|  |  |  |
| --- | --- | --- |
|  | | |
| **Závěrečná studijní práce**  **dokumentace** | | |
| **Arduino - třídička** | | |
| Jan Wolf | | |
|  | | |
|  | |  |
| **Obor:** | 18-20-M/01 INFORMAČNÍ TECHNOLOGIE  se zaměřením na počítačové sítě a programování | |
| **Třída:**  **Školní rok:** | IT4  2021/2022 | |

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité   
informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové   
a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2021

*podpis autora práce*

**ANOTACE**

Cílem projektu bylo vytvořit třídičku barev s využitím RGB senzoru. Tento sensor funguje na principu zachycení a filtrace základních barev jakými jsou červená, zelená a modrá. Toto je realizováno za pomocí barevných filtrů, fotodiod a převodníku, které jsou součástí RGB senzoru TCS3200. Práce se zabývá testováním měřených hodnot senzoru, algoritmizací zápisu a měření barev a také realizací jednoduchého uživatelského menu s výpisem na LCD display.

Na projekt byl využit mikrokontrolér Arduino UNO s procesorem ATmega328P. Programová část je řešena v jazyce Wiring, který je odnož jazyka C++ a s využitím externích knihoven pro zjednodušení a přehlednost. Třídička dokáže zapsat barvu do dvojrozměrného pole a následně ji rozlišovat. Toto je realizováno uživatelským menu ovládaným tlačítky.

**Klíčová slova**

Arduino UNO, mikrokontroler, RGB, TCS3200, dvojrozměrné pole

OBSAH

[Úvod 4](#_Toc89967789)

[1 výroba a vývoj třídičky 5](#_Toc89967790)

[1.1 Počátky a testování 5](#_Toc89967791)

[1.2 Výroba krabičky 5](#_Toc89967792)

[1.3 Konečný vývoj 6](#_Toc89967793)

[2 Využité technologie 8](#_Toc89967794)

[2.1 Hardware 8](#_Toc89967795)

[2.1.1 Arduino UNO 8](#_Toc89967796)

[2.1.2 Detektor barev TCS3200 8](#_Toc89967797)

[2.1.3 LCD display s I2C převodníkem 9](#_Toc89967798)

[2.1.4 Tlačítkový modul 9](#_Toc89967799)

[2.1.5 Mikro servo SG90 10](#_Toc89967800)

[2.2 Software 10](#_Toc89967801)

[2.2.1 Programovací jazyk Wiring 10](#_Toc89967802)

[2.2.2 Visual Studio Code a plugin PlatformIO 10](#_Toc89967803)

[2.2.3 KiCad 5.1.10 10](#_Toc89967804)

[3 Způsoby řešení a použité postupy 11](#_Toc89967805)

[3.1 Využité součástky 11](#_Toc89967806)

[3.2 Schématické zapojení 11](#_Toc89967807)

[3.3 Zápis barev 12](#_Toc89967808)

[3.4 Rozlišování barev 12](#_Toc89967809)

[3.5 Knihovna CMBmenu.h 14](#_Toc89967810)

[4 Výsledky řešení 15](#_Toc89967811)

[4.1 Hardwarové a mechanické provedení 15](#_Toc89967812)

[4.2 Softwarové provedení 15](#_Toc89967813)

[Závěr 17](#_Toc89967814)

[Seznam použitýCH INFORMAČNÍCH ZDROJů 18](#_Toc89967815)

[Seznam příloh 19](#_Toc89967816)

Úvod

K výběru mé závěrečné práce mě určitě vedl můj zájem o elektrotechniku, hardware a programování mikrokontrolerů. V rámci mé práce se věnuji hlavně testování a programování barevného senzoru TCS3200 s využitím vývojové desky Arduino UNO. S Arduinem jsem měl již zkušenosti z hodin pana učitele Godovského.

Cílem bylo vytvořit program sloužící k uživatelskému přidávání barev a následnému třídění. Toto bude řízeno pomocí tlačítek a menu na displayi. Mimo jiné jsem se také věnoval mechanické realizaci pro automatický posun a roztřídění předmětů.

V průmyslu jsou barevné senzory hojně využívány například pro detekci a třídění podle barevné značky, kontrolu etiket, popisů nebo správného obsahu. Nalezneme je nejčastěji ve strojním průmyslu na montážních linkách a kontrolách kvality.

V mé dokumentaci se věnuji popisu vývoje mého projektu, využitým technologiím a mojí principiální realizací. Součástí je také obrázková dokumentace průběhu vývoje třídičky, co se týče vizuálního provedení, využitých komponentů a uživatelského menu.

# výroba a vývoj třídičky

## Počátky a testování

Začátky mého závěrečného projektu spočívaly ve výběru vhodného hardware, navrhnutí zapojení komponentů, dosavadní představu funkčnosti a postupy testování součástek.

Jako mikrokontrolér jsem zvolil vývojovou desku Arduino UNO, se kterou jsem měl zkušenosti z hodin pana učitele Godovského. Při výběru sensoru pro zjišťování barev mě přesvědčil sensor TCS3200 kvůli jeho dobré dostupnosti a využití v podobných aplikací.

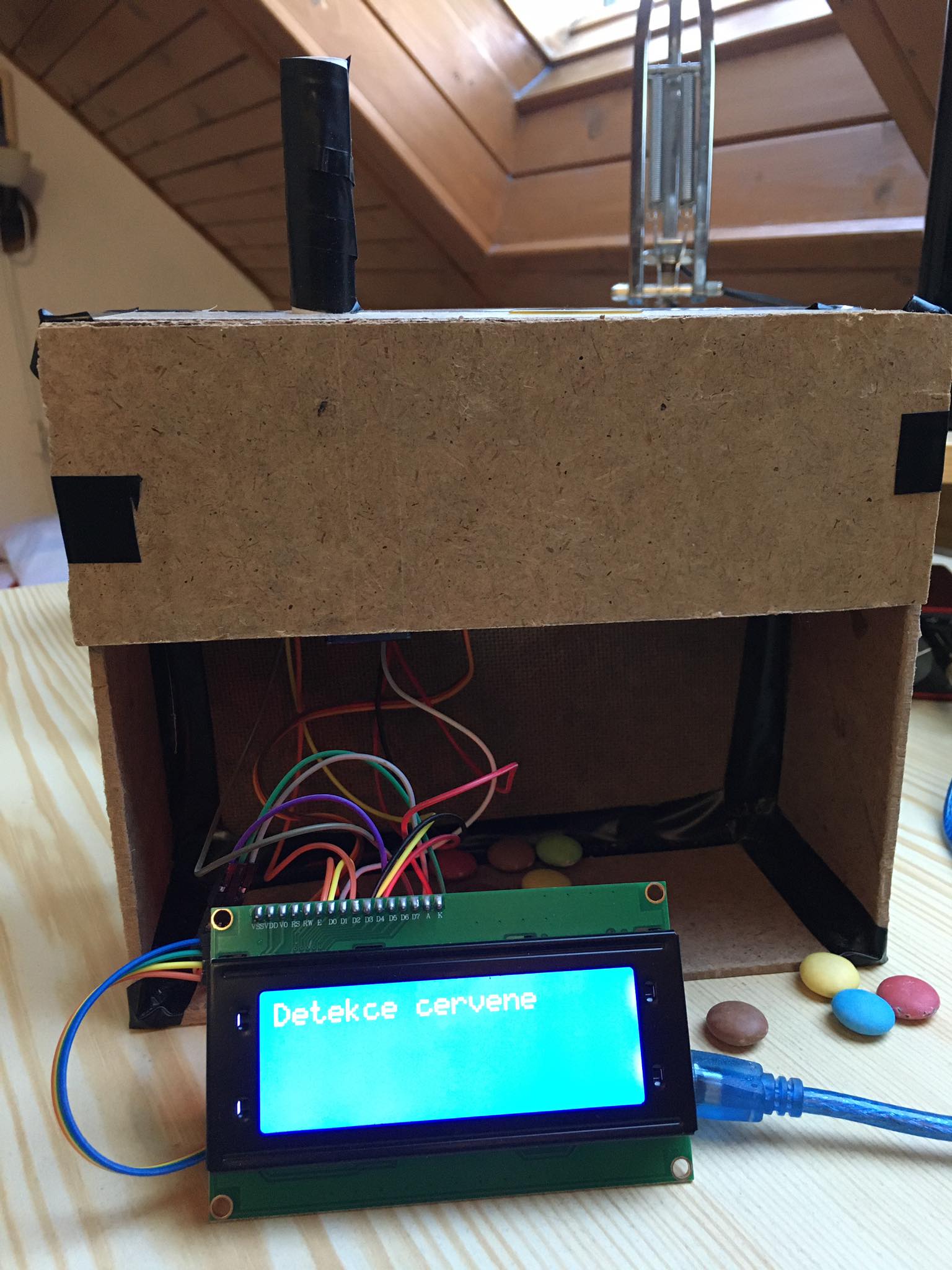
Moje začátky testování probíhaly v programu VS Code v kombinaci s pluginem PlatformIO, ve kterém jsem programoval celou aplikaci. Začínal jsem krok po kroku a přidával další součástky. Prvním byl ale test již zmíněného barevného sensoru. Funkčnost jsem ověřoval jednoduchým zápisem jednotlivých frekvencí RGB do pole a následným výpisem na sériovou linku. Tento proces jsem zkoušel na barevných papírech a následně ručně zapisoval hodnoty do dvojrozměrného pole.

