Systém monitorování a řízení bezpečnostních procesů

Anton Džima

Bakalářská práce
2024

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav bezpečnostního inženýrství

Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Anton Džima**
Osobní číslo:	**A21215**
Studijní program:	**B1032A020001 Bezpečnostní technologie, systémy a management**
Forma studia:	**Prezenční**
Téma práce:	**Systém monitorování a řízení bezpečnostních procesů**
Téma práce anglicky:	**Monitoring and Control System of Security Processes**

Zásady pro vypracování

1. Vypracujte literární rešerši současného stavu v oblasti hardwarové signalizace.
2. Navrhněte prototyp hardwarové signalizace pro dveřní systémy.
3. Realizujte a ověřte funkčnost prototypu.
4. Vytvořte aplikaci obsahující půdorys objektu, měření časového intervalu otevření dveří a trasování průchodů objektem.
5. Ověřte funkčnost propojení hardwarové a softwarové části práce.

Forma zpracování bakalářské práce: **tištěná/elektronická**
Jazyk zpracování: **Slovenština**

Seznam doporučené literatury:

1. KAMENÍK, Jiří a BRABEC, František. *Komerční bezpečnost*. 2. vydání. Praha: Wolters Kluwer, 2019. ISBN 978-80-7598--303-9.
2. Projektování integrovaných systémů. Druhé vydání. Online, Skriptá, vedoucí Valouch, Jan. Zlín: Univerzita Tomáše Bati, Fakulta aplikované informatiky, 2015. Dostupné z: https://digilib.k.utb.cz/handle/10563/18616. [cit. 2023-11--13].
3. LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management. 1. vydání. Zlín: Radim Bačuvčík – VeRBuM, 2013. ISBN 978-80-87500-35-4.
4. ČSN EN 50131: Poplachové systémy – Poplachové zabezpečovací a tísňové systémy. Praha: Český normalizační institut, 2015.
5. LUNA, José Ignacio Vega; SÁNCHEZ-RANGEL, Francisco Javier; COSME-ACEVES, José Francisco. Monitoring System for Doors and Windows of a Data Center with IoT. *Ingenius*, 2019, 22: 72.

Vedoucí bakalářské práce: **Ing. Stanislav Kovář, PhD.**
Ústav bezpečnostního inženýrství

Datum zadání bakalářské práce: **8. prosince 2023**
Termín odevzdání bakalářské práce: **28. května 2024**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN
L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

Ing. Jan Valouch, Ph.D. v.r.
ředitel ústavu

Ve Zlíně dne 8. prosince 2023

Jméno, příjmení: Anton Džima

Název bakalářské práce: Systém monitorování a řízení bezpečnostních procesů

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

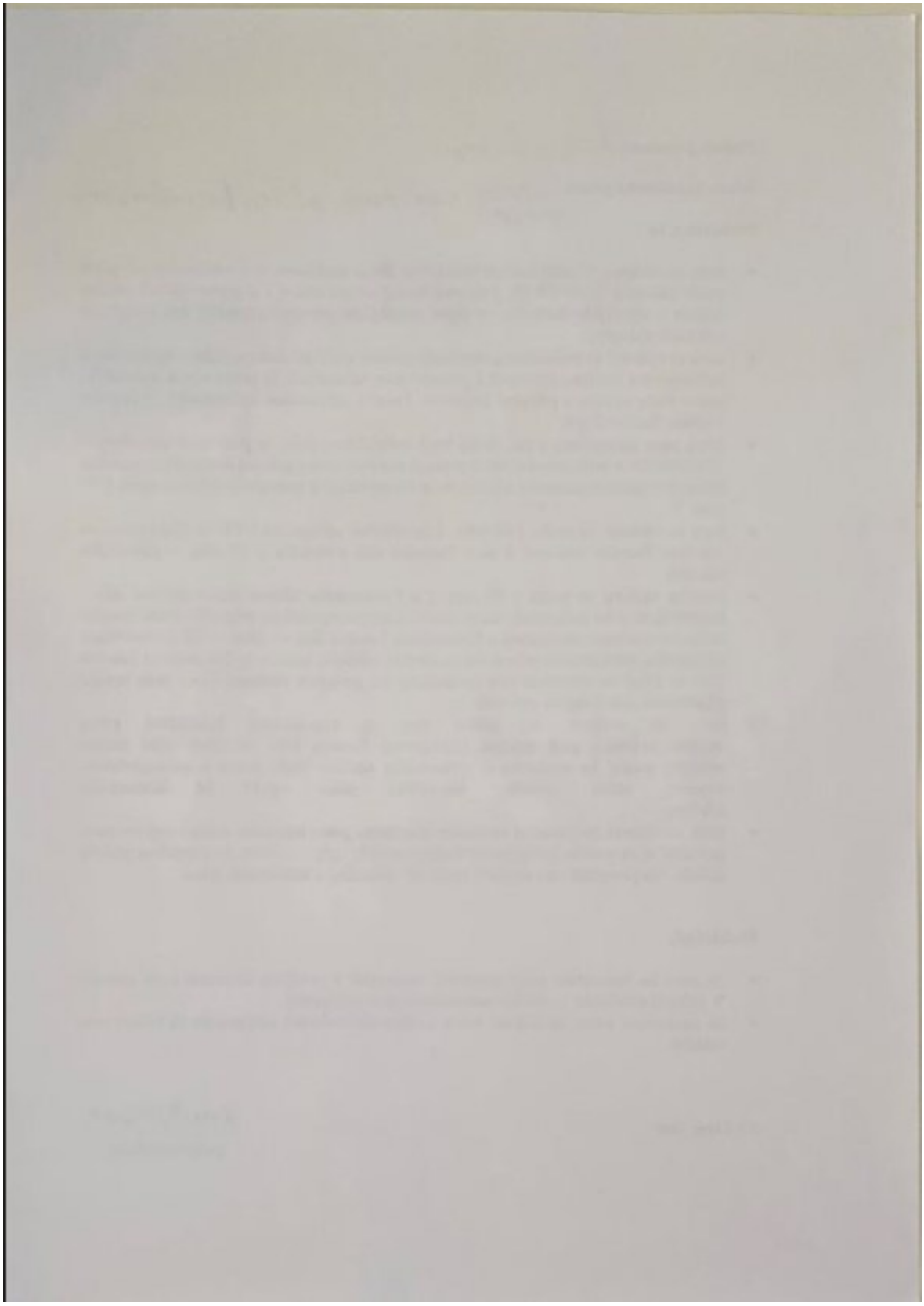
Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne

Anton Džima v.r.

podpis studenta



ABSTRAKT

Táto bakalárska práca sa zameriava na systém monitorovania a riadenia bezpečnostných procesov s dôrazom na plášťovú ochranu v komerčnom prostredí. Skúma nedostatky súčasných systémov a navrhuje možné zlepšenia. Práca preskúma relevantnú literatúru, analyzuje možnosti zabezpečenia dverí a iných vstupných bodov, a skúma vzťah medzi mechanickými zabezpečovacími systémami a poplachovými zabezpečovacími a tiesňovými systémami v kontexte moderných technológií. Zvláštny dôraz je kladený na využitie technológie „Internet of Things“ (IoT) v komerčnej bezpečnosti a hardvérovú signalizáciu.

Kľúčové slová: systém monitorovania, riadenie bezpečnostných procesov, komerčná bezpečnosť, mechanické zabezpečovacie systémy, poplachové zabezpečovacie systémy, Internet of Things, hardvérová signalizácia, plášťová ochrana.

ABSTRACT

This bachelor's thesis focuses on the system of monitoring and controlling security processes, emphasizing perimeter protection in the commercial sector. It investigates the shortcomings of current systems and proposes potential improvements. The thesis explores relevant literature, analyzes security options for doors and other entry points, and examines the relationship between mechanical security systems and alarm and emergency systems in the context of modern technologies. Special emphasis is given to applying "Internet of Things" (IoT) technology in commercial security and hardware signalling.

Keywords: monitoring system, security process management, commercial security, mechanical security systems, alarm security systems, Internet of Things, hardware signaling, perimeter protection.

„Bezpečnosť je ako zdravie ľudia sa o ňu začnú zaujímať až keď je neskoro.“

Ďakujem všetkým ľudom ktorý ma počas tvorenia bakalárskej práce podporovali, tých ktorý ma nepodporovali, nepoznám.

Prohlašuji, že odevzdaná verze bakalářské/diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

Bezpečnosť v komerčnom prostredí je kritickým aspektom, ktorý výrazne ovplyvňuje ochranu majetku a osôb. S narastajúcimi hrozbami a technickými možnosťami je nevyhnutné neustále zlepšovať systémy monitorovania a riadenia bezpečnostných procesov. Táto bakalárska práca sa zameriava na vývoj a implementáciu moderného systému monitorovania bezpečnostných procesov s dôrazom na plášťovú ochranu. V teoretickej časti tejto práce budú preskúmané existujúce systémy a technológie v oblasti bezpečnosti, pričom sa osobitná pozornosť bude venovať mechanickým zabezpečovacím systémom, poplachovým zabezpečovacím a tiesňovým systémom, ako aj využitiu technológie Internet of Things (IoT). Analyzované budú tiež aktuálne trendy v hardvérovej signalizácii a legislatívne požiadavky súvisiace s bezpečnostnými systémami. Praktická časť práce sa sústredí na vývoj prototypu bezpečnostného systému, ktorý bude monitorovať stav dverí a ďalších vstupných bodov. Použitím komponentov ako Arduino vývojová doska, OLED zobrazovadlo, relé modul a magnetické kontakty, bude navrhnutý a zostrojený systém, ktorý umožní efektívne monitorovanie a riadenie bezpečnostných procesov. Prototyp bude testovaný v simulovanom prostredí s cieľom overiť jeho funkčnosť a spoľahlivosť. Cieľom tejto práce je identifikovať nedostatky súčasných systémov a navrhnúť zlepšenia, ktoré zvýšia úroveň bezpečnosti v komerčnom prostredí. Na záver sa vyjadruje poďakovanie vedúcemu práce, doktorovi Stanislavovi Kovářovi, za jeho neoceniteľné rady a podporu počas celého procesu tvorby tejto bakalárskej práce.

I. TEORETICKÁ ČÁST

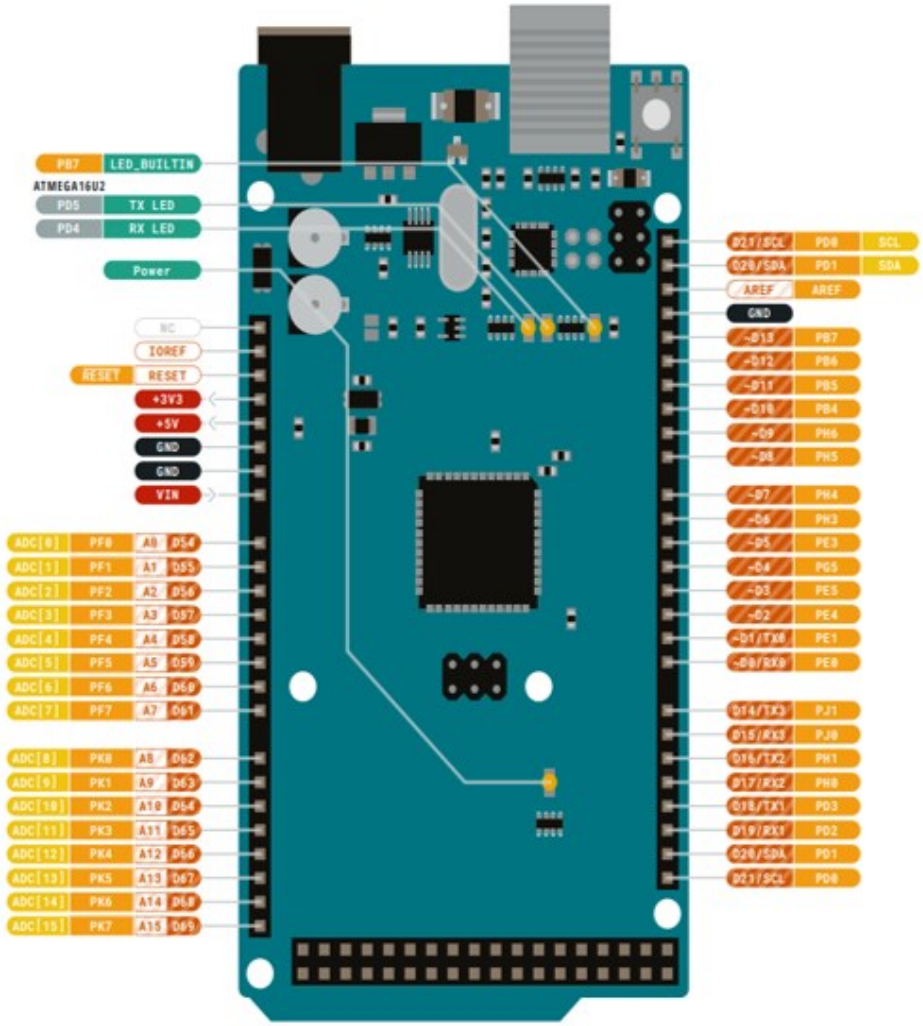
1 ČINNÉ SÚČIASTKY POUŽITÉ PRI TVORBE PROTOTYPU

Súčasťou tejto kapitoly budú popisy HW súčiastok ktoré boli použité v rámci vytvárania HW prototypu. Opísaná bude ich činnosť, fyzické parametre.

1.1 Arduino vývojová doska

Pri tvorbe HW prototypu bola použitá Arduino vývojová doska po označením Arduino MEGA 2560. Kódové číslo 2560 označuje číslo procesora ktorý bol použitý pri výrobe vývojovej dosky a to konkrétne ATmega2560. Arduino MEGA je jedna z najväčších dosiek ktorá ponúka najviac digitálnych a analógových pinov z celej rady Arduino. Taktiež disponuje väčšou pamäťou oproti iným doskám od daného výrobcu. Pre porovnanie s Arduino UNO(najnákladnejšia doska od z výberu od firmy Arduino). "Uno" disponuje 32Kb Flash typu pamäťového úložiska, ktoré je dostačujúce pre jednoduché programy a riešenia. Arduino MEGA disponuje až 256Kb Flash typu pamäťového úložiska. Všeobecné zhrnuté doska Arduino MEGA2560 je vhodná pre masívnejšie výpočtovo náročnejšie a väčšie projekty ako iné dosky od firmy Arduino. Celé portfólio firmy Arduino je Open-Source, čiže kompletne otvorený projekt s prístupom k všetkým „datasheetom", schmátam a dokumentom opisujúcu funkčnosť a funkcionalitu vývojovej dosky. Je dôležité povedať že sa jedná o výbornú voľbu pre všetky druhy projektov a prototypov, z dôvodu vysokého výkonu a veľkej konektivity. Poskytuje dostačujúcu redundanciu pri menších projektoch a dostačujúce kapacity naopak pri projektoch menších. [1]

Arduino Mega tak bolo teda zvolené ešte pri plánovacej činnosti bakalárskej práce a bolo zvolené práve pre svoju predispozíciu byť pre projekt redundantné a teda z pohľadu zostrojiteľa bakalárskej práce vyhovujúcejšie pre prototypizáciu oproti iným doskám od firmy Arduino. Ďalšou nespornou výhodou bola dispozícia výrobku priamo v škole a materiáloch poskytnutých od Vedúceho práce Pána Doktora Stanislava Kováča.



Obrázok 1 Arduino MEGA 2560[1]

Na obrázku je možné vidieť veľké množstvo pinov Arduino Mega. Ktoré môžu slúžiť širokej škále funkčností. Medzi použité piny a porty počas zostrojovania prototypu patrí: USB-B port, Digitálne piny(pod označením DXX), Analógové piny(pod označením AXX), Digitálne piny pre zapojenie zobrazovadla(SDA,SDL).[1]

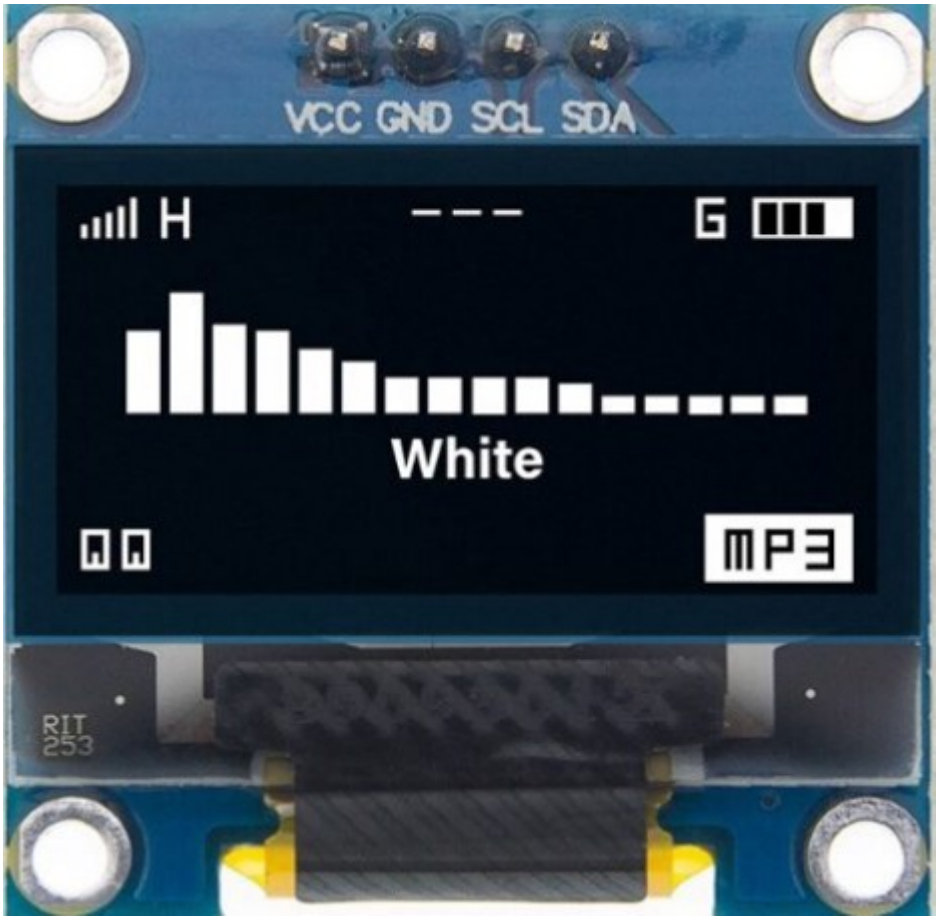
1.2 OLED zobrazovadlo

OLED zobrazovadlo bolo použité v prototype ako komunikácia s používateľom na HW úrovni bez zásahu SW časti prototypu. OLED zobrazovadlo je svojimi pinmy priamo napojené na Arduino Mega, z tohto dôvodu je možné sledovať činnosť a správnosť fungovania HW celku. Na zobrazovadle je pomocou Arduina vykresľovaná „tabuľka“ so

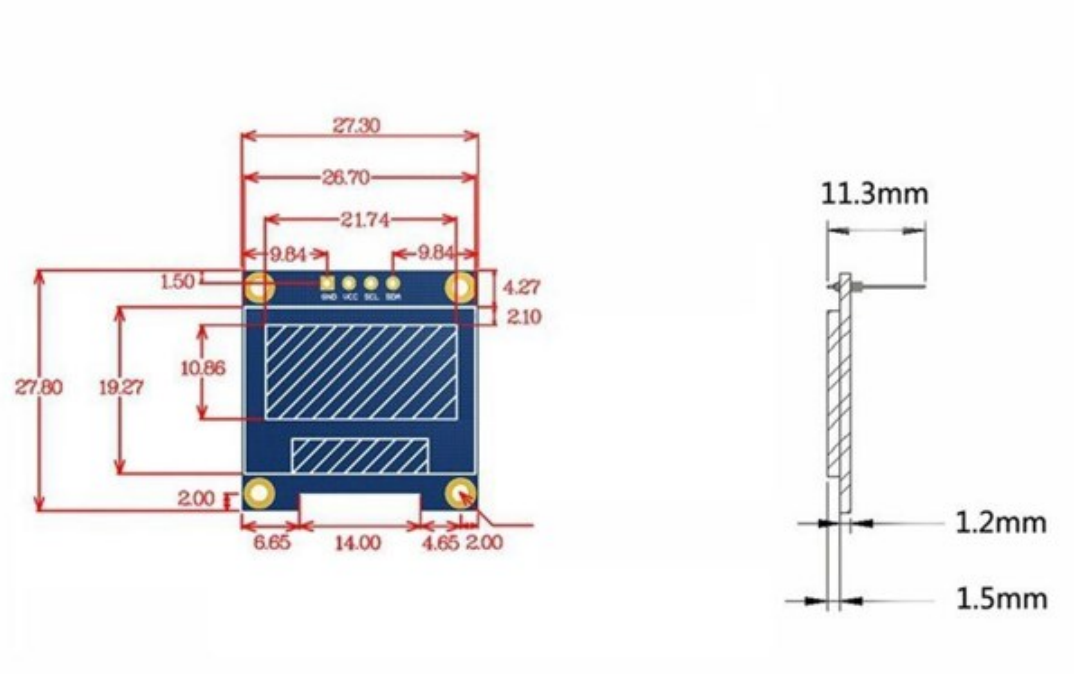
stavom dverí a teda, v akom stave sa dvere nachádzajú, zatvorenom alebo otvorenom. Takéto použitie OLED zobrazovadla bolo použité aj z dôvodu samostatného fungovania HW časti bez nutného pripojenia na SW časť.

OLED zobrazovadlo ponúka biele farby na čiernom podklade, tým vzniká veľký kontrast a ľahká čitateľnosť zobrazovaného čísla. Rozlíšenia displeja je 128x64, čo je vysoké rozlíšenie na 0,96“ (po prepočte 0.96x2,54= 2,44 cm) uhlo priečkovom displeji. Displej sa na Arduino pripája pomocou I2C komunikačného rozhrania, ktoré zjednodušuje komunikáciu a pripojenie displeja na Arduino dosku. Pre fyzické zapojenie stačia 4 vodiče pod označením VCC,GND,SDA,SLC. Konkrétnejšie zapojenie je možné k nahliadnutiu v praktickej časti bakalárskej práce.

