Výukový modul pro předmět Mikropočítače a PLC: elektronický zámek

Jiří Trojan

Bakalářská práce
2021

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav bezpečnostního inženýrství

Akademický rok: 2020/2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Jiří Trojan**
Osobní číslo:	**A20960**
Studijní program:	**B3902 Inženýrská informatika**
Studijní obor:	**Bezpečnostní technologie, systémy a management**
Forma studia:	**Kombinovaná**
Téma práce:	**Výukový modul pro předmět Mikropočítače a PLC: elektronický zámek**
Téma práce anglicky:	**Educational Module for Microcontrollers and PLC Course: Electronic Lock**

Zásady pro vypracování

1. Zpracujte literární rešerši na dané téma.
2. Proveďte hardwarový návrh rozšiřujícího modulu pro vývojovou desku Arduino.
3. Realizujte navržený modul.
4. Vytvořte podpůrné programové vybavení modulu a ukázkovou aplikaci.
5. Ověřte funkci modulu elektronického zámku.

Forma zpracování bakalářské práce: **Tištěná/elektronická**

Seznam doporučené literatury:

1. ATMEL CORPORATION. *Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V datasheet: Technical Data*, [online]. 2014 [cit. 2020-11-24]. Dostupné z: http://www.microchip.com
2. ID INNOVATIONS. *ID Series Datasheet: Classic RFID module products, Rev. 28*, [online]. 2015 [cit. 2020-11-24]. Dostupné z: http:// www.id-innovations.com
3. MANN, Burkhard. *C pro mikrokontroléry: ANSI-C, kompilátory C, spojovací programy – linkery, práce s ATMEL AVR a MSC-51, příklady programování v jazyce C, nástroje pro programování, tipy a triky*. Praha: BEN – technická literatura, 2003. μC & praxe. ISBN 80-7300-077-6.
4. PINKER, Jiří. *Mikroprocesory a mikropočítače.* Praha: BEN – technická literatura, 2004. ISBN 80-7300-110-1
5. VODA, Zbyšek. *Průvodce světem Arduina.* Bučovice: Martin Stříž, 2015. ISBN 978-80-87106-90-7

Vedoucí bakalářské práce: **Ing. Petr Dostálek, Ph.D.**
Ústav automatizace a řídicí techniky

Datum zadání bakalářské práce: **26. ledna 2021**
Termín odevzdání bakalářské práce: **19. května 2021**

doc. Mgr. Milan Adámek, Ph.D. v.r.
děkan

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

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

Ve Zlíně dne 15. ledna 2021

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 bakalářské 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 Univerzita Tomáše Bati 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 14. 5. 2021

Jiří Trojan v. r.

podpis studenta

ABSTRAKT

Bakalářská práce se zabývá návrhem a sestrojením elektronického zámku založeného na mikropočítačové platformě Arduino. Identifikace uživatele se provádí za pomocí RFID čtečky a může být kombinována se zadáním pinu na klávesnici. Elektronický zámek má implementovány dva režimy činnosti. Prvním režimem je autonomní zámek, který funguje bez potřeby komunikace s jiným zařízením. Druhou je zámek, který plní pokyny od master zařízení, se kterým komunikuje přes sériovou linku RS-485. Hardware i software zámku budou sloužit jako výuková pomůcka pro studenty Fakulty aplikované informatiky ve Zlíně.

Klíčová slova: elektronické systémy kontroly vstupu, RFID, mikrokontrolér, Arduino, sériová komunikace

ABSTRACT

This bachelor's thesis deals with design and construction of electronic lock based on microcontroller platform Arduino. User identification is performed by using RFID reader which can be combined with entering a pin on the keypad. Electronic lock has implemented two modes of operation. First mode is autonomous lock, which works without any need of communication with any other device. Second is lock, which carry out commands given by master device, with whom communicates using serial interface RS-485. Hardware and software of a lock will serve as teaching aids for students of Faculty of Applied Informatics in Zlín.

Keywords: electronic access control systems, RFID, microcontroller, Arduino, serial communication

Tímto bych chtěl poděkovat vedoucímu bakalářské práce Ing. Petru Dostálkovi Ph.D. za jeho profesionální přístup, výbornou komunikaci, nesčetné odborné rady a připomínky, které mi poskytl během vytváření bakalářské práce.

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

OBSAH

ÚVOD

Neustálý vývoj nových technologií znamená i neustálé zlepšování elektronických systémů kontroly vstupu. V současné době největší inovací prochází především biometrické přístupové systémy. Elektronické přístupové systémy jsou velmi důležitou součástí zabezpečovacích systémů, které by měla každá menší či větší firma využívat. Přispívají ke zvýšení bezpečnosti objektů díky možnosti kontrolovat přístup do zabezpečených prostor objektu. Dříve tento přístup byl kontrolován pomocí lidských zdrojů. Každá osoba, která do objektu vstupovala, musela být touto osobou zkontrolována. Tento způsob je časově i finančně náročný, proto stále více firem začíná využívat elektronické přístupové a docházkové systémy. Autentizace osob tímto způsobem probíhá téměř okamžitě.

Tato bakalářská práce má za úkol navrhnout a vytvořit výukový modul elektronického zámku. Výukový modul se skládá z vývojové desky Arduino Nano, která je připojena k desce plošných spojů. Deska plošných spojů obsahuje další rozšiřující moduly jako: čtečka RFID tagů, klávesnice 3x4, budič RS-485, DIP spínač, piezoměnič, LED dioda a elektromagnetické relé. Všechny tyto moduly jsou přes desku plošných spojů propojeny s deskou Arduino Nano, která je hlavní řídící jednotkou elektronického zámku. Výukový modul je určený především pro studenty Fakulty aplikované informatiky ve Zlíně, studující předmět Mikropočítače a PLC. Studenti si budou moci vyzkoušet funkci elektronického zámku a za pomocí předem připravených funkcí zámek sami naprogramovat.

Na začátku teoretické části bakalářské práce jsou popsány elektronické systémy kontroly vstupu, jaké jsou způsoby autentizace uživatelů, co je to RFID technologie, jaké má výhody a využití. Dále jsou zde popsány hardwarové prvky jako RFID čtečka ID-12LA a budič RS-485, který umožňuje sériovou komunikaci. Hlavním hardwarovým prvkem je vývojová deska Arduino Nano od společnosti Arduino, o které je zde napsaná kratší historie. V poslední kapitole teoretické části jsou popsány programy, jejichž pomocí byl vytvořen návrh desky plošných spojů a vývojové prostředí, ve kterých probíhala tvorba programů.

V první polovině praktické části jsou popsány jednotlivé kroky návrhu a vyrobení desky plošných spojů elektronického zámku. Druhá polovina praktické části se zabývá vytvořením programového vybavení pro oba režimy zámku. Nejdříve jsou zde popsané funkce, které jsou pro oba zámky společné. Následně byl pro každý režim zámku vytvořen vývojový diagram, ve kterém je zobrazena posloupnost dotazů a jejich vyhodnocení. Na základě vývojových diagramů byl vytvořen zdrojový kód pro oba typy zámku. Pro funkční programy byl udělán stručný uživatelský manuál.

I. TEORETICKÁ ČÁST

1 ELEKTRONICKÉ SYSTÉMY KONTROLY VSTUPU

Elektronické systémy kontroly vstupů (EACS) je soubor opatření, který řídí a eviduje informace o přístupu do určených prostor objektu. EACS přispívají ke zvýšení bezpečnosti objektu. Systémy poskytují oprávněným osobám nebo entitám vstupovat do zabezpečeného prostoru nebo jej opustit. Ověření oprávněnosti probíhá jedním ze způsobů autentizace. V případě, že se o vstup pokoušejí neoprávněné osoby nebo entity, dojde k zamítnutí přístupu. [7]
Elektronické systémy kontroly vstupu nejen zvyšují bezpečnost objektu, ale také šetří peníze provozovatele objektu, protože není nutné zaměstnávat osobu, která má na starosti kontrolu vstupu.
Od roku 2016 jsou EACS definované normou ČSN EN 60839: Poplachové a elektronické bezpečnostní systémy, která obsahuje čtyři normy.

- ČSN EN 60839-11-1 - Poplachové a elektronické bezpečnostní systémy - Elektronické systémy kontroly vstupu - Požadavky na systém a komponenty.
- ČSN EN 60839-11-2 - Poplachové a elektronické bezpečnostní systémy - Elektronické systémy kontroly vstupu - Pokyny pro aplikace.
- ČSN EN 60839-11-31 - Poplachové a elektronické bezpečnostní systémy - Elektronické systémy kontroly vstupu - Implementace IP interoperability na základě webových služeb - Základní specifikace.
- ČSN EN 60839-11-32 - Poplachové a elektronické bezpečnostní systémy - Elektronické systémy kontroly vstupu - Implementace IP interoperability na základě webových služeb - Specifikace systému kontroly vstupu. [7]

Nejrozšířenější zařízení EACS jsou čtečky (RFID a magnetické) a kódové klávesnice. Hlavní předností těchto systémů je jejich jednoduché používání, nízká cena a odolnost vůči vlivům počasí. Mezi další zařízení patří biometrické systémy jako čtečky otisků prstů nebo systémy rozpoznávání obličeje. Ovšem cena těchto systémů je výrazně vyšší, proto jsou také méně používané.
Elektronické systémy kontroly vstupu se dle rozsahu a topologie dělí do dvou typů:

- Autonomní systémy kontroly vstupu
- Modulární systémy kontroly vstupu [8]

Jednotlivé typy jsou blíže popsány v dalších kapitolách.

1.1 Autonomní systémy kontroly vstupu

Autonomní systémy kontroly vstupu zabezpečují kontrolu vstupu pouze nad jedním přístupovým místem. Obsahují řídící jednotku a snímací zařízení (čtečka, klávesnice nebo biometrie). Nejčastěji se používají kombinace klávesnice a čtečky nebo biometrie. Řídící jednotka čte informace ze snímacího zařízení a poté je vyhodnocuje. Na základě vyhodnocených informací buď přístup udělí nebo odmítne. Autonomní systémy jsou vhodné do objektů, ve kterých je malý počet přístupových míst. [8]

1.2 Modulární systémy kontroly vstupu

Modulární systémy kontroly vstupu se využívají, jestliže má objekt velký počet přístupových míst. Snímací zařízení jsou opět připojeny k řídící jednotce, ta však nepracuje autonomně, ale plní příkazy centrální řídící jednotky. Propojení jednotek nejčastěji probíhá pomocí sběrnice RS-485. [8]
Modulární systémy kontroly vstupu umožňují mnohem lepší kontrolu nad vstupem, kde povolení nebo odmítnutí přístupu určuje centrální řídící jednotka. Přidávání nových uživatelů do systému probíhá na jednom místě. U autonomních systémů by se uživatel musel přidat u každé jednotky samostatně. Informace o přístupech mohou být zobrazeny na centrální řídící jednotce.

1.3 Autentizace uživatelů

Jedná se o předem definovaný proces, podle kterého uživatel prokazuje svoji identitu. Autentizace má tři základní způsoby identifikace.

- Autentizace znalostí - V případě přístupových systému se jedná o zadání číselného kódu neboli hesla na klávesnici.
- Autentizace předmětem - Osoba se prokazuje vlastnictvím předmětu. Mezi tyto předměty obvykle patří čárové kódy, magnetické karty, RFID tagy či NFC (použití v mobilních telefonech).
- Autentizace biometrií - Fyzický znak dané osoby, jenž má každá osoba jedinečný (otisk prstu, rozpoznání obličeje).
- Více-faktorová autentizace - Jedná se o kombinaci minimálně dvou různých způsobů autentizace. [5]

1.3.1 Autentizace dle stupně zabezpečení

Druhy ověření podle normy ČSN EN 60839-11-1 vycházejí ze stupňů zabezpečení. Některé způsoby ověřování nejsou dostatečně bezpečné, proto je není možné použít pro

všechny stupně zabezpečení. Ovšem kombinace dvou a více autentizací je dostatečně bezpečná pro všechny stupně zabezpečení. [6]
Systémové požadavky na zabezpečovací systémy dle normy ČSN EN 50131-1 uvádí čtyři stupně zabezpečení podle rizika:

- Stupeň 1: Nízké riziko
- Stupeň 2: Nízké až střední riziko
- Stupeň 3: Střední až vysoké riziko
- Stupeň 4: Vysoké riziko [6]

Tabulka 1.1 Stupeň zabezpečení dle typu autentizace [6]

Stupeň zabezpečení / Typ autentizace	Stupeň 1	Stupeň 2	Stupeň 3	Stupeň 4
PIN - čtyřmístný (10 000 kombinací)	Povolené	Zakázané	Zakázané	Zakázané
PIN - pětimístný (100 000 kombinací)	Povolené	Povolené	Zakázané	Zakázané
Magnetické karty/RFID tagy	Povolené	Povolené	Povolené	Povolené
Biometrie	Povolené	Povolené	Povolené	Povolené
Dvoufaktorová autentizace	Povolené	Povolené	Povolené	Povolené

V praktické části bakalářské práce je pro autentizaci uživatele nebo správce použita dvoufaktorová autentizace (RFID karta + čtyřmístný pin), což odpovídá čtvrtému stupni zabezpečení.

1.4 RFID technologie

RFID (Radio Frequency Identification) je bezdrátová technologie používaná k automatické identifikaci objektů pomocí radiofrekvenční komunikace. RFID systémy se skládají z RFID čtečky a RFID tagu. Čtečka vysílá rádiové vlny a tím v okolí vytváří čtecí zónu. Tag má své sériové číslo, které je možné číst nebo přepisovat pomocí rádiových vln. Jakmile tag vstoupí do čtecí zóny, jsou jeho data zachyceny a dekódovány čtečkou. Tato data jsou předána počítači.
RFID systémy je vhodné používat tam, kde je zapotřebí rychlé a přesné zpracování informací ke zvýšení efektivnosti procesů. [12]
Mezi hlavní výhody RFID technologie patří:

- Každý tag má jedinečné sériové číslo.
- Čtečka komunikuje s tagy na dálku bez nutnosti přímé viditelnosti.
- Čtečka může načíst více tagů současně a velmi rychle (až 100 tagů/1 sec).

- Malé rozměry tagů. Nejčastěji v podobě klíčenky nebo karty.
- Poměrně nízká cena tagů (nákladnější v porovnání s čárovými kódy).
- Velká odolnost čipů vůči vlhkosti, teplotě a opotřebení. [12]

RFID technologie je nejčastěji používaná v logistice, výrobě, ochraně a evidenci majetku.