Zde jsem ale narazil na první velký problém a tím byl zápis složek barev vůči okolnímu světlu, který se mi ovšem později podařilo vyřešit. Jednoduše řečeno zapsaná barva měla jiné RGB složky při silném denním světle než například v šeru, protože zároveň na snímač dopadalo i okolní světlo. Sensor sice disponuje čtyřmi LE diodami, které mají za úkol daný předmět nasvítit. Toto nasvícení ovšem „nepřebije“ okolní světlo. Tím pádem mě napadla myšlenka dynamického přidávání barev.

## Výroba krabičky

Jako další krok mě čekala výroba samotné krabičky, kde budou umístěny součástky. Třídičku jsem od počátku chtěl vyrobit pro účel třídění lentilek či jiných malých předmětů. Na projektu jsem začal pracovat již o velkých prázdninách neměl jsem ale možnost 3D tisku krabičky. Abych mohl na projektu pokračovat rozhodnul jsme se krabičku vyrobit po svém.

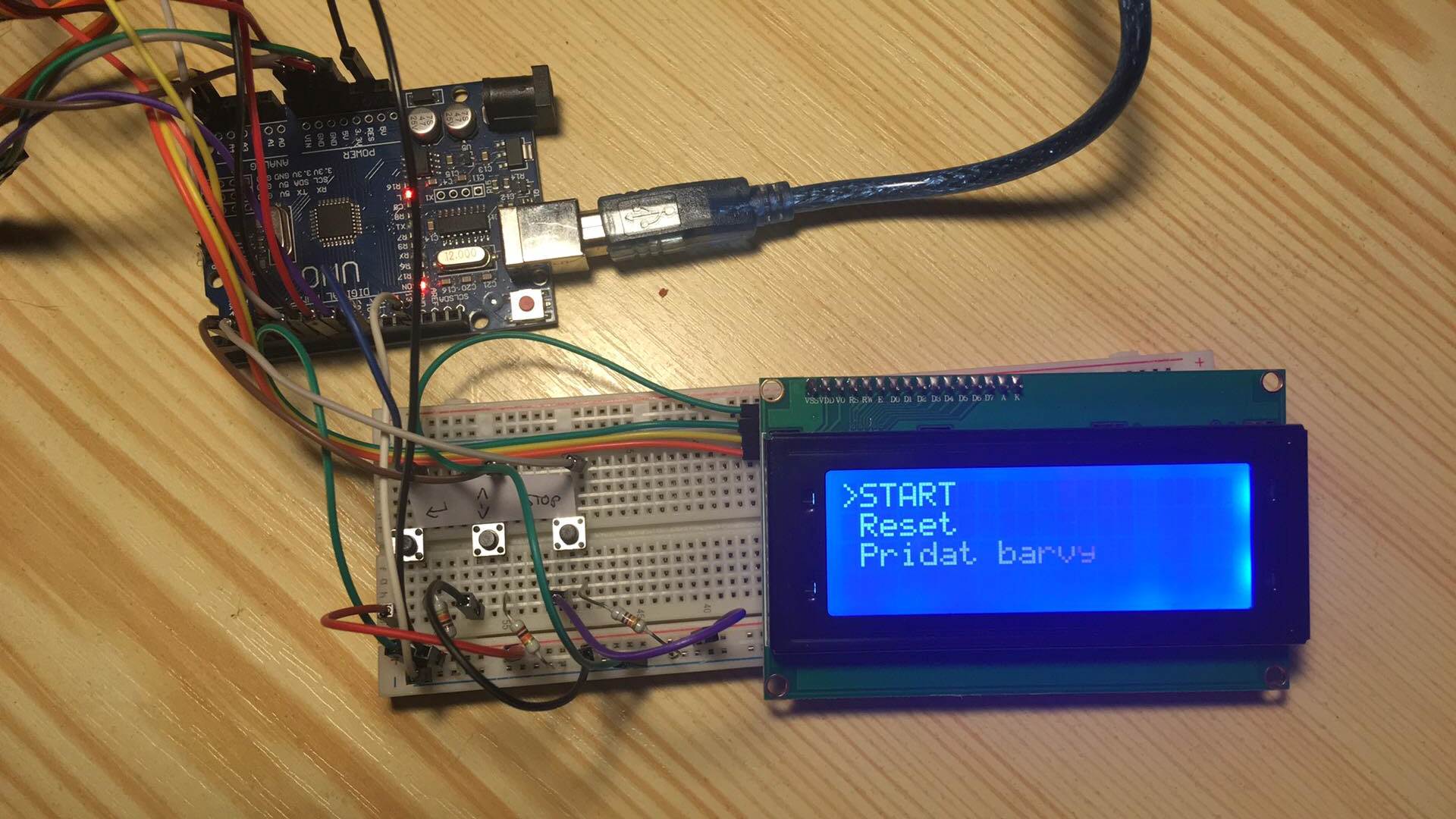
Věděl jsem, že je hodně důležité, aby lentilky padaly na sensor pokaždé ve stejné poloze, z důvodu správného snímání barvy. Toto se pro mě stalo velkou výzvou, jelikož jsem zároveň chtěl, aby se lentilky automaticky pomocí servo motoru posouvaly. Toto se tedy stalo mým druhým velkým problémem, protože kvůli ručně vyráběných mechanických prvků lentilky nepadaly přesně na místo snímání sensoru nebo se při manipulaci zasekávaly. Snímání barev se proto stalo méně spolehlivé.



Z těchto důvodů a také na doporučení pana učitele Grussmanna jsem přidal další funkci, a to *Přidat barvu*, která umožnovala uživatelsky přidat jakoukoliv barvu. Tím jsem částečně vyřešil problém s okolním světlem, protože barva se nasnímala a uložila vůči okolnímu svitu.

## Konečný vývoj

Po vytvoření krabičky jsem se vrhnul na programování tlačítek, displaye a menu. Menu jsem poprvé vytvořil bez pomocí jakýchkoli knihoven tudíž docházelo k opakování a nepřehlednosti kódu. Toto prozatímní menu jsem vypisoval na display a ovládal pomocí tlačítek, připojených na nepájivém kontaktním poli.



Následně jsem napájel tlačítka a rezistory na plošný spoj a zabudoval společně s diplayem do předního krytu krabičky. Tím jsem dokončil výrobu krabičky.