Bolo možné požiť širokú škálu zobrazovadieľ napríklad LCD displeje ale pre jednoduchosť použitia a vysokú mieru nastavenia OLED displeja bol vybraný práve OLED displej pre svoj vysoký kontrast a nízku cenu. [2]



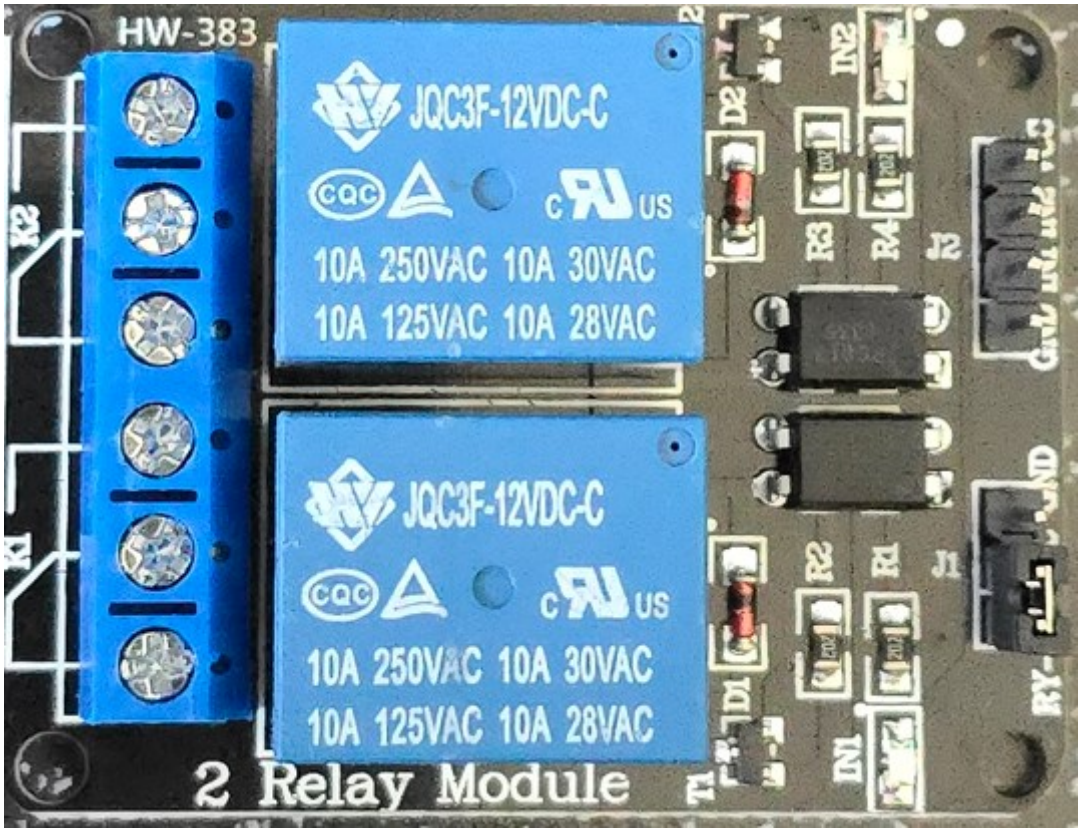
Obrázok 2 OLED zobrazovadlo[2]



Obrázok 3 Rozmery z OLED displeja [2]

1.3 Relé modul

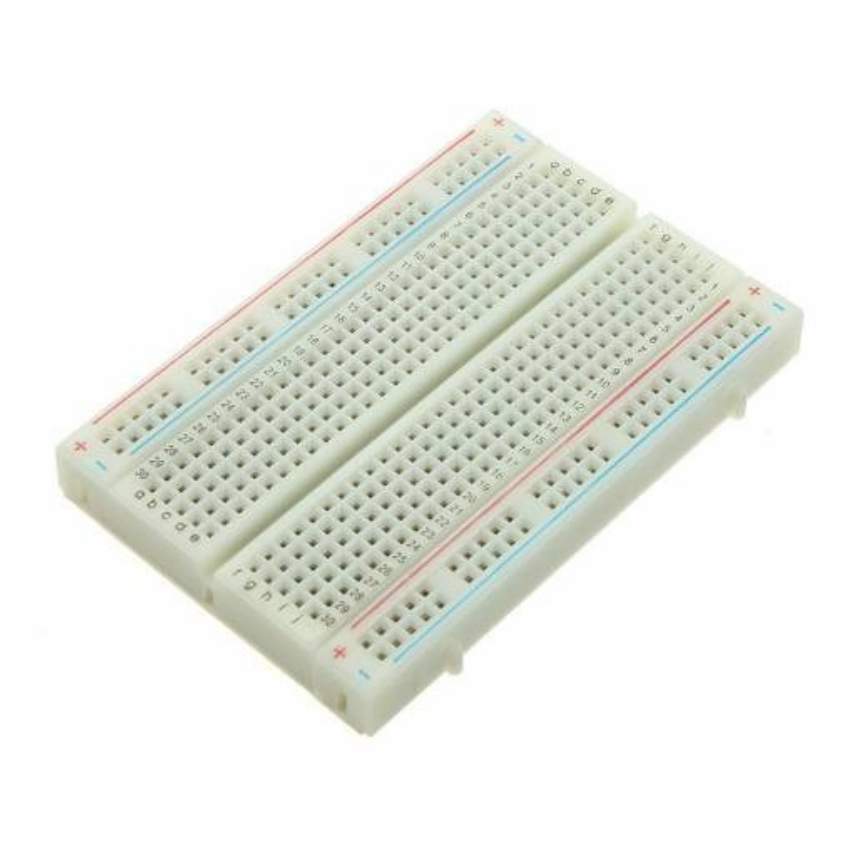
Relé modul, ktorý bol použitý, sa skladá z obvodu magnetického relé a vonkajšieho obvodu, ktorý zaručuje správne fungovanie magnetického relé segmentu. Použitý bol modul umožňujúci nezávisle spínať 2 oddelené magnetické relé. Použitie magnetického relé galvanicky oddeľuje obvod signalizácie od zvyšku práce, zároveň dovoľuje použiť oddelené napájanie na signalizačné prvky. Presné zapojenie relé modulu je možné k nahliadnutiu v praktickej časti bakalárskej práce. Magnetický relé modul umožňuje spínať až 10A pri 250V striedavého napätia. Z toho vyplýva že je možné pomerne nízkym napätím pochádzajúceho z Arduino vývojovej dosky spínať napájacie napätia HW signalizácie. Relé moduly sa v „Arduino svete“ bežne používajú pre podobné prípady. [3]



Obrázok 4 Relé modul[Vlastné]

1.4 Nespájkové pole

Nespájkové pole bolo použité ako alternatíva k cínovému spájkovaniu. Jedná sa o zjednodušenie prototypizácie pri častom prepojovaní a iným zmenám spájaným k prepojovaním súčiastok ktoré sa pri zostave systému použili. Nespájivé pole je prispôsobené k použitiu prototypizačnej kabeláže. Skladá sa zo 4 vertikálnych liniek ktoré sú medzi sebou prepojené. Vonkajšie dve linky po oboch stranách označené červenou farbou sa zapájajú do napájania Arduina, vnútorné dve linky po oboch stranách označené modrou farbou sa zapájajú na GND (uzemnenie) Arduina. Medzi priestor ktorý vznikol je označený písmenami A-J. Na tieto zdierky je možne zapájať súčiastky bez nutnosti spájkovania.



Obrázok 5 Nespájkové pole[4]

1.5 Magnetické kontakty

Magnetické kontakty ktoré boli požité sú jedny z najekonomickejších možností. Sú zložené z dvoch častí. Prvá časť je zapojená do digitálnych pinov na Arduine a zachytená v zárubni dverí a druhá časť ktorá je zachytená priamo o dvere. Funkčnosť magnetických kontaktov je možné opísať nasledovne: Pokiaľ sú obe časti magnetických kontaktov vzájomnej blízkosti ich vzájomné magnetické pole kontakt zopne a obvod sa uzavrie, v opačnom prípade kedy sú obe časti magnetické kontaktu ďaleko od seba je kontakt rozopnutý a obvod nie je uzatvorený. Vďaka tejto logike je možné naprogramovať Arduino tak aby zopnutý alebo rozopnutý obvod snímala ako stav dverí.

Obrázok 6 Magnetický kontakt [5]

1.6 Kabeláž

V celej prototypizácií boli použité dva typy kabeláže. Kabeláž „samec-samec“ ktoré slúžia na prepájaní portov na nespájkovom poli a zároveň pri prepojení nespájkového poľa s Arduinom, druhý typ je kabeláž „samica-samec“ ktoré slúžili na prepojenie výstupov z modulárnych súčiastok ako napríklad OLED zobrazovadlo a relé modul s nespájkovaným poľom alebo priamo s Arduinom. Použitá bola kabeláž v rôznom množstve. Všeobecné zapojenie sa dá upravovať podľa potrieb zhotoviteľa a jeho materiálnym možnostiam. Pri prepojení Arduina s počítačom a SW časťou bol použitý dátový kábel typu USB-B-A.

Obrázok 7 Kabeláž[6]

Obrázok 8 USB dátový kábel USB-B-A[7]

1.7 Poplachová siréna

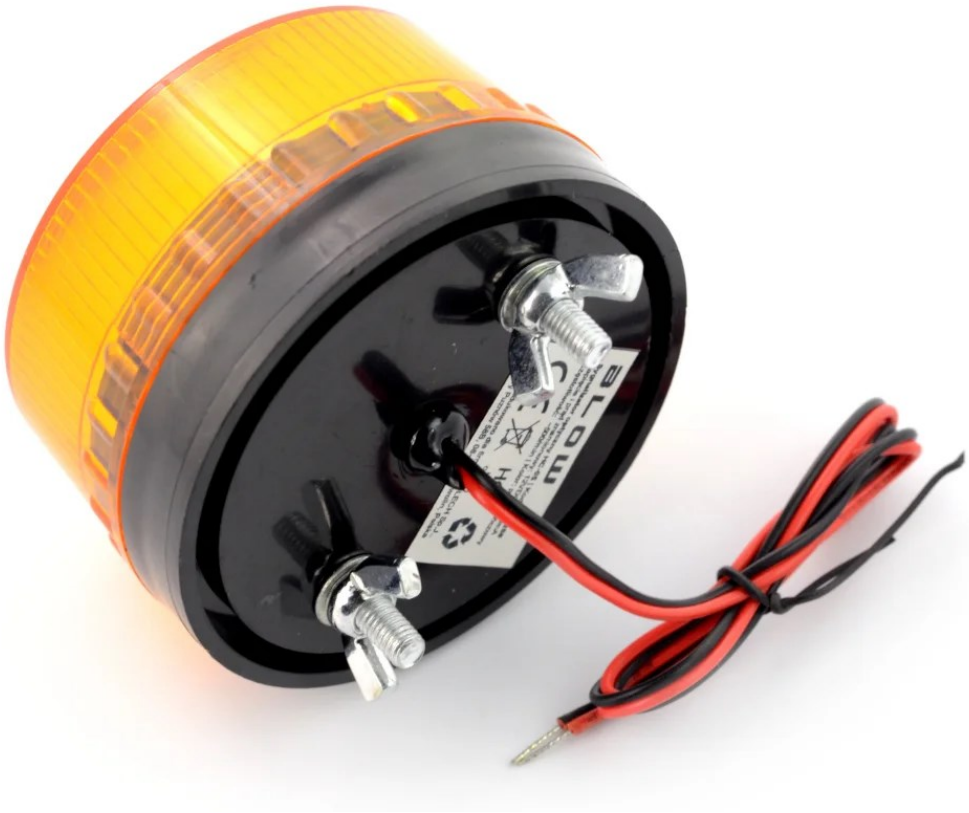
Pre prototyp bola použitá siréna pod označením S2 od výrobcu OEM. Oficiálny datasheet nebolo možné dohľadať, poskytnuté boli len informácie od predajcu ktoré boli nasledovné: „Jednoduchá výstražná siréna napájená napětím od 6 V do 14 V se spotřebou proudu až 250 mA. Generuje 6 po sobě jdoucích zvukových tónů. Pro připojení zařízení se používají dva vodiče s odstraněnou izolací. průměr je 54 mm, výška je 55 mm.“[8] Vybraná bola z dôvodu

jednoduchosti inštalácie a samostatnej náture celkového produktu. Pre prototypizáciu je dostačujúca. Telo sirény je prispôsobené na pripevnenie na pevné povrchy.[8]

Obrázok 9 Siréna[8]

1.8 Signalizačné LED svetlo

Pre optickú signalizáciu bola vybraná siréna taktiež od výrobcu OEM pod označením HC-05. Oficiálny datasheet nebolo možné dohľadať, poskytnuté boli len informácie od predajcu ktoré boli nasledovné: „Signální lampa oblíbená v automatizaci bran. Zařízení je okamžitě viditelné a zajišťuje vysokou úroveň zabezpečení. Funguje jako stroboskop s frekvencí cca 500 / min.“ [9] Napája sa pomocou jednosmerného prúdu o veľkosti 12V spotreba pri funkčnosti je cca 140mA. Je určená na vnútorné použitie no napriek tomu disponuje Triedou tesnosti IP44. [9]



Obrázok 10 Signalizačné osvetlenie[9]

Celkové počty použitých materiálov

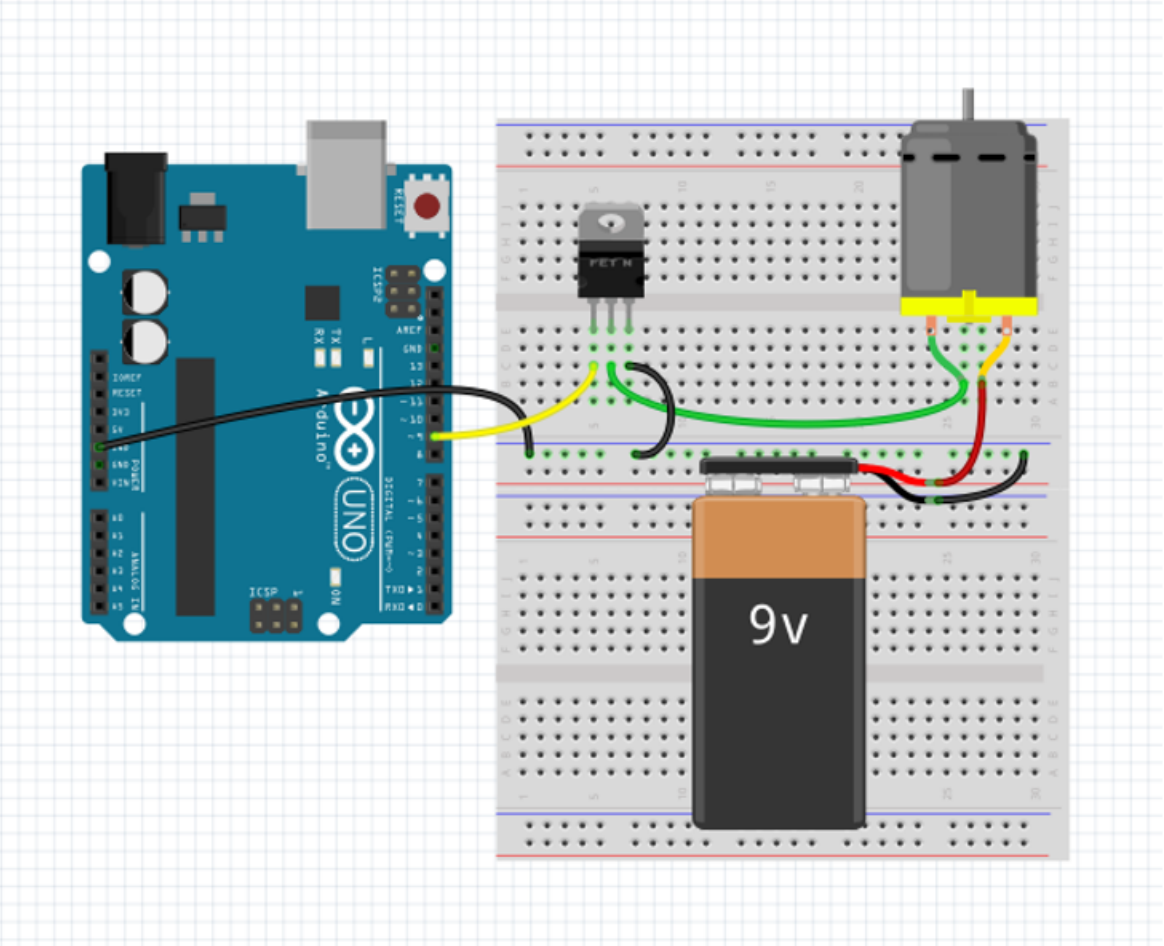
V tabuľke je možné vidieť množstvá materiálov ktoré boli použité na zostavenie prototypu.

Tabuľka 1 Materiál [Vlastné]

Použitá súčiastka	množstvo
Kabeláž samec-samec	12 ks
Kabeláž samica-samec	7 ks
Arduino	1 ks
Relé modul	1 ks
USB-B-A	1 ks
Nespájkové pole	1 ks
OLED zobrazovadlo	1 ks
Magnetický kontakt	5 ks
Poplachová siréna	1 ks
Optická siréna	1 ks

2 FRITZING

Všetky schémy zapojenia ktoré budú použité v celej bakalárskej práci sú vytvorené v programe Fritzing. Táto aplikácia je opensource a jej celý zdrojový kód je prístupný na GIT-hube. Fritzing sa snaží o vytvorenie aplikácie ktorá sprístupní modelovanie, navrhovanie a písanie Arduino kódu pre všetky vekové kategórie a pre všetky úrovne chápanie techniky a programovania. Fritzing bol pri príprave zvolený pre svoje grafické zobrazenia elektrotechnických schém. Grafické zobrazenie zapojenia výrazne zjednodušuje čitateľnosť a chápanie zostrojeného obvodu. Ďalšou z nesporných výhod SW Fritzing je jednoduchá zmena a spravovanie súčiastok zapojených v elektrotechnickej schéme, pri prototypovaní je táto vlastnosť veľmi vítaná. Posledným vymenovaným dôvodom bude to že bakalársku prácu ozvláštni a dodá elektrotechnickým schémam požadovaný „lesk“. Na obrázku 11 je možné vidieť k nahliadnutiu jednu z „demo“ elektrických schém ktorý Fritzing ponúka.[10]



Obrázok 11 Fritizng[Vlastné]

3 LEGISLATÍVA

Legislatíva upravujúca signalizáciu poplachových systémov sa nevyskytuje vo veľkom rozsahu. Pri uvedení výrobku na európsky trh sa očakáva dodržanie všetkých noriem ktoré platia v členských štátoch Európskej únie a týkajúc sa elektrických zariadení pre príklad normy ČSN 33 0010 ED.2 platnej v Českej republike, ktorá upravuje kategorizáciu elektrických zariadení. Ďalšou dôležitou normou je ČSN EN 50131, konkrétne tabuľka č. 10, ktorá stanovuje požiadavky na hlásenia. Výstražné zariadenie môže byť v prípade potreby oneskorené maximálne o 10 minút, pričom jeho činnosť môže byť zastavená len v prípade, že poplachové hlásenie bolo úspešne prenesené. V prípade detekcie poruchy musí byť akékoľvek oneskorenie činnosti zariadenia zrušené, v prípade ak sú poruchy detegované na všetkých prenosových trasách. Tak akustické výstražné zariadenie musí byť v prevádzke minimálne 90 sekúnd, v prípade ak nie sú stanovené kratšie časy miestnymi alebo národnými predpismi. Maximálna doba prevádzky nesmie presiahnuť 15 minút no iba pokiaľ nie sú predpismi stanovené kratšie časy. Hlásenie poruchy napájacieho zdroja môže byť oneskorené maximálne o 1 hodinu. Hlásenia môžu byť doplnené nepovinnými prostriedkami, ak neovplyvňujú správnu činnosť povinných zariadení, napríklad siréna s batériovým napájaním alebo zariadenia znižujúce viditeľnosť. [11],[12]

Signalizácie sa ďalej týka norma ČSN EN 50131-4 ed. 2 ktorá pojednáva o Výstražných zariadeniach v ktorej je uvedená tabuľka nižšie ktorá je vo forme obrázku.

Funkce	Nezávislé (autonomní) napájení				Dálkové napájení			
	Stupeň				Stupeň			
	1	2	3	4	1	2	3	4
Aktivační povel	M	M	M	M	M	M	M	M
Generování sabotážního signálu nebo zprávy	M	M	M	M	M	M	M	M
Generování poruchového signálu nebo zprávy	Op [b]	Op [b]	M	M	Op	Op	Op	Op
Monitorování vzdáleného napájení [a]	M	M	M	M	Op	Op	Op	Op
Monitorování integrity propojení aktivačního povelu	Op	OP	M	M	Op	Op	Op	Op
Místní autonomní test	Op [b]	Op [b]	M	M	Op	Op	Op	Op
Vstup pro dálkově řízený test	Op	Op	Op	M	Op	Op	Op	Op

Legenda

Op volitelné *(Optional)*

M povinné *(Mandatory)*

[a] Monitorování vzdáleného napájení se používá u výstražných zařízení se vzdáleným napájecím zdrojem a s interním zálohovacím zařízením, viz typy X a Z definované v tabulce 9.

[b] Povinné pro zařízení typu W definovaného v tabulce 9.