2 HARDWARE

Pod pojmem hardware je označeno veškeré fyzické vybavení určitého zařízení. V případě této bakalářské práce se jedná o všechny součástky elektronického zámku. V této kapitole budou popsány použité součástky.
Základ systému tvoří deska Arduino Nano, která řídí komunikaci s ostatními komponenty. Mezi další komponenty patří čtečka RFID tagů ID-12LA a budič RS485. Dále je použit čtyřpólový DIP spínač, 3x4 klávesnice KB304 (12 znaků), elektromagnetické relé, piezoměnič a dvoubarevná LED dioda.

2.1 ID-12LA

ID-12LA je RFID čtečka od společnosti ID-innovations o čtecí frekvenci 125 kHz. Napájecí napětí čtečky je v rozmezí 2,8-5 V. Dokáže číst formát karty EM 4001 nebo s ní kompatibilní, čtecí vzdálenost karet je až 12 centimetrů od čtečky. Výstup čtečky je 9600 baudů (1 baud je rovno 1 bit/s) po sériové lince RS232. [2]

Tabulka 2.1 Formát výstupu čtečky [2]

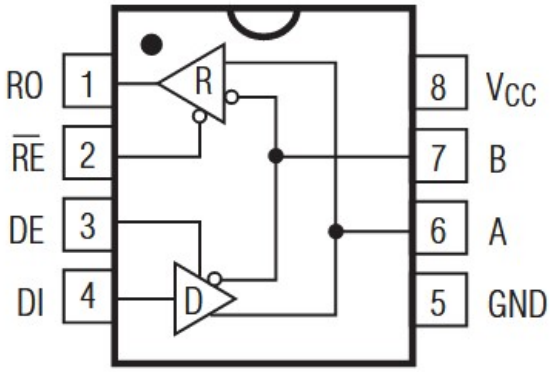
STX (02h)	DATA (10 ASCII)	CHECKSUM(2 ASCII)	CR+LF	ETX (03h)
Začátek přenosu	ID karty	Kontrolní součet	Nový řádek	Konec přenosu

Obrázek 2.1 ID-12LA [2]

2.2 Sběrnice RS-485

RS-485 sériové je rozhraní, které umožňuje digitální komunikaci na velkou vzdálenost (až 1200 metrů), je vhodné i do prostředí s elektronický šumem a podporuje více zařízení na jedné lince (32 a v případě, že se použije více opakovačů tak až 256 zařízení). RS-485 může mít dva druhy přenosu, a to poloviční a plný duplex. Komunikace v polovičním duplexu umožňuje jednotce buď jen příjem nebo jen odesílání informací. Plný duplex umožňuje zároveň příjem i odesílání informací. Aby mohl plný duplex fungovat, je potřeba používat dva signálové páry namísto jednoho.

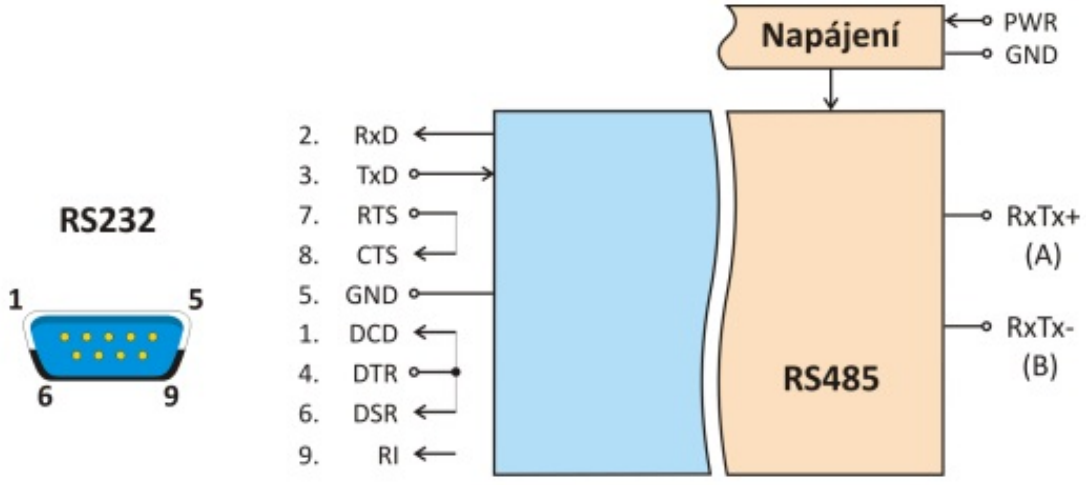
RS-485 se nejčastěji používá právě v poloviční duplexu, kdy má centrální řídící jednotka neboli master kontrolu nad sériovou linkou. Vysílá příkaz na sériovou linku, jenž obsahuje informaci o tom, pro kterou jednotku je směřován a typ dotazu. Jakmile tento příkaz odešle, tak se přepne do stavu čtení a čeká na odpověď. Příkaz čtou všechny ostatní řídící jednotky nazývající se slave. Jestliže je příkaz směřován na danou jednotku, tak se jednotka přepne ze stavu příjem na stav odesílání a odpoví.
Zařízení můžou přepínat mezi příjmem a odesíláním pomocí logické úrovně na pinu, který je připojen k pinům $\overline{RE}$,DE budiče RS-485 viz obrázek 2.2.[11]



Obrázek 2.2 Schéma budiče RS-485 [11]

2.3 Převodník TC485

Jedná se o obousměrný převodník linky RS232 na sběrnici RS485. Vyrábí jej česká společnost Papouch s.r.o. V bakalářské práci se používá pro zprostředkování komunikace mezi počítačem a budičem RS485. Převodník je možné propojit s počítačem pomocí sériové linky RS232. [9]



Obrázek 2.3 Blokové zapojení převodníku TC485 [9]

2.4 Arduino

V roce 2005 se lidé z italského Interaction Design Institute rozhodli vytvořit jednoduchý a levný vývojový set. Hlavním cílem vývojového setu byli studenti, kteří si v té době nemohli dovolit mnohem dražší desky BASIC Stamp. Arduino se mezi studenty rychle uchytilo, proto se jej tvůrci rozhodli poskytnout širší veřejnosti, a tím se Arduino postupně dostalo do celého světa. [4]
Výrobci se rozhodli Arduino vydat pod open source licencí. To způsobilo rychlejší vývoj platformy, ale také možnost Arduino vyrábět a distribuovat. Z toho důvodu společně s hlavními modely Arduina vznikají i neoficiální typy, které se nazývají klony např. Funduino, Femtoduino, FreeDuino a další. [4]

2.4.1 Arduino Uno

Arduino Uno je v současné době nejčastěji používaný typ desky. Na desce se nachází mikrokontrolér ATmega328, jehož parametry jsou: frekvence 16 MHz, 32 kB flash paměti, 1 kB EEPROM, pull-up rezistory o hodnotě 20 kΩ-50 kΩ a 2 kB SRAM. [1]
Na desce je umístěno klasické USB. Deska dále obsahuje 14 digitálních I/O pinů (6 z těchto pinů může být použito jako PWM) a 6 analogových pinů. Arduino Uno je ideální model pro výukové účely, díky jednoduchosti připojovat komponenty do female pinů. [4]

Obrázek 2.4 Arduino Uno [13]

2.4.2 Arduino Nano

Arduino Nano je druhou nejmenší oficiální verzí Arduina, navrženou pro úsporu místa. Výkon je srovnatelný s deskou Arduino Uno. Na desce je, stejně jako u Arduino Uno umístěn mikrokontrolér ATmega328. Hlavním rozdílem mezi deskami je kromě velikosti použití mini USB portu a male pinů. Jestliže bychom chtěli používat model Nano

pro výukové účely, museli bychom jej připojit k nepájivému poli. Díky malým rozměrům je vhodný k použití v dálkových ovladačích, elektronických zámcích a finálních konstrukcích, ve kterých se již nepředpokládá změna zapojení. [4]

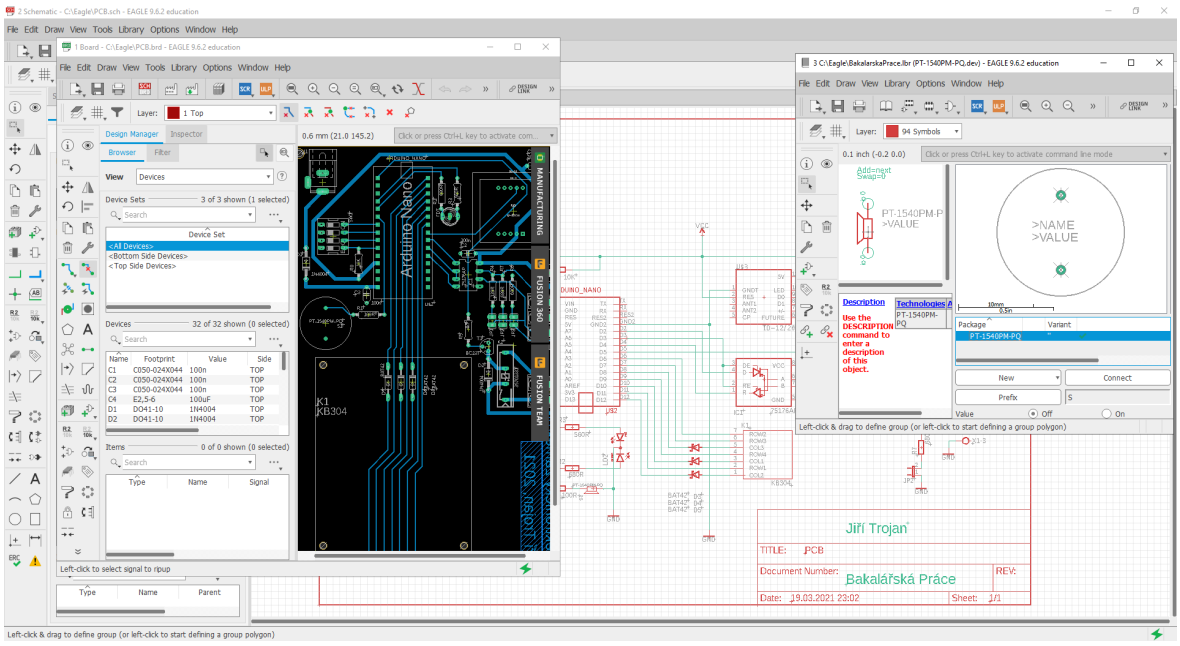
Obrázek 2.5 Arduino Nano [14]

3 SOFTWARE

V této kapitole jsou popsané jednotlivé části programu EAGLE, který sloužil k vytvoření návrhu schéma a vytvoření návrhu desky plošných spojů elektronického zámku. Dále jsou zde popsány použité programovací jazyky, vývojové prostředí Arduino IDE, PlaformIO a editor zdrojového kódu Microsoft Visual Studio Code, které byly použity pro tvorbu jednotlivých programů.

3.1 EAGLE

Software EAGLE, jehož novým majitelem je od roku 2017 společnost Autodesk Inc. (původně jím byla Německá společnost CadSoft Computer GmbH), je nejpoužívanější návrhový systém pro vytváření desek plošných spojů.



Obrázek 3.1 Prostředí programu EAGLE

Software se skládá ze tří podprogramů:

- Editor knihoven - Program obsahuje velké množství knihoven, ve kterých jsou obsaženy definice jednotlivých součástek. Knihovny s dalšími součástkami je možné volně stáhnout z internetu. Každá součástka v knihovně má dvě části, první je schematická značka (symbol) a druhou je pouzdro. Editor knihoven se používá v případě, že je potřeba upravit nebo vytvořit novou součástku.
- Editor schémat slouží k vytváření kompletního schéma zapojení. Vytváření schéma probíhá tak, že si z knihovny vybereme součástku a umístíme její schematickou

značku do volného místa v editoru. Jednotlivé symboly potom propojíme pomocí příkazu NET. Jakmile je hotové schéma zapojení, poté lze jedním příkazem vytvořit desku plošných spojů.

- Editor plošných spojů je posledním z nástrojů pro návrh desky plošných spojů. V editoru se vytváří závěrečné rozmístění pouzder součástek a jejich propojení. Propojení může uživatel udělat sám nebo je možné použít funkci autorouter. Program je schopen pracovat s návrhy až šestnácti vrstev plošných spojů.

3.2 Programovací jazyky

Při tvorbě programu pro Arduino je použit jazyk, který se nazývá Arduino Programming Language. Tento jazyk byl vytvořen na základě jazyka Wiring. Ovšem základ obou těchto jazyků tvoří programovací jazyky C a C++. Jedná se tedy v podstatě o programování v jazyce C++, které je obohaceno o funkce komunikující s mikrokontrolérem. [4]

3.3 Vývojové prostředí

Vývojové prostředí (IDE) je softwarový nástroj, který umožňuje programátorovi efektivní tvorbu počítačového programu. Kombinuje běžné aktivity vytváření softwaru do jedné aplikace:

- Editor zdrojového kódu pomáhá programátorovi při psaní kódu. Barevně zvýrazňuje slova, které mají určitý význam. Při psaní textu nabízí automatické doplnění slova, neboli autocomplete pro urychlení psaní.

- Kompilátor neboli překladač přeloží zdrojový text programu do strojového kódu pro počítačový procesor. Ve strojovém kódu se nachází instrukce a data, se kterými dané instrukce pracují.

- Debugger je nástroj pro hledání programátorských chyb v programu ve fázi ladění. Umožňuje vykonávat program řádek po řádku (krokování). Při krokování je možné sledovat hodnoty proměnných, registrů nebo míst v paměti. Debugger také umožňuje do programu vkládat zarážky, pomocí nichž lze zastavit běh programu v místě, ve kterém je daná zarážka umístěna. [10] [3]

3.3.1 Arduino IDE

Vývojové prostředí Arduino IDE je open-source software, který je volně ke stažení na webových stránkách Arduina. Aplikace je napsaná v jazyce Java a lze ji používat na všech běžných operačních systémech (Windows, macOS, Linux). Jedná se o software,

který vznikl na základě výukového prostředí Processing. [4]
Arduino IDE obsahuje jednoduchý textový editor, který podporuje jazyky C a C++, jejichž syntaxi barevně zvýrazňuje. Tlačítka pro kompilaci a nahrávání zdrojového kódu jsou umístěny v levém horním rohu programu. Program obsahuje také manažer knihoven pro případ, že by uživatel chtěl používat další knihovny, které nejsou součástí základního programu. Velmi užitečnou součástí vývojového prostředí je sériový monitor, jenž slouží pro zobrazení sériové komunikace.