Na radu pana učitele Grussmanna jsem přeprogramoval menu pomocí knihovny *CMBMenu*, která umožnuje více úrovňového menu a přehlednější kód. Pro větší spolehlivost tlačítek a dalších funkcí jsem upravil tlačítka přes knihovnu *OneButton*, která obsahuje funkce jako *doublepress* nebo *longpress*.

Jako jedna z posledních funkcí byla čítač jednotlivých barev zobrazený na displayi při třídění a funkce *Reset* pro resetování čítače.

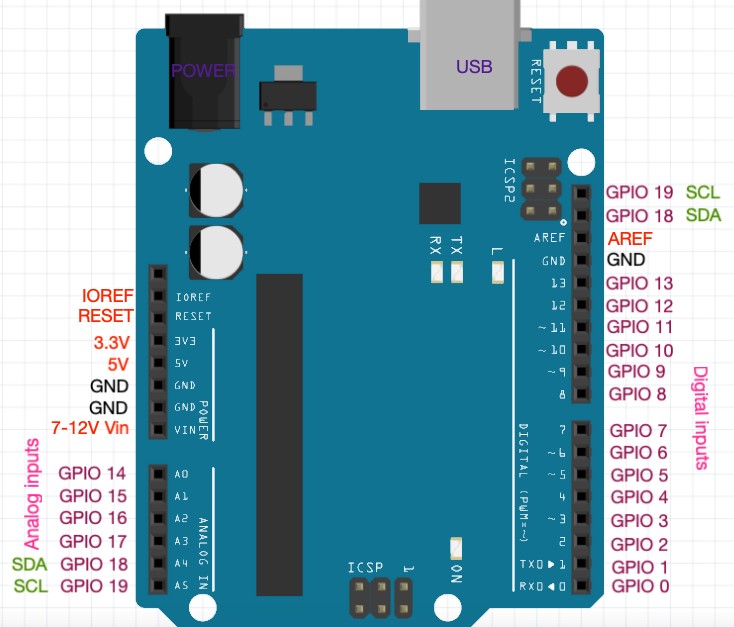
Projekt byl tímto téměř hotov. Bohužel v průběhu vývoje jsem narazil na další problém a tím byla malá citlivost sensoru. Barvy, které jsou si svými složkami RGB podobné, nedokáže sensor dostatečně citlivě nasnímat a následně rozlišit. S tímto jsou spojeny i výše zmíněné problémy s mechanickými prvky. Pokud lentilky spadly na sensor například obráceně či kousek vedle, tak je sensor špatně nasnímal. Kvůli těmto problémům jsem musel omezit počet přidávaných barev. I přes to ale třídička funguje mnohdy nespolehlivě.

# Využité technologie

## Hardware

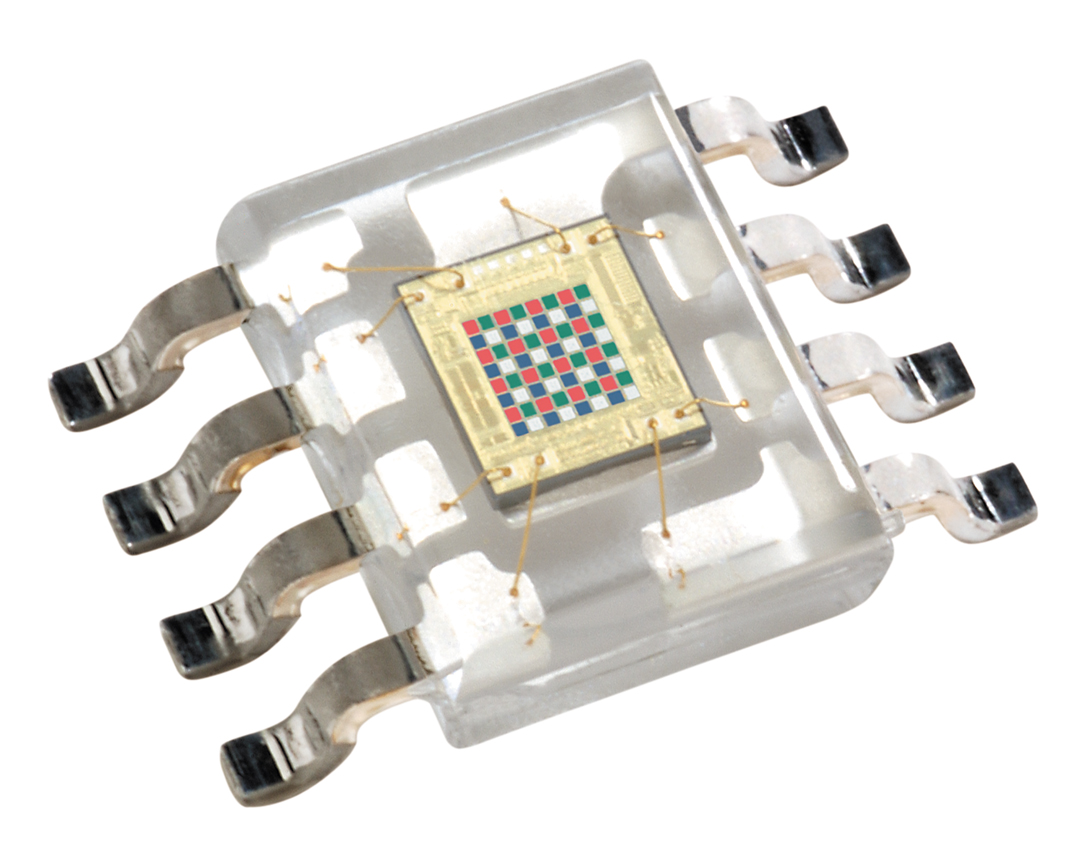
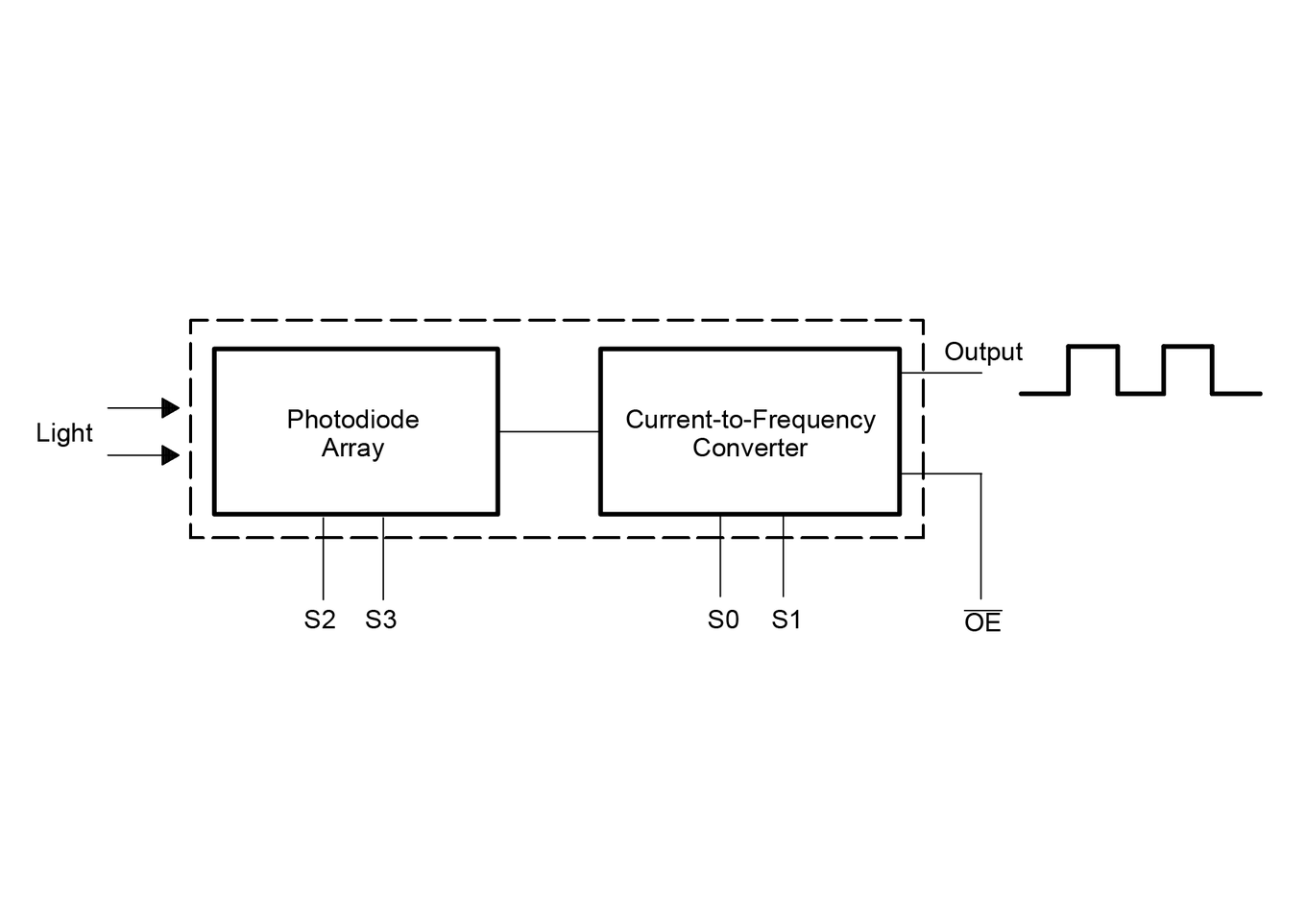
### Arduino UNO