Obrázok 12 Výstražné zariadenia[12]

Z tabuľky je možné vyčítať aké činnosti výstražného systému sú povinné v rôznych stupňoch zabezpečenia. Ďalej norma stanovuje úrovne akustického výkonu ktorý je potrebný dosiahnuť vo vonkajšom ale i vnútornom umiestení výstražného zariadenia.[12]

	Vnitřní výstražné zařízení	Vnější výstražné zařízení
Minimální úroveň střední hodnoty akustického výkonu	80 dB (A)	100 dB (A)
Minimální úroveň jednotlivé hodnoty akustického výkonu	75 dB (A)	95 dB (A)

Obrázok 13 Úrovne akustického výkonu[12]

Norma ďalej stanovuje ďalšie podmienky na výstražné zariadenia napríklad ako: *Konštrukcia krytu, rozmery nástroja k ochrane proti sabotáži, rozmery nástroja k detekcii sabotáže, detekcia sabotáže, odstránenie z montážneho krytu, doba trvania pohotovostného stavu akumulátora-batérie.* [12]

Bakalárskej práce sa týkajú aj normy ktoré sú orientované na detektory stavu otvorenia týmto zariadeniam sa venuje norma ČSN EN 50131-2-10 ktorá stanovuje medzi inými aj udalosti, ktoré majú byť zapracované a hlavné funkcie. Tabuľka k nahliadnutiu nižšie.[12]

Událost, která má být zpracována a funkce, které mají být poskytnuty	Stupeň			
	1	2	3	4
Překročení maximální pracovní vzdálenosti	M	M	M	M
V pracovní vzdálenosti	M	M	M	M
Vzdálené povolení indikace detekce[a]	Op	Op	M	M
Narušení spojovacího pole	Op	Op	M	M
Prevence a detekce přístupu k vnitřku detektoru	Op[b]	M[c]	M[c]	M[c]
Demontáž z montážního povrchu[d]	Op	M	M	M
Nízké napájecí napětí[e]	Op	Op	M	M
Celková ztráta externího napájení[f]	M	M	M	M
Odpovídající dvojice, kódování a šifrování	Op	Op	M[g]	M[g]

Legenda
M = Povinné (*Mandatory*)
Op = Volitelné (*Optional*)

[a] Vyžaduje se, pouze pokud je přítomen indikátor detekce.
[b] Pro otevření normálním způsobem není nutné aktivovat sabotážní signál.
[c] Není nutné pro utěsněné kontakty v souladu s 4.5.2.
[d] Povinné pro bezdrátové detektory, stupně 2, 3 a 4; povinné pro všechny detektory stupně 4, nepovinné pro drátové detektory stupně 1, 2 a 3.
[e] Viz 4.6.7.
[f] Viz 4.6.6.
[g] Viz 4.5.5.

Obrázok 14 Udalosti, ktoré majú byť zapracované a hlavné funkcie[12]

Ďalej sa zaoberá odolnosťou voči magnetickému poľu ktorá je nesmierne dôležitá pretože magnetické kontakty fungujú na zmene magnetické poľa vo svojej najbližšej blízkosti. V neposlednej rade sa norma zameriava na zabezpečenie pri sabotáži a elektrickým skúškam ktoré sa týkajú len detektorov ktoré sú napájané externým zdrojom elektrickej energie.[12]

4 PLÁŠŤOVÁ OCHRANA

Bakalárska práca opíše zostrojený prototyp signalizácie otvorených dverí a práve vďaka tejto skutočnosti je pre bakalársku prácu vhodné opísať plášťovú ochranu. Cieľom plášťovej ochrany je zabrániť neoprávnenému (škodlivému) prístupu do vnútra objektu. Plášťová ochrana pre svoje účely používa dva koncepty a to fyzickú alebo virtuálnu bariéru. Takáto bariéra dokáže svojím pôsobením odhaliť neoprávnený prístup a minimalizovať riziká. [12] Pojem plášťová ochrana označuje metódy MZS a PZTS. Pre Metódy MZS to sú konkrétne tieto prvky: ploty, steny, zámky, dvere. Z pohľadu PZTS to sú znova: infračervené závory, mikrovlnné závory, mikrofóne alebo optické kabely snímajúce vibrácie, štrbinové kabely, deformačné senzory, tlakové podzemné hadice. [13]

Poplachové zabezpečovacie a núdzové systémy (PZTS) slúžia ako komplexné riešenia určené na identifikáciu neautorizovaného vstupu, na aktiváciu poplachov pri takomto vniknutí a na vyvolanie núdzového signalizovania.

Primárnym poslaním PZTS je posilniť bezpečnosť chráneného priestoru prostredníctvom zistenia prítomnosti neautorizovanej osoby a spustenia núdzového poplachu. Systémy sú navrhnuté na ochranu životov, zdravia a majetku pred krádežou, vlámaním alebo poškodením. PZTS neposkytuje možnosť fyzického zásahu proti narušiteľovi, jeho hlavnou úlohou je detekcia, signalizácia a prenos informácií. Pre dosiahnutie najvyššej efektivity by mali byť PZTS integrované s Mechanickými zábranovými systémami a postupmi.[14]

Do roku 2007 boli systémy, známe dnes ako PZTS (poplachové zabezpečovacie a núdzové systémy), bežne označované akronymom EZS, čo predstavuje „elektrické zabezpečovacie signalizačné zariadenia“ alebo „elektrické zabezpečovacie systémy“. Avšak, v tomto roku došlo k zmene terminológie, pričom EZS bolo nahradené termínom PZTS, čo v anglickom jazyku zodpovedá I&HAS (Intrusion and Hold-Up Alarm Systems). [15],[13]

5 SIGNALIZÁCIA

Signalizovanie poplachu je jeden z najdôležitejších aspektov celého PZTS komplexu. Signalizácia narušenia objektu plní nasledujúce úlohy a to:

- upozornenie ostrahy a/alebo majiteľa objektu na neautorizovaný vstup,
- stresovú stimuláciu na osobu ktorá je neautorizovaná v objekte.

5.1 HW signalizácia

Pri správnom použití HW signalizácie dokonca môže plniť aj funkciu dezorientujúceho prvku pre osobu ktorá sa v objekte nevyzná ale iba za predpokladu súčinnosti viacerých prvkov. Svetelného osvetlenia napojeného na PZTS akustických a vizuálnych signalizačných prvkov.[16] Postup takéhoto dezorientujúceho prvku môže vyzerať nasledovne. V narušenom segmente zhasne osvetlenie ktoré je napojené na PZTS. Rozoznejú sa sirény ktoré indikujú narušenie pre ostrahu alebo majiteľa. Spôsobujú zvýšenú mieru hluku v priestoroch. Zapnutie stroboskopického blikania v objekte ktorý tiež plní aj funkciu oznámenia narušenia objektu. Takáto súčinnosť signalizačných a osvetľujúcich prvkov v objekte môže výrazne zhoršiť navigáciu a pôsobiť stresovo na neautorizovanú osobu v objekte.[16], [17]

Hardwarová signalizácia v rozsahu bakalárskej práce môže byť rozdelená na nasledujúce prvky:

- akustická,
- vizuálna.

V aktuálnej dobe je možné zakúpiť produkt ktorý spája oba prvky do jedného.

Konkrétne parametre nasledujúceho modelu B710-720 produktu od značky Mucco Warning Lamps sú nasledujúce.[19]

Model	Voltaj/Power Supply	Akım/Current(mA)Max.
SNT-B710	24V/DC	150mA
SNT-B720	40-260V AC/DC	37mA

Melodi/Melody	Ses Şiddeti /Sound Intensity(dB) (10cm)	Ses Şiddeti /Sound Intensity(dB)(1m)
M1	128	103
M2	118	100

Obrázok 15 Parametre[19]

Z datasheetu je možné vyvodiť nasledujúce informácie:
Jeden meter vzdialenosti od produktu je akustická hluková hodnota na úrovni 103dB čo pre predstavu znamená hluk porovnateľný s hlukom ktorý vydáva reťazová motorová píla. Takáto hodnota hluku zaručene osobu s neautorizovaným prístupom zaručene zmätie.[18]

Tabuľka 2 Decibely [20]

Decibely [dB]	Približná hodnota hluku
90	Idúci vlak
100	Maximálne zaťaženie motora motorky
100	Reťazová motorová píla
110	Zbíjačka
110	Rockový koncert
130	Štart tryskového lietadla

Pri výbere a návrhu signalizačného systému je dôležité zohľadniť charakteristiku objektu, očakávané riziká a potreby konkrétneho používateľa. Výber správneho typu signalizácie a jej správna integrácia do celkového bezpečnostného systému môže výrazne prispieť k ochrane majetku a zvýšeniu bezpečnosti.

5.2 Trendy HW signalizácie

Posledné desaťročie je trendom naprieč všetkými elektrotechnickými, technickými a bezpečnostným prostrediami, zmenšovanie a integrácia všetkých prvkov do čo najmenšieho balíčku. Tento trend sa nevyhol ani HW signalizácií, zmeny ktoré je možné si povšimnúť, je integrácia viacerých prvkov napríklad ako zlúčenie optickej a akustickej signalizácie do jedného zdroja z ktorého obe vychádzajú. Taktiež sa posledné roky dostali do popredia majáky, ktoré pri svojej funkcii používajú svetlo vyžarujúce diódy (LED).

5.2.1 Použitie LED

Majáky ktoré používajú pre svoju činnosť LED diódy sa vyznačujú menšími rozmermi, nižšou spotrebou a ľahšími spôsobmi napájania. LED diódy taktiež umožňujú kombinovanie funkcií. Medzi príklady je možné uviesť LED panely ktoré slúžia ako HW signalizácia no kombinujú ju s komunikáciou s davmi a kamerovým dohľadom, jedným z výrobcom takýchto popredných panelov HW signalizácie je firma amena display S.R.O. [21]

Obrázok 16 Amena displej[21]

5.2.2 Pokusy o integorvanie PZTS do IoT

Počas literárnej rešerše k bakalárskej práci bol nájdený dokument popisujúci stavbu systému pod názvom „Monitoring and Controlling of Home Security System using IoT and LabVIEW“. Účelom tohto prototypu je zabezpečenie domu za pomoci IoT a LabVIEW, ktoré monitorujú a kontrolujú bezpečnostný systém pomocou senzorov a kamier. Systém

využíva Raspberry Pi ako riadiacu jednotku a rôzne senzory, vrátane PIR senzorov a LDR senzorov. PIR senzory sú umiestnené pred dverami a vo vnútri domu na detekciu pohybu. Ak PIR senzor zaznamená pohyb, kamera pripojená k systému zachytí tvár osoby a porovná ho s databázou. Ak je osoba známa, dvere sa automaticky otvoria pomocou DC motora. Ak je osoba neznáma, systém upozorní majiteľa pomocou SMS. LDR senzor je použitý na detekciu otvorenia a zatvorenia okien, pričom operácia je riadená DC motorom. Pre kontrolu systému bola vyvinutá mobilná aplikácia, ktorá umožňuje všetky tieto operácie ovládať na diaľku. Výsledky sú zobrazované pomocou softvéru LABVIEW, čo umožňuje efektívne a presné sledovanie.[22]

Ďalší dokument, „Monitoring System for Doors and Windows of a Data Center with IoT“, z roku 2019 opisuje podobný systém, ktorý signalizuje otvorené dvere a okná v dátovom centre za použitia IoT bezdrôtových senzorov. Tieto senzory komunikujú so serverom cez internet a PyBoard, pričom server vykresľuje užívateľské prostredie. Je možné konštatovať, že na bezpečnostné prvky sú kladené čoraz väčšie nároky na integráciu s ostatnými zariadeniami, najmä na integráciu s IoT. Takáto integrácia poskytuje prepojenie medzi prvkami, ktoré môžu ovládať rôzne aspekty domu, ako sú elektrické inštalácie, vzduchotechnika, či osvetlenie. Je však nevyhnutné preskúmať bezpečnosť komunikačných štandardov pri bezdrôtovej komunikácii IoT a všeobecnú bezpečnosť bezpečnostných IoT prvkov. [23], [24]

Pre budúcnosť je dôležité, aby bezpečnostné firmy, agentúry a výrobcovia boli pripravení rýchlo a moderne reagovať na zmeny v technickom svete a na želania stále náročnejších zákazníkov. Integrácia IoT technológií hrá kľúčovú úlohu pri zvyšovaní bezpečnosti a pohodlia užívateľov, čo predstavuje významný krok vpred v oblasti domácej automatizácie a bezpečnosti.

6 DETEKTORY

V aktuálnej časti práce budú opísané vybrané typy detektorov, s ktorými je možné sa stretnúť.

6.1 Mechanické detektory

Mechanické detektory ktoré opisujú odborné prednášky od Pána Doktora Inžiniera Rudolfa Drgu opisuje mechanické spínacie detektory nasledovne: Mikrospínače, o ktorých sa hovorí, sú navrhnuté tak, aby sa dali integrovať do rámov dverí, kde budú pôsobiť proti západke. Ide o zariadenia, ktoré využívajú mechanický pohyb na uzavretie alebo prerušenie elektrického obvodu, čím detegujú otvorenie dverí.[25]

Druhý typ mikrospínačov je navrhnutý ako pružinové hroty, ktoré pri kontakte s vodivými plochami uzavierajú elektrický obvod. Tento mechanizmus sa často používa na monitorovanie stavu okien a dverí. Pri ich otvorení dôjde k prerušeniu obvodu, čo vyvoláva alarm. Tieto mikrospínače sú teda kľúčovými komponentmi bezpečnostných systémov, kde zabezpečujú okamžitú reakciu na neoprávnené vstupy alebo pokusy o vniknutie.[25]

6.2 Elektrické detektory

Odborná literatúra rozdeľuje elektrické detektory takhľa:

„• elektromechanické detektory,

• elektromagnetické detektory,

• elektroakustické detektory,

• pasivní infračervené detektory.“[26]

Princíp činnosti detektora je možné opísať ako snímač ktorý vyhodnocuje fyzikálnu veličinu ktorá sa mení v čase v závislosti od okolného prostredia a jej zmenu je možné vyhodnotiť ako narušenie prostredia ktoré detektor sníma. Takýto detektor môžeme charakterizovať ako vstupné zariadenie ktorého výstup je podnet k spusteniu alarmu. Základné rozdelenie stavov ktoré môžu pri svojej činnosti detektory nadobúdať sú nasledovné:

„• stav klidu - detektor je ve stavu klidu, je pouze napájen, nestřeží prostor,

• stav poplachu - detektor hlásí poplach, narušení prostoru,

• stav sabotáže - detektor hlásí, že je sním manipulováno,

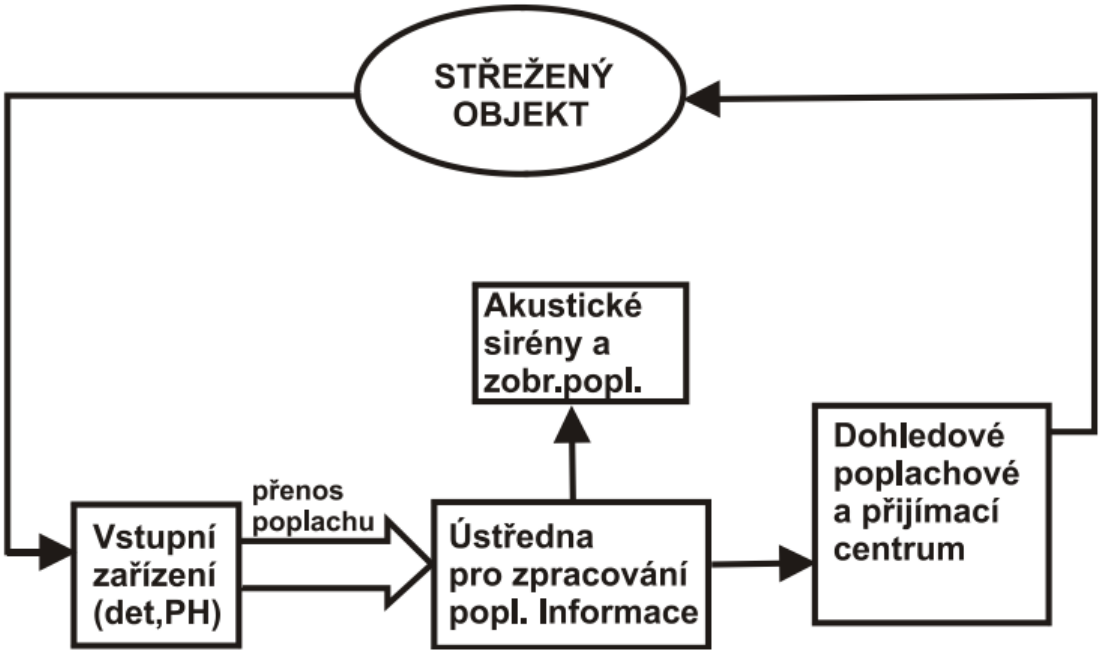
• stav zastřeženo - detektor předává informace o poplachu, sabotáži a poruše,

• stav porucha - detektor hlásí poruchu,

• stav test - detektor má blokovány všechny signály kromě sabotáže.“[25]

Pri vypracovaní prototypu bude nastavené správne sebahodnotenie činnosti detektora. Teda prototyp by mal nadobúdať stavy: Kľudu, Poplachu, Zastraženia.

6.3 Elektromechanické detektory



Obrázok 17 Elektromechanické detektory [25]

Elektromechanické detektory patria medzi najstaršie typy detektorov. Práve vďaka tejto skutočnosti boli používané už v prvých systémoch technickej ochrany a sú stále veľmi široko využívané aj dnes. Ich princíp spočíva v mechanickej zmene prostredia (jav, ktorý nastáva ako dôsledok narušenia bezpečnosti), ktorá je premenená na elektrický signál. Práve tento jednoduchý princíp fungovania ich robí veľmi jednoduchými v aplikovaní naprieč širokou škálou alternatív nasadenia.[25]

II. PRAKTICKÁ ČÁST

7 BODY PRAKTICKEJ ČASTI

Praktická časť si dáva za úlohy:

1. vytvorenie HW prototypu ktorý bude ovládať HW signalizáciu otvorenia dverí,
2. overenie HW prototypu,
3. zobrazovanie stavu dverí na pôdoryse objektu ktorý bude spracovávať informácie časového intervalu otvorenia dverí a trasovanie priechodov v objekte,
4. overenie funkčnosti SW prototypu,
5. overenie prepojenia HW a SW časti.

Zostrojený prototyp by mal splňovať všetky činné úlohy. Predpoklad prototypu je schopnosť fungovať správne v simulovanom prostredí. Z dôvodu že sa jedná o prototyp nemusí spĺňať normy ktoré sú naň kladené a očakávane že spĺňať bude. Pri práci s prototypom bola dodržovaná základná etika pri práci s elektrickým zariadením. Pri tvorbe prototypu bude kritické správne zapojenie Arduino vývojovej dosky, jej následné naprogramovanie a úprava kódu, ktorý bude zohľadňovať všetky nároky, ktoré naň budú kladené zo strany zadanie bakalárskej práce. Táto fáza bude zahŕňať dôkladné naplánovanie zapojenia, medzi ne patrí napríklad správne pripojenie magnetických kontaktov, magnetického relé modulu a OLED zobrazovadla k Arduino doske. Každý komponent musí byť presne integrovaný, aby bola zabezpečená spoľahlivosť prototypu.

Programovanie Arduino vývojovej dosky bude vyžadovať napísanie kódu, ktorý nielenže riadi jednotlivé komponenty, ale aj optimalizuje výkon celého systému. To zahŕňa vytváranie efektívnych algoritmov, ktoré minimalizujú latenciu a maximalizujú odozvu systému.

Ďalšou problematikou, ktorou sa bude praktická časť zaoberať, je vytvorenie komunikačného systému, ktorý bude zodpovedný za komunikáciu medzi HW a SW časťou prototypu. Tento systém musí byť navrhnutý tak, aby bol čo najjednoduchší na spracovanie pre obe časti. Komunikácia musí byť rýchla a efektívna, aby nedochádzalo k veľkým čakacím dobám, ktoré by mohli negatívne ovplyvniť výkon a odozvu systému. To zahŕňa výber vhodného komunikačného protokolu a implementáciu mechanizmov na synchronizáciu dát medzi HW a SW komponentmi.

Praktická časť bude ďalej riešiť správne fungovanie softvérovej časti, ktorá musí splňovať všetky požiadavky zadané v úlohách bakalárskej práce. Bude obsahovať užívateľské rozhranie pre monitorovanie a ovládanie prototypu, ako aj algoritmy a príkazy pre spracovanie dát a komunikáciu s hardvérom.

Celkový úspech prototypu bude závisieť na starostlivom naplánovaní a realizácii všetkých týchto krokov, pričom každá časť musí byť správne vykonaná a otestovaná. Cieľom je vytvoriť spoľahlivý systém, ktorý spĺňa všetky stanovené požiadavky a poskytuje požadovanú funkcionalitu.

8 PROTOTYPIZÁCIA

V praktickej časti sa bakalárska práca zameria na prototypizáciu, testovanie a overenie funkčnosti prototypu.

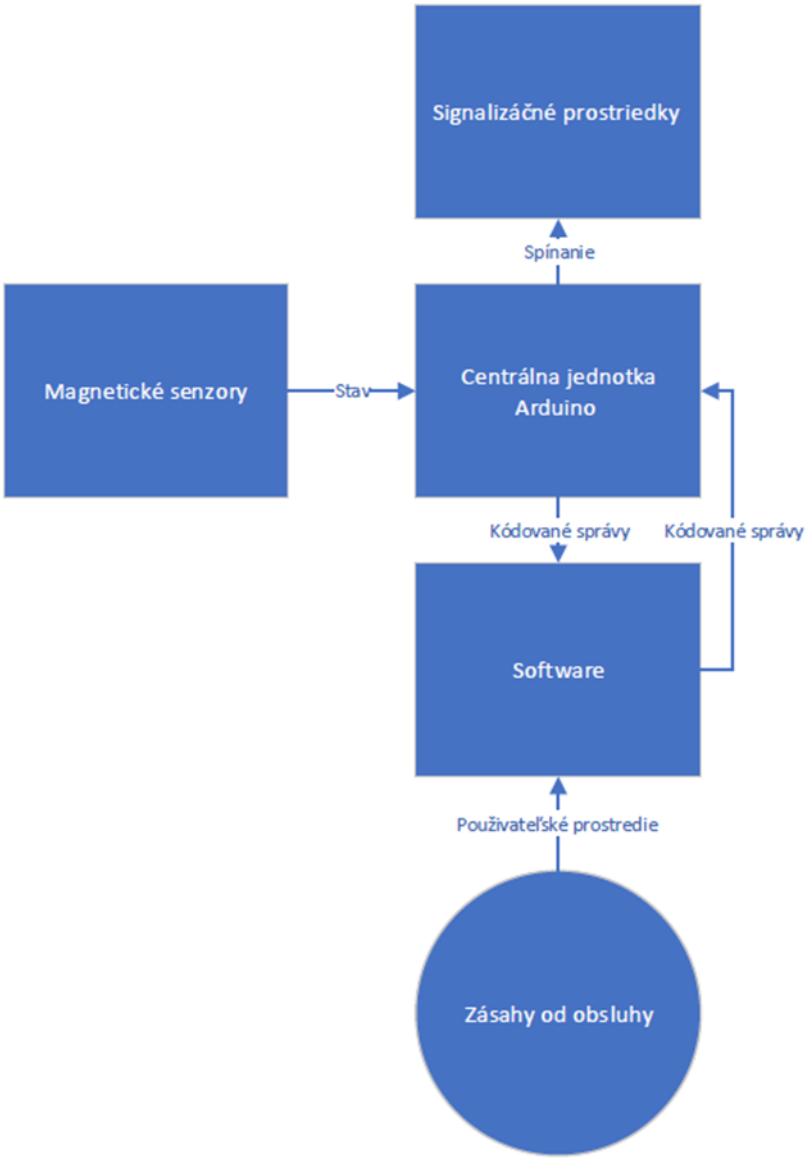
Prvá fáza prototypizácie bola plánovanie, v tejto časti boli vytýčené ciele ktoré by mal fyzický prototytp splňovať a to:

1. snímanie stavu dverí, otvorené/zatvorené,
2. zobrazovanie stavu na fyzickom zobrazovači pripojenému k Arduino doske,
3. odosielanie dát na serial-linku pre ďalšie spracovávanie v programe,
4. spínanie HW signalizácie pri splnení podmienok pre spustenie poplachu.