3.3.2 Microsoft Visual Studio Code

Microsoft Visual Studio Code (VS Code) je efektivní editor zdrojového kódu s vývojářskými operacemi jako ladění a spouštění kódu. Poskytuje nástroje pro rychlé vytváření a ladění kódu. Nejedná se ovšem o plnohodnotné vývojové prostřední jako Visual Studio IDE. Základní verze VS Code, kvůli co největší jednoduchosti, obsahuje pouze podporu jazyků jako JavaScript, TypeScript, CSS, a HTML. Existuje však velké množství rozšíření, které je možné stáhnout přímo v programu.
Tyto rozšíření přidávají nejen podporu téměř všech programovacích jazyků, ale třeba také code runner, který slouží k jednoduchému spouštění kódu. V případě, že je nainstalované rozšíření s podporou jazyků, lze využívat technologii IntelliSense. IntelliSense nabízí funkce jako: dokončování slov, seznam členů, informace o parametrech a rychlé informace. Za zmínku stojí také rozšíření, které přidávají možnost úpravy vzhledu a formátování kódu.

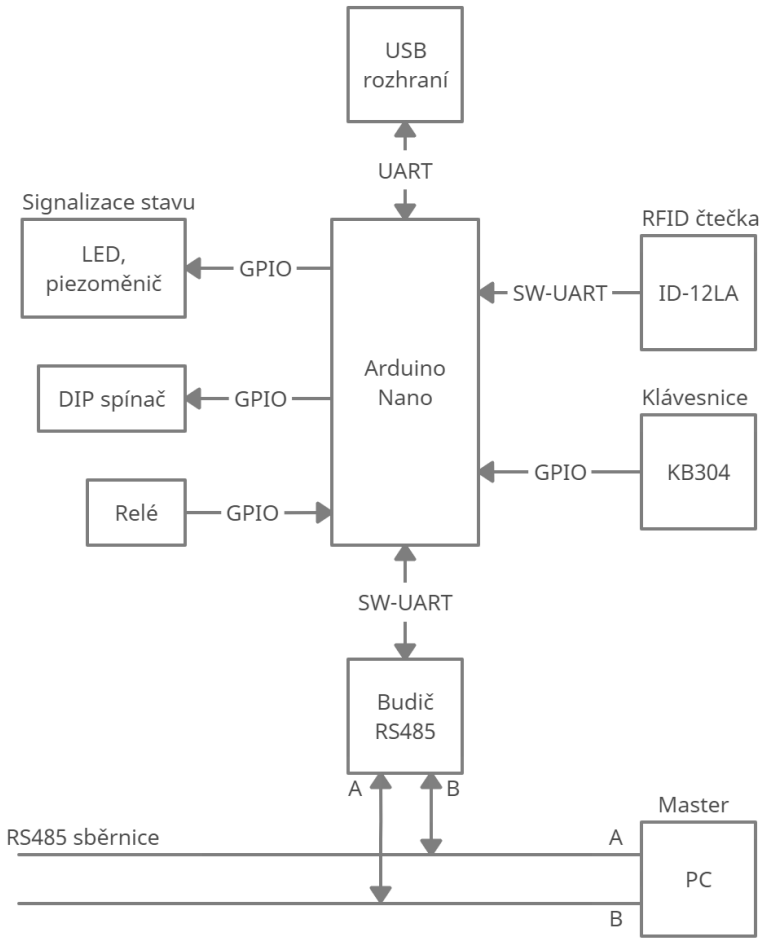
3.3.3 PlatformIO

Jedním z obsáhlejších rozšíření pro Microsoft Visual Studio Code je PlatformIO IDE. Jedná se o vývojové prostředí pro vytváření aplikací embedded zařízení. PlatformIO IDE funguje ve všech operačních systémech, na kterých je možné nainstalovat VS Code. Programování elektronického zámku probíhalo právě v PlatformIO. Počáteční instalace a nastavení vývojového prostředí byla časově náročnější než velmi jednoduché nastavení Arduina IDE. Ovšem čas strávený touto instalací se díky lepšímu přehledu programu společně s technologií IntelliSense, které zvýšily celkovou efektivitu a kvalitu programování, se bezpochyby vyplatil.

II. PRAKTICKÁ ČÁST

4 NÁVRH A TVORBA MODULU ELEKTRONICKÉHO ZÁMKU

Úkolem praktické části bakalářské práce je vytvořit elektronický zámek včetně funkčního programového vybavení. V této kapitole bude popsán návrh a vytváření hardwarové části zámku. Hlavní hardwarovou součástí je Arduino Nano, která je připojena k desce plošných spojů na které jsou umístěny všechny ostatní komponenty.
Návrh fyzického sestavení elektronického zámku spočívá především ve výběru komponentů dle funkčních požadavků zámku. V teoretické části bylo zmíněno, že existují dva typy elektronických systémů kontroly vstupu. Elektronický zámek byl tedy navržen tak, aby jej bylo možné použít v obou těchto typech. V prvotní fázi návrhu bylo vytvořeno blokové schéma elektronického zámku.



Obrázek 4.1 Blokové schéma elektronického zámku

Blokové schéma obsahuje všechny důležité hardwarové komponenty pro oba typy zámku. Šipky ukazují směr komunikace a zda se jedná o jednostrannou nebo oboustrannou komunikaci. Popis u šipek značí typ propojení:

- GPIO neboli univerzální vstupní/výstupní pin je označení digitálního rozhraní,

jehož využití je zcela na programátorovi.

- SW-UART je sériové komunikační rozhraní sloužící k asynchronnímu sériovému přenosu dat. Funkce rozhraní je zajištěna prostřednictvím běžných GPIO pinů bez použití dedikovaného UART rozhraní. Programová obsluha je zajištěna knihovnou SoftwareSerial. Formát a rychlost přenosu jsou nastavitelné. V bakalářské práci se pro všechny komunikace používá rychlost 9600 baudů.

V tabulce Tab. 4.1 je uveden seznam všech komponentů z blokového schéma. U každého komponentu je uvedeno, zda se s daným komponentem pracuje v jednotlivých typech zámku:

Tabulka 4.1 Seznam komponentů

Název komponentu	Autonomní zámek	Modulární zámek
Čtečka ID-12	ANO	ANO
Klávesnice KB304	ANO	ANO
Budič RS-485 (75176AP)	NE	ANO
Elektromagnetické relé	ANO	ANO
DIP spínač	NE	ANO
LED dioda	ANO	ANO
Piezoměnič	ANO	ANO

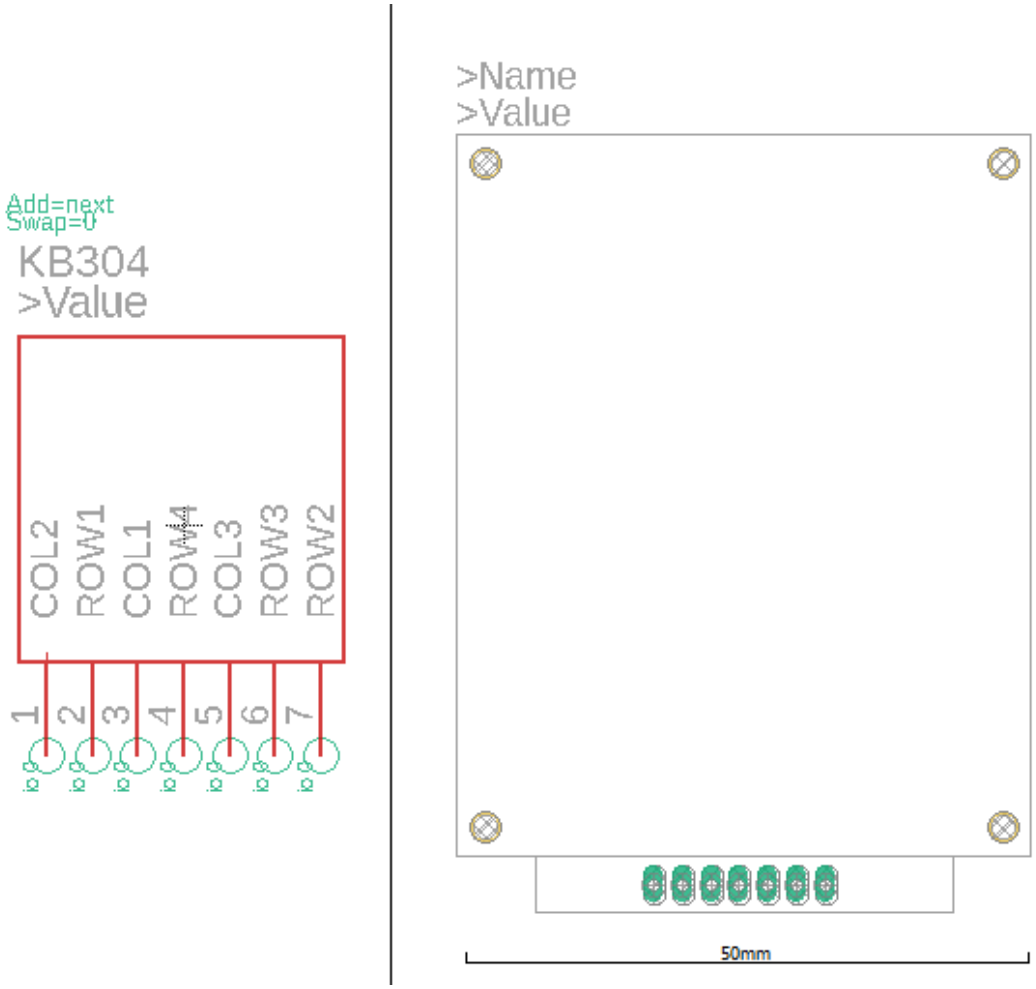
Z tabulky vyplývá, že většina komponentů je využívaná v obou typech zámku. U autonomního zámku nejsou použity komponenty jako budič RS-485 pro sériovou komunikaci a DIP spínač, jehož pomocí lze zámku vytvořit jeho unikátní identifikátor. Všechny použité součástky jsou běžně dostupné ve specializovaných obchodech s elektronikou.

4.1 EAGLE - Návrh schéma

Další část návrhu desky se provádí v programu EAGLE. Nejprve je v programu na základě dříve vytvořeného blokového schématu vytvořeno schéma, které obsahuje detailní zapojení jednotlivých součástek. Ve schématu jsou také obsaženy menší součástky jako: rezistory, kondenzátory a diody, které jsou nezbytně nutné pro správné fungování hlavních komponentů.
Pro vkládání jednotlivých součástek je potřeba stisknout tlačítko ADD PART. Po jeho stisknutí se otevře okno, ve kterém jsou zobrazeny všechny přístupné knihovny. Uvnitř těchto knihoven se nachází požadované součástky. Potom co jsou součástky umístěny do schématu, tak se musí propojit. K jejich propojení se používá příkaz NET.
Při vytváření schéma bylo zjištěno, že nejsou dostupné knihovny pro klávesnici a piezoměnič. Pro tyto součástky se tedy musely vytvořit vlastní knihovny.

4.1.1 Vytváření knihoven

Vytváření knihoven probíhá v editoru knihoven programu EAGLE. Pro každou součástku je nejdříve vytvořeno pouzdro dle rozměrů z datasheetu. Vhodné je také vybrat správný typ a velikost padů pro jednodušší pájení součástky v pozdější části výroby desky. Jakmile je dokončeno pouzdro součástky, následuje vytváření její schematické značky. Při vytváření schematické značky lze zvolit její velikost, ovšem výsledný symbol součástky by měl být v souladu s normou: ČSN EN 61082-1 ed. 3: Zhotovování dokumentů používaných v elektrotechnice. Důležité je také správně označit, kterému padu náleží daný pin.



Obrázek 4.2 Schematická značka a pouzdro klávesnice KB304

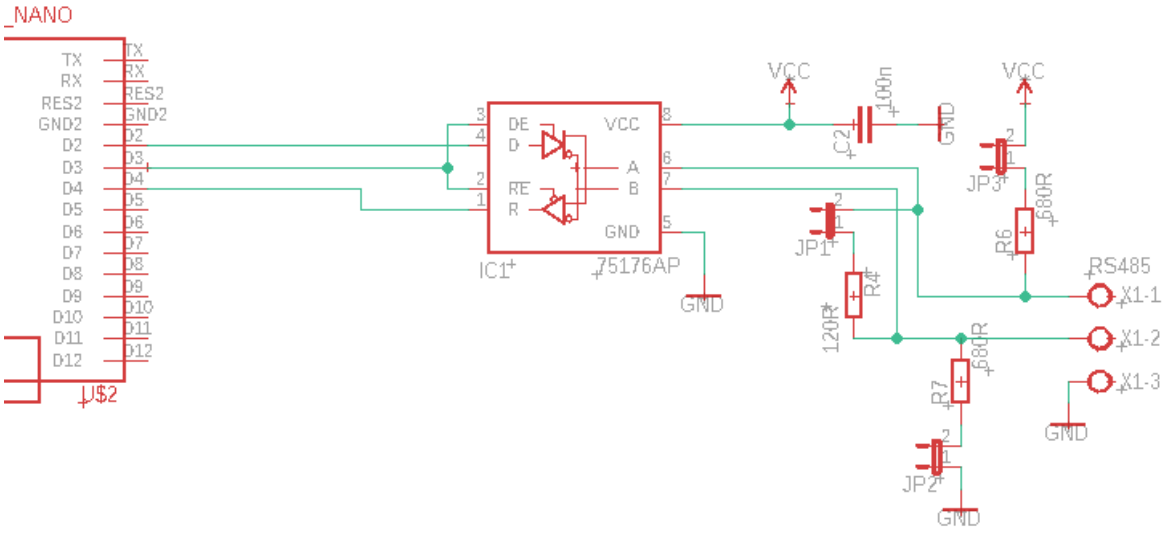
4.1.2 Klávesnice KB304

Jedná se o maticovou klávesnici s 12 tlačítky, které jsou v konfiguraci 3x4 (tři sloupce a čtyři řádky). Výstup každého sloupce a řádku musí být vyveden samostatně. Klávesnice má tedy výstup hodnot ze sedmi pinů. Mezi propojením pinů sloupců s piny Arduina

jsou použity polovodičové diody. Polovodičové diody zabraňují zkratu v případě, že by bylo stisknuto více kláves než jedna.

4.1.3 Budič RS-485

V teoretické části byla na obrázku 2.2 zobrazena schématická značka budiče. Z této schématické značky se vycházelo při zapojování budiče v programu EAGLE. Piny řídící směr komunikace jsou připojeny do pinu D3 v Arduinu. V případě, že je logická úroveň na tomto pinu rovna HIGH (rovna hodnotě 1), tak je budič ve stavu odesílání. Logická úroveň LOW značí stav přijímání. Piny pro odesílání a přijímaní dat jsou připojeny do pinů D2 a D4. Datové piny A (RxTx+), B (RxTx-) jsou připojeny ke svorkovnici, ke které je možné v případě potřeby připojit RS485 sběrnici. Mezi propojením datových pinů a svorkovnice je umístěn terminátor sběrnice R4 a bias rezistory R7 a R6 pro nastavení klidové napěťové úrovně na sběrnici. Pomocí propojek JP1 až JP3 lze dle aktuálního propojení modulu zámku se sběrnicí RS485 tyto rezistory odpojit. U budiče je, stejně jako u čtečky potřeba připojení k napětí 5 V.