Srdce projektu tvoří vývojová deska Arduino UNO stojící na procesoru ATmega328P pracující na frekvenci 16MHz a disponuje flash pamětí 32KB. Tato deska nabízí 14 digitálních vstupně výstupních pinů a 6 vstupních analogových pinů. Arduino disponuje USB konektorem pro propojení s počítačem, *vin* konektor pro připojení AC/DC zdroje, nebo konektor pro 9 V baterii. Jednočip pracuje s napětím 5 V ale umožnuje jako výstupní napětí také 3,3 V.



### Detektor barev TCS3200

Tento senzor funguje na principu přeměny intenzity světla odraženého od barevného předmětu na frekvenci jednotlivých složek RGB. Provozní napětí je 3 V–5 V. Doporučuju využít 5 V, protože LE diody poté vyzařují více světla, čili detekce se stává spolehlivějším. TCS3200 disponuje čipem který obsahuje pole fotodiod s filtrem pro červenou, zelenou, modrou a žádnou barvu. Tyto diody spolu s převodníkem zajišťují přeměnu intenzity světla na elektrický proud.

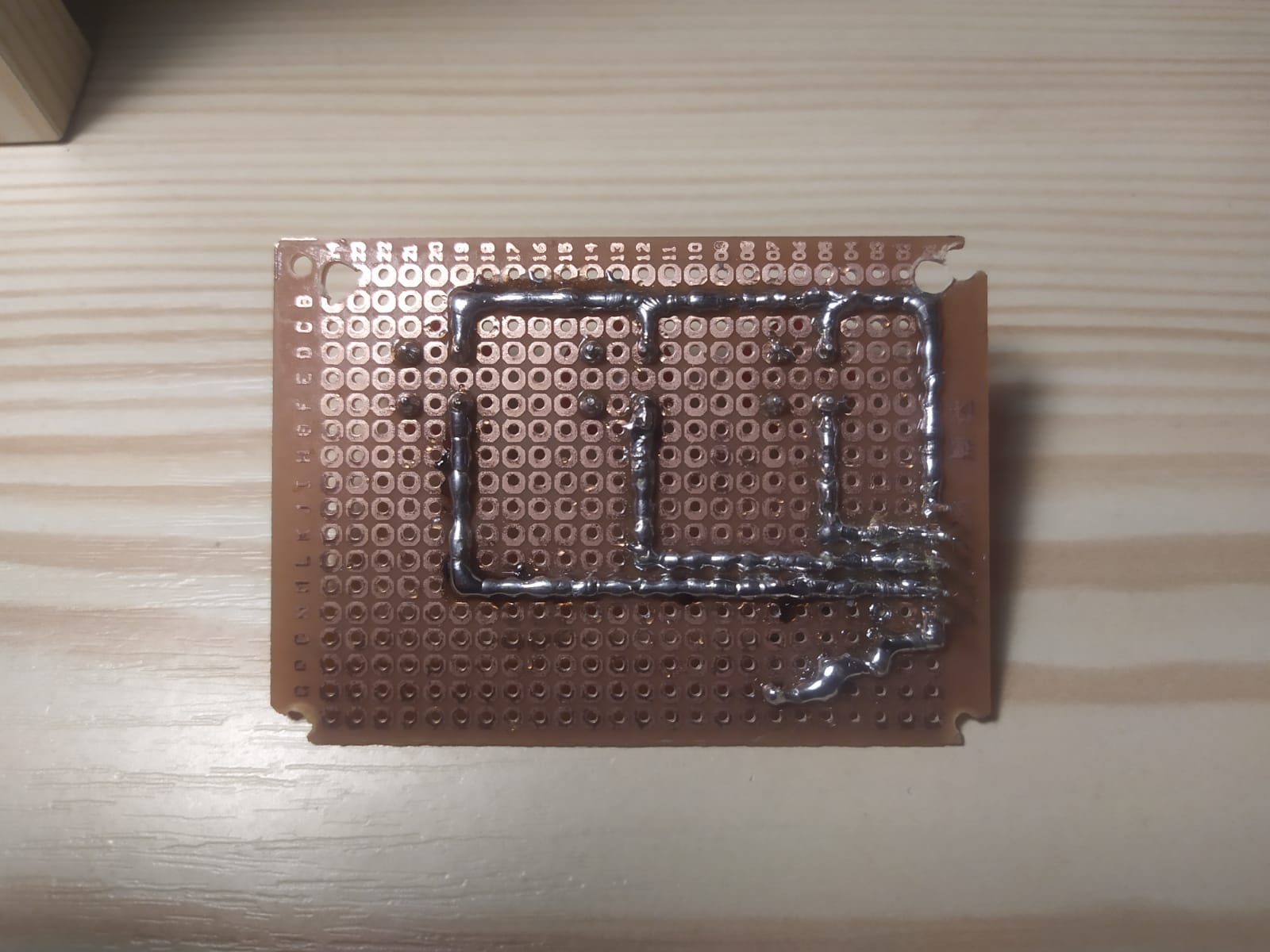
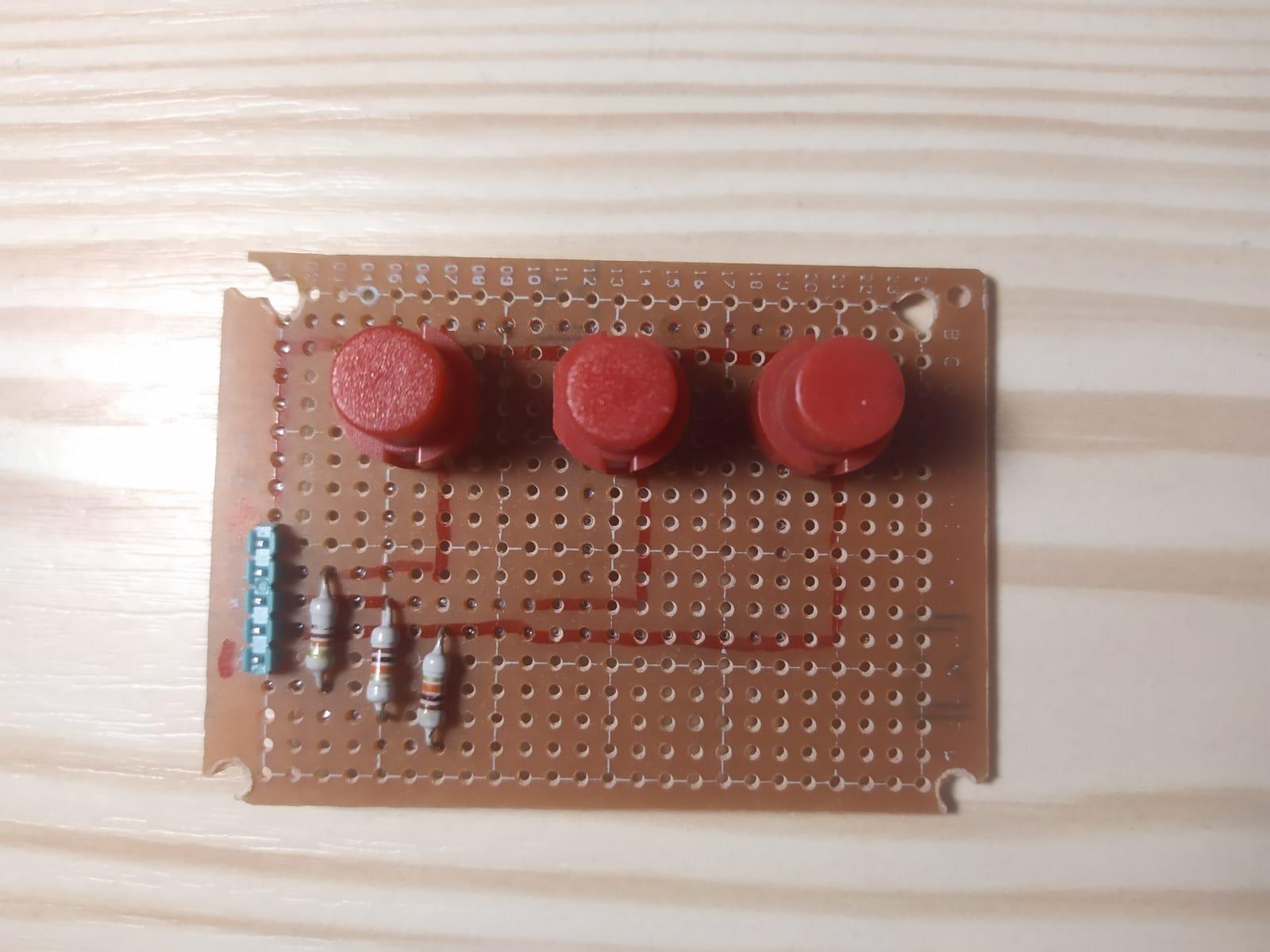