8.1 Činnosť prototypu

Pre celkové fungovanie systému budú použité 3 schémy. Prvá uvedená schéma bude zobrazovať všeobecné fungovanie celého systému. Druhá schéma sa zameria na priblíženie a konkrétnejšie zobrazenie a vysvetlenie HW časti Bakalárskej práce a tretia schéma sa zameria a priblíženie SW časti bakalárskej práce.

Všeobecná funkcia systému

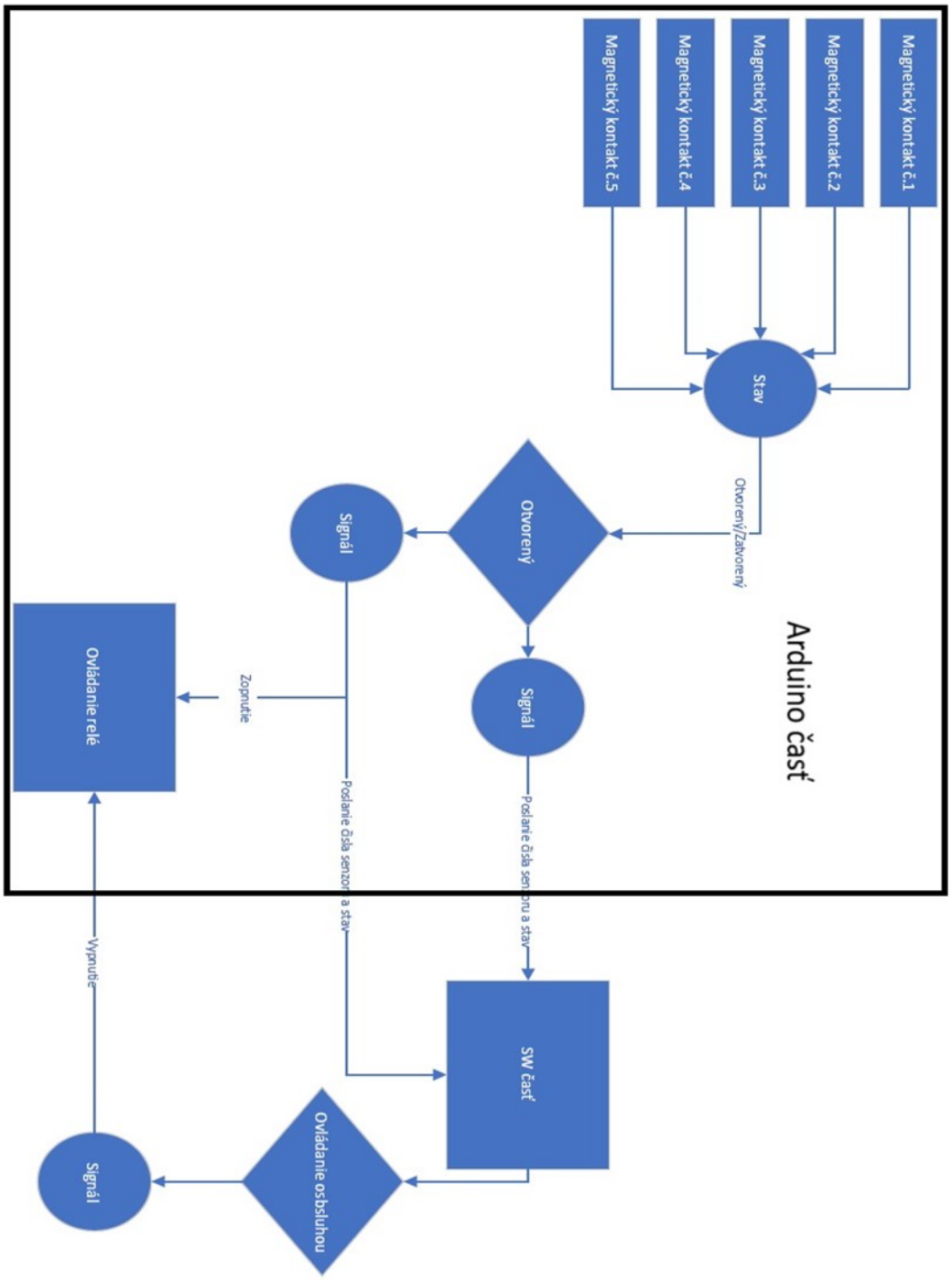


Obrázok 18 Blokové schéma [Vlastné]

Vysvetlenie blokovej schémy je nasledovné: Magnetické kontakty pri otvorených dverách uzatvoria obvod. Tento stav Arduino vyhodnotí formou poslania kódovanej správy do Softwareu aplikácie. A zároveň bez zásahu SW časti spúšťa poplach zopnutím relé modulu pripojeným na Arduino. SW časť kódované správy vyhodnocuje pomocou svojho kódu a na pôdoryse objektu zobrazuje aktuálny stav dverí. Do svojich logovacích panelov. V paneloch sa zároveň vypočítavajú prechody v objekte a ukladajú sa stavy dverí SW má možnosť

vypnúť alarm pomocou tlačidla v programe. Jedná sa o jedinú možnosť ako signalizáciu zastaviť.

Činnosť Arduino vývojovej dosky

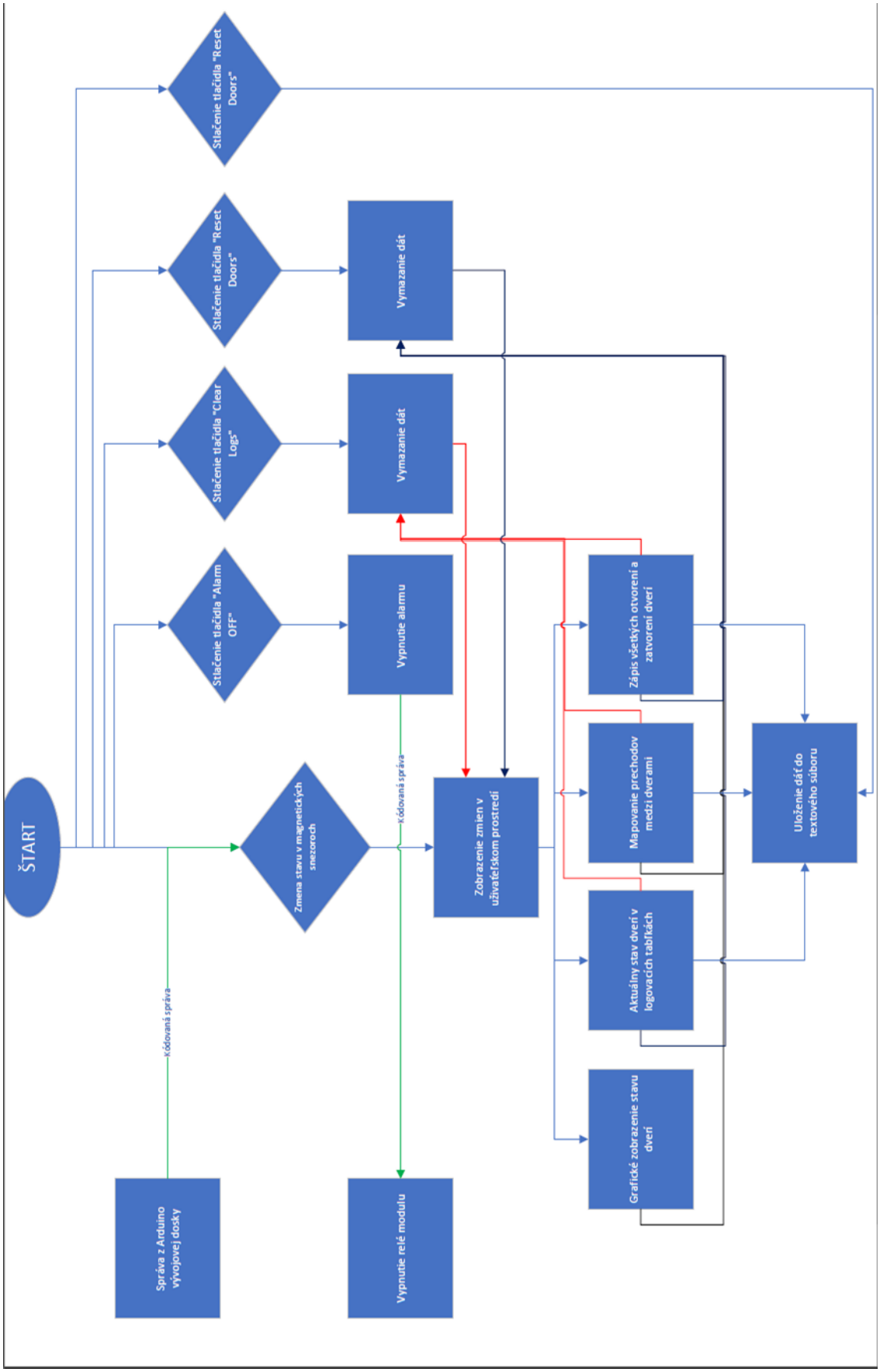


Obrázok 19 Činnosť Arduino vývojovej dosky[Vlastné]

Pre lepšiu predstavu fungovania Arduino časti prototypu je možné upriamiť svoju pozornosť na Obrázok 19 Činnosť Arduino vývojovej dosky, ktorý znázorňuje už vyššie vysvetlenú Funkčnosť prototypu.

Činnosť SW Časti

Pre lepšie pochopenie SW časti bakalárskej práce je na Obrázok 20 Činnosť SW časti. Vyobrazený vývojový diagram. Program pracuje náležite: Signály z Arduino vývojovej dosky vo forme kódovanej správy sú spracovávané SW a sú z nich zobrazené tieto dáta. Grafické zobrazenie stavu dverí, aktuálny stav dverí, mapovanie prechodov medzi dverami, zápis všetkých stavov dverí. Užívateľ má možnosť vypnúť alarm, SW časť v tomto prípade odošle kódovanú správu do sériovej linky a Arduino ju spracuje. Má možnosť premazať všetky logové tabuľky nimi sú Aktuálny stav dverí, Mapovanie prechodov medzi dverami a zápisy všetkých stavov dverí. Má možnosť resetovať stavy dverí, v tomto prípade budú premazané všetky dáta ktoré sú aktuálne zobrazované. A v neposlednej rade ma používateľ možnosť uložiť zobrazované logy do textového súboru.



Obrázok 20 Činnosť SW časti[Vlastné]

8.2 Snímanie stavu dverí

Pre splnenie prvého cieľa boli zvolené magnetické kontakty ktoré sú opísane v teoretickej časti práce. Magnetické kontakty boli napojené priamo na Arduino jednotku. Pri otvorených dverách sú kontakty nezopnuté a obvod je prerušený, pri dverách zatvorených sa kontakty spoja a obvod sa otvorí. Arduino bude tieto stavy spracovávať a vyhodnocovať či sú dvere zatvorené alebo otvorené.

Zobrazovanie stavu na fyzickom zobrazovači.

Pri plánovaní bola zvolená táto funkcionalita pre dôvod kontroly a vyhodnocovania činnosti Arduina na HW úrovni. Displej bude mať za úlohu zobrazovať v reálnom čase stav dverí(otvorené zatvorené). Pri hotovom prototype sa bude jednať o funkčnosť prototypu, ktorá bude zjednodušovať kontrolu stavu dverí pre používateľa systému.

Odosielanie dát na sériovú-linku pre ďalšie spracovávanie v programe.

Odosielanie dát ktoré budú niesť informácie o stave dverí je dôležité pre zaručenie funkčnosti SW časti prototypu. Predispozícia na túto symbiózu je odosielať informácie v takom formáte aby sa ľahko vyhodnocovali v SW.

Spínanie HW signalizácie pri splnení podmienok pre spustenie poplachu.

Spínanie HW signalizácie bude nutné uskutočňovať pomocou relé modulu pretože sirény ktoré poskytujú dostačujúcu výkonnosť sa vymykajú napájacím možnostiam Arduino vývojovej dosky. Relé modul bude spínaný pomocou otvorenia alebo zatvorenia obvodu. Touto činnosťou sa relé modul bude otvárať alebo spínať a tým otvárať alebo zatvárať galvanicky oddelený obvod so sirénou.

Vyhodnotenie

Návrhy na prototypizáciu HW časti by pri správnom nastavení, zapojení a nastavení programu pre Arduino, mali spĺňať všetky podmienky ktoré sú speté so správnou funkčnosťou prototypu.

8.3 Tvorba HW prototypu

V tejto kapitole sa práca zameriava na vytvorenie prototypu podľa bodov vytýčených v kapitole Prototypizácia. Všeobecná metodika vytvárania prototypu vychádzala z poznatkov a skúseností nadobudnutých počas štúdia na strednej a vysokej škole. Počas zapájania

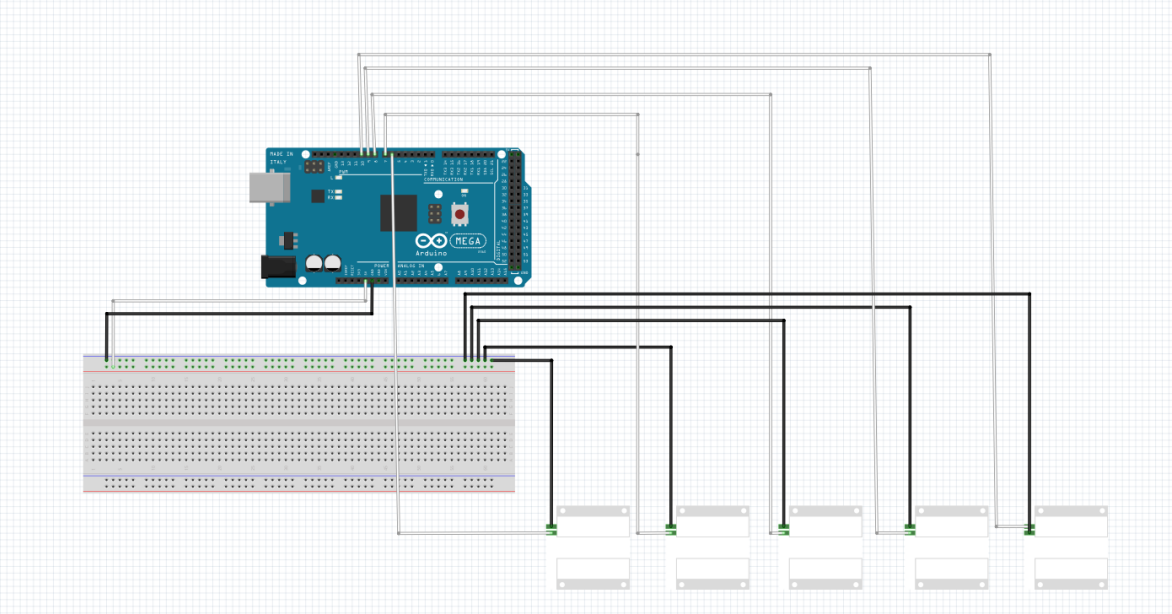
prototypu sa nevyskytli žiadne väčšie ani menšie problémy a vyhotovenie postupovalo podľa plánu.

8.4 Zapojenia prototypu

Zapojenie prototypu bude rozdelené na časti:

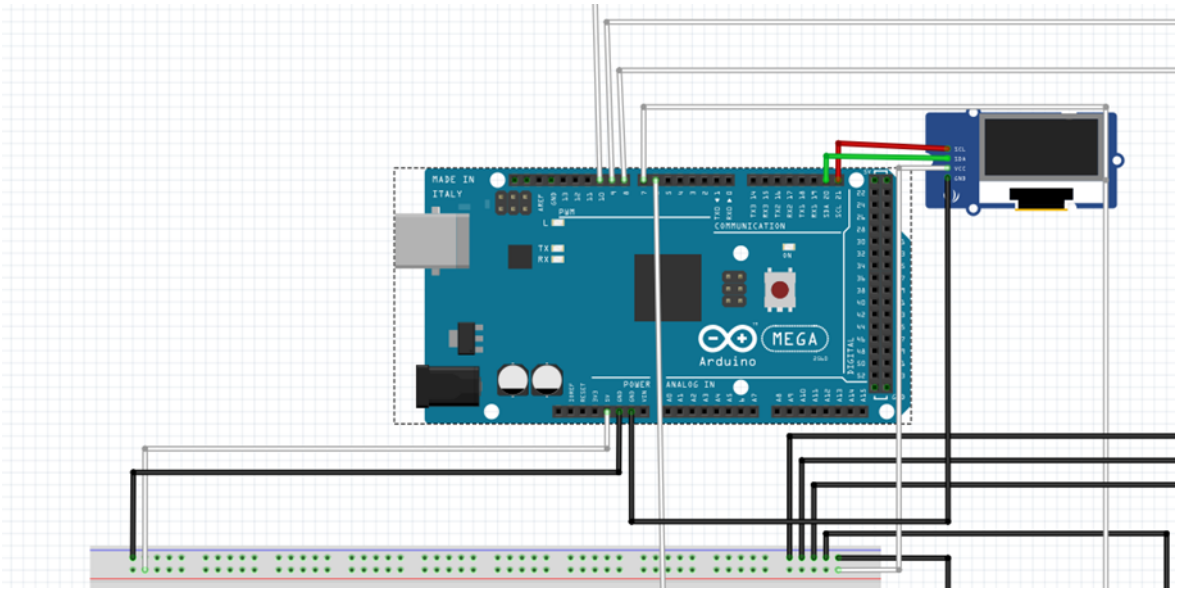
1. snímanie dverí,
2. zobrazovanie stavu dverí na displeji,
3. spínanie výstražného systému,
4. prepojenie so SW častou.

V poslednom zapojení budú všetky segmenty spojené do jedného prehľadného schématu. Bielou farbou sa budú označovať vodiče smerujúce do 5V napájacieho Arduino portu. Čiernou farbou sa budú označovať vodiče smerujúce do GND Arduino portu.



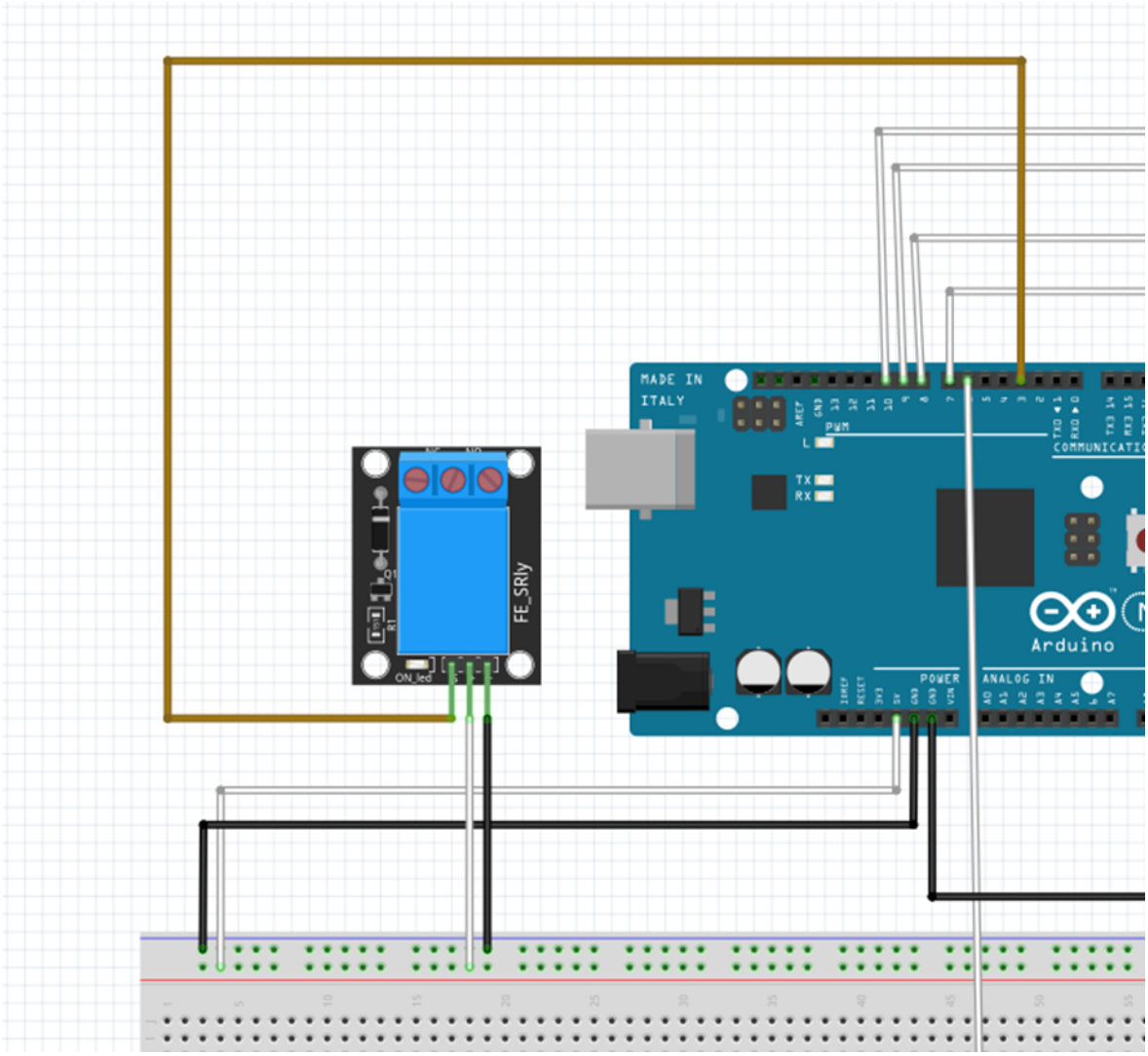
Obrázok 21 Zapojenie magnetických kontaktov[Vlastné]

Schéma zapojenia zobrazuje zapojenie magnetických kontaktov. Sú uzemnené cez GND(uzemňujúci port Arduino vývojovej dosky). Druhý vývod magnetického kontaktu je zapojený v digitálnych portoch 6,7,8,9,10.