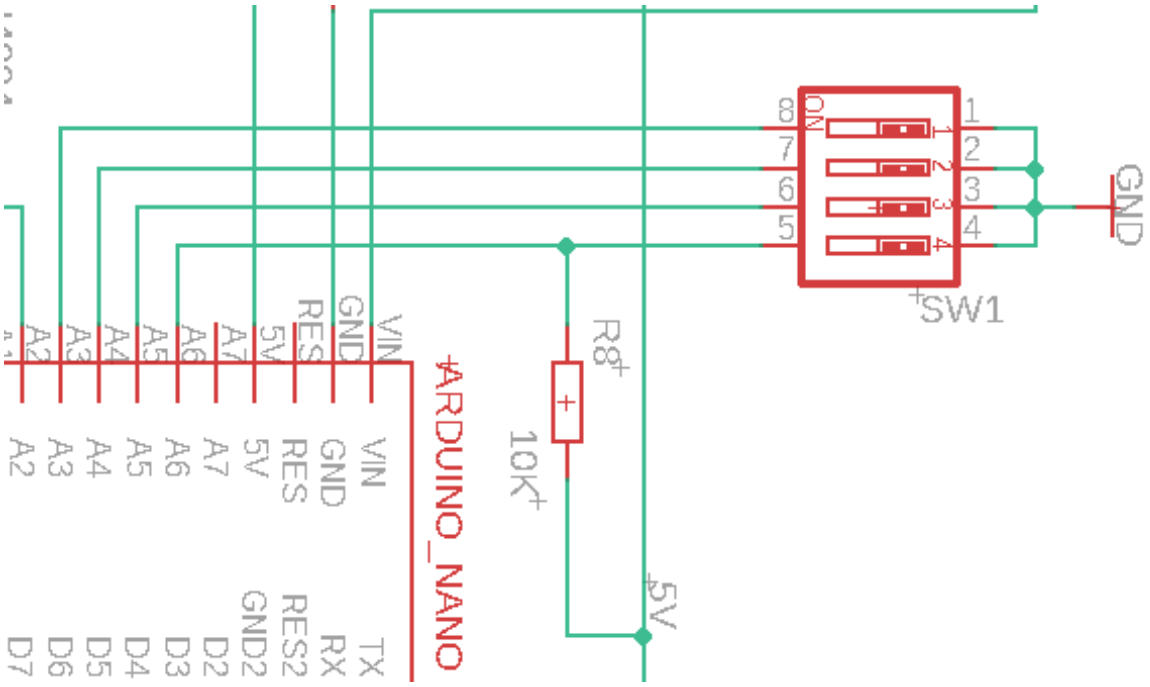


Obrázek 4.3 Schéma zapojení budiče RS-485

4.1.4 DIP spínač

V bakalářské práci je použit čtyřpólový DIP spínač, který slouží k identifikaci až šestnácti zámků. Pomocí spínače lze zámku přidělit jeho unikátní číslo bez nutnosti zásahu do jeho programu. Hodnoty na spínači se přepínají ručně mezi 0 a 1. Převodem binárního čísla na decimální tedy lze získat číslo 0-15. Důvodem zvolení jen čtyřpólového spínače je omezený počet pinů na desce Arduino Nano. Pokud by byl použit osmipólový spínač, bylo by možné mít až 256 kombinací.

Hodnoty na DIP spínači se u Arduina dají zjistit pomocí funkcí digitalRead() a analogRead(). Pro správné čtení aktuálního nastavení jednotlivých spínačů je zapotřebí na odpovídajících vstupech aktivovat pullup rezistory. V případě, že by tento rezistor nebyl použit, tak by vstupy neměly při rozpojeném spínači definované logické úrovně. U digitálních pinů Arduina lze použít pullup rezistor pomocí funkce pinMode(), která se nastaví jako INPUT_PULLUP. Použitím této softwarové funkce se přistupuje k pullup rezistorům, které jsou integrované na čipu mikrokontroléru ATmega. Jakmile jsou tyto piny nastavené, tak se pro čtení těchto pinů používá funkce digitalRead(). Arduino Nano u pinů A6, A7 umožňuje použití pouze analogového vstupu. Hodnota se tedy čte za pomocí funkce analogRead(). U analogového vstupu není možné použití pullup rezistoru na čipu mikrokontroléru. Z toho důvodu je nutné pro tento pin použít fyzický pullup rezistor. Hodnota odporu tohoto rezistoru byla zvolena 10 kΩ.



Obrázek 4.4 Schéma zapojení DIP spínače včetně pullup rezistoru

4.2 EAGLE - Návrh plošných spojů

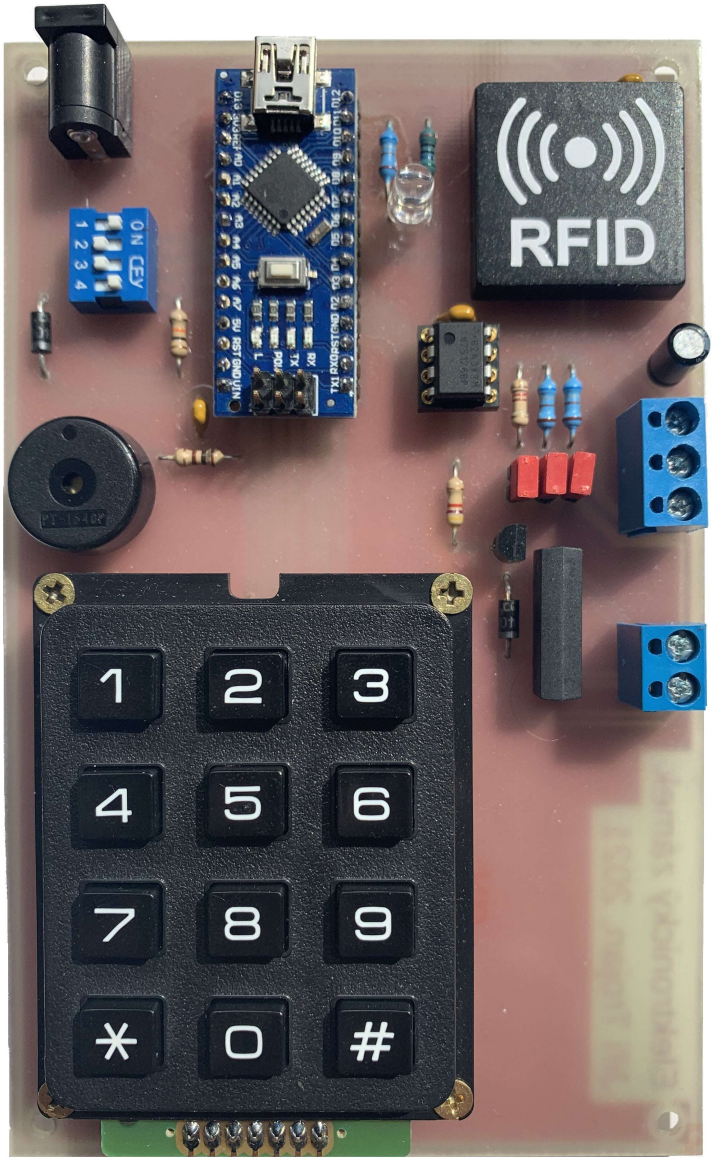
Potom co byl zhotoven návrh schéma zapojení, bylo možné přikročit k vytvoření návrhu desky plošných spojů. Prvním krokem bylo vygenerování desky pomocí příkazu GENERATE. Po stisku tohoto příkazu se otevře nové okno, ve kterém jsou vygenerovány pouzdra všech součástek. Elektrické připojení součástek je značeno vzdušnými spoji (airwires). Přeměnění vzdušných spojů na fyzické plošné spoje umožňuje příkaz ROUTE. Při užívání tohoto příkazu je potřeba počítat s tím, že se fyzické plošné spoje nemůžou navzájem překrývat. Po zhotovení všech propojení následovalo vytvoření po-

lygonu. Polygon, sloužící jako GND, byl nakreslen okolo celé desky. Posledním krokem návrhu je příkaz RATSNEST. Příkaz vytvoří vzdušné spoje v místech okolo plošných spojů. Šířka tohoto vzdušného spoje se dá zvolit pomocí příkazu DRC. V bakalářské práci je šířka vzdušných spojů zvolena 0,33 mm. Výsledný návrh se vytiskne na fólii.

4.3 Vytváření desky plošných spojů

Vytváření desky plošných spojů probíhalo fotochemickou cestou neboli fototechnikou. Je zapotřebí deska s měděnou folií, na které je nanesena vrstva emulze citlivé na světlo. Celá výroba probíhala v 10 krocích:

1. Osvit plošných spojů - Do UV osvitové jednotky se na pět minut vloží měděná deska společně s fólií, která byla vytisknuta na základě návrhu z EAGLU.
2. Osvětlená deska se vloží do vývojky (roztok, který obsahuje okolo 1,5% hydroxidu sodného NaOH), ve které se osvětlená emulze začne vyvolávat (emulze se začne odplavovat). Emulze, která nebyla osvětlena, tak zůstává v celku. Deska se vyvíjí do té doby, než jsou meziprostory viditelné jako čistá měď (pár minut).
3. Leptání v chloridu železitém $FeCl_3$ - Deska se ponoří do leptací lázně. V místech, ve kterých byla emulze odstraněna, začne docházet k leptání mědi. Leptání probíhá do té doby, než je měď zcela odstraněna (okolo dvaceti minut).
4. Vyleptaná deska se opláchne pod tekoucí vodu a osuší.
5. Vrtání otvorů pro součástky - Do desky jsou vyvrtány otvory s požadovaným průměrem, které se liší dle velikosti padů.
6. Osvit plošných spojů - Nyní se do osvitové jednotky na pět minut vloží deska bez fólie.
7. Osvětlená deska se vloží do vývojky - Stejný postup jako u kroku 2.
8. Pro lepší pájení a krátkodobou ochranu před oxidací byla na desku nanesena pájecí kapalina (kalafuna rozpuštěná v lihu), která se nechala zaschnout.
9. Pájení jednotlivých součástek - Součástky jsou vloženy do určeného místa na desce. Nožičky umístěných součástek jsou k desce připájeny pomocí cínu. Nejprve jsou připájeny nízké součástky (rezistory, diody, kondenzátory) a postupně se připájí součástky vyšší (Arduino, klávesnice)
10. Deska se očistí a je na ni nanesen lak, který se nechá uschnout (desítky hodin).



Obrázek 4.5 Vzhled dokončené desky elektronického zámku

5 PROGRAMOVÉ VYBAVENÍ

Jakmile byla výroba zámku dokončena, mohlo se začít s vytvářením dvou programů. Nejdříve byl vytvořen program pro autonomně fungující zámek a poté program pro zámek, jenž by bylo možné použít do modulárního systému kontroly vstupu. Při vytváření prvního programu byla snaha vytvořit funkce, které by bylo možné použít v obou programech.

5.1 Knihovny

Knihovny jsou balíčky, jenž přidávají funkce nebo operace, které slouží k zjednodušení kódu. Bez nich by se muselo psát mnohdy velmi zdlouhavý kód, který je jejich pomocí nahrazen jednou funkcí. Knihovny se do programu vkládají pomocí direktivy #include. Měly by se do hlavičky programu přidat jako první. Jestliže se používá knihovna, která nebyla nainstalována společně s instalací vývojového prostředí, je nutné ji před používáním nainstalovat. Inicializace knihoven v obou programech vypadá následovně:

```
#include SoftwareSerial.h>
#include Keypad.h>
//Pouze u autonomního zámku
#include EEPROM.h>
```

Knihovna SoftwareSerial zprostředkovává použití sériové komunikace na digitálních pinech. Knihovna EEPROM je použita pouze u autonomního zámku. Slouží pro práci s EEPROM (čtení a zápis). Keypad je jedinou knihovnou, která není přístupná mezi základními knihovnami. Slouží pro čtení maticových klávesnic všech možných konfigurací. Verze knihovny 3.0 a novější (v současnosti je nejnovější verze 3.1.1) podporují stisk více kláves najednou. Knihovna využívá integrované pullup rezistory mikrokontroléru. Jestliže by se vždy používala tato knihovna, nebylo by nutné zapojení polovodičových diod.

5.2 Deklarace proměnných a konstant

Potom co jsou do programu vloženy knihovny, je na čase deklarovat konstanty, proměnné a definovat makra. Konstanty se používají v případech, ve kterých v průběhu programu nebude potřeba změnit jejich hodnotu. V programu se jedná například o pin součástky. Existují dva způsoby, jak konstanty definovat. První způsob je použití klíčového slova const před definicí dané proměnné. Hodnotu v proměnné již nebude možné změnit. Druhou je definice makra. Definice makra se píše ve tvaru #**define název hodnota**. Hodnotou může být číslo pinu, ale také například výpočet maximálního počtu uživatelů nebo zavolání funkce, která vrací nějakou hodnotu. Vždy když program narazí na některý z názvů maker, tak je název pomocí preprocesoru změněn na danou hodnotu. Konstanty a makra slouží ke zvýšení přehlednosti programu, protože

si programátor nemusí pamatovat na jaký pin je daná součástka připojena, ale stačí znát její název. Definice konstant v obou programech vypadají následovně:

```
//Společné definice maker obou zámků
#define LEDRed A0
#define LEDGreen A1
#define piezo A2
#define relay 13
#define idTX A7
#define idRX 5
//Jedinečné makra autonomního zámku
#define maxUsers 20
#define maxMemoryEEPROM (maxUsers*12+1)
//Jedinečné makra modulárního zámku
#define RXTXswitch 3
#define dip1 A3
#define dip2 A4
#define dip3 A5
#define dip4 A6
#define rsTX 2
#define rsRX 4
#define rs485TX HIGH
#define rs485RX LOW
#define doorNumber getDip() //Zámku přidělí identifikátor dle
   hodnoty na DIP spínači
```

Proměnné se používají v případě, že se v průběhu programu bude měnit jejich hodnota. Při deklaraci proměnné je na prvním místě zapsán její typ (int, bool, char, long, atd.) a na druhém místě její název. Krok, ve kterém se do proměnné vkládá hodnota, se nazývá inicializace. Inicializace proměnné může probíhat zároveň s její deklarací. Příklad deklarace a inicializace proměnných:

```
int x = 0;
unsigned long ledTimer; //Unsigned nemůže nabývat záporných
   hodnot.
char password[5]; //Znakové pole o velikosti pěti znaků.
char key;
char card[13];
bool allowCard = false;
```

Navázání komunikace mezi klávesnicí a Arduinem bylo zprostředkováno pomocí knihovny Keypad. Aby bylo možné klávesnici správně číst, je potřeba definovat a inicializovat tři proměnné. První proměnnou je pole, které definuje, jaké znaky se odešlou při stisknutí příslušného tlačítka. Znaky jsou v poli rozloženy, stejně jako na klávesnici. Další dvě proměnné nastavují pro řádky a sloupce čísla pinů tak, jak jsou do připojeny do Arduina. Zde je důležité dodržet správné pořadí pinů. Jakmile jsou proměnné inicializovány, tak následuje příkaz pro vytvoření instance.

```
//Pole charakterů, ve kterém jsou znaky rozloženy, stejně jako
   na klávesnici.
char keyList[4][3] =
{
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};
byte rowPin[4] = {11, 6, 7, 9}; //Piny Arduina, do kterých jsou
   připojeny piny řádků klávesnice.
```

```
byte columnPin[3] = {10, 12, 8}; //Piny Arduina, do kterých jsou
    připojeny piny sloupců klávesnice.
Keypad keypad = Keypad(makeKeymap(keyList), rowPin, columnPin,
   4, 3);
```

Část kódu deklarující klávesnici byla převzatá z příkladu knihovny Keypad. [15]
V hlavičce programu probíhá také vytváření nové instance SoftwareSerial objektu. Jejich vytváření probíhá následujícím způsobem:

```
//Nastavení sériová komunikace, pro komunikaci s RFID čtečkou.
SoftwareSerial RFID(idRX, idTX);
//Nastavení sériová komunikace, pro komunikaci s budičem RS-485.
SoftwareSerial RS485(rsRX, rsTX);
```

Jejich aktivace a nastavení přenosové rychlosti probíhá ve funkci setup().