### LCD display s I2C převodníkem

Tento modře podsvícený display umožňuje vypisování znaků ve 20 sloupcích a 4 řadách. Kvůli jednoduchosti zapojení jsem se rozhodl využit I2C převodník čili k zapojení jsem využil pouze 2 vodiče SDA a SCL a samozřejmě vodiče pro napájení. Intenzita podsvícení se reguluje trimrem na I2C převodníku. Ke komunikaci jsem využil knihovnu pro jednoduché ovládání *LiquidCrystal\_I2C*.

### Tlačítkový modul

Pro ovládání menu v diplayi jsem využil trojice tlačítek a rezistorů. Tyto součástky jsem kvůli efektivnějšímu provedení následně napájel na univerzální plošný spoj a zabudoval do krabičky třídičky. Tlačítka jsou uzemněné 10 KΩ rezistorem, mají 3 výstupní konektory pro čtení hodnot z tlačítek a společnou anodu a katodu.



### Mikro servo SG90

Pro posuv předmětů ze zásobníku a pro otáčení jsem využil dvou servo motorů. Provozní napětí je 5 V–6 V.

## Software

### Programovací jazyk Wiring

Celý program je napsaný v jazyce Wiring, který byl vyvinut pro programování mikrokontroléru a nejčastěji je spojen s vývojovým prostředím Arduino IDE. Wiring je odnož jazyků C a C++ a využívá knihovnu *Arduino.h* pro funkce pro práci s mikrokontrolerem.

### Visual Studio Code a plugin PlatformIO

Jako integrované vývojové prostředí jsem využil program Visual Studio Code který umožňuje nainstalování mnoha pluginů pro rychlejší a efektivnější vývoj jako například tzv. *našeptávač*, nebo profesionální plugin pro programovaní aplikací pro mikrokontrolery PlatformIO, který jsem využil.

### KiCad 5.1.10

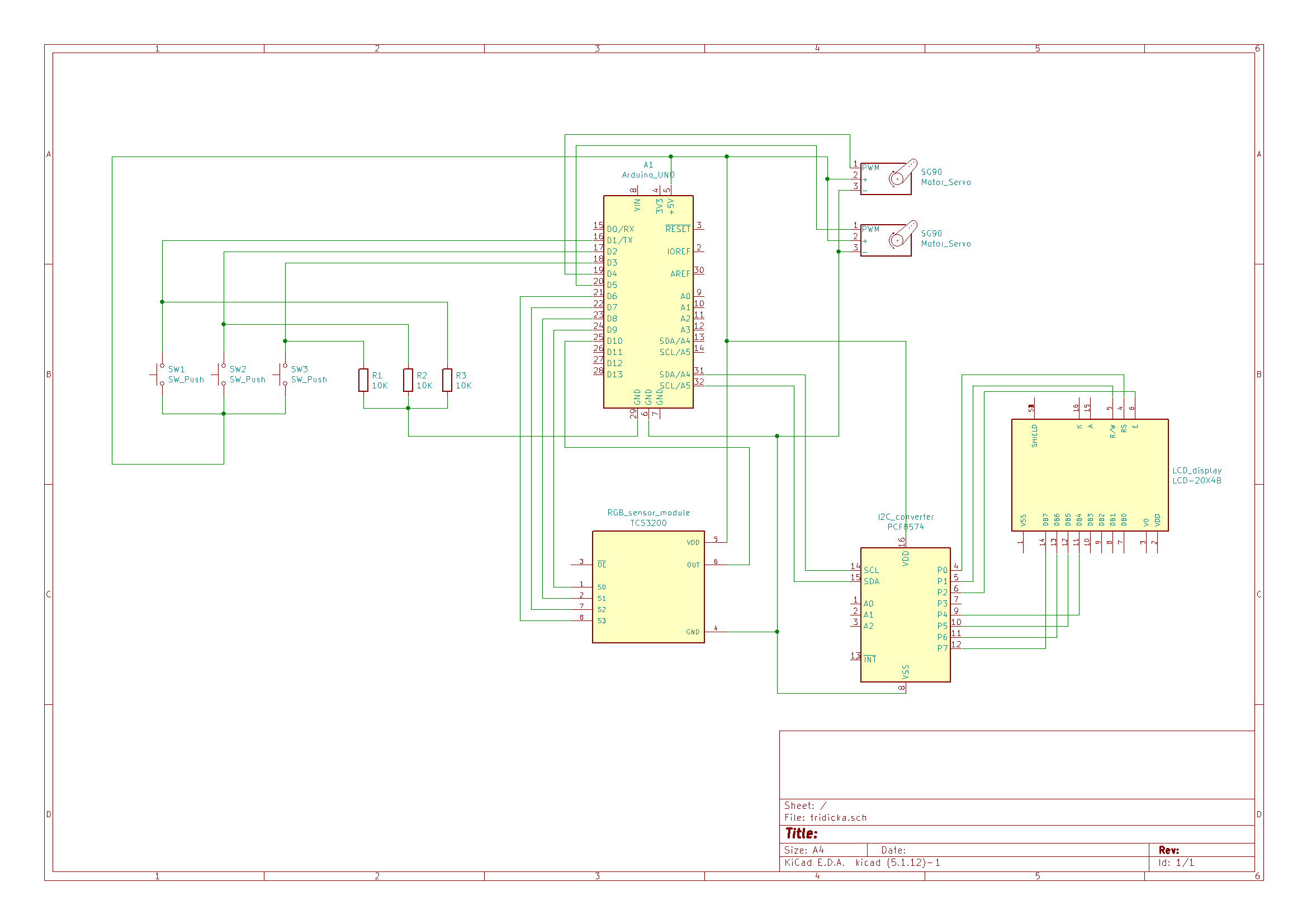
Pro nákres schématu zapojení jsem si vybral oper-source software KiCad, který umožňuje projektování plošných spojů a disponuje obrovskou databází nejrůznějších elektrotechnických součástek.