Obrázok 22 Zapojenie zobrazovadla[Vlastné]

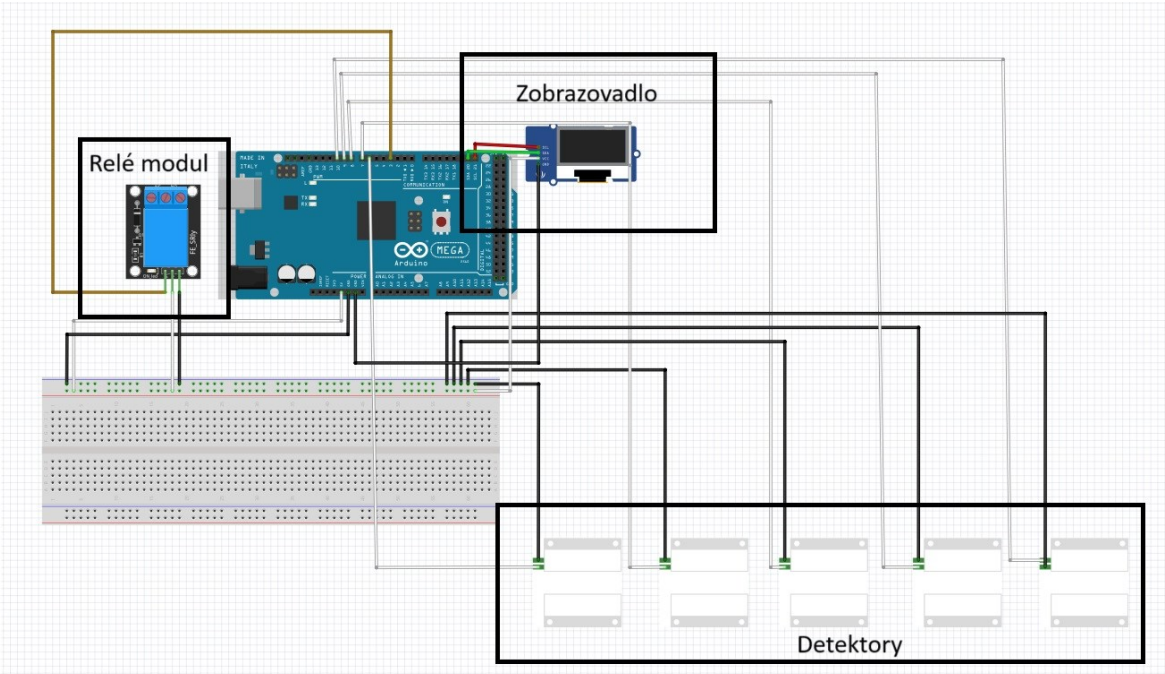
Port VCC je zapojený do 5V portu Arduina. Port GND je zapojený do GND portu Arduina. Port SCL do portu SCL Arduina.Port SDA do portu SDA Arduina. Takéto zapojenie zaručuje fungovanie OLED displeja.



Obrázok 23 Relé modul[Vlastné]

Spínanie výstražného systému zabezpečuje relé modul. Spínací port je pripojený na Digitálny port Arduino vývojovej dosky číslo 3. GND port relé portu je pripojený na GND port Arduina. VCC port relé modulu je pripojený na 5V port Arduino vývojovej dosky.

Prepojenie so SW častou prototypu je zabezpečené dátovým prepojením. Pomocou USB kábla s koncovkami USB-B => USB-A.



Obrázok 24 Celkové zapojenie[Vlastné]

9 TVORBA PROGRAMU PRE HW ČASŤ

Program ktorý bude opísaný zabezpečuje správny chod prototypu z pohľadu funkčnosti. Celý kód bude podrobne opísaný a vysvetlená bude každá jeho funkcionalita

Programovanie Arduino vývojovej dosky

Programovanie prebieha v SW priamo od výrobcov Arduino vývojovej dosky pod názvom:

„Arduino IDE“. IDE je anglická skratka pre výraz “integrated development environment“.

Čiže pre integrované vývojové prostredie. Pre programovanie sa používa jazyk pod názvom C++ s drobnými zmenami a úpravami, zmenami programovacej syntaxe.

Program

Kód bude rozčlenený do úsekov ktoré je možné vyhodnocovať naraz. Kód bol zhotovovaný vlastnoručne s pomocou AI chatu ktorý je priamo integrovaný vo vývojových prostrediach pre programátorov. A jedná sa o bežnú prax pri tvorbe a optimalizácií kódu.

Syntax kódu bude pre lepšiu čitateľnosť vyznačená *kurzívou* a tučným **písmom.**

#include <Wire.h>

#include <Adafruit_GFX.h>

#include <Adafruit_SSD1306.h>

V týchto riadkoch je inicializácia a načítanie knižníc potrebných pre pridanie funkcionality kódu a pre ovládanie OLED zobrazovadla.

Knižnica Wire.h – Je knižnica ktorá umožňuje Arduino vývojovej doske komunikovať cez I2C rozhranie

#define SCREEN_WIDTH 128

#define SCREEN_HEIGHT 64

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

Definovanie výšky a šírky OLED displeja, ktorý je popísaný v technickej dokumentácií. Siedmy riadok inicializuje zadané veľkosti a celý OLED displej.

const int SENSOR_PIN_1 = 6;

const int SENSOR_PIN_2 = 7;

const int SENSOR_PIN_3 = 8;

const int SENSOR_PIN_4 = 9;

const int SENSOR_PIN_5 = 10;

const int RELAY_PIN = 3;

Segment nastavuje mená pre porty číslo 6-10 ktoré budeme používať len pod menami ktoré im boli priradené. Kód by fungoval správne aj bez týchto pridelení mien. Jedná sa len o zjednodušenie postupov a rozlišovaní pinov pri programovaní.

bool lastStates[5] = {false, false, false, false, false};

bool relayActivated = false;

Tieto „tvrdé“ nastavenia false štatútov v Bool premennej zaručuje reštartovanie a prepísanie premenných uložených v pamäti a medzi pamäti Arduina. Zároveň sa jedná o uvedenie stavu senzorov do pôvodného stavu – „zatvorené“

void setup() {

Serial.begin(9600);

pinMode(SENSOR_PIN_1, INPUT_PULLUP);

pinMode(SENSOR_PIN_2, INPUT_PULLUP);

pinMode(SENSOR_PIN_3, INPUT_PULLUP);

pinMode(SENSOR_PIN_4, INPUT_PULLUP);

pinMode(SENSOR_PIN_5, INPUT_PULLUP);

Nastavenie senzorov na INPUT mód ktorý umožňuje snímanie stavu napätia na pine, PULLUP zapína interný rezistor ktorý bráni skratu Arduino vývojovej dosky.

pinMode(RELAY_PIN, OUTPUT);

digitalWrite(RELAY_PIN, LOW);

Relay teda relé pin sa stará o spínanie HW signalizácie otvorených dverí, inicializuje sa ako vypnutý teda vo východnej pozícií.

```
 display.display();
  delay(2000);
  display.clearDisplay();
}
```

Display.display(); kód je zodpovedný za premazania medzi pamäte OLED displeja a jeho inicializáciu. Delay čaká 2 sekundy kým sa display inicializuje. Display.clearDisplay(); premaže všetky informácie aktuálne zobrazené na displeji.

```
void loop() {
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
```

Nastavenie veľkosti písma na displeji, nastavenie farby zobrazovaného textu, a umiestnenie „kurzora“ na počiatok displeja.

```
  if (Serial.available() > 0)
   {
    String command = Serial.readStringUntil('\n');
    if (command == "ROFF")
    {
      digitalWrite(RELAY_PIN, LOW);
      relayActivated = false;  // Reset relay control flag
    }
  }
```

Program ktorý je zodpovedný za čítanie serial-linky ktorá slúži na komunikovanie HW a SW časti prototypu. Pri načítaní kódového slova „ROFF“ ktorá je iniciálovou skratkou pre Relay-OFF vypne relay pin a do kontrolnej premennej Bool relayActivated načíta hodnotu false.

```
//písanie na diplej
 for (int i = 0; i < 5; i++) {
   int pin = SENSOR_PIN_1 + i;
   bool newState = digitalRead(pin);
   display.print("Sensor ");
   display.print(i + 1);
   display.print(": ");
   display.println(newState ? "Open" : "Closed");
```

Funkcia FOR sa stará o sledovanie stavu na senzorových pinoch. Začína pri čísle 1 a končí pri čísle 5.Funkcia ukladá stav pinu do premennej newState ktorá bude snímaná v ďalšej funkcii Zároveň sa funkcia stará o vypisovanie stavu dverí na displej v skrátenej funkcií ktorá pri Bool premennej TRUE bude vypísaná správa „Open“, ak je Bool premenná False bude vypísané „Closed“.

```
if (newState != lastStates[i]) {
    lastStates[i] = newState;  // Update posledný state
    Serial.print(i + 1);  // Sensor číslo
    Serial.println(newState ? "O" : "C");  // 'O' pre otvorený, 'C' pre zatoverný
  }
```

Funkcia IF zisťuje zmeny v premennej newState ak je tento stav iný ako stav predchádzajúci stav tak tento stav uloží ako nový stav, zároveň do serial linky pomocou Bool premennej zašle do serial.linky číslo pinu a „O“ pre senzor dverí ktoré sú otvorené a „C“ pri senzore dverí ktoré sú zatvorené. Príklad správy je 2C takže stav na senzore 2 je Closed – zatvorené.

```
  // zapínanie relay
    if (newState && !relayActivated) {
      digitalWrite(RELAY_PIN, HIGH);
      relayActivated = true;
    }
  }
```

Ďalšia funkcia IF spína relé pri splnení podmienky a tým celú HW signalizáciu. A do premennej Bool relayActivated zapisuje hodnotu true. Zároveň sa funkcia stará o to aby bolo relé zopnuté len raz pri prvom otvorení dverí.

```
  display.display();
  delay(100);
}
```

V závere programu sa medzi pamäť OLED dipleja premaže a inicializuje znova aby sa na display neobjavovali artefakty alebo nesprávne údaje. V poslednom kroku program čaká 100 milisekúnd aby sa dokončili všetky úkony ktoré boli uvedené do činnosti. Funkcia Loop sa po tomto čakaní znova spúšťa od znova. Celý kód je k nahliadnutiu v prílohe 1.

10 TVORBA PROGAMU PRE SW ČASŤ

Program bol tvorený vo vývojovom prostredí Miscrosoft Visual Studio ktorý je voľne prístupný. Počas tvorenia kódu bolo vývojové prostredie prispôsobené používateľovi a boli stiahnuté voľne prístupné rozšírenia napríklad ako: Visual chatGPT Studio. Ktorý za pomoci umelej inteligencii pomáha automaticky optimalizovať kód, jedná sa o bežnú prax pri tvorbe kódu.

Program

V kapitole bude opísaný program ktorý vytvára SW časť bakalárskej práce. Program bude rozdelený do segmentov ktoré na seba nadväzujú a majú spoločnú funkciu alebo sa funkčne podobajú. Cieľom kapitoly je podrobnejšie demonštrovať celú programovaciu syntax.

Pre lepšie pochopenie umiestnení je dôležité uviesť aj celkovú veľkosť okna ktorá je 1579x 1012 pixelov. Táto veľkosť je fixovaná, pri maximalizovaní okna v systéme Windows ostávajú všetky užívateľské rozhrania na konštantne určenom mieste. Voľný priestor získaný maximalizovaním je vyplnený prázdnym priestorom.

Syntax kódu bude pre lepšiu čitateľnosť zvýraznená *kurzívou* a **tučným písmom**.

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.IO;
using System.IO.Ports;
using System.Text;
using System.Windows.Forms;
```

Kód importuje dôležité knižnice pre prácu so základnými prvkami Windows Froms, pre generické kolekcie napríklad ako funkcie Dictionary, prácu s grafikou a vzhľadom, pre prácu so súbormi a sériovými portami, pre prácu s textom a na záver pre prácu a tvorbu Windows Form aplikácií.

```
namespace SW_časť
{
    public partial class Form1 : Form
```

```
{
    TextBox outputTextBox, timestampsTextBox, transitionsTextBox;
    Label outputLabel, timestampsLabel, transitionsLabel;
    Button clearLogsButton, resetDoorsButton, alarmOffButton, saveLogsButton;
    Dictionary<int, Panel> doorPanels;
    Dictionary<int, Label> doorLabels;
    Dictionary<int, string> doorOpenTimestamps;
    Dictionary<int, string> doorCloseTimestamps;
    Dictionary<int, DateTime> lastDoorOpenTimes;
    SerialPort serialPort1;
    int lastDoorOpened = 0;
    DateTime lastTimeOpened = DateTime.MinValue;
```

V aktuálnej časti kódu sa deklarujú funkcie ktoré budú používané v celom projekte. „TextBox“ a „Label“ sú objekty ktoré sa používajú pre zobrazovanie a manipuláciu s textovými poľami a štítkami. „Button“ slúži pre definovanie tlačidiel v aplikácií. „Dictionary“ slúži pre ukladanie premenných z ktorých následne vyplýva mapovanie dverí, grafické reprezntácie a popisy. „SerialPort“ pre prácu so sériovým portom, komunikáciou s HW časťou. „lastDoorOpened“ a „DateTime lastTimeOpened“ pre sledovanie posledných otvorených dverí a času otvorenia.

```
public Form1()
    {
        InitializeComponent();
        InitializeSerialPort();
        SetupOutputTextBox();
        SetupTimestampsTextBox();
        SetupTransitionsTextBox();
        SetupDoorPanels();
        SetupButtons();
```

```
        doorOpenTimestamps = new Dictionary<int, string>();
        doorCloseTimestamps = new Dictionary<int, string>();
        lastDoorOpenTimes = new Dictionary<int, DateTime>();
        InitializeDoorStates();
    }
```

Časť kódu ktorá ma za úlohu správne nastavenie inštalácie, obsahuje inicializačné metódy ktoré majú za úlohu pridanie funkčnosti a komponentov s ktorými následne užívateľ interaguje. „InitializeComponent()“ inicializuje komponenty formulára, „InitializeSerialPort()“ nastavenie sériového portu, „SetupOutputTextBox()“, „SetupTimestampsTextBox()“, „SetupTransitionsTextBox()“, „SetupDoorPanels()“, „SetupButtons()“ sú inicializácie ktoré nastavujú konkrétne prvky s ktorými bude užívateľ fungovať. „doorOpenTimestamps“, „doorCloseTimestamps“, „lastDoorOpenTimes“ slúži na inicializáciu slovníkov do ktorých sa budu následne ukladať časové údaje potrebné pre fungovanie SW časti práce. „InitializeDoorStates()“ inicializuje dvere vo východzej pozícií „zatvorené“.

```
private void SetupDoorPanels()
    {
        doorPanels = new Dictionary<int, Panel>();
        doorLabels = new Dictionary<int, Label>();
        int startingY = transitionsTextBox.Location.Y + transitionsTextBox.Height + 10;
        for (int i = 1; i <= 5; i++)
        {
            Panel doorPanel = new Panel
            {
                Size = new Size(50, 50),
                Location = new Point(10, startingY + 60 * (i - 1)),
                BackColor = Color.Green
            };
```

```
            Label doorLabel = new Label
            {
                Text = $"Door {i}",
                Location = new Point(70, startingY + 60 * (i - 1) + 15),
                Size = new Size(70, 20)
            };
            Controls.Add(doorPanel);
            Controls.Add(doorLabel);
            doorPanels[i] = doorPanel;
            doorLabels[i] = doorLabel;
        }
    }
```

Inicializujú sa doorPanels a doorLabels ako prázdne slovníky. „startingY“ vypočíta počiatočnú vertikálnu polohu pre panely dverí. Cykly pre čísla dverí (od 1 do 5) vytvárajú Panel a Label pre každé dvere, nastavujú ich vlastnosti (veľkosť, umiestnenie, farbu) a pridávajú ich do formulára pomocou Controls.Add.

```
private void SetupOutputTextBox()
    {
        outputTextBox = new TextBox
        {
            Location = new Point(100, 40),
            Size = new Size(300, 200),
            Multiline = true,
            ScrollBars = ScrollBars.Vertical
        };
        outputLabel = new Label
        {
```

```
        Text = "Door Status and Last Open/Close Time:",
        Location = new Point(100, 20),
        Size = new Size(300, 20)
    };
    Controls.Add(outputTextBox);
    Controls.Add(outputLabel);
}
```

Metóda „SetupOutputTextBox“ vytvára a nastavuje „TextBox“ pre výstup stavu dverí a ich posledných otvorení/zatvorení, používa „Label“ na označenie daného textového poľa. „TextBox“ je umiestnený na formulári na pozícii (100, 40 pixelov) s veľkosťou 300x200 pixelov, je viacriadkový a má vertikálny posuvník. Label je umiestnený nad TextBox na pozícii (100, 20 pixelov) s textom označujúcim obsah TextBox vo formáte “ Door Status and Last Open/Close Time:“.

```
private void SetupTimestampsTextBox()
{
    timestampsTextBox = new TextBox
    {
        Location = new Point(450, 40),
        Size = new Size(300, 200),
        Multiline = true,
        ScrollBars = ScrollBars.Vertical
    };
    timestampsLabel = new Label
    {
        Text = "Timestamp Log for Door Events:",
        Location = new Point(450, 20),
        Size = new Size(220, 20)
```

```
        };
        Controls.Add(timestampsTextBox);
        Controls.Add(timestampsLabel);
    }
```

Podobne ako predošlá metóda, SetupTimestampsTextBox vytvára a nastavuje TextBox a Label pre časové značky udalostí dverí. TextBox je umiestnený na pozícii (450, 40 pixelov) s veľkosťou 300x200 pixelov, je viacriadkový a má vertikálny posuvník. Label je umiestnený nad TextBox na pozícii (450, 20) a má text označujúci obsah TextBox.

```
private void SetupTransitionsTextBox()
    {
        transitionsTextBox = new TextBox
        {
            Location = new Point(800, 40),
            Size = new Size(300, 200),
            Multiline = true,
            ScrollBars = ScrollBars.Vertical
        };
        transitionsLabel = new Label
        {
            Text = "Door Transitions Log:",
            Location = new Point(800, 20),
            Size = new Size(200, 20)
        };
        Controls.Add(transitionsTextBox);
        Controls.Add(transitionsLabel);
    }
```

SetupTransitionsTextBox vytvára a nastavuje TextBox a Label pre záznam prechodov dverí. TextBox je umiestnený na pozícii (800, 40 pixelov) s veľkosťou 300x200 pixelov, je viacriadkový a má vertikálny posuvník. Label je umiestnený nad TextBox na pozícii (800, 20) a má text označujúci obsah TextBox.

```
private void SetupButtons()
    {
        clearLogsButton = new Button
        {
            Text = "Clear Logs",
            Location = new Point(450, 260),
            Size = new Size(100, 30)
        };
        clearLogsButton.Click += ClearLogsButton_Click;
        Controls.Add(clearLogsButton);

        resetDoorsButton = new Button
        {
            Text = "Reset Doors",
            Location = new Point(800, 260),
            Size = new Size(100, 30)
        };
        resetDoorsButton.Click += ResetDoorsButton_Click;
        Controls.Add(resetDoorsButton);

        alarmOffButton = new Button
        {
            Text = "Alarm OFF",
```

```
            Location = new Point(650, 260),
            Size = new Size(100, 30)
        };
        alarmOffButton.Click += AlarmOffButton_Click;
        Controls.Add(alarmOffButton);

        saveLogsButton = new Button
        {
            Text = "Save Logs",
            Location = new Point(1000, 260),
            Size = new Size(100, 30)
        };
        saveLogsButton.Click += SaveLogsButton_Click;
        Controls.Add(saveLogsButton);
    }
```

Metóda SetupButtons vytvára a nastavuje štyri tlačidlá:

Tlačidlo „Clear Logs Button“ slúži na vymazanie záznamov a je umiestnené na súradniciach (450, 260 pixelov). Kliknutím na toto tlačidlo sa spustí metóda „ClearLogsButton_Click“.

Tlačidlo „Reset Doors Button“ slúži na resetovanie dverí a je umiestnené na súradniciach (800, 260 pixelov). Kliknutím na toto tlačidlo sa spustí metóda „ResetDoorsButton_Click“.

Tlačidlo „Alarm Off Button“ slúži na vypnutie alarmu a nachádza sa medzi ostatnými dvoma tlačidlami na súradniciach (650, 260 pixelov). Kliknutím na toto tlačidlo sa spustí metóda „AlarmOffButton_Click“.

Tlačidlo „Save Logs Button“ slúži na uloženie záznamov a je umiestnené na súradniciach (1000, 260 pixelov). Kliknutím na toto tlačidlo sa spustí metóda „SaveLogsButton_Click“.

```
private void ClearLogsButton_Click(object sender, EventArgs e)
    {
```

```
    timestampsTextBox.Clear();
    transitionsTextBox.Clear();
}
```

Metóda ClearLogsButton_Click sa vykoná po kliknutí na tlačidlo Clear Logs. Vymaže obsah timestampsTextBox a transitionsTextBox.

```
private void ResetDoorsButton_Click(object sender, EventArgs e)
{
    foreach (var door in doorPanels)
    {
        door.Value.BackColor = Color.Green;
    }
    lastDoorOpened = 0;
    lastTimeOpened = DateTime.MinValue;
    doorOpenTimestamps.Clear();
    doorCloseTimestamps.Clear();
    lastDoorOpenTimes.Clear();
    outputTextBox.Clear();
    timestampsTextBox.Clear();
    transitionsTextBox.Clear();
}
```

Metóda ResetDoorsButton_Click sa vykoná po kliknutí na tlačidlo Reset Doors. Pre každé dvere nastaví farbu panelu na zelenú (zatvorené dvere), vymaže uložené časové údaje a vymaže obsah textových polí.