5.3 Funkce setup()

Příkazy ve funkci setup() jsou vykonány pouze jednou a to při startu nebo restartu mikropočítače. Funkce setup se nejčastěji využívá k nastavení parametrů sériové komunikace např.: Serial.begin(9600) sloužící k nastavení výpisu na sériovou linku. Také se zde nastavují piny Arduina na vstup nebo výstup. Dále se zde dá nastavit logická úroveň tohoto pinu. To je vhodné v případě, že chceme při spuštění programu rozsvítit LED diodu.

5.4 Funkce loop()

Potom co se provedou všechny příkazy funkce setup(), následuje funkce loop(). V této funkci se příkazy opakují v nekonečné smyčce. Jedná se o jádro všech programů Arduina, ve kterém se vykonává naprostá většina příkazů nebo funkcí.

5.5 Společné funkce pro oba typy zámku

Přestože oba režimy zámku fungují velmi odlišně. Funkce getRFID() pro čtení karet, lightLED() pro rozsvěcování LED diod a knihovna Keypad pro čtení klávesnice jsou v obou programech stejné.

5.5.1 Funkce lightLED()

Fukce lightLED(barva,čas) se používá v případě, že programátor chce rozsvítit LED diody na určitý čas. Vstupními hodnotami funkce jsou barva ('r' pro červenou, 'g' pro zelenou, 'o' pro oranžovou = obě najednou) a délka rozsvícení dané LED diody.
Pro rozsvěcováni LED diod je využita funkce millis(). Tato funkce vrací čas (v milisekundách), který uplynul od spuštění programu. Použití této funkce je poněkud složitější než jen jednoduché použití funce delay(), která by vždy pozastavila celý program.

```
void lightLED(char color, int time)
{
//Nastavení času a časovače, které jsou použity v další funkci.
  LEDTimerStart = millis();
  timer = time;
  digitalWrite(LEDGreen, LOW);
  digitalWrite(LEDRed, LOW);
  switch (color)
  {
    case 'r':
    {
      turnOffLED = true;
      digitalWrite(LEDRed, HIGH);
      break;
    }
    case 'g':
    {
      turnOffLED = true;
      digitalWrite(LEDGreen, HIGH);
      break;
    }
    case 'o':
    {
      turnOffLED = true;
      digitalWrite(LEDRed, HIGH);
      digitalWrite(LEDGreen, HIGH);
      break;
    }
    default:
    {
      break;
    }
  }
}
```

Ve funkci lightLED() probíhá pouze rozsvěcováni LED diod, jejich zhasínání probíhá v další společné funkci.

5.5.2 Funkce checkTimer()

Funkce kontroluje, zda uplynul čas, který byl nastaven při zapínání LED a zda byla nějaká z LED diod zapnutá. V případě, že byla daná podmínka splněna, zhasne LED diody a resetuje timer. Dojde také k uzavření dveří, proto jejich otevírání probíhá společně s funkcí, která spouští LED diody.

```
void checkTimer()
{
  LEDTimer = millis();
//Jestliže uplynul nastavený čas a zároveň byla zapnutá LED
   dioda, tak je podmínka splněna.
  if (LEDTimer - LEDTimerStart >= timer && turnOffLED)
  {
    turnOffLED = false;
    LEDTimerStart = ledTimer;
    digitalWrite(ledRed, LOW);
    digitalWrite(ledGreen, LOW);
    digitalWrite(relay, LOW);
  }
}
```

5.5.3 Funkce getRFID()

Funkce getRFID(card) se volá v případě, že se požaduje získání ID přečtené karty. Funkci se předává ukazatel na textový řetězec, do kterého bude uloženo ID přečtené karty. Funkce postupně čte všechny znaky, které čtečka posílá. V případě, že je přečtenou hodnotou platný znak, tak je uložen.

```
void getRFID(char * save)
{
  flushR = 0; //int flushR
  x = 0; //int x
//Čte ID karty do té doby než přečte 12znaků.
  while (save[11] == '\0')
  {
    i = RFID.read(); //int i
//Do ID karty chceme ukládat pouze znaky jejichž decimální
   hodnota je větší než 47 (čtečka posílá pouze 0-9, A-F).
      if (i > 47)
      {
        save[x] = char(i);
        x++;
      }
  }
//Jakmile je celé ID přečteno, přečte a smaže zbytek znaků, kter
   é čtečka posílá.
  while (flushR < 5)
  {
    i = RFID.read();
      if (i < 0)
        flushR++;
  }
  Serial.print(F("Karta byla načtena, její ID je: "));
  Serial.println(save);
}
```

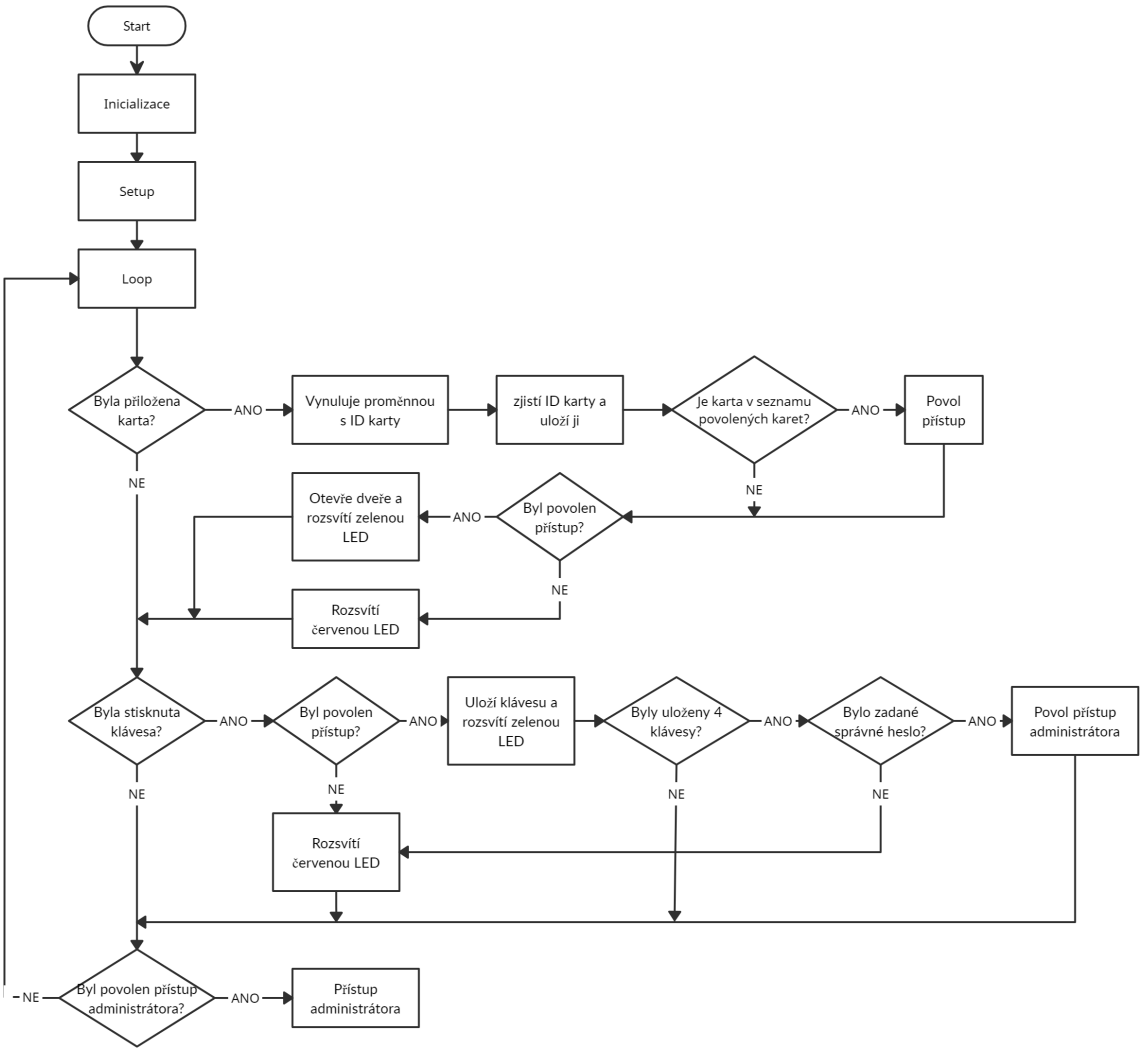
5.6 Autonomní elektronický zámek

Program autonomního elektronického zámku funguje zcela samostatně. To znamená, že celý přístupový systém je řízen přímo ze zámku. Zámek tedy umí číst RFID karty a vyhodnocovat, zda dané kartě přístup udělí nebo ne. Zámek má v programu deklarované dvě administrátorské karty a jejich piny. V případě přiložení této karty a zadání správného pinu se spustí přístup administrátora. Zde je možné zobrazit karty, přidat nové karty a nebo smazat stávající karty. Práce s těmito kartami probíhá v paměti EEPROM. Zámek vypisuje současný stav na sériový monitor.
Ve funkci setup() se nastavují piny pro LED diody, relé a piezoměnič. Také je zde zahájena sériová komunikace pro výpis po sériové lince a čtení RFID. Celá funkce vypadá následovně:

```
void setup()
{
//Zahájení sériové komunikace pro výpis na sériový monitor.
   Rychlost přenosu je nastavena na 9600 bitů/s.
  Serial.begin(9600);
```

```
//Zahájení sériové komunikace pro čtení RFID čtečky. Rychlost př
    enosu musí být nastavena na 9600 bitů/s.
  RFID.begin(9600);
  RFID.listen();
//Piny jsou nastaveny, jako výstup a poté jsou nastaveny na
    logickou úroveň 0.
  pinMode(piezo, OUTPUT);
  pinMode(relay, OUTPUT);
  pinMode(LEDGreen, OUTPUT);
  pinMode(LEDRed, OUTPUT);
  digitalWrite(relay, LOW);
  digitalWrite(LEDGreen, LOW);
  digitalWrite(LEDRed, LOW);
}
```

Na základě požadavků, jak by měl autonomní zámek fungovat, byl vytvořen vývojový diagram. Vývojový diagram graficky znázorňuje jednotlivé kroky vykonávání programu.



Obrázek 5.1 Vývojový diagram autonomního elektronického zámku

Prvním dotazem zámku je, zda byla přiložena karta na čtečku RFID. Pokud byla, tak

zjistí ID karty a pomocí funkce getAccessCard(card) získá informaci o tom, zda je daná karta v paměti nebo ne. V případě, že odpoví ANO, je přístup udělen. Odpověď NE znamená zamítnutí přístupu. Funkce pracuje s ukazatelem na proměnnou card, který se nazývá userID. Funkce vrací datový typ bool (TRUE nebo FALSE).

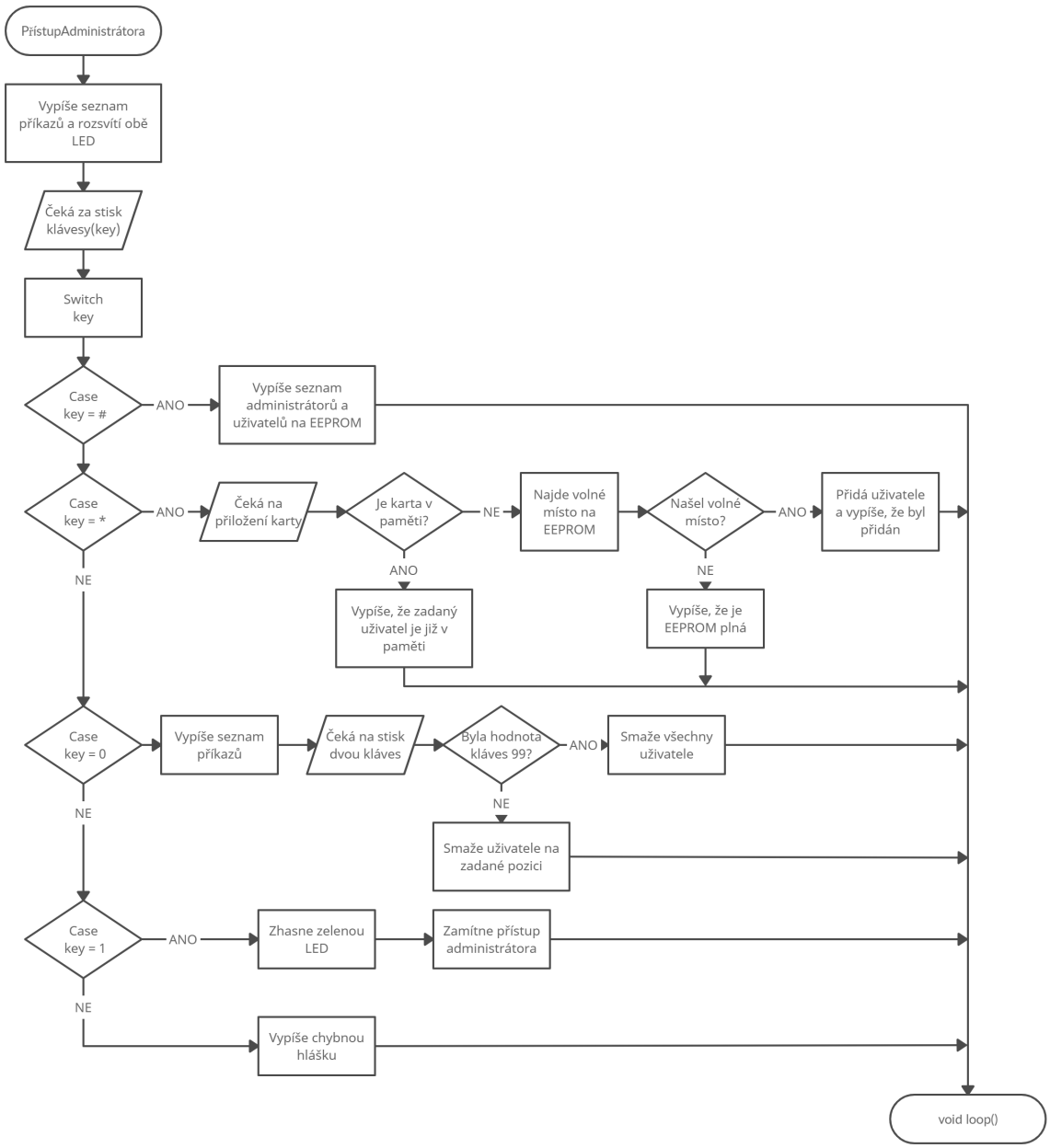
```
bool getAccessCard(char * userID)
{
  accessC = false;
  for(x = 0; x<2 ; x++)
  {
    if (strcmp(userID, LockUsers[x].card_id) == 0)
    {
      accessC = true;
      return accessC;
    }
  }
  x = 0;
  while(x<maxMemoryEEPROM)
  {
    i = x;
    x = x+12;
    pos = 0;
    if(EEPROM.read(i) != 255)
    {
      while (pos < 12)
      {
        readEE[pos] = char(EEPROM.read(i));
        i++;
        pos++;
      }
      if (strcmp(userID, readEE) == 0)
      {
        accessC = true;
        return accessC;
      }
    }
  }
  return accessC;
}
```

