int ServoB = 12;
const int ButtonA = 11;
const int ButtonB = 10;

// nastavení propojovacích pinů
#define RST 8
#define CE 7
#define DC 6
#define DIN 5
#define CLK 4

// inicializace LCD displeje z knihovny
Adafruit_PCD8544 display = Adafruit_PCD8544(4, 5, 6, 7,8);
```

```
// Interrupt
void coinInterrupt(){
// pokazde kdyz pulz je poslan z akceptoru minci, přidá se 5 korun
korun = korun + 5;
}

//////////////////VOID SETUP////////////////////////
void setup()   {
korun = 0;
credits = 0;

//////////////////TLACITKA/////////////////
pinMode(ButtonA, INPUT_PULLUP);
pinMode(ButtonB, INPUT_PULLUP);
//////////////////AKCEPTORMINCI/////////////////
Serial.begin(9600);
attachInterrupt(digitalPinToInterrupt(coinpin), coinInterrupt, RISING);
////////////////////////////DISPLAY////////////////////////////
display.begin();
display.setContrast(40);
display.display(); // zobrazeni
delay(2000);
display.clearDisplay();   // vymazani displeje


////////////////////////////DISPLEJE////////
///UVODNI STRANKA
display.setTextSize(1);
display.setTextColor(BLACK);
delay(2000);
display.drawRect(0, 0, 84, 48, BLACK);
```

```
delay(2000);
display.setCursor(20,10);
display.println("PRODEJNI");
display.setCursor(23,20);
display.println("AUTOMAT");
display.setCursor(20,30);
display.println("VAS VITA");

display.display();
delay(2000);

// Nastaveni DISPLEJE PRO VLOZENI PENEZ
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(BLACK);
display.drawRect(0, 0, 84, 48, BLACK);
display.setCursor(4,5);
display.println("VLOZTE CASTKU");
display.setTextSize(2);
display.drawRect(8, 15, 70, 25, BLACK);
display.setCursor(15,20);
display.println("20 KC");
display.display();
}
////////////////////////////VOID LOOP NASTAVENI//////////////////////////////
void loop() {

// Pokud jsme dosáhli cílového množství mincí, zvýšíme počet kreditů a
vynulujeme počítadlo korun.
if (korun >= celek) {
credits = credits + 1;
korun = korun - celek;
```

```
}

// jestli ne tak čekáme
else {
}

Serial.print(korun);
Serial.print(" korun toward current credit and ");
Serial.print(credits);
Serial.println(" credit(s) earned so far.");
delay(1000);

if (credits > 0) {
//////////////////////////DISPLAJ ZOBRAZUJICI VYBER MOZNOSTI////////////
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(BLACK);
display.drawRect(0, 0, 84, 48, BLACK);
display.setCursor(28,2);
display.println("VYBER");

display.setTextSize(3);
display.setCursor(17,15);
display.println("A");
display.drawRect(10, 10, 30, 30, BLACK);

display.setCursor(53,15);
display.println("B");
display.drawRect(45, 10, 30, 30, BLACK);
display.display();
```

```
//////////////////////Pokud tlacitko bude
stisknuty//////////////////////////////
if ((digitalRead(ButtonA) == LOW) ) {
disvyber1();

//////////////////////POHYB SERVOMOTORU//////////////////////////////
for(int i=0; i<70; i++)  // SERVO
{
digitalWrite(ServoA,HIGH);
delayMicroseconds(clockwise);
digitalWrite(ServoA,LOW);
credits = 0;
korun = 0;
delay(18.5); // 18.5ms



}
dispenize();
}
//////////////////////Pokud 2 tlacitko bude
stisknuty//////////////////////////////
if ((digitalRead(ButtonB) == LOW) ) {
disvyber2();

//////////////////////POHYB SERVOMOTORU//////////////////////////////
for(int i=0; i<70; i++)  // SERVO
{
digitalWrite(ServoB,HIGH);
delayMicroseconds(clockwise);
digitalWrite(ServoB,LOW);
credits = 0;
korun = 0;
```

```
delay(18.5); // 18.5ms




}
dispenize();
}
}
}

void dispenize() {
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(BLACK);
delay(2000);
display.drawRect(0, 0, 84, 48, BLACK);
delay(2000);
display.setCursor(4,5);
display.println("VLOZTE CASTKU");
display.setTextSize(2);
display.drawRect(8, 15, 70, 25, BLACK);
display.setCursor(15,20);
display.println("20 KC");
display.display();
}

void disvyber1() {
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(BLACK);
```

```
display.drawRect(0, 0, 84, 48, BLACK);
display.setCursor(28,2);
display.println("VYBER");

display.setTextSize(3);
display.setCursor(17,15);
display.println("A");
display.drawRect(10, 10, 30, 30, BLACK);

display.setCursor(53,15 );
display.println("");
display.drawRect(45, 10, 30, 30, BLACK);
display.display();
}

void disvyber2() {
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(BLACK);
display.drawRect(0, 0, 84, 48, BLACK);
display.setCursor(28,2);
display.println("VYBER");

display.setTextSize(3);
display.setCursor(17,15);
display.println("");
display.drawRect(10, 10, 30, 30, BLACK);