```
private void AlarmOffButton_Click(object sender, EventArgs e)
{
    if (serialPort1.IsOpen)
    {
```

```
            serialPort1.WriteLine("ROFF");
        }
        else
        {
            MessageBox.Show("Serial port is not connected.");
        }
    }
```

Metóda AlarmOffButton_Click sa vykoná po kliknutí na tlačidlo Alarm OFF. Skontroluje, či je sériový port otvorený, a ak áno, odošle príkaz "ROFF" na vypnutie alarmu. Ak sériový port nie je pripojený, zobrazí správu.

```
    private void SaveLogsButton_Click(object sender, EventArgs e)
    {
        SaveLogsToFile();
    }
```

Metóda SaveLogsButton_Click sa vykoná po kliknutí na tlačidlo Save Logs. Zavolá metódu SaveLogsToFile na uloženie záznamov do súboru.

```
    private void SaveLogsToFile()
    {
        try
        {
            using (SaveFileDialog saveFileDialog = new SaveFileDialog())
            {
                saveFileDialog.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*";
                saveFileDialog.FilterIndex = 1;
                saveFileDialog.RestoreDirectory = true;

                if (saveFileDialog.ShowDialog() == DialogResult.OK)
```

```
            {
                string filePath = saveFileDialog.FileName;

                using (StreamWriter writer = new StreamWriter(filePath))
                {
                    writer.WriteLine("Door Status and Last Open/Close Time:");
                    writer.WriteLine(outputTextBox.Text);
                    writer.WriteLine();

                    writer.WriteLine("Timestamp Log for Door Events:");
                    writer.WriteLine(timestampsTextBox.Text);
                    writer.WriteLine();

                    writer.WriteLine("Door Transitions Log:");
                    writer.WriteLine(transitionsTextBox.Text);
                }

                MessageBox.Show("Logs saved successfully!");
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Error saving logs: {ex.Message}");
    }
}
```

Metóda SaveLogsToFile otvára dialógové okno pre uloženie súboru pomocou SaveFileDialog. Ak používateľ vyberie cestu a súbor, uloží obsah outputTextBox, timestampsTextBox a transitionsTextBox do zvoleného súboru. V prípade chyby zobrazí správu s popisom chyby.

```
private void InitializeDoorStates()
{
    foreach (var door in doorPanels)
    {
        door.Value.BackColor = Color.Green; // Closed door
    }
}
```

Metóda InitializeDoorStates nastaví farbu všetkých panelov dverí na zelenú, čo znamená, že všetky dvere sú zatvorené.

```
private void InitializeSerialPort()
{
    serialPort1 = new SerialPort
    {
        BaudRate = 9600,
        Parity = Parity.None,
        StopBits = StopBits.One,
        DataBits = 8,
        Handshake = Handshake.None,
        PortName = "COM6"
    };

    serialPort1.DataReceived += SerialPort1_DataReceived;
```

```
        try
        {
            serialPort1.Open();
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Error opening serial port: {ex.Message}");
        }
    }
```

Metóda InitializeSerialPort nastaví parametre sériového portu (BaudRate, Parity, StopBits, DataBits, Handshake, PortName) a pripojí metódu SerialPort1_DataReceived na obsluhu udalostí prijatia dát. Pokúsi sa otvoriť sériový port a v prípade chyby zobrazí správu.

```
    private void SerialPort1_DataReceived(object sender, SerialDataReceivedEventArgs
e)
    {
        SerialPort sp = (SerialPort)sender;
        string indata = sp.ReadExisting();
        if (!string.IsNullOrEmpty(indata))
        {
            UpdateDoorStates(indata.Trim());
        }
    }
```

Metóda SerialPort1_DataReceived sa vykoná, keď sú prijaté dáta zo sériového portu. Prečíta prijaté dáta a ak nie sú prázdne, zavolá metódu UpdateDoorStates na aktualizáciu stavu dverí.

```
    private void UpdateDoorStates(string data)
    {
```

```
        if (data.Length >= 2)
        {
            int doorNumber = data[0] - '0';
            char state = data[1];
            if (doorPanels.ContainsKey(doorNumber))
            {
                bool isOpen = state == 'O';
                DateTime now = DateTime.Now;
                string timestamp = now.ToString("HH:mm:ss");
                this.Invoke(new MethodInvoker(delegate
                {
                    doorPanels[doorNumber].BackColor = isOpen ? Color.Red : Color.Green;
                    if (isOpen)
                    {
                        doorOpenTimestamps[doorNumber] = timestamp;
                        if (lastDoorOpened != 0 && lastDoorOpened != doorNumber)
                        {
                            TimeSpan duration = now - lastTimeOpened;
                            transitionsTextBox.AppendText($"Door {lastDoorOpened} to Door
{doorNumber}: {duration.TotalSeconds} seconds\r\n");
                        }
                        lastDoorOpened = doorNumber;
                        lastTimeOpened = now;
                        UpdateTimestampsTextBox(doorNumber, "opened", timestamp);
                    }
                    else
```

```
                {
                    doorCloseTimestamps[doorNumber] = timestamp;
                    UpdateTimestampsTextBox(doorNumber, "closed", timestamp);
                }
                RefreshOutputDisplay();
            }));
        }
    }
}
```

Metóda UpdateDoorStates aktualizuje stav dverí podľa prijatých dát kódovanej správy. Dáta obsahujú číslo dverí a ich stav (otvorené/zatvorené). Ak sa dvere otvoria, aktualizuje čas otvorenia a zaznamená prechod medzi dverami. Ak sa dvere zatvoria, aktualizuje čas zatvorenia. V oboch prípadoch aktualizuje zobrazenie stavu dverí.

```
private void UpdateTimestampsTextBox(int doorNumber, string action, string timestamp)
{
    string text = $"Door {doorNumber} {action} at {timestamp}\r\n";
    if (timestampsTextBox.InvokeRequired)
    {
        timestampsTextBox.Invoke(new MethodInvoker(delegate { timestampsTextBox.AppendText(text); }));
    }
    else
    {
        timestampsTextBox.AppendText(text);
    }
}
```

Metóda UpdateTimestampsTextBox aktualizuje timestampsTextBox novými časovými značkami pre dvere. Ak je potrebné, vykonáva aktualizáciu na hlavnom vlákne pomocou Invoke.

```
private void RefreshOutputDisplay()
{
    StringBuilder displayText = new StringBuilder();
    foreach (var door in doorPanels)
    {
        string status = door.Value.BackColor == Color.Red ? "Open" : "Closed";
        string lastActionTime = door.Value.BackColor == Color.Red ?
            "Last opened at: " + (doorOpenTimestamps.ContainsKey(door.Key) ? doorOpenTimestamps[door.Key] : "N/A") :
            "Last closed at: " + (doorCloseTimestamps.ContainsKey(door.Key) ? doorCloseTimestamps[door.Key] : "N/A");
        displayText.AppendLine($"Door {door.Key}: {status} - {lastActionTime}");
    }

    UpdateUIThread(displayText.ToString());
}
```

Metóda RefreshOutputDisplay prechádza všetky dvere a vytvorí text zobrazujúci aktuálny stav každých dverí a čas poslednej akcie (otvorenie/zatvorenie). Potom zavolá metódu UpdateUIThread na aktualizáciu outputTextBox.

```
private void UpdateUIThread(string text)
{
    if (outputTextBox.InvokeRequired)
    {
        outputTextBox.Invoke(new MethodInvoker(delegate { outputTextBox.Text = text; }));
```