Následuje dotaz o tom, zda byla na klávesnici stisknuta klávesa a zároveň, zda byl dříve povolen vstup. Pokud je odpověď TRUE, tak se zadaná klávesa uloží. Potom co se uloží čtyři klávesy, je zavolána funkce giveAccessAdministrator(password, card), která porovnává, zda přiložená karta a zadané heslo souhlasí. V případě, že je odpověď opět TRUE, tak dojde k povolení přístupu administrátora (do proměnné allowAdministrator se uloží hodnota TRUE).

```
bool giveAccessAdministrator(char * password,char * userID)
{
  accessA = false;
//Funkce kontroluje pouze piny administrátorských karet.
  for(x = 0; x<2 ; x++)
  {
    if (strcmp(password, LockUsers[x].pin) == 0 && strcmp(userID
       , LockUsers[x].card_id) == 0)
    {
      accessA = true;
    }
  }
  return accessA;
}
```

5.6.1 Funkce přístup administrátora

Posledním dotazem programu je zda byl povolen přístup administrátora. Dotaz, zda je hodnota proměnné allowAdministrator TRUE nebo FALSE. V případě hodnoty TRUE, se spustí obsáhlá funkce accessAdministrator(). Pro funkci byl vytvořen vlastní vývojový diagram.



Obrázek 5.2 Vývojový diagram přístupu administrátora

Přístup administrátora se signalizuje rozsvícením oranžové barvy na LED diodě (obě diody najednou). Při přístupu administrátora je program pozastaven a čeká na zadání příkazu z klávesnice.
Funkce má celkem čtyři různé příkazy:

Klávesa #: Slouží pro vypsání všech karet. Nejdříve vypíše administrátorské karty včetně jejich pinů a poté všechny karty uživatelů uložených v paměti EEPROM.

```
//Výpis karet administrátorů a pinů na terminál
for(i = 0; i<2 ; i++)
{
  Serial.print(F("Id karty: "));
  Serial.println(LockUsers[i].card_id);
  Serial.print(F("pin karty: "));
  Serial.println(LockUsers[i].pin);
}
x = 0;
//Výpis karet uživatelů v paměti EEPROM na terminál
while(x<maxMemoryEEPROM)
{
  i = x;
  x = x+12;
  pos = 0;
  if(EEPROM.read(i) != 255)
  {
    while (pos < 12)
    {
      readEE[pos] = char(EEPROM.read(i));
      i++;
      pos++;
    }
    Serial.print(F("Index: "));
    Serial.println(i/12-1);
    Serial.print(F("Id karty: "));
    Serial.println(readEE);
  }
}
```

Klávesa 0: Při stisku klávesy 0 dochází k odebírání karet z paměti EEPROM. Funkce zavolá další funkci, která čte stisk dvou kláves. Karty můžou být na pozicích 00 - 20. V případě, že by administrátor chtěl smazat všechny uživatele na EEPROM, stačí zadat číslo 99.

```
//Čte příští dvě tlačítka na klávesnici.
userPos = getUserPosition();
if (userPos == 99)
{
  for(x = 0; x < 241 ; x=x+12)
  {
    EEPROM.update(x,255);
    delay(2);
  }
}
//Uloží hodnotu 255 do prvního znaku karty, při čtení se hodnota
    255 počítá za prázdné místo.
EEPROM.update(userPos*12,255);
```

Klávesa *: Slouží pro přidání nové karty do paměti EEPROM. Nejdříve je potřeba přiložit kartu. Poté co je karta přiložena a je přečteno její ID, tak se pomocí funkce getAccessCard() zkontroluje, zda je kartě udělen přístup nebo ne. Zde platí, že vrácená hodnota FALSE (přístup nebyl udělen) znamená přidání karty, protože karta ještě není v paměti. Karty se vždy ukládají na první volné místo v paměti EEPROM.

```
//Čeká než se přiloží karta.
while (RFID.available() <= 0)
{
}
card[11] = '\0';
delay(20);
getRFID(card);
//Kontroluje jestli je karta již v seznamu. Jestliže není, tak
   se přidá.
if(!getAccessCard(card))
{
  x = 0;
  //Hledá první volné místo na EEPROM.
  while(EEPROM.read(x) != 255 && x<240)
  {
    x++;
  }
  pos = 0;
  //Přidává postupně kartu do EEPROM.
  if(x<240)
  {
    while (pos < 12)
    {
      EEPROM.update(x,card[pos]);
      x++;
      pos++;
    }
    Serial.println(F("Karta byla přidána."));
  }
  else
  {
    Serial.println(F("Pamět EEPROM je plná."));
  }
}
else
{
  Serial.println(F("Karta nebyla přidána, protože je již v pamě
     ti."));
}
```

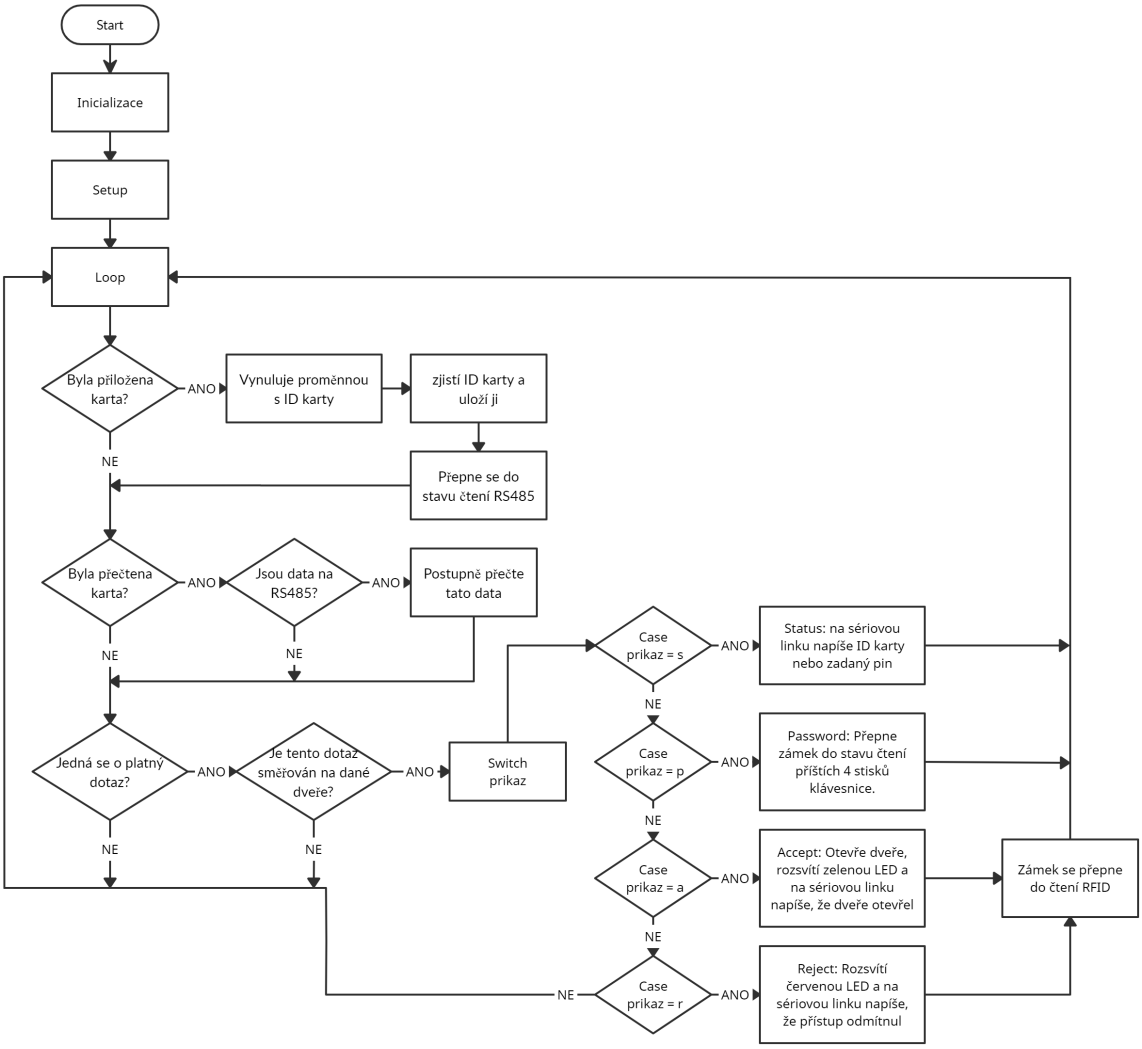
Klávesa 1: Zhasne LED diody a ukončí přístup administrátora tím, že do proměnné allowAdministrator uloží hodnotu FALSE.

```
Serial.println(F("Přístup administrátora ukončen."));
allowAdministrator = false;
digitalWrite(LEDGreen, LOW);
digitalWrite(LEDRed, LOW);
```

Ostatní klávesy: Program vypíše, že pro danou klávesu není žádný příkaz.

5.7 Zámek pro modulární systémy kontroly vstupu

Program tohoto zámku plní příkazy od master zařízení, které jsou vysílány přes sériovou linku RS485. Při vytváření programu bylo zjištěno, že při použití knihovny SoftwareSerial nelze číst data z více software serial portů najednou. To znamená, že zámek může číst pouze z RFID čtečky nebo budiče RS485 sběrnice. To bylo vyřešeno tak, že zámek nejdříve čte RFID čtečku. Jakmile je ke čtečce přiložena karta a zámek ji přečte, tak se přepne do stavu čtení RS485. Pokud se master zařízení dotáže zámku na stav a zámek neodpoví do 10 milisekund, tak se předpokládá, že nebyla přiložena karta. Může se také stát, že zámek neodpovídá, protože nastala chyba ve spojení.



Obrázek 5.3 Vývojový diagram modulárního zámku

V hlavičce programu je zámku při spuštění přiděleno číslo (definice konstant na 31. straně). Pro správnou funkci modulárního systému kontroly vstupu toto číslo musí být pro každý zámek jedinečné. Číslo zámku se určuje hodnotou na DIP spínači, která se čte pomocí funkce getDIP().

5.7.1 Funkce getDIP()

Funkce getDIP() čte jednotlivé spínače na DIP spínači. Převede jejich binární hodnotu na decimální číslo a toto číslo vrátí.

```
int getDip()
{
  port = 0;
  port = (digitalRead(dip1) + digitalRead(dip2)*2 + digitalRead(
     dip3)*4);
  if (analogRead(dip4) > 0)  //analogRead nevrací hodnotu 0 nebo
      1
    port = port + 8;
return port; //Vrátí hodnotu 0 - 15
}
```

Funkce loop() začíná příkazem pro čtení RFID karet, která je stejná, jako u předchozího programu. Další část kódu se dotazuje, zda byla přečtena karta a zda jsou data na RS485. V případě, že je odpověď TRUE, tak dojde k přečtení a uložení těchto dat.

```
//Zatím nebyly přečteny žádné data z RS485 a byla přečtena karta
if(readTerminal[0] == '\0' && card[11] != '\0')
  {
    //Jsou data na budiči RS485
    if(RS485.available())
    {
      delay(1);
      while(readTerminal[3] == '\0')
      {
        character = RS485.read();
        if(character > 0 && character < 125)
        {
          readTerminal[i] = char(character);
          i++;
          //Uloží číslo dveří do proměnné checkDoor
          if(readTerminal[2] != '\0')
          {
            number[0] = readTerminal[1];
            number[1] = readTerminal[2];
            checkDoor = atoi(number);
          }
        }
      }
    }
  }
```

Poté co byla uložena data ze sériové linky, tak dochází k poslednímu dotazu a to, zda je příkaz platný (první přečtený znak musí být *) a zda se shoduje číslo příslušného zámku s číslem zámku v příkazu. Pokud je u obou odpověď TRUE, tak se spustí příkaz switch. Možnosti uvnitř tohoto příkazu jsou podrobněji zobrazeny na obrázku vývojového digramu na obrázku 5.3.

6 OVĚŘENÍ FUNKCE MODULU ZÁMKU

Při vytváření každého programu může dojít k chybám, se kterými programátor nepočítal. Pro odhalení těchto chyb byl proveden velmi důležitý proces testování programu a ověřování, jestli k nějakým chybám dochází. Při testování je ideální vyzkoušet co nejvíce kombinací používání zámku. V případě, že nějaká chyba nastane, tak se musí zjistit, ve které části zdrojového kódu se nachází. Poté co je chyba nalezena, je potřeba provést úpravu kódu k její odstranění. Pro testování programu bylo k dispozici deset RFID karet. Dvě z těchto karet byly u autonomního zámku zvoleny, jako administrátorské.