# Způsoby řešení a použité postupy

## Využité součástky

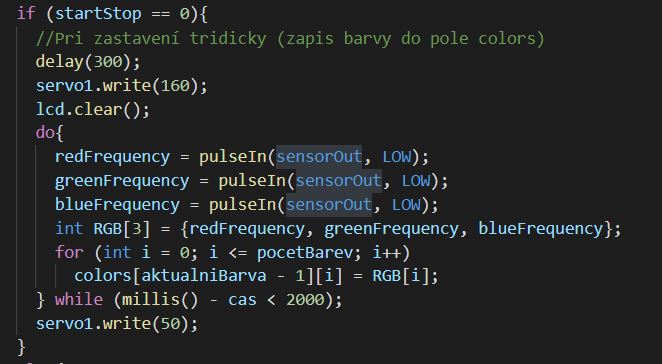
* Mikrokontroler Arduino UNO
* RGB senzor TCS3200
* LCD display 20x4 znaků
* I2C převodník PCF8574
* 2x mikro servo SG90
* 3x tlačítko
* 3x Rezistor 10KΩ
* Univerzální plošný spoj
* Malé nepájivé kontaktní pole

## Schématické zapojení



## Zápis barev

Hlavní částí projektu byla práce se sensorem barev. RGB složky byly ukládány do statického dvojrozměrného pole. V první části vývoje jsem vypsané složky na sériovou linku ručně zapisoval do tohoto pole. Toto se brzy projevilo jako nepraktické. Později jsem toto zapisování udělal automaticky pomocí stisknutí tlačítka.

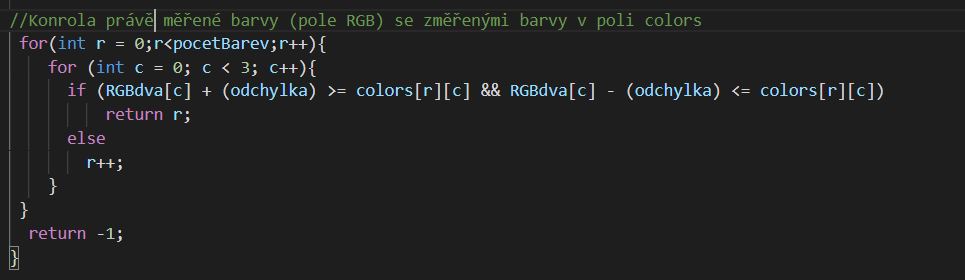


V této části kódu je jednoduchá podmínka kontrolující, zda je třídička ve stavu *Stop* (hodnota 0). Dále proběhne načtení frekvencí ze sensoru pomocí funkce **pulseIn** a uložení těchto proměnných do pole *RGB*. Následně v cyklu **for** se tato trojice hodnot uloží do globálního dvojrozměrného pole *colors*. Řádky pole se mění podle globální proměnné *akutalniBarva*, která se inkrementuje při opětovném volání funkce.

Pole colors jsem kontroloval pomocí pomocné funkce *vypisPole()* na sériové lince.

## Rozlišování barev

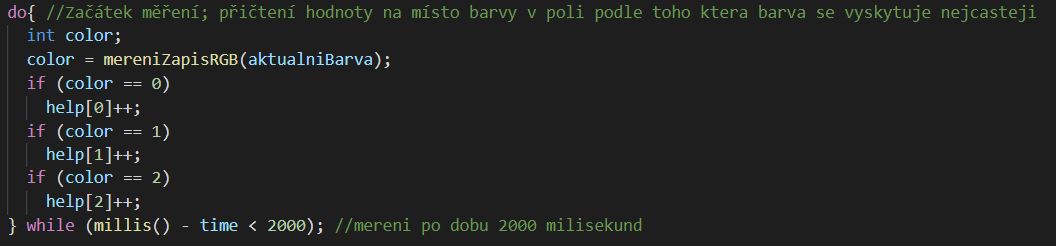
Jakmile byly barvy zapsané do pole colors následovalo jejich rozlišování. V začátcích projektu jsi jednoduše změřil a zapsal složky RGB například pro červenou barvu. Dále v programu se tyto složky kontrolovali a vypisovali na display „Detekce červené barvy“. Toto statické zapisování se ukázalo velmi nespolehlivé, hlavně kvůli problémům s okolním světlem a také samozřejmě nedynamičností. Toto se dělo díky jinému nasvícení například ve tmě či šeru se a barvy se tím pádem rozlišovali špatně (jak už bylo zmíněno výše).



Nyní program za pomocí dvou cyklů a jedné podmínky kontroluje, zda aktuální RGB změřené ze sensoru rovná některému ze zapsaných v poli. Zde jsem musel brát v potaz odchylku při měření. Ovšem kvůli malé citlivosti sensoru tato odchylka „zasahovala“ do jiných barev. Pokud podmínka platí, funkce vrátí číslo řádku, což odpovídá změřené barvě. Pokud ne, inkrementuje se řádek a cyklus kontroluje, zda se nejedná o další barvu.

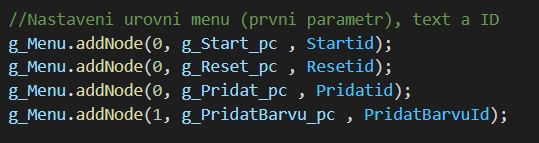
Vzhledem k zvýšení spolehlivosti jsem toto měření prováděl po dobu dvou sekund. Návratová hodnota funkce *mereniZapisRGB()* (viz ukázky kódu výše) představuje index pole právě změřené barvy. Tato barva se ale může lišit od té, která se zde skutečně vyskytuje. Proto se tato hodnota zapisuje do pole *help*. Prvek pole *help* s nevyšší hodnotou se zde vyskytuje nejčastěji. Jestliže například první index pole *help* obsahuje číslo kolem tisíce a hodnoty na jiných indexech jsou menší, jako výsledná barva se vybere první zapsaná.

Tento princip měření je přesnější, než kdyby se se RGB frekvence ze sensoru kontrolovali pouze pomocí jedné podmínky. Tento sensor bohužel na toto není dostatečně citlivý\*.



## Knihovna CMBmenu.h

Na doporučení pana učitele Grussmanna jsem naprogramoval jednoduché víceúrovňové uživatelské menu ovládaného pomocí tlačítek. Využil jsem knihovnu *CMBmenu.h*, čímž se zvýšila přehlednost a jednoduchost kódu. Prvním krokem při tvorbě menu je přidání tzv. **node**, čili uzle. K tomuto uzlu je přiřazeno číslo úrovně menu, text, který chceme vypisovat a jedinečné ID.



Dále jednoduše pomocí tlačítek vyvoláváme funkce *left(), right(), enter(), exit()*, pro pohybování v menu. Musíme před tím ještě sestavit a vypsat menu pomocí funkcí *buildMenu()* a *printMenu()*.