```
        }
        else
        {
            outputTextBox.Text = text;
        }
    }
```

Metóda UpdateUIThread aktualizuje outputTextBox daným textom. Ak je potrebné, vykonáva aktualizáciu na hlavnom vlákne pomocou Invoke. Celý kód je k nahliadnutiu v prílohe 2.

11 TESTOVANIE HW PROTOTYPU

V tejto kapitole práce budú vykonané Testy prototypu, ako časti HW tak aj SW z hľadiska fungovania a funkčnosti, ako celku tak aj ako jednotlivých segmentov prototypu.

Testovanie snímania dverí a zobrazovania správ na OLED displeji.

Snímanie dverí a zobrazovanie správ na displeji je najdôležitejšia funkcia ktorú Arduino vývojová doska vykonáva, pokiaľ nie je toto snímanie presné a nefunguje tak aký je predpoklad, nie je možné pokračovať zo žiadnej z ďalších funkcii ktoré má systém vykonávať.

Postup testovania.

1. Systém sa oživí pripojením USB-A-B zbernice k počítaču
2. Vyčká sa na pripravenosť(po inicializácií OLED displeja)
3. Skúšobným kontaktom sa vykonáva skúška magnetických kontaktov
    a. Jednotlivo
    b. Po dvojiciach
    c. V náhodnom poradí
    d. V náhodnom poradí a intervaloch
    e. Zapnutie systému s obsadenými kontaktmi(simulácia zatvorenia všetkých dverí pri zapnutí systému)

Predpoklad.

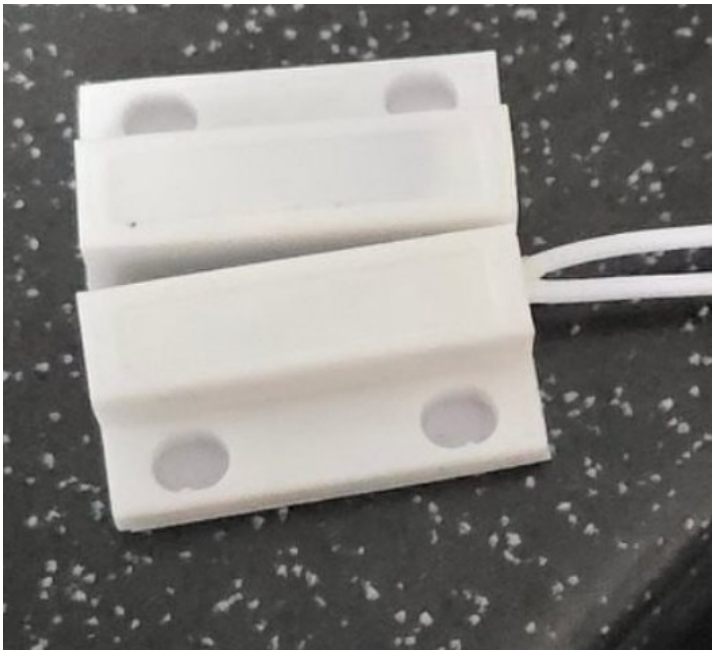
Segment systému by mal na skúšobný kontakt reagovať bez žiadneho oneskorenia, z chovania magnetických kontaktov nie je možné v aktuálnom zložení prototypu pozorovať zmeny. Vykonávane testovanie musí byť pozorované na OLED displeji ktorý zobrazuje stav magnetických kontaktov. Pri priložení skúšobného kontaktu k magnetickému kontaktu č.x sa na OLED displeji zobrazí správa: Sensor x: Closed.

Výsledné správanie segmentu.

Výsledné spávanie segmentu je nasledovné.

Jednotlivo.

Pri jednotlivom prikladaní skúšobného kontaktu k magnetickým kontaktom sa systém správal podľa očakávania a pri priložení ku kontaktom 1-5 sa na OLED displeji zobrazovali správy správne priradené ku kontaktom. Pre ilustráciu je možné na Obrázok 26 LED vidieť správanie sa pri priložení ku kontaktu č. 4.



Obrázok 25 Magnetické kontakty[Vlastné]

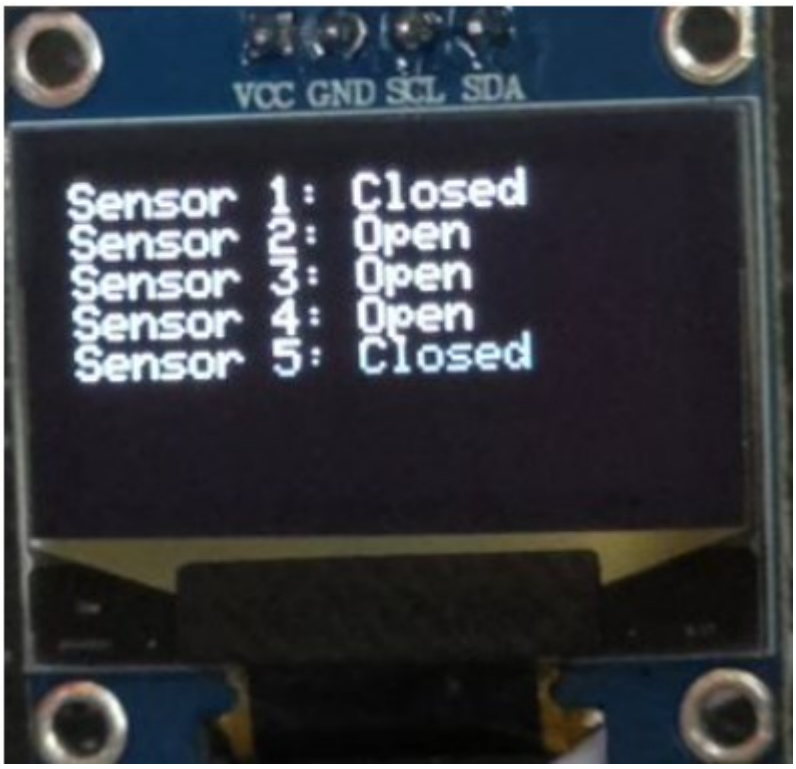
Obrázok 26 OLED displej[Vlastné]

Test prebehol úspešne.

Po dvojiciach.

Pri možnosti prikladania kontaktoch po dvojiciach sa systém čakal podľa predpokladu a správne. Pri priložení skúšobného kontaktu na magnetické kontakty č.1 a č.5 sa na OLED displeji zobrazili správy Sensor 5: Closed a Senzor 1:Closed.

Obrázok 27 Stav magnetických senzorov[Vlastné]



Obrázok 28 Stav OLED displeja[Vlastné]

Test prebehol úspešne.

V náhodnom poradí.

Pri náhodnom prikladaní skúšobného kontaktu k magnetickým kontaktom sa systém správal podľa očakávania. Reakčná doba systému na zmenu stavu magnetických kontaktov, bola nízka až nemerateľná a pre vonkajšieho pozorovateľa nulová.

Pre charakteristiku testu nie je možné priložiť fotodokumentáciu ktorá by správne vyobrazovala výsledky a postup testu.

Test prebehol úspešne

V náhodnom poradí a intervale.

Počas testovania prototypu a prikladaní skúšobného magnetického kontaktu v náhodnom poradí a intervale sa systém správal podľa očakávaní, s veľmi nízkou reakčnou dobou. Pri náhodnom priložení ku kontaktu č.2 zároveň priložení kontaktu č.5 a č.3 odobraní skúšobného kontaktu z magnetického kontaktu č.2 a č.5 a opätovným priložením kontaktu

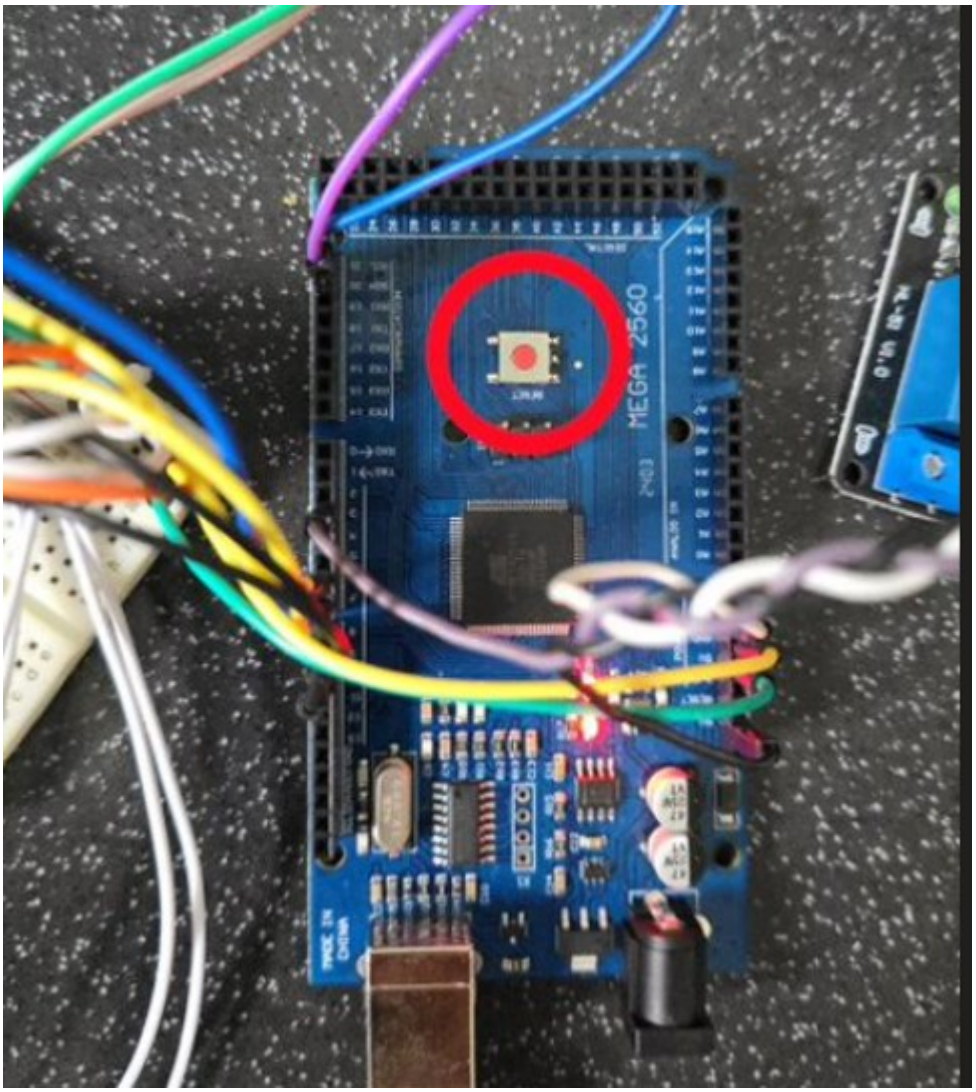
č.2 a zároveň odobratím kontaktu č.3 systém zobrazoval skutočné stavy v súlade s pozorovanou skutočnosťou.

Pre charakteristiku testu nie je možné priložiť fotodokumentáciu ktorá by správne vyobrazovala výsledky a postup testu.

Test prebehol úspešne

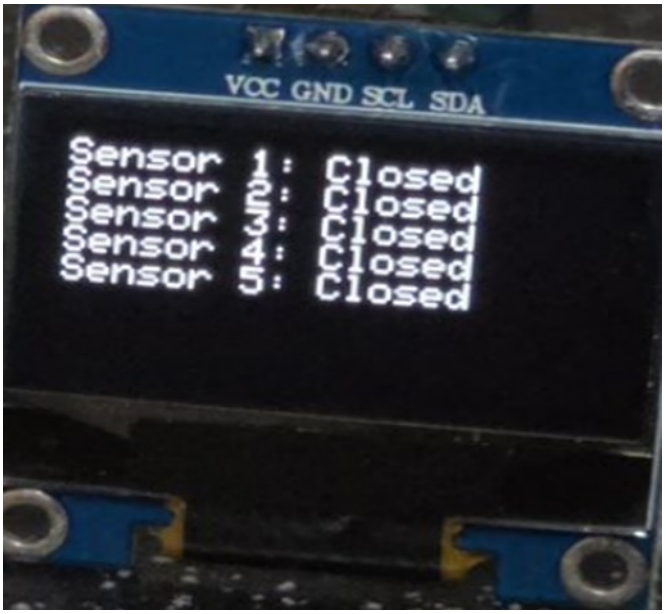
Zapnutie systému s obsadenými kontaktmi. (simulácia zatvorenia všetkých dverí pri zapnutí systému)

Posledný test ktorý bol v tejto kapitole vykonaný je oživenie systému v stave zatvorenia všetkých dverí. Boli obsadené všetky magnetické kontakty skúšobnými kontaktmi a uviedli Arduino vývojovú dosku do stavu reštartu stlačením tlačidla označeným na Obrázok 29 Tlačidlo resetu.



Obrázok 29 Tlačidlo resetu[Vlastné]

Po úspešnom resetovaní sa systém ustálil v očakávanom stave. Ktorý je možné k nahliadnutiu na Obrázok 30 Stav OLED displeja.



Obrázok 30 Stav OLED displeja[Vlastné]

Test prebehol úspešne.

Testovanie spínania relé.

Spínanie relé modulu slúži na spínanie externej HW signalizácie. Ktorá sa skladá z akustickej sirény a LED majáku.

Predpoklad

Systém je nastavený tak aby relé zoplo len raz behom celého cyklu bez zásahu SW časti a to, pri prvom otvorení ktorýchkoľvek dverí(magnetický senzor bude v stave OPEN).

Postup

1. Pred oživením systému budú priložené všetky kontakty osadené skúšobnými kontaktmi
2. Systém sa oživí pripojením USB-A-B zbernice k počítaču
3. Vyčká sa na pripravenosť(po inicializácií OLED displeja)
4. Odoberie sa ľubovoľný skúšobný kontakt
5. Relé svojím zopnutím vydá charakteristický zvuk a kontrolná LED dióda ktorá svieti červenou farbou, zhasne.

Výsledné správanie segmentu.

Stav pred oživením segmentu bol nastavený. Počas oživovania systému a pri načítaní zdrojového kódu sa relé modul viacero krát zopol a rozopol z dôvodu napájania relé modulu z 5V pinu Arduina, ktorá sa pri inicializácií striedavo zapína a vypína z dôvodu inicializácie, no po úspešnej inicializácií sa relé modul ustálil v očakávanom stave, ktorý je možný vidieť na Obrázok 31 Stav relé modulu pred zopnutím.

Obrázok 31 Stav relé modulu pred zopnutím[Vlastné]

Po inicializácií bol odobraný náhodný skúšobný kontakt a relé modul bol zopnutý. Stav relé modulu je možné vidieť na:

Obrázok 32 Stav relé modulu po zopnutí[Vlastné]

Test prebehol úspešne, správanie nebolo ale plne predvídateľné

Z dôvodu plnej funkčnosti podľa zamýšľaného správanie, jediný aspekt, ktorý bol nepredvídateľné bola inicializácia Arduino vývojovej dosky ktorú nie je možné nastaviť.

Možnosti na odstránenie spínania behom inicializácie.

Pre odstránenie spínania relé modulu by bolo možné relé modul napájať samostatným 5V zdrojom. Toto riešenie by predchádzalo spínaniu relé modulu no bola by narušená fyzická kompaktnosť systému.

Autor bakalárskej práce a strojca prototypu hodnotí správanie systému za dostačujúce a neboli podniknuté žiadne kroky k odstráneniu tejto problematiky.

Strojca toto správanie považuje za vyhovujúce z dôvodu zachovania malých fyzických rozmerov. Zároveň je predpokladané že, pri prevádzke prototypu neprichádzalo k resetovaniu celého prototypu, z dôvodu kvalitného vyhotovenia a naprogramovania všetkých častí. Pri alternatíve kedy prototyp môže byť po úprave používaný ako učebná pomôcka alebo pri dni otvorených dverí taktiež nijak neznehodnocuje celkovú hodnotu a prínos výrobku.

Testovanie zasielania správ do sériového portu.

Zasielanie kódovaných správ ma za účel komunikáciu so SW časťou bakalárskej práce. Arduino vývojová doska ma odosielať kódované správy pri činnosti magnetických kontaktov. Správa obsahuje číslo senzoru a aktuálny stav. Príklad správy je uvedený v Kapitole Program. Táto kapitola nie je priradená ku kapitole testovanie SW časti z dôvodu testovanie HW so SW. Takže sa z pohľadu bakalárskej práce sa jedná o testovanie HW časti.

Predpoklad

Predpoklad správania systému je nasledovný pri zmene stavu na magnetických kontaktoch bude odoslaná kódovaná správa ktorá bude obsahovať číslo senzoru a stav v ktorom sa po zmene nachádza.

Postup

1. Systém sa oživí pripojením USB-A-B zbernice k počítaču
2. Vyčká sa na pripravenosť(po inicializácií OLED displeja)
3. Pripojenie všetkých skúšobných kontaktov na magnetické kontakty
4. Sledovanie Sériovej linky

Výsledné správanie

Po pripojení všetkých skúšobných kontaktov bol výsledok ktorý bol odoslaný na sériovú linku nasledujúci.

Obrázok 33 Stav sériovej linky po pripojení kontaktov[Vlastné]

Po opakovanom priložení všetkých skúšobných kontaktov bol stav na sériovej linke nasledovný.

Obrázok 34 Stav sériovej linky po odpojení kontaktov[Vlastné]

Test prebehol úspešne.

Testovanie spracovania príkazov zo SW časti

SW časť odosiela správy na ovládanie relé modulu pomocou kódovanej správy, celkovú funkciu je možné nájsť v kapitole Program. Kapitola je umiestnená v testovaní HW pretože je testovanie sústredenie na **správanie HW časti** pri prijatí SW správy z ideálne pracujúceho programu.

Predpoklad

Po zopnutí relé bude stlačené SW tlačidlo pod názvom „Alarm OFF“ ktoré relé modul pomocou kódovanej správy vo formáte „ROFF“ rozopne relé modul.

Postup

1. Systém sa oživí pripojením USB-A-B zbernice k počítaču
2. Vyčká sa na pripravenosť(po inicializácií OLED displeja)
3. Z ľubovoľného kontaktu sa odstráni skúšobný magnetický kontakt
4. Zopnutie relé modulu
5. Stlačenie SW tlačidla pod názvom „Alarm OFF“
6. Kontrola stavu relé modulu

Výsledné správanie

Po oživení a inicializácií systému bol odstránený ľubovoľný magnetický kontakt, zvolený bol kontakt č.4. Relé modul sa podľa očakávania zopol. V SW časti bolo stlačené SW tlačidlo pod názvom „Alarm OFF“. Relé modul sa podľa predpokladu rozopol.

Z technických dôvodov nie je možné priložiť fotku stavu seriál linky. Technické obmedzenia obmedzujú obsluhu sériovej linky z dvoch SW inštancii.

Sériová linka ponúka otvorenie len jednej inštancie zobrazovania, to znamená že ju môže obsluhovať iba jedna inštancia. Jedna a jediná inštancia, ktorá linku obsluhovala bola SW časť komunikovaná s linkou pomocou príkazu alarm vypla. Z tohto dôvodu je možné potvrdiť správnu komunikácia medzi HW a SW časťou.

Test prebehol úspešne.

12 UKÁŽKA SW ČASTI A TESTOVANIE SW ČASTI

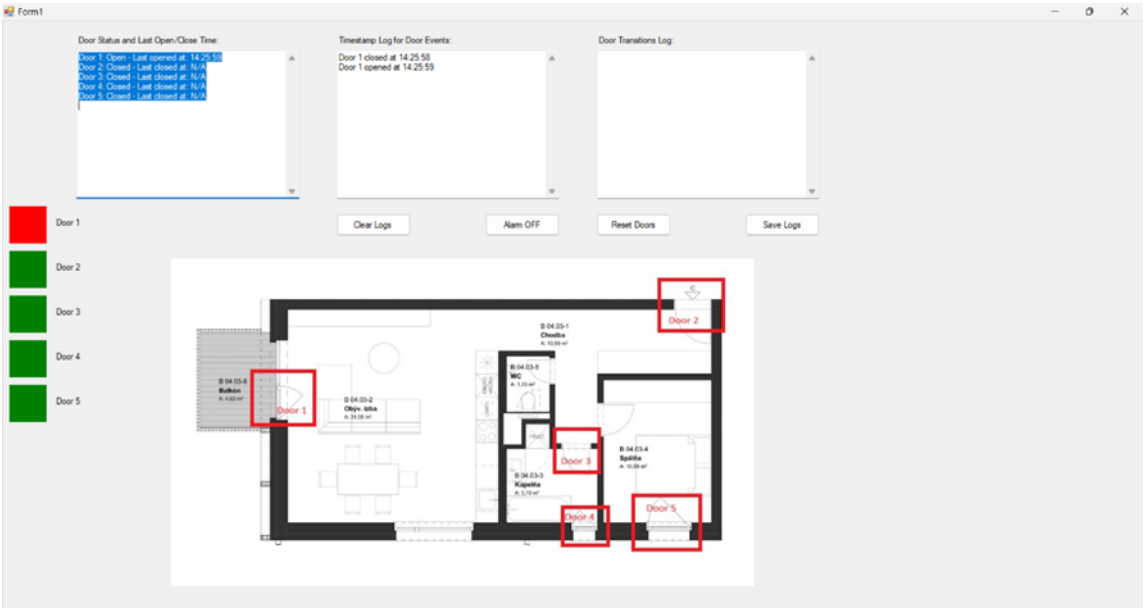
V danej kapitole bude opísane obecné zoznámenie sa s aplikáciou SW časti, kapitola sa zameria na ukážky činnosti SW časti ktorá je prepojená s časťou HW. Zároveň kapitola spĺňa obsahom aj testovaciu stránku SW časti bakalárskej práce.

Inicializácia aplikácie

Aplikáciu je možné inicializovať spustením aplikácie. Pred spustením aplikácie je možné pripojiť Arduino vývojovú dosku k počítaču v ktorom sa bude aplikácia spúšťať pomocou USB kábla. Po spustení aplikácie je možné vidieť nasledujúce okno:

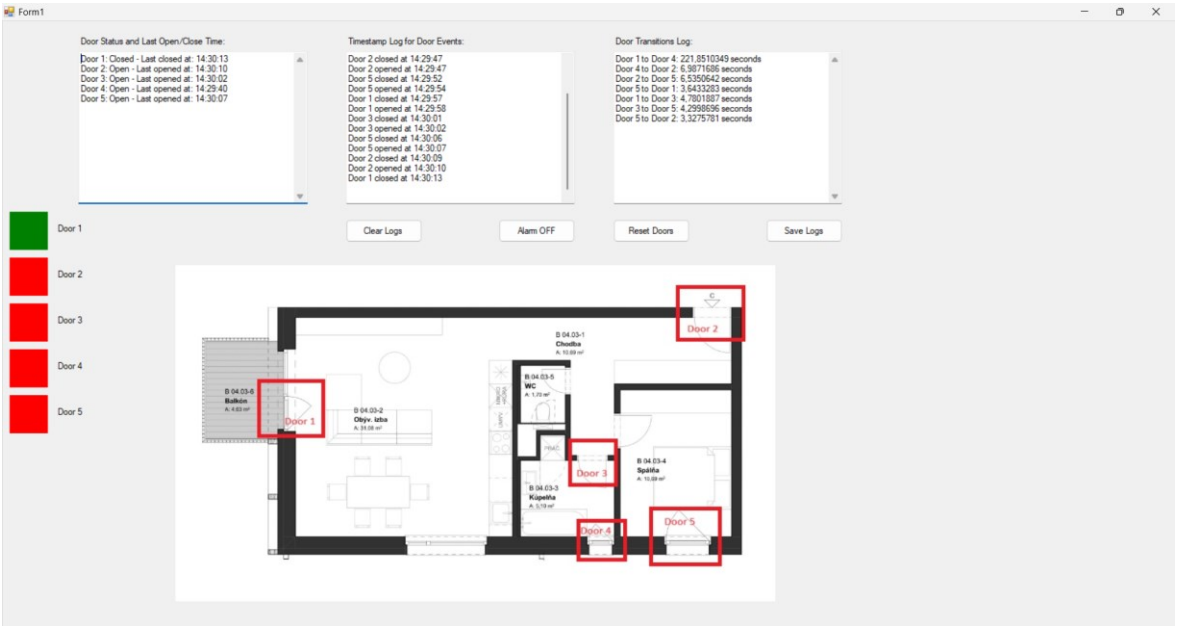
Obrázok 35 Stav po otvorení aplikácie[Vlastné]

Pri prvom otvorení ľubovolnych dverí sa stav aplikácie zmení následovne:



Obrázok 36 Stav po prvom otvorení dverí[Vlastné]

Je možné si všimnúť ze v tabuľke „Door Status and Last Open/Close Time“ prišlo k zmene vo forme vyplnenenia informácií podľa činnosti magnetických kontaktov. Červená farba značí že su dvere otovrené, zelená farba značí že dvere sú v otvorenom stave. Pre lepšiu demonštráciu budú dvere otvárané a zatvárané v náhodnom poradí.

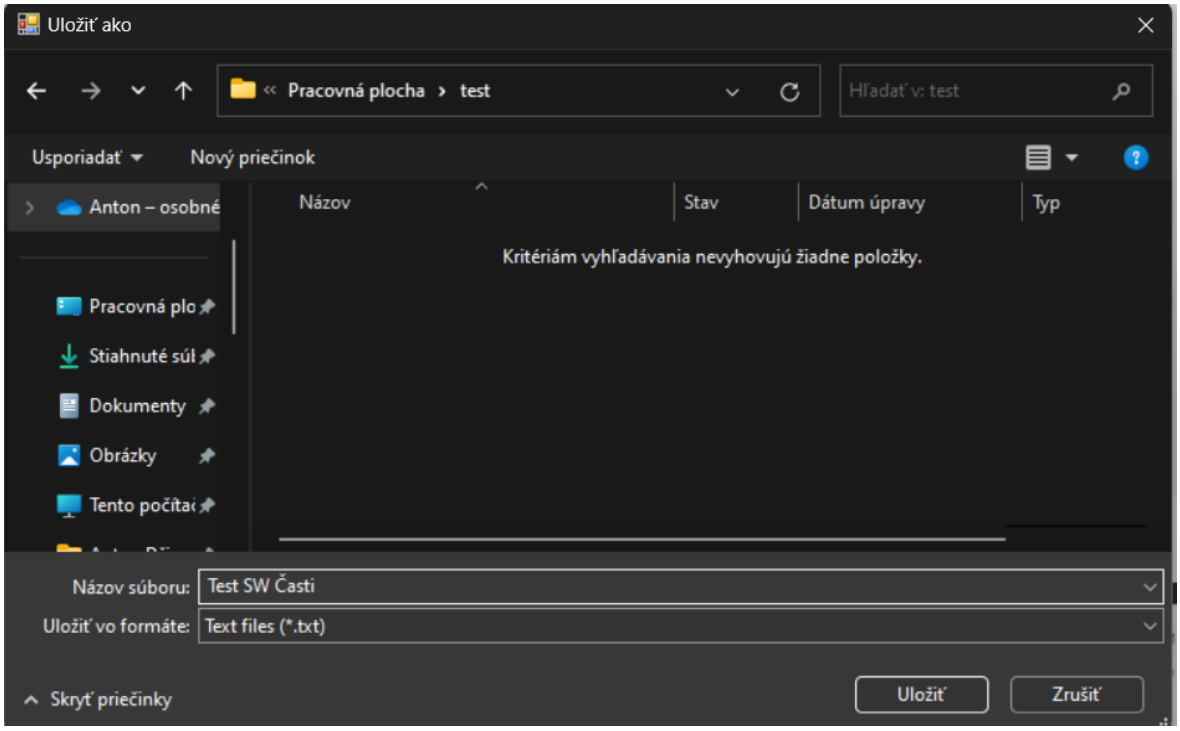


Obrázok 37 Demonštrácia[Vlastné]

Po naplnení všetkých tabuliek dátami. Je možné pokračovať v ukážke SW časti programu.

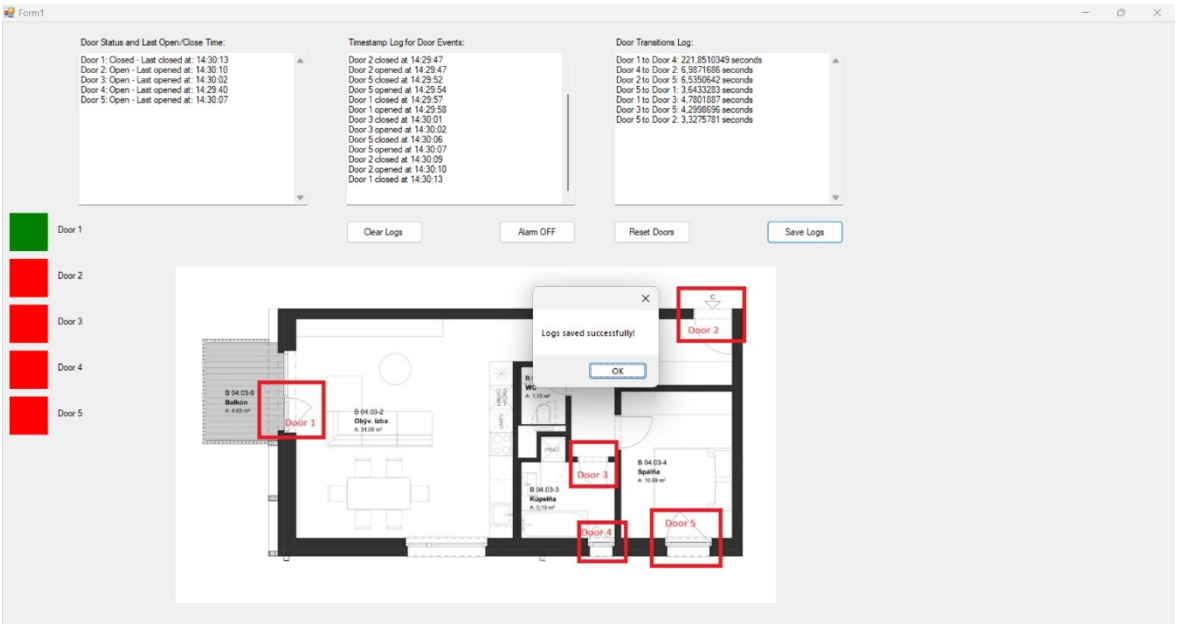
12.1 Save Logs

Aplikácia má funkčnosť uložiť všetky uložené dáta do textového súboru. Proces ukladania je nasledovný: Stlačenie SW tlačidla „Save Logs“ po stlačení tlačidla je zobrazené dialógové okno.



Obrázok 38 Dialógové okno uloženia dát[Vlastné]

Ďalším krokom je uloženie dát. Stlačenie tlačidla „Uložiť“. Po stlačení tlačidla. Je stav nasledujúci.



Obrázok 39 Stav po uložení[Vlastné]

Je zobrazené dialógové okno, ktoré udeľuje informáciu o úspešnom uložení textového súboru. Výsledný textový súbor má nasledujúcu štruktúru.

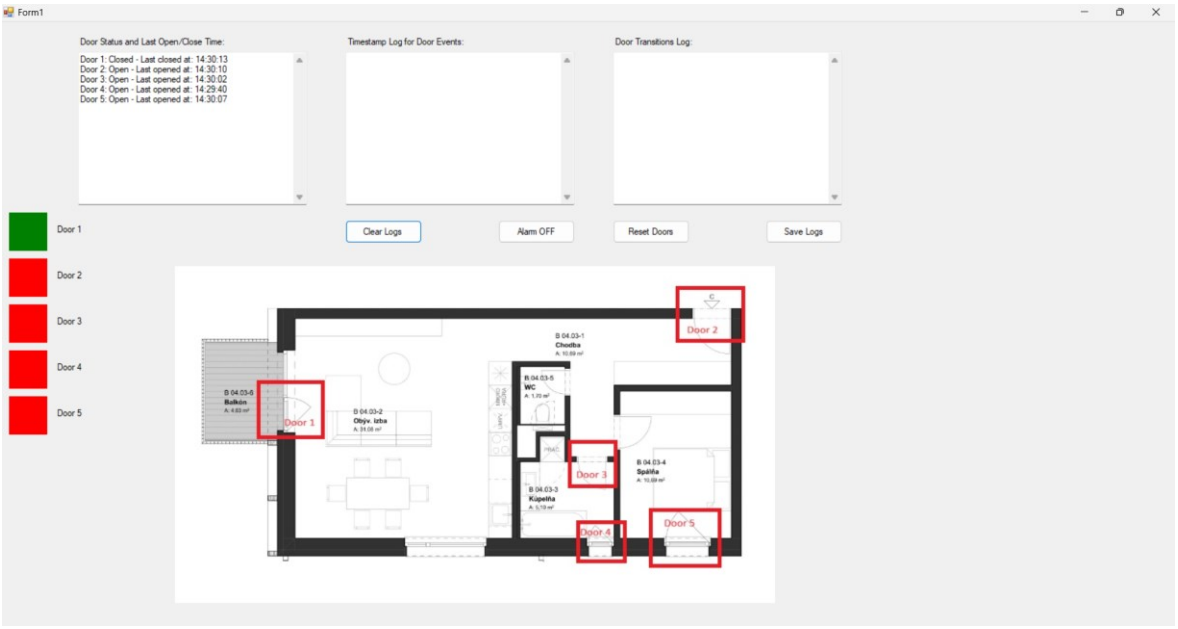
File Edit View

```
Door Status and Last Open/Close Time:
Door 1: Closed - Last closed at: 14:30:13
Door 2: Open - Last opened at: 14:30:10
Door 3: Open - Last opened at: 14:30:02
Door 4: Open - Last opened at: 14:29:40
Door 5: Open - Last opened at: 14:30:07


Timestamp Log for Door Events:
Door 1 closed at 14:25:58
Door 1 opened at 14:25:59
Door 4 closed at 14:29:39
Door 4 opened at 14:29:40
Door 2 closed at 14:29:47
Door 2 opened at 14:29:47
Door 5 closed at 14:29:52
Door 5 opened at 14:29:54
Door 1 closed at 14:29:57
Door 1 opened at 14:29:58
Door 3 closed at 14:30:01
Door 3 opened at 14:30:02
Door 5 closed at 14:30:06
Door 5 opened at 14:30:07
Door 2 closed at 14:30:09
Door 2 opened at 14:30:10
Door 1 closed at 14:30:13