6.1 Autonomní elektronický zámek

V prvotních verzích programu autonomního zámku byly obsaženy chyby, které zasekávaly program. Postupem času se všechny odhalené chyby podařilo odstranit. V současné době nejsou známy žádné chyby, ke kterým by při chodu programu docházelo.

6.2 Modulární elektronický zámek

Pro ověření funkce zámku a zajištění jeho komfortního ovládání byly v jazyce Python vytvořeny dva programy. Prvním z nich je terminál, který umožňuje ovládání zámku pomocí stisknutí tlačítek. Po stisku jednoho ze čtyř tlačítek se vytvoří a odešle na sériovou linku zpráva. Zprávy, které jsou vytvořeny a odeslány pomocí tlačítek terminálu na obrázku 6.1 (zpráva je zde vysílána na zámek číslo 13) vypadají následovně:

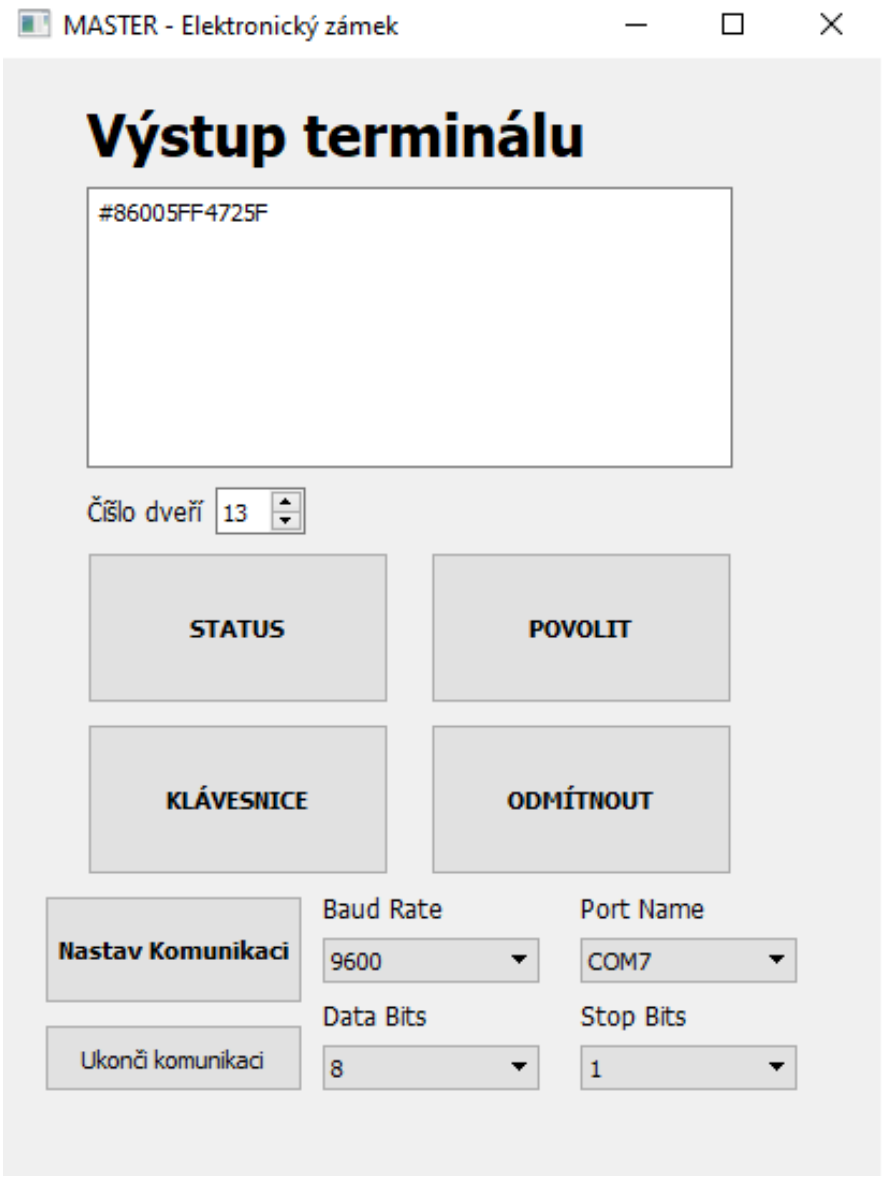
- STATUS - *13s
- POVOLIT - *13a
- ODMÍTNOUT - *13r
- KLÁVESNICE - *13p

Aby bylo možné program ovládat je potřeba nejdříve zahájit komunikaci mezi počítačem a převodníkem TC485. Zde je potřeba nastavit pouze název portu počítače podle toho, do kterého je převodník připojen. Ostatní parametry sériového přenosu jsou přednastaveny na požadované hodnoty. Potom co je komunikace zahájena, tak se může za použití tlačítek začít s vysíláním příkazů na jednotlivé zámky. Výběr čísla zámku nebo dveří se určuje pomocí tzv. spin boxu.
Výstup terminálu vypisuje zprávy, které odeslal zámek jako odpovědi na určitý příkaz. Tyto zprávy začínají vždy mřížkou a obsahují vždy jednu ze čtyř možností odpovědi.

- #IDkarty - Jedná se o odpověď, kterou zámek vypíše na základě příkazu *13s.

- #Pin - Jedná se o odpověď, kterou zámek vypíše na základě příkazu *13s.
- #ACCEPT - Jedná se o odpověď, kterou zámek vypíše na základě příkazu *13a.
- #REJECT - Jedná se o odpověď, kterou zámek vypíše na základě příkazu *13r

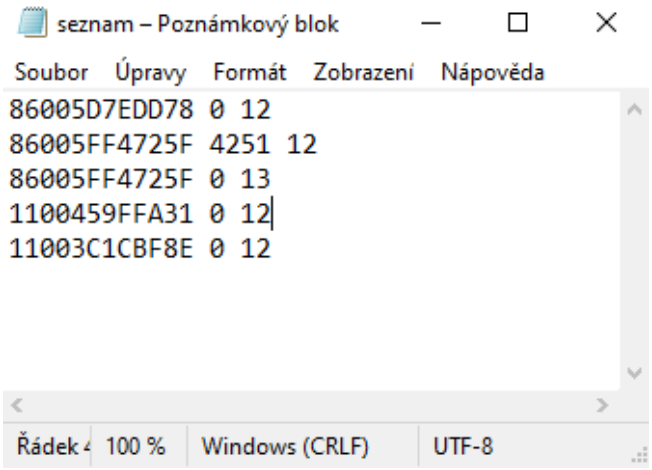
Příkazy ACCEPT a REJECT potvrzují, že byl požadovaný příkaz proveden. Pro příkaz klávesnice zámek nevypíše, žádnou odpověď. Pouze se na zámku zobrazí příslušná indikace. Zadaný pin je možno zobrazit příkazem STATUS. Pokud zámek neodpoví do 10 milisekund, tak se vypíše zpráva, že zámek neodpovídá.



Obrázek 6.1 Vzhled grafického rozhraní prvního programu

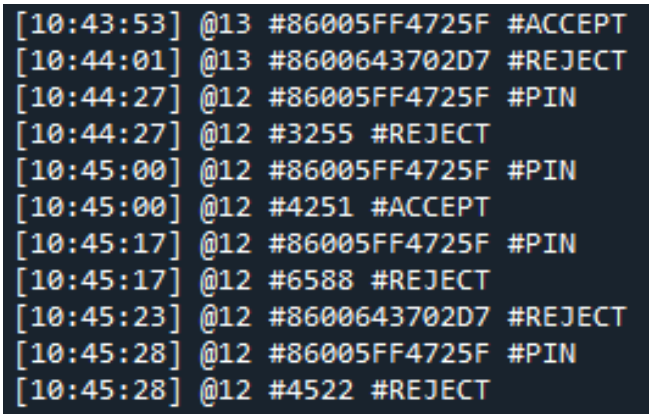
Druhý program vysílá zprávy ve stejném formátu, ovšem funguje samostatně bez jakéhokoli zásahu uživatele. Sériová komunikace je zahájena automaticky, její parametry

lze nastavit v souboru setup.ini. Program se postupně dotazuje všech zámků na stav. V případě, že některý ze zámků na dotaz odpoví přečteným ID karty nebo pinem, tak program na základě informací z textového dokumentu přístup udělí nebo zamítne. Textový dokument na prvním místě obsahuje ID karty, na druhém místě je požadovaný čtyřmístný pin (0 znamená, že se nevyžaduje žádný pin) a poslední je v seznamu číslo dveří. Pokud má jedna karta udělený přístup do více zámků, bude v textovém dokumentu uvedena víckrát.



Obrázek 6.2 Textový dokument pro udělení přístupu

V případě, že došlo ke společné komunikaci mezi zámkem a programem, je výsledek této komunikace vypsán do konzole.
Formát výpisu vypadá následovně: **[čas] @číslo dveří #ID karty #typ příkazu**. Pokud se na zámku zadával pin, tak je vypsán místo ID karty.

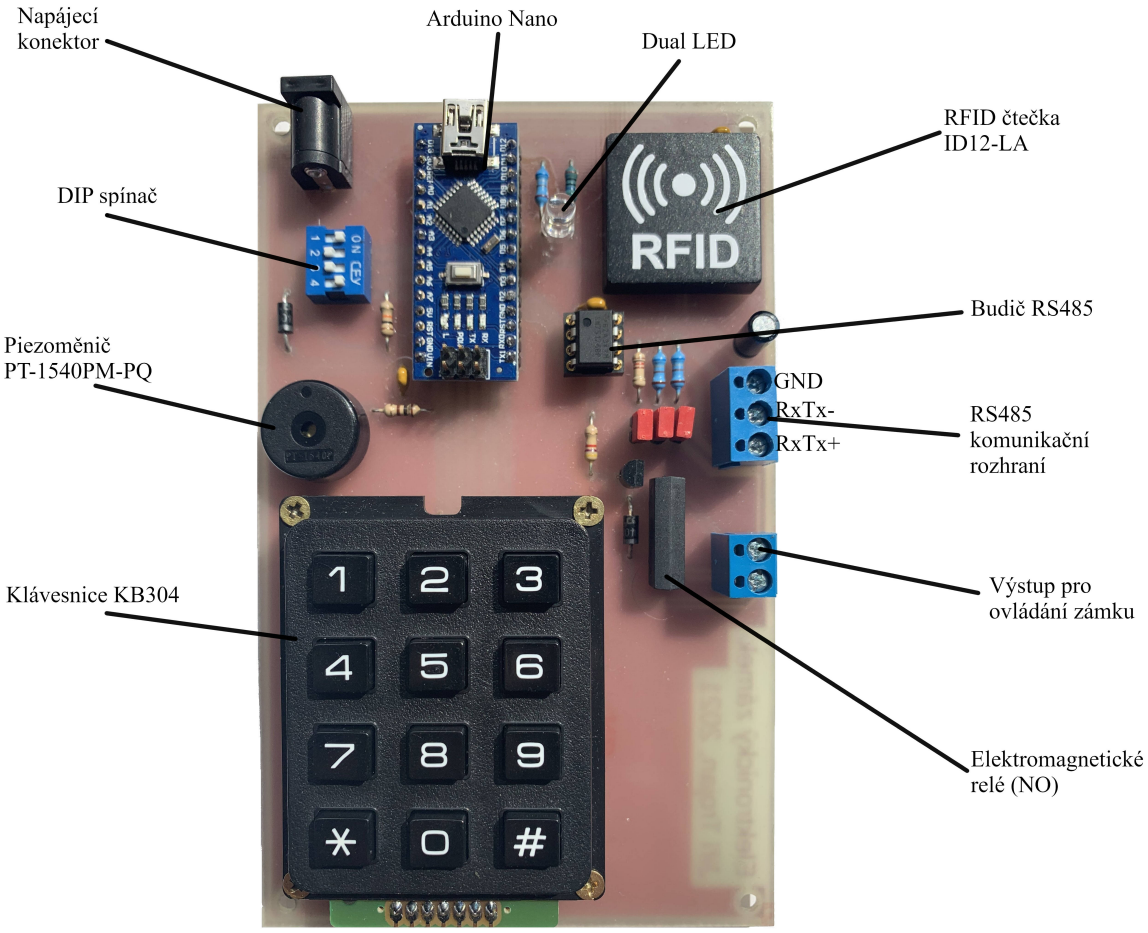


Obrázek 6.3 Výpis konzole druhého programu

V současné době nejsou u modulárního zámku známy, žádné závažné chyby, stejně jako u autonomního zámku.

7 UŽIVATELSKÝ MANUÁL PRO OVLÁDÁNÍ ZÁMKU

Aby bylo možné zámek používat, je potřeba jej připojit ke zdroji napětí o hodnotě 5 V. Zámek může být napájen pomocí power jack kabelu nebo připojením mini USB do Arduina. Při výběru napájení z USB je možné zobrazit výpis programu na sériový monitor.



Obrázek 7.1 Označení jednotlivých součástek umístěných na desce

7.1 Autonomní režim zámku

1. Přiložení karty ke čtečce RFID. V případě, že se jedná o platnou kartu, tak je přístup udělen. Sepne se relé pro otevření dveří a rozsvítí se zelená LED dioda. Po celou dobu otevření dveří je spuštěn piezoměnič (tři sekundy). Přiložením jakékoli platné karty se umožní zadávání pinu na klávesnici. Červená LED indikuje neplatnou kartu.

2. Zadávání pinu slouží pro přístup do režimu administrátora. Existují pouze dvě administrátorské karty, pro které je zadávání pinu relevantní.

3. Přístup administrátora je indikován rozsvícením obou LED diod (oranžová barva). Příkazy v přístupu administrátora se zvolí stiskem příslušných kláves na klávesnici. Seznam možných příkazů:

   - `#` - Vypíše na sériový monitor všechny uživatele.
   - `*` - Slouží k přidání nové karty. Po stisku této klávesy se čeká na přiložení karty ke čtečce. Tato karta se uloží na první volné místo v EEPROM.
   - 0 - Slouží k odebrání uživatele ze EEPROM. Po stisku této klávesy se čeká na stisk dalších dvou kláves, které určují index uživatele v EEPROM.
   - 1 - Ukončí přístup administrátora a program se přepne do stejného stavu jako po spuštění.

7.2 Modulární režim zámku

1. Přiložení karty ke čtečce RFID. Přečtení karty se indikuje rozsvícením zelené LED diody společně se zvukem z piezoměniče.

2. Jestliže master zařízení požaduje zadání pinu, tak se na LED diodě zámku rozsvítí oranžová barva.

3. V případě, že se jedná o platnou kartu, tak je přístup udělen. Sepne se relé pro otevření dveří a rozsvítí se zelená LED dioda. Po celou dobu otevření dveří je spuštěn piezoměnič (tři sekundy). Červená LED indikuje neplatnou kartu nebo pin.