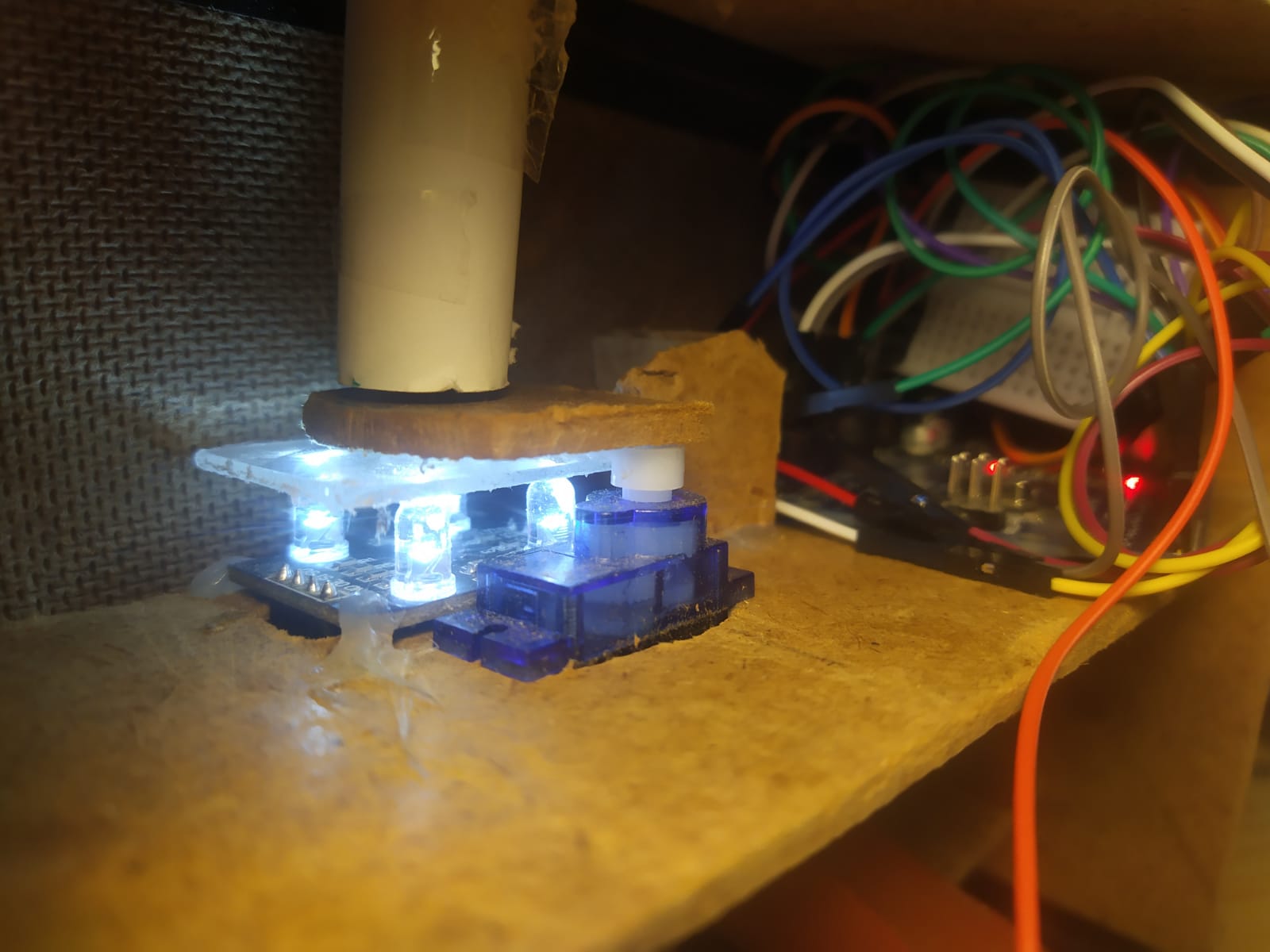


Při stisku tlačítka reprezentující funkci *enter()* se kontroluje ID, podle kterého je vyvolána určitá funkce.

# Výsledky řešení

## Hardwarové a mechanické provedení

Výsledné zařízení, co se týče vzhledu a provedení, odpovídá mým představám, které jsem měl na začátku práce. Třídička má vestavěný LCD display, tlačítkový modul, zásobník na lentilky a oba servo motory tak, jak jsem si představoval.

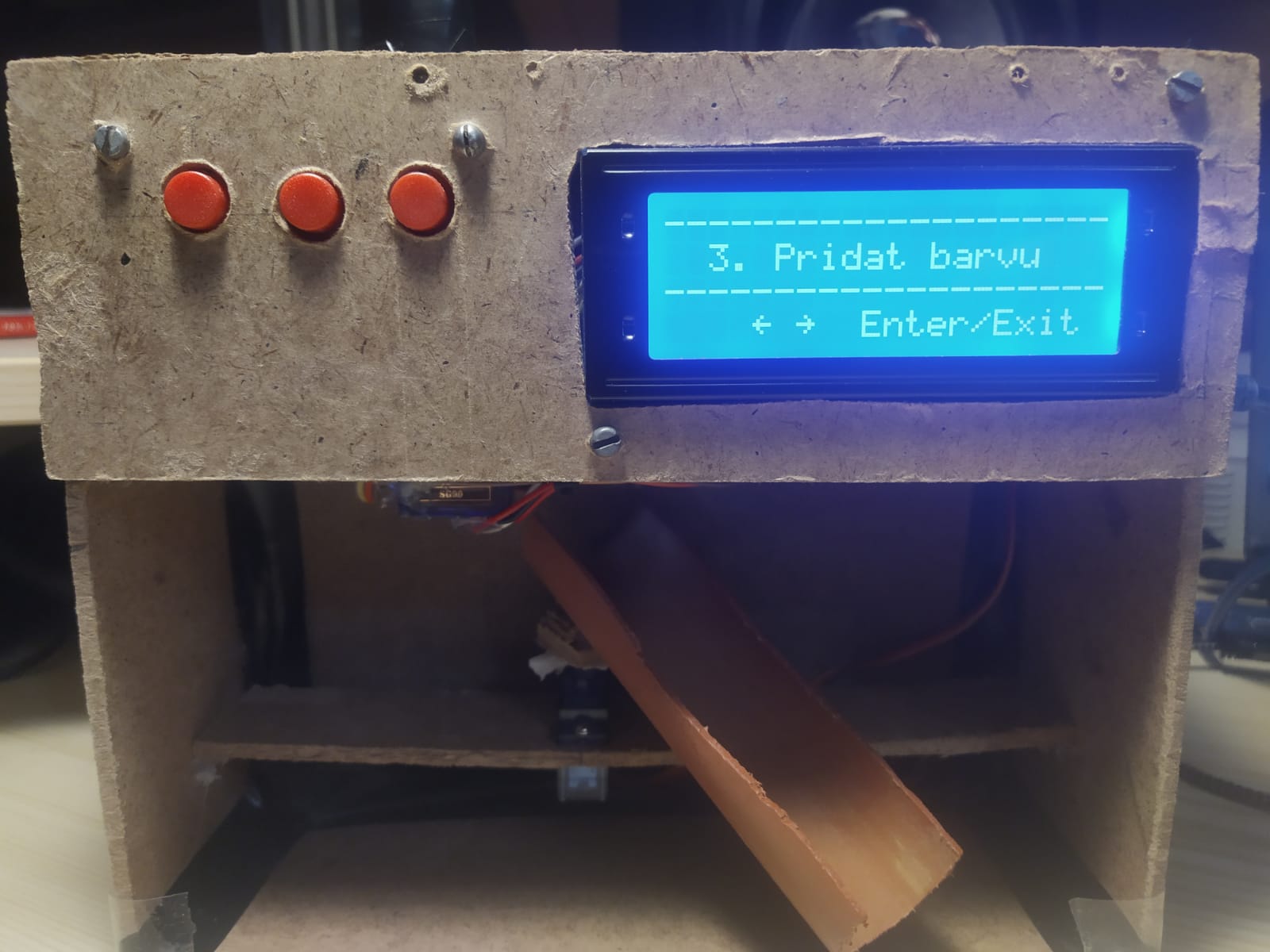
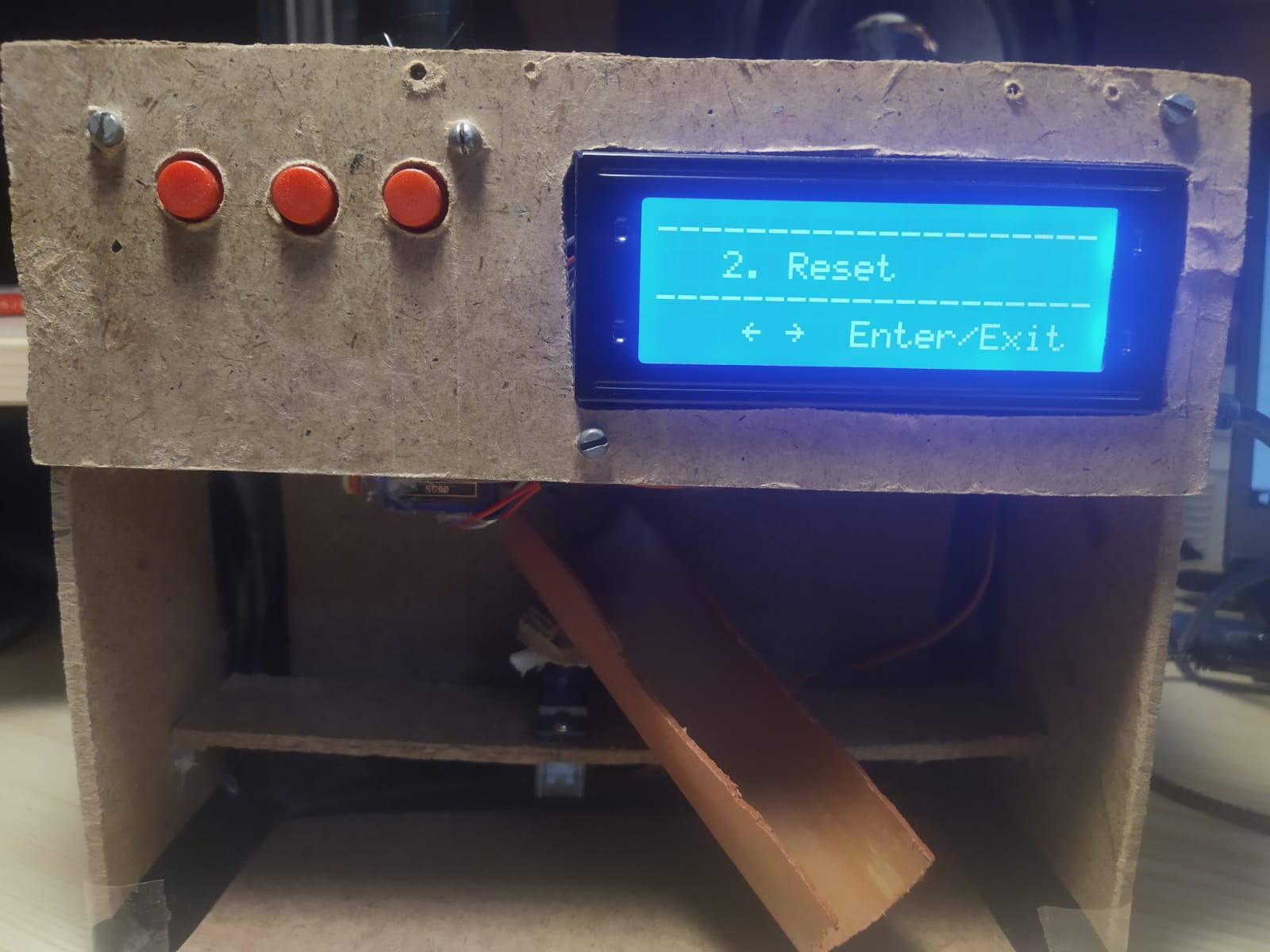