Door Transitions Log:
Door 1 to Door 4: 221,8510349 seconds
Door 4 to Door 2: 6,9871686 seconds
Door 2 to Door 5: 6,5350642 seconds
Door 5 to Door 1: 3,6433283 seconds
Door 1 to Door 3: 4,7801887 seconds
Door 3 to Door 5: 4,2998696 seconds
Door 5 to Door 2: 3,3275781 seconds
```

Obrázok 40 Textový súbor[Vlastné]

12.2 Clear Logs

Funkciou programu „Clear Logs“ je možné premazať všetky zobrazované údaje o dverách, ktorými SW časť disponuje. Po stlačení tlačidla je stav nasledujúci.



Obrázok 41 Stav po stlačení tlačidla Clear Logs[Vlastné]

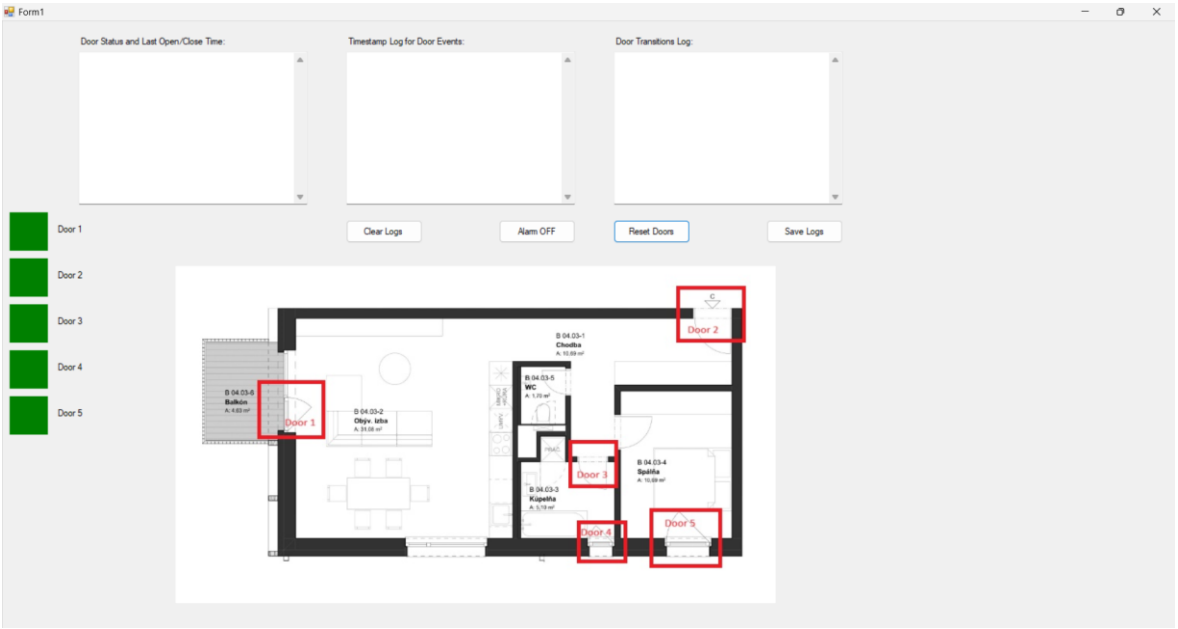
Je možné všimnúť si že stavy dverí v grafickej ale aj písomnej forme ostali neporušené, no dátové logy mali všetky svoje dáta premazané.

12.3 Alarm OFF

Testovanie spracovania príkazov zo SW časti. Pre uskutočnenie tejto akcie je potrebné stlačenie SW tlačidla.

12.4 Reset Doors

Funkcia SW časti Reset Doors má za úlohu kompletný reset systému a premazanie všetkých dát. Po stlačení tlačidla je stav nasledujúci.



Obrázok 42 Stav po stlačení tlačidla Reset Doors[Vlastné]

ZÁVĚR

V tejto bakalárskej práci bol skúmaný a vyvinutý systém monitorovania a riadenia bezpečnostných procesov s dôrazom na plášťovú ochranu v komerčnom prostredí. V teoretickej časti boli analyzované aktuálne systémy a technológie používané v oblasti komerčnej bezpečnosti, vrátane mechanických zabezpečovacích systémov, poplachových zabezpečovacích a tiesňových systémov a integrácie technológie Internet of Things (IoT).V praktickej časti bol vyvinutý prototyp bezpečnostného systému, ktorý zahŕňal hardvérové komponenty ako Arduino vývojovú dosku, OLED zobrazovadlo, relé modul a magnetické kontakty. Cieľom bolo vytvoriť funkčný prototyp schopný monitorovať stav dverí, vizualizovať tento stav a spínať alarm v prípade neautorizovaného prístupu. Testovanie prototypu preukázalo jeho schopnosť efektívne monitorovať stav dverí a signalizovať narušenie. Z výsledkov práce vyplýva, že využitie moderných technológií a integrácia IoT môže významne zlepšiť efektivitu a spoľahlivosť bezpečnostných systémov. Práca zároveň úspešne identifikovala niekoľko oblastí, kde je potrebné ďalšie výskumy a zlepšenia, napríklad v oblasti zabezpečenia komunikácie medzi jednotlivými komponentmi systému a zvýšenia odolnosti voči sabotáži. Verí sa, že výsledky tejto práce poskytnú cenné poznatky pre ďalší vývoj a implementáciu pokročilých bezpečnostných systémov v komerčnom prostredí. Práca taktiež poukazuje na dôležitosť neustáleho zlepšovanie a aktualizácie bezpečnostných systémov, aby boli schopné čeliť novým výzvam a hrozbám.

SEZNAM POUŽITÉ LITERATURY

[1] Arduino.cc [online]. Italy: © 2023 Arduino, 2023 [cit. 2023-05-06]. Dostupné z: https://www.arduino.cc

[2] OLED zobrazovadlo. Online. Dratek. 2024. Dostupné z: https://dratek.cz/docs/produkty/0/114/1487765029.pdf. [cit. 2024-05-18].

[3] Relé modul. Online. Dratek. 2024. Dostupné z: https://dratek.cz/arduino/2954-modul-rele-5v-1-kanal-opticky-oddeleno.html. [cit. 2024-05-18].

[4] Nespájkové pole. Online. Dratek. 2024. Dostupné z: https://dratek.cz/arduino/1226-eses-nepajive-pole-400-pinu.html. [cit. 2024-05-18].

[5] Jazýčkový magnetický kontakt. Online. Dratek. 2024. Dostupné z: https://dratek.cz/arduino/7700-jazyckovy-magnetický-kontakt.html?gad_source=1&gclid=CjwKCAjwo6GyBhBwEiwAzQTmc4hL8JHJ3zkfOoxNOGZCRlK1s4U60XOUEVZximqwqL3PTZ5siNiFQxoCFyIQAvD_BwE. [cit. 2024-05-18].

[6] Kabeláž. Online. Dratek. 2024. Dostupné z: https://dratek.cz/arduino/121747-40-x-m-f-dupont-kabel-40-cm.html?gad_source=1&gclid=CjwKCAjwo6GyBhBwEiwAzQTmczcMtLfLcpNLS1pvK95zdYAr8e1kfrw8p0Pyw4F4u_OXwFgCTlv5ZhoC8y0QAvD_BwE. [cit. 2024-05-18].

[7] AlzaPower LinkCore USB-A to USB-B 1m černý. Online. Alza. 2024. Dostupné z: https://www.alza.cz//alzapower-linkcore-usb-a-b-1m-cerny-d7245511.htm?kampan=adwacc_prislusenstvi-pro-mt_pla_all_ad-top_ad-top_c_1003744___APWCB020a_694226775811_~165758106688~&gclid=CjwKCAj-wo6GyBhBwEiwAzQTmc4W2ow0eRqOwmPfDmpAiyEikWxEslVNYserJ3YOVPXfRJshtuql0FRoCpiMQAvD_BwE. [cit. 2024-05-18].

[8] Poplachová siréna S2 - 6-14V 105 dB 6 tónů. Online. Botland. 2024. Dostupné z: https://botland.cz/poplachove-sireny/3040-poplachova-sirena-s2-6-14v-105-db-6-tonu-5900804011804.html. [cit. 2024-05-18].

[9] Blikající žárovka HC-05 - LED 12V - oranžová. Online. Botland. 2024. Dostupné z: https://botland.cz/led-signalni-lampy/8714-blikajici-zarovka-hc-05-led-12v-oranzova-5900804081241.html. [cit. 2024-05-18].

[10] Fritzing. Online. Fritzing. 2024. Dostupné z: https://fritzing.org/. [cit. 2024-05-22].

[11] Súlad výrobkov s právnymi predpismi. Online. Europa.eu. 2024. Dostupné z: https://europa.eu/youreurope/business/product-requirements/compliance/index_sk.htm. [cit. 2024-05-22].

[12] ČSN EN 50131: Poplachové systémy - Poplachové zabezpečovací a tísňové systémy. Praha: Český normalizační institut, 2015.

[13] Projektování integrovaných systémů. Druhé vydání. Online, Skriptá, vedoucí Valouch, Jan. Zlín: Univerzita Tomáše Bati, Fakulta aplikované informatiky, 2015. Dostupné z: https://digilib.k.utb.cz/handle/10563/18616. [cit. 2023-11-13]

[14] LUKÁŠ, Luděk. Bezpečnostní technologie, systémy a management. 1. vydání. Zlín: Radim Bačuvčík - VeRBuM, 2013. ISBN 978-80-87500-35-4.

[15] KAMENÍK, Jiří a BRABEC, František. Komerční bezpečnost. 2. vydání. Praha: Wolters Kluwer, 2019. ISBN 978-80-7598-303-9.

[16] RADA, Vojtěch. Poplachové zabezpečovací a tísňové systémy. Bakalárska práca. Zlín: Univerzita Tomáše Bati, Fakulta aplikované informatiky, 2021.

[17] KOVÁŘ, Stanislav. Systemizace bezpečnosti. Online, Prezentácia. 2021.

[18] MILLER, James D. Effects of noise on people. The Journal of the Acoustical So-ciety of America, 1974, 56.3: 729-764.

[19] MUCCO. SNT - B720 SERİSİ / SERIES MODEL: B710 - B720, SNT-B710-B720 Data sheet. 2020. Istambul.

[20] Příklady hladin hluku + kdy je potřeba chránit sluch? Online. Earplugs. 2019. Dostupné z: https://www.earplugs.cz/tezka-veda/priklady-hladin-hluku/. [cit. 2024-05-22].

[21] Amenadisplays. Online. 2024. Dostupné z: http://amenadisplays.com/. [cit. 2024-05-18].

[22] OBHEROI, Kunwardeep Singh, et al. Economical home monitoring system using IOT. In: Proceedings of the Second International Conference on Computational Intelligence and Informatics: ICCII 2017. Springer Singapore, 2018. p. 627-637.

[23] LUNA, José Ignacio Vega; SÁNCHEZ-RANGEL, Francisco Javier; COSME-ACEVES, José Francisco. Monitoring System for Doors and Windows of a Data Center with IoT. Ingenius, 2019, 22: 72.

[24] Co to je IoT? Online. IoTPORT. 2019. Dostupné z: https://www.iotport.cz/iot-novinky/ostatni-clanky-o-iot/co-to-je-iot. [cit. 2024-05-18].

[25] DRGA, Rudolf. TECHNICKÉ PROSTŘEDKY BEZPEČNOSTNÍHO PRŮMYSLU Vstupní zařízení – detektory. 1. 2013.

[26] DRGA, Rudolf. TECHNICKÉ PROSTŘEDKY BEZPEČNOSTNÍHO PRŮMYSLU Elektromechanické detektory. 1. 2013.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

MZS	Mechanické zábranné systémy
PZTS	Poplachové Zabezpečovacie a Tiesňové systémy
IoT	Internet of Things (Slovenský preklad: Internet vecí)
SW	Software (program)
HW	Hardware
GND	Uzemnenie Arduino vývojovej dosky
OLED	Organic light emiting diode
LED	Light emmiting diode (svetlo vyžarujúca dióda)
I^2C	Inter-Integrated Circuit
USB	Universal Serial Bus (Univerzálna sériová zbernica)
PIR	Pasívne infračervené čidlo
LDR	Svetlo citlivý rezistor
DC	Direct current (jednosmerný prúd)
SMS	Služba krátkych textových správ

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: ARDUINO KÓD

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire);

const int SENSOR_PIN_1 = 6;
const int SENSOR_PIN_2 = 7;
const int SENSOR_PIN_3 = 8;
const int SENSOR_PIN_4 = 9;
const int SENSOR_PIN_5 = 10;

const int RELAY_PIN = 3;

bool lastStates[5] = {false, false, false, false, false};
bool relayActivated = false;

void setup() {
  Serial.begin(9600);

  pinMode(SENSOR_PIN_1, INPUT_PULLUP);
  pinMode(SENSOR_PIN_2, INPUT_PULLUP);
  pinMode(SENSOR_PIN_3, INPUT_PULLUP);
  pinMode(SENSOR_PIN_4, INPUT_PULLUP);
  pinMode(SENSOR_PIN_5, INPUT_PULLUP);

  pinMode(RELAY_PIN, OUTPUT);
  digitalWrite(RELAY_PIN, LOW);

  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3C for
128x64
    Serial.println(F("SSD1306 allocation failed"));
    for (;;); // Don't proceed, loop forever
  }
```

```
  display.display();
  delay(2000);
  display.clearDisplay();
}

void loop() {
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);

  if (Serial.available() > 0) {
    String command = Serial.readStringUntil('\n');
    if (command == "ROFF") {
      digitalWrite(RELAY_PIN, LOW);
      relayActivated = false;  // Reset relay control flag
    }
  }

  // Writing to display and serial
  for (int i = 0; i < 5; i++) {
    int pin = SENSOR_PIN_1 + i;
    bool newState = digitalRead(pin);
    display.print("Sensor ");
    display.print(i + 1);
    display.print(": ");
    display.println(newState ? "Open" : "Closed");

    // Writing to serial
    if (newState != lastStates[i]) {
      lastStates[i] = newState;  // Update posledný state
      Serial.print(i + 1);  // Sensor number
      Serial.println(newState ? "O" : "C"); // 'O' pre otvorený, 'C' pre
zatoverný
    }

    // zapínanie relay
    if (newState && !relayActivated) {
```

```
      digitalWrite(RELAY_PIN, HIGH);
      relayActivated = true;
    }
  }

  display.display();
  delay(100);
}
```

PŘÍLOHA P II: SW KÓD

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.IO;
using System.IO.Ports;
using System.Text;
using System.Windows.Forms;
namespace SW_časť
{
   public partial class Form1 : Form
   {
      TextBox outputTextBox, timestampsTextBox, transitionsTextBox;
      Label outputLabel, timestampsLabel, transitionsLabel;
      Button clearLogsButton, resetDoorsButton, alarmOffButton, saveLogsButton;
      Dictionary<int, Panel> doorPanels;
      Dictionary<int, Label> doorLabels;
      Dictionary<int, string> doorOpenTimestamps;
      Dictionary<int, string> doorCloseTimestamps;
      Dictionary<int, DateTime> lastDoorOpenTimes;
      SerialPort serialPort1;
      int lastDoorOpened = 0;
      DateTime lastTimeOpened = DateTime.MinValue;
      public Form1()
      {
         InitializeComponent();
         InitializeSerialPort();
         SetupOutputTextBox();
         SetupTimestampsTextBox();
         SetupTransitionsTextBox();
         SetupDoorPanels();
         SetupButtons();
         doorOpenTimestamps = new Dictionary<int, string>();
```

```
    doorCloseTimestamps = new Dictionary<int, string>();
    lastDoorOpenTimes = new Dictionary<int, DateTime>();
    InitializeDoorStates();
}
private void SetupDoorPanels()
{
    doorPanels = new Dictionary<int, Panel>();
    doorLabels = new Dictionary<int, Label>();
    int startingY = transitionsTextBox.Location.Y + transitionsTextBox.Height + 10;
    for (int i = 1; i <= 5; i++)
    {
        Panel doorPanel = new Panel
        {
            Size = new Size(50, 50),
            Location = new Point(10, startingY + 60 * (i - 1)),
            BackColor = Color.Green
        };
        Label doorLabel = new Label
        {
            Text = $"Door {i}",
            Location = new Point(70, startingY + 60 * (i - 1) + 15),
            Size = new Size(70, 20)
        };
        Controls.Add(doorPanel);
        Controls.Add(doorLabel);
        doorPanels[i] = doorPanel;
        doorLabels[i] = doorLabel;
    }
}
private void SetupOutputTextBox()
{
    outputTextBox = new TextBox
    {
        Location = new Point(100, 40),
```

```
        Size = new Size(300, 200),
        Multiline = true,
        ScrollBars = ScrollBars.Vertical
    };
    outputLabel = new Label
    {
        Text = "Door Status and Last Open/Close Time:",
        Location = new Point(100, 20),
        Size = new Size(300, 20)
    };
    Controls.Add(outputTextBox);
    Controls.Add(outputLabel);
}
private void SetupTimestampsTextBox()
{
    timestampsTextBox = new TextBox
    {
        Location = new Point(450, 40),
        Size = new Size(300, 200),
        Multiline = true,
        ScrollBars = ScrollBars.Vertical
    };
    timestampsLabel = new Label
    {
        Text = "Timestamp Log for Door Events:",
        Location = new Point(450, 20),
        Size = new Size(220, 20)
    };
    Controls.Add(timestampsTextBox);
    Controls.Add(timestampsLabel);
}

private void SetupTransitionsTextBox()
{
```

```
    transitionsTextBox = new TextBox
    {
        Location = new Point(800, 40),
        Size = new Size(300, 200),
        Multiline = true,
        ScrollBars = ScrollBars.Vertical
    };
    transitionsLabel = new Label
    {
        Text = "Door Transitions Log:",
        Location = new Point(800, 20),
        Size = new Size(200, 20)
    };
    Controls.Add(transitionsTextBox);
    Controls.Add(transitionsLabel);
}
private void SetupButtons()
{
    clearLogsButton = new Button
    {
        Text = "Clear Logs",
        Location = new Point(450, 260),
        Size = new Size(100, 30)
    };
    clearLogsButton.Click += ClearLogsButton_Click;
    Controls.Add(clearLogsButton);

    resetDoorsButton = new Button
    {
        Text = "Reset Doors",
        Location = new Point(800, 260),
        Size = new Size(100, 30)
    };
    resetDoorsButton.Click += ResetDoorsButton_Click;
```

```
    Controls.Add(resetDoorsButton);

    alarmOffButton = new Button
    {
        Text = "Alarm OFF",
        Location = new Point(650, 260),  // Positioned between the other two buttons
        Size = new Size(100, 30)
    };
    alarmOffButton.Click += AlarmOffButton_Click;
    Controls.Add(alarmOffButton);

    saveLogsButton = new Button
    {
        Text = "Save Logs",
        Location = new Point(1000, 260),
        Size = new Size(100, 30)
    };
    saveLogsButton.Click += SaveLogsButton_Click;
    Controls.Add(saveLogsButton);
}

private void ClearLogsButton_Click(object sender, EventArgs e)
{
    timestampsTextBox.Clear();
    transitionsTextBox.Clear();
}

private void ResetDoorsButton_Click(object sender, EventArgs e)
{
    foreach (var door in doorPanels)
    {
        door.Value.BackColor = Color.Green; // Closed door
    }
    lastDoorOpened = 0;
```

```
    lastTimeOpened = DateTime.MinValue;
    doorOpenTimestamps.Clear();
    doorCloseTimestamps.Clear();
    lastDoorOpenTimes.Clear();
    outputTextBox.Clear();
    timestampsTextBox.Clear();
    transitionsTextBox.Clear();
}

private void AlarmOffButton_Click(object sender, EventArgs e)
{
    // Send the "ROFF" command to the serial port
    if (serialPort1.IsOpen)
    {
        serialPort1.WriteLine("ROFF");
    }
    else
    {
        MessageBox.Show("Serial port is not connected.");
    }
}

private void SaveLogsButton_Click(object sender, EventArgs e)
{
    SaveLogsToFile();
}

private void SaveLogsToFile()
{
    try
    {
        using (SaveFileDialog saveFileDialog = new SaveFileDialog())
        {
            saveFileDialog.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*";
```

```
            saveFileDialog.FilterIndex = 1;
            saveFileDialog.RestoreDirectory = true;

            if (saveFileDialog.ShowDialog() == DialogResult.OK)
            {
                string filePath = saveFileDialog.FileName;

                using (StreamWriter writer = new StreamWriter(filePath))
                {
                    writer.WriteLine("Door Status and Last Open/Close Time:");
                    writer.WriteLine(outputTextBox.Text);
                    writer.WriteLine();

                    writer.WriteLine("Timestamp Log for Door Events:");
                    writer.WriteLine(timestampsTextBox.Text);
                    writer.WriteLine();

                    writer.WriteLine("Door Transitions Log:");
                    writer.WriteLine(transitionsTextBox.Text);
                }

                MessageBox.Show("Logs saved successfully!");
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Error saving logs: {ex.Message}");
    }
}

private void pictureBox1_Click(object sender, EventArgs e)
{
```

```
}

private void InitializeDoorStates()
{
    foreach (var door in doorPanels)
    {
        door.Value.BackColor = Color.Green; // Closed door
    }
}

private void InitializeSerialPort()
{
    serialPort1 = new SerialPort
    {
        BaudRate = 9600,
        Parity = Parity.None,
        StopBits = StopBits.One,
        DataBits = 8,
        Handshake = Handshake.None,
        PortName = "COM6"
    };

    serialPort1.DataReceived += SerialPort1_DataReceived;

    try
    {
        serialPort1.Open();
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Error opening serial port: {ex.Message}");
    }
}
```

```
    private void SerialPort1_DataReceived(object sender, SerialDataReceivedEventArgs
e)
    {
        SerialPort sp = (SerialPort)sender;
        string indata = sp.ReadExisting();
        if (!string.IsNullOrEmpty(indata))
        {
            UpdateDoorStates(indata.Trim());
        }
    }

    private void UpdateDoorStates(string data)
    {
        if (data.Length >= 2)
        {
            int doorNumber = data[0] - '0';
            char state = data[1];
            if (doorPanels.ContainsKey(doorNumber))
            {
                bool isOpen = state == 'O';
                DateTime now = DateTime.Now;
                string timestamp = now.ToString("HH:mm:ss");
                this.Invoke(new MethodInvoker(delegate
                {
                    doorPanels[doorNumber].BackColor = isOpen ? Color.Red : Color.Green;
                    if (isOpen)
                    {
                        doorOpenTimestamps[doorNumber] = timestamp;
                        if (lastDoorOpened != 0 && lastDoorOpened != doorNumber)
                        {
                            TimeSpan duration = now - lastTimeOpened;
                            transitionsTextBox.AppendText($"Door {lastDoorOpened} to Door
{doorNumber}: {duration.TotalSeconds} seconds\r\n");
                        }
```

```
                    lastDoorOpened = doorNumber;
                    lastTimeOpened = now;
                    UpdateTimestampsTextBox(doorNumber, "opened", timestamp);
                }
                else
                {
                    doorCloseTimestamps[doorNumber] = timestamp;
                    UpdateTimestampsTextBox(doorNumber, "closed", timestamp);
                }
                RefreshOutputDisplay();
            }));
        }
    }
}

    private void UpdateTimestampsTextBox(int doorNumber, string action, string timestamp)
    {
        string text = $"Door {doorNumber} {action} at {timestamp}\r\n";
        if (timestampsTextBox.InvokeRequired)
        {
            timestampsTextBox.Invoke(new MethodInvoker(delegate { timestampsTextBox.AppendText(text); }));
        }
        else
        {
            timestampsTextBox.AppendText(text);
        }
    }

    private void RefreshOutputDisplay()
    {
        StringBuilder displayText = new StringBuilder();
        foreach (var door in doorPanels)
```

```
        {
            string status = door.Value.BackColor == Color.Red ? "Open" : "Closed";
            string lastActionTime = door.Value.BackColor == Color.Red ?
                "Last opened at: " + (doorOpenTimestamps.ContainsKey(door.Key) ? doorOpenTimestamps[door.Key] : "N/A") :
                "Last closed at: " + (doorCloseTimestamps.ContainsKey(door.Key) ? doorCloseTimestamps[door.Key] : "N/A");
            displayText.AppendLine($"Door {door.Key}: {status} - {lastActionTime}");
        }

        UpdateUIThread(displayText.ToString());
    }

    private void UpdateUIThread(string text)
    {
        if (outputTextBox.InvokeRequired)
        {
            outputTextBox.Invoke(new MethodInvoker(delegate { outputTextBox.Text = text; }));
        }
        else
        {
            outputTextBox.Text = text;
        }
    }
}}
```