ZÁVĚR

Cílem bakalářské práce bylo navrhnout a sestrojit hardwarové a softwarové vybavení pro výukový modul elektronického zámku, který je určený do předmětu Mikropočítače a PLC. Studenti se pomocí tohoto modulu budou moci seznámit s tím, jak jednotlivé typy elektronického zámku fungují v praxi.

Modul elektronického zámku je založen na mikropočítačové platformě Arduino. Pro modul byl zvolen model desky Arduino Nano, který je pomocí desky plošných spojů propojen s rozšiřujícími moduly. V modulu elektronického zámku jsou obsaženy všechny komponenty, které jsou potřebné k správné funkci obou typů elektronických systémů kontroly vstupu. Tyto typy se dělí na autonomní systémy kontroly vstupu a modulární systémy kontroly vstupu. Při vytváření desky plošných spojů a programového vybavení byly zjištěny nedostatky v návrhu, které způsobily především obtížnější osazování desky. Mezi tyto nedostatky patří:

- Příliš malá velikost padů u pouzdra piezoměniče - problém s usazením součástky do desky a poté obtížné pájení.
- Pady u pouzdra Arduino Nano by mohly být také větší - obtížné pájení.
- Zapojení elektromagnetického relé do pinu D13 - při spouštění Arduina se logická úroveň na tomto pinu několikrát změní z 0 na 1. Docházelo by tedy k odemykání zámku při výpadku elektřiny nebo odpojení Arduina od zdroje napětí a opětovnému připojení (restartování). Tento problém by mohl být vyřešen modifikací bootloaderu mikropočítače.

V případě, že by se tento modul v budoucnu realizoval znova, bylo by ideální s uvedenými nedostatky při návrhu počítat a řešit je lépe.

Pro oba typy zámku bylo vytvořeno odpovídající programové vybavení. Nejdříve bylo vytvořeno pro autonomní zámek. V programovém vybavení pro tento typ zámku je obsaženo velké množství funkcí. Pomocí využití těchto funkcí si mohou studenti zámek naprogramovat sami. V případě, že by některý ze studentů měl zájem, tak by si mohl upravit nebo vytvořit jednotlivé funkce dle jeho představ. Druhým programem je zámek, který by bylo možné použít do modulárních systémů kontroly vstupu. Pro tento program si studenti mohou vyzkoušet vytvořit nejen daný zámek, ale také zařízení, které tento zámek bude ovládat. Při vytváření tohoto programu bylo zjištěno, že knihovna SoftwareSerial neumožňuje číst více portů najednou. Pokud by zámek byl připojen ve velkých modulárních systémech kontroly vstupu, nebylo by možné rychle zjistit, jestli zámek neodpovídá z důvodu problému v připojení, nebo protože nepřečetl žádnou kartu.

SEZNAM POUŽITÉ LITERATURY

[1] ATMEL CORPORATION. Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V datasheet: Technical Data, [online]. 2014 [cit. 2020-11-24]. Dostupné z: http://www.microchip.com

[2] ID INNOVATIONS. ID Series Datasheet: Classic RFID module products, Rev. 28, [online]. 2015 [cit. 2020-11-24]. Dostupné z: http://www.id-innovations.com

[3] MANN, Burkhard.C pro mikrokontroléry: ANSI-C, kompilátory C, spojovací programy – linkery, práce s ATMEL AVR a MSC-51, příklady programování v jazyce C, nástroje pro programování, tipy a triky. Praha: BEN – technická literatura, 2003. μC praxe. ISBN 80-7300-077-6.

[4] VODA, Zbyšek.Průvodce světem Arduina. Bučovice: Martin Stříž, 2015. ISBN 978-80-87106-90-7

[5] JAŠEK, doc. Mgr. Roman, Ph.D. a MALANÍK, Ing. David, Ph.D. Bezpečnost infor-mačních systémů. Zlín, 2013. ISBN 978-80-7454-312-8.

[6] BITARA, Peter. Konstrukce systému kontroly vstupu pomocí Arduina. Zlín: Univerzita Tomáše Bati ve Zlíně, 2017, 54 s. Dostupné také z: http://hdl.handle.net/10563/43271

[7] VALOUCH, Jan. Projektování bezpečnostních systémů [online]. Zlín, 2019 [cit. 2021-04-10]. Dostupné z: https://digilib.k.utb.cz/handle/10563/45863

[8] VALOUCH, Jan. Funkce a požadavky na systémy kontroly vstupu [online]. Zlín, 2014 [cit. 2021-04-14]. Dostupné z: https://publikace.k.utb.cz/handle/10563/1005843

[9] TC485: Převodník RS232 na RS485 [online]. Praha, 2013 [cit. 2021-04-20]. Dostupné z: https://papouch.com/tc485-prevodnik-rs232-na-rs485-p1927/

[10] MARTÍNEK, David. Základní nástroje a pojmy. Jak na projekty v jazyce C [online]. 2017 [cit. 2021-4-24]. Dostupné z: http://jaknaprojekty.davidm.cz/nastroje.html

[11] KELLY, Jason. RS-485 Serial Interface Explained [online]. [cit. 2021-04-01]. Dostupné z: https://www.cuidevices.com/blog/rs-485-serial-interface-explained#what-is-rs-485-

[12] HAVLÍČEK, Jakub. Komunikace v blízkém poli Near Field Communication [online]. Zlín, 2015 [cit. 2021-03-23]. Dostupné z: https://digilib.k.utb.cz/handle/10563/33885. Diplomová. Univerzita Tomáše Bati ve Zlíně.

[13] ARDUINO UNO REV3. In: Arduino store [online]. [cit. 2021-03-21]. Dostupné z: https://store.arduino.cc/arduino-uno-rev3

[14] ARDUINO NANO. In: Arduino store [online]. [cit. 2021-03-21]. Dostupné z: https://www.microchip.com/wwwproducts/en/ATmega328

[15] STANLEY, Mark a Alexander BREVIG. Keypad library for Arduino. In: Github [online]. 2020 [cit. 2021-04-23]. Dostupné z: https://github.com/Chris--A/Keypad

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

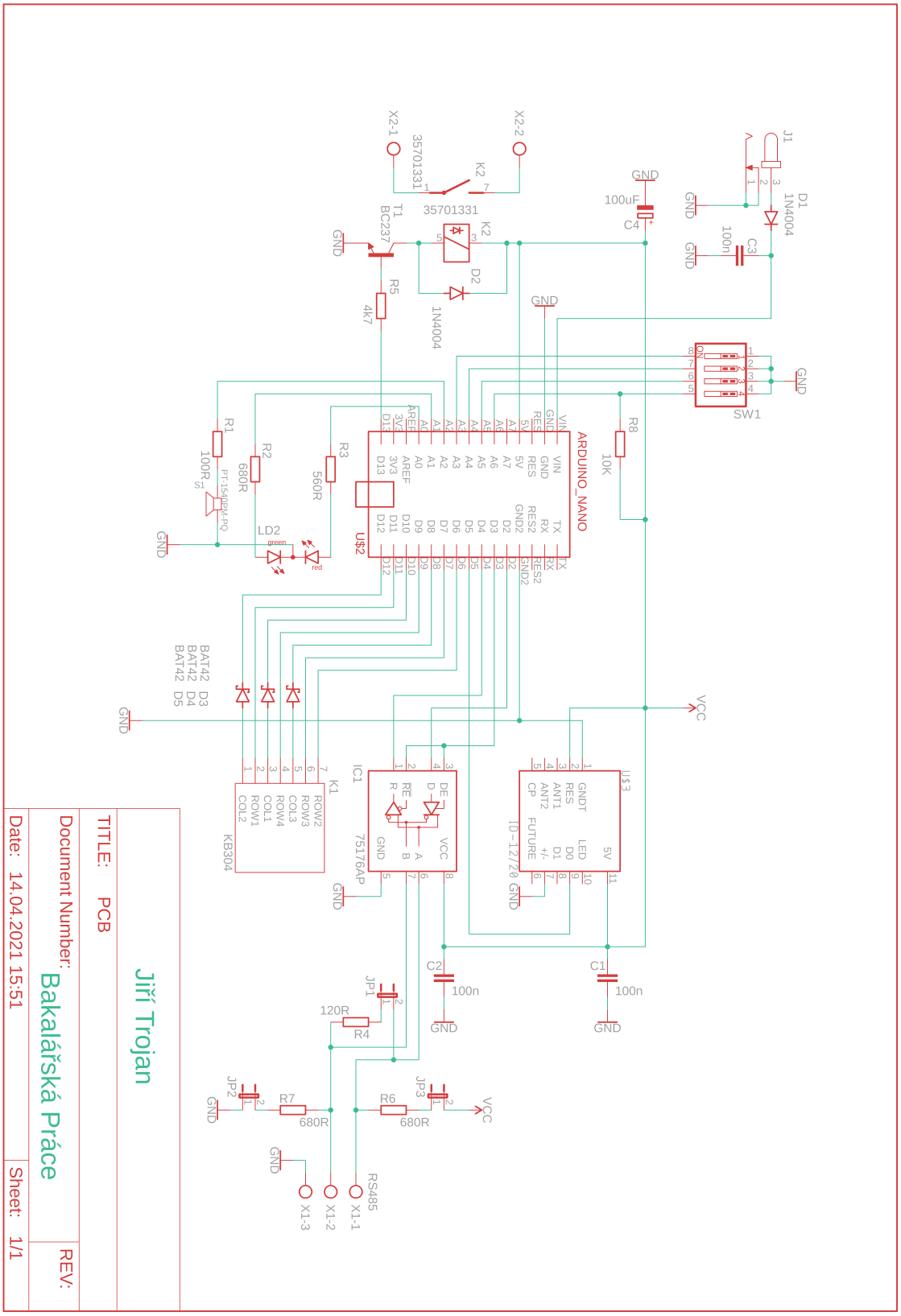
RFID	Radio Frequecy Identification
LED	Light-Emitting Diode
I/O	Input/Output
ČSN	Česká technická norma
CR	Carriage Return
LF	Line Feed
IDE	Integrated development environment
PWM	Pulse Width Modulation
IDE	Integrated Development Environment
ACS	Access Control System
EACS	Electronic Access Control System
VS	Visual Studio
GPIO	General-Purpose Input/Output
UART	Universal Asynchronous Receiver-Transmitter
SW	Software
HW	Hardware
RX	Receive
TX	Transmit
DRC	Design Rule Checking
DIP	Dual In-line Package
NO	Normally open

SEZNAM OBRÁZKŮ

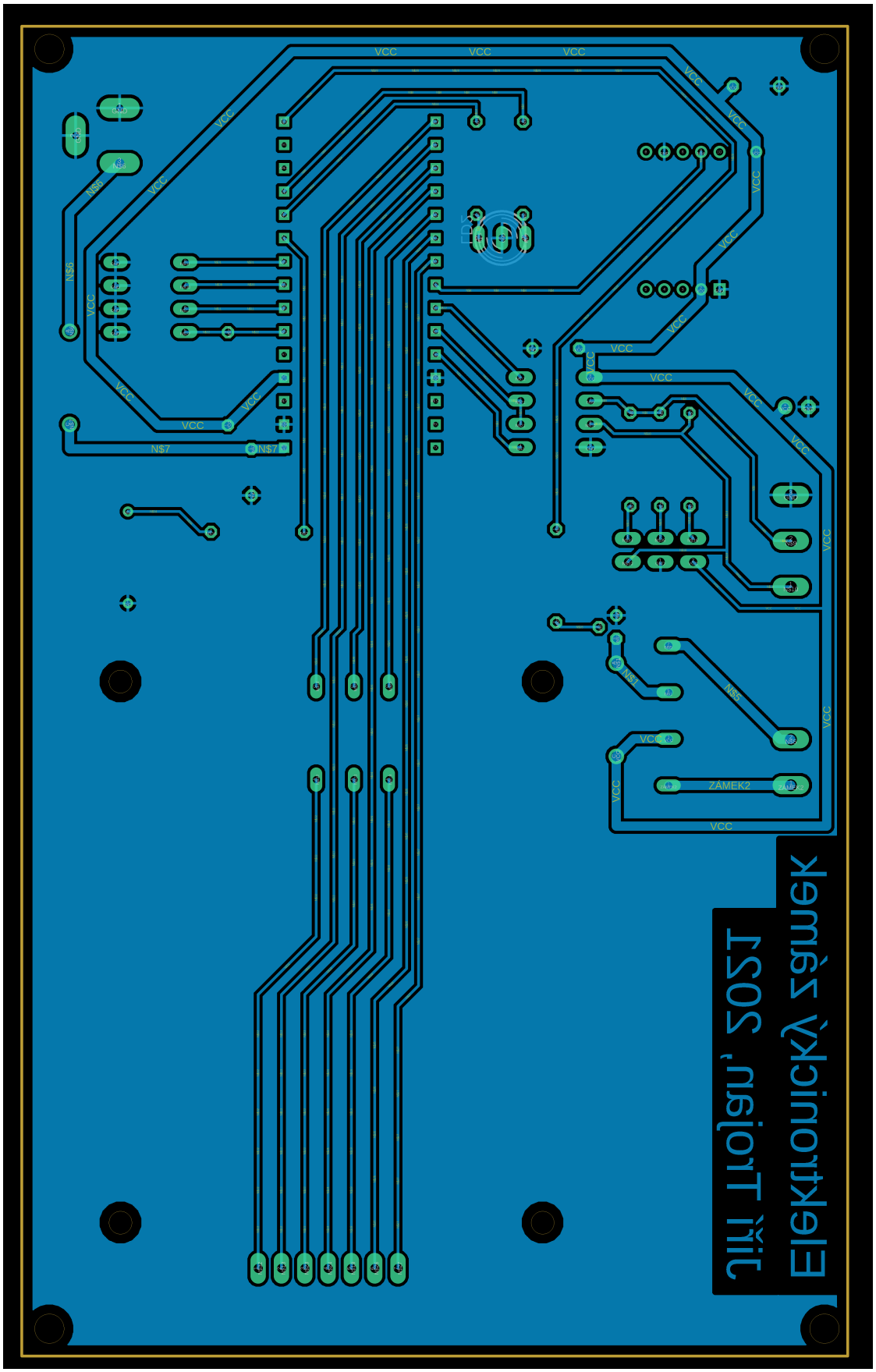
SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I. SCHÉMA ELEKTRONICKÉHO ZÁMKU



PŘÍLOHA P II. MOTIV DESKY PLOŠNÝCH SPOJŮ ZÁMKU



PŘÍLOHA P III. OSAZOVACÍ PLÁN DESKY PLOŠNÝCH SPOJŮ ZÁMKU