Na druhou stranu je zde pár věcí, které bych udělal jinak nebo lépe, kdybych je byl věděl na začátku mé práce na projektu. Jedna z nich je, že bych celou třídičku vytvořil na třídění větších předmětů, než jsou lentilky. Na snímač by se totiž nasnímalo více barevného světla, které se odráží od předmětu, a to z důvodu jeho vetší plochy. Tím pádem by vznikl větší kontrast mezi barevnými předměty. Jako další věc bych upravil mechanické prvky, vymodeloval a vytisknul je na 3D tiskárně, tak aby se předmět dostal na snímač vždy do stejného místa a ten pokaždé správně přečetl barvu. Vhodné by bylo také napájení komponentů na plošný spoj a změna vývojové desky na Arduino Nano, aby došlo k redukci vodičů.

Vzhledem k výše uvedeným problémům, které se týkají mechanických částí a velikosti předmětů, se třídička chová často nespolehlivě a nepřesně.

## Softwarové provedení

Co se týče funkčnosti samotného programu, tak za pomocí ošetřených tlačítek včetně funkcí *doublepress* a *longpress* je možno se pohybovat v menu. Program dokáže zapsat barvu do pole a následně po programátorem určený čas měřit a rozlišovat dané barvy. Čítač barev zobrazovaný na displayi a pole zapsaných barev se dá vynulovat funkcí *Reset*.



# **Závěr**

Úkolem bylo vytvořit třídičku s využitím RGB senzoru TCS3200 s možností dynamického přidávání vlastních barev. Zadaný cíl byl splněn včetně výroby krabičky, osazení komponentů, napájení tlačítek na plošný spoj a automatickému posuvu lentilek.

Je zde určitě pár věci, které by vyžadovali vylepšení či jinou realizaci, jak jsem již zmínil v předchozích kapitolách. Jednou z nich je realizace třídičky na vetší předměty, než jsou lentilky a vytisknutí potřebných mechanických prvků na 3D tiskárně. Například vytvoření dopravníkového pásu, po kterém by se předměty posouvaly a třídili, jak se to realizováno v průmyslu. Jako další vylepšením by mohlo být navrhnutí PCB desky plošného spoje pro redukci vodičů. Z důvodu malé citlivosti barevného sensoru by bylo dobré uvažovat o jeho výměně za přesnější. Z těchto důvodů se jedná spíše o amatérský projekt a jeho realizaci, tudíž by pravděpodobně nenašel své místo v praxi.

Seznam použitýCH INFORMAČNÍCH ZDROJů

[1] BOHMAN, Ludvík. Zákon o pojistné smlouvě. Praha: Linde Praha a. s., 2004. 381 s. ISBN80-7201-504-4

[2] DUCHÁČKOVÁ, Eva. Principy pojištění a pojišťovnictví. 3. aktualizované vydání. Praha: Ekopress 2009. 224 s. ISBN 978-80-86929-51-4

[3] KUBALA, Petr. Planetární dvojcata - Věda a technika (Český rozhlas) [online].   
Č. 2000-2008, poslední revize 19. 3. 2008 [cit. 2008-03-20].  
<http://www.rozhlas.cz/veda/vesmir/\_zprava/435849>.

[4] KULDOVÁ, O., FLEISCHMANNOVÁ, E. Metodická příručka k technice administrativy a obchodní korespondence. 1.vyd. Praha: Fortuna 1998. 111 s.   
ISBN 80-7168-574-7. Kapitola 6, Metody nácviku psaní hmatovou metodou,   
s. 28-29.

[5] VLACH, J. JE Temelín a zásobování teplem. Energetika, 2001, roč. 51, č. 3, s. 84 -85. ISSN 0375-8842.

<https://www.arduino.cc/reference/en/language/functions/advanced-io/pulsein/>

<https://create.arduino.cc/projecthub/SurtrTech/color-detection-using-tcs3200-230-84a663>

<https://github.com/mchlbrnhrd/mbLib>

<https://github.com/mathertel/OneButton>

https://www.hwkitchen.cz/navody-hwkitchen/prace-s-lcd-je-jednoducha-arduino-navody/

Seznam příloh

č. 1 Fotodokumentace

**Příloha č. 1: Fotodokumentace**