display.setCursor(53,15 );
display.println("B");
display.drawRect(45, 10, 30, 30, BLACK);
```

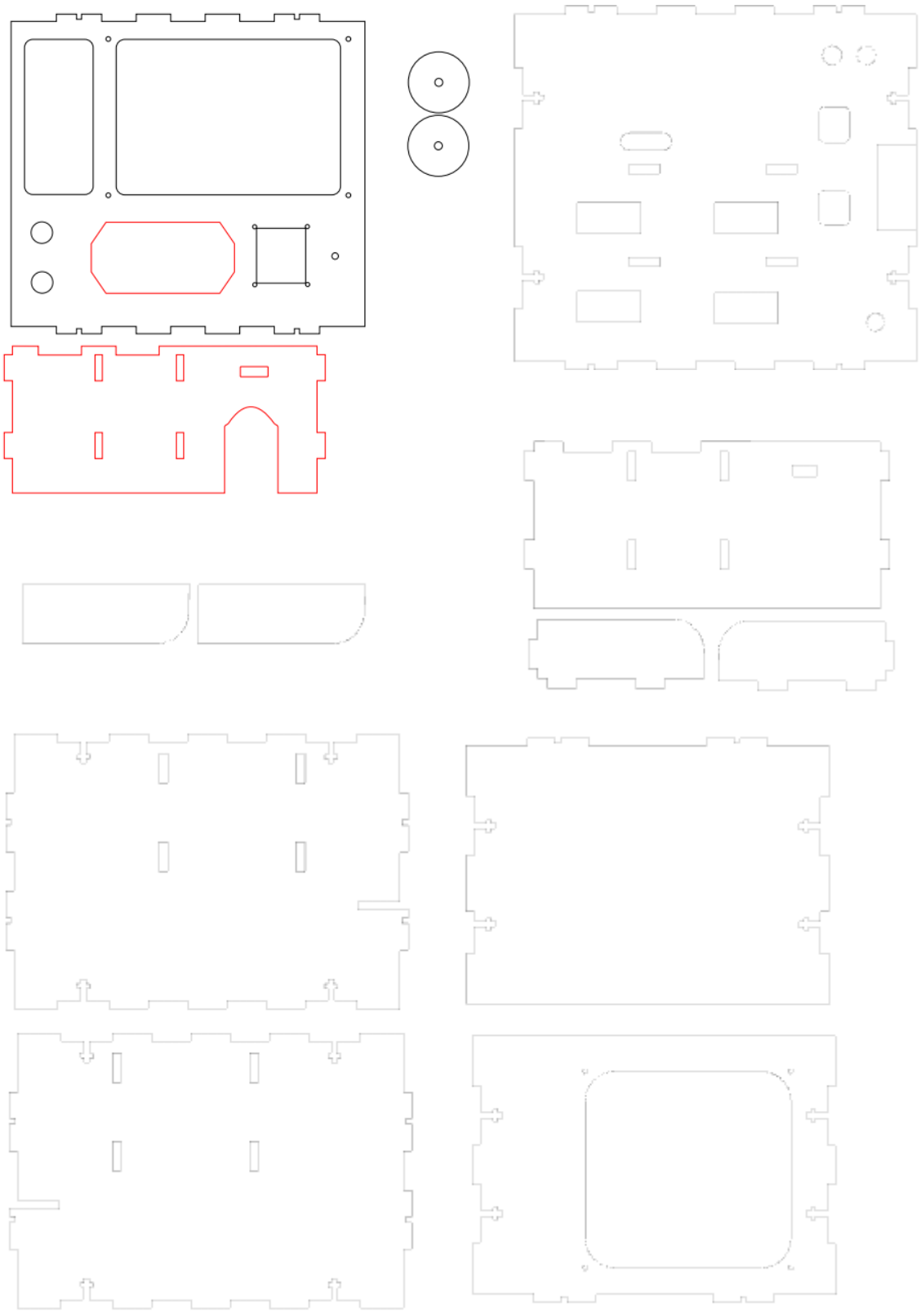
```
display.display();
}
```

Příloha II.

```
#include <Adafruit_NeoPixel.h>

#define PIN A1
#define NUM_LEDS 8
const int TlacitkoLED = 3;
Adafruit_NeoPixel rgb_leds(NUM_LEDS, PIN, NEO_GRB + NEO_KHZ800);

int ledState = 0; // 0 = vypnuto, 1 = bílá, 2 = modrá

//////////////////VOID SETUP////////////////////////
void setup() {
pinMode(TlacitkoLED, INPUT_PULLUP);
rgb_leds.begin();
delay(10);
}

//////////////////////////VOID LOOP NASTAVENI//////////////////////////
void loop() {
if ((digitalRead(TlacitkoLED) == LOW)) {
if (ledState == 0) {
// LED jsou vypnuté, nastaví se bílá barva
setColorLED(rgb_leds.Color(255, 255, 255));
ledState = 1;
} else if (ledState == 1) {
// LED jsou nastavené na bílou, nastaví se modrá barva
setColorLED(rgb_leds.Color(0, 0, 255));
ledState = 2;
} else {
// LED jsou nastavené na modrou, vypnou se
```

```
setColorLED(rgb_leds.Color(0, 0, 0));
ledState = 0;
}
delay(500); // oddálení pro snížení šumu
}
}

void setColorLED(uint32_t color) {
for (int led_number = 0; led_number < NUM_LEDS; led_number++) {
rgb_leds.setPixelColor(led_number, color);
}
rgb_leds.show();
}
```

Příloha III.



Příloha IV.