TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Návrh řídicích algoritmů jednotky osvětlení experimentálního elektromobilu

Bakalářská práce

Studijní program: B2646 – Informační technologie
Studijní obor: 1802R007 – Informační technologie

Autor práce: **Václav Veleba**
Vedoucí práce: Ing. Pavel Jandura, Ph.D.

Liberec 2016

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

Front-lighting system design for electric vehicle

Bachelor thesis

Study programme: B2646 – Information Technology
Study branch: 1802R007 – Information Technology

Author: **Václav Veleba**
Supervisor: Ing. Pavel Jandura, Ph.D.

Liberec 2016

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: 2015/2016

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Václav Veleba**
Osobní číslo:	**M11000134**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Návrh řídicích algoritmů jednotky osvětlení experimentálního elektromobilu**
Zadávající katedra:	**Ústav mechatroniky a technické informatiky**

Zásady pro vypracování:

1. Seznamte se s možnostmi řešení čelního osvětlení u moderních osobních automobilů, zaměřte se pak zejména na specifika bateriových elektromobilů.
2. Pro systém čelního osvětlení vozidla navrhněte algoritmy řízení moderních funkcí jako jsou např. "trojblik" směrových ukazatelů nebo "corner" funkce mlhových světlometů apod.
3. Zvolte vhodný vývojový kit, který bude plnit funkci řídicí jednotky osvětlení vozu, a navržené algoritmy/protokoly implementujte.
4. Vlastní řídicí jednotka osvětlení bude komunikovat s řídicí jednotkou vozidla po sběrnici CAN.

Rozsah grafických prací:	**dle potřeby dokumentace**
Rozsah pracovní zprávy:	**cca 30–40 stran**
Forma zpracování bakalářské práce:	**tištěná/elektronická**

Seznam odborné literatury:

[1] **TOMSA, Jan. Elektrická zařízení elektromobilu eTUL. Liberec, 2014. Diplomová práce. Technická univerzita v Liberci. Fakulta mechatroniky, informatiky a mezioborových studií**

[2] **Ministerstvo dopravy a spojů ČR. Vyhláška MDS ČR č. 341/2002 Sb. Sbírka zákonů a Sbírka mezinárodních smluv. [Online] [Citace: 20. únor 2014.] http://aplikace.mvcr.cz/sbirka-zakonu/start.aspx**

[3] **TEXAS INSTRUMENTS. Automotive Adaptive Front-lighting System Reference Design: System Application Engineering/MCU. 2013, 42 s. Dostupné z: http://www.ti.com/lit/ug/spruhp3/spruhp3.pdf**

Vedoucí bakalářské práce:	**Ing. Pavel Jandura** Ústav mechatroniky a technické informatiky
Datum zadání bakalářské práce:	**10. října 2015**
Termín odevzdání bakalářské práce:	**16. května 2016**

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

Technická univerzita v Liberci - 14 -

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci dne 10. října 2015

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 16. 5. 2016

Podpis:

Poděkování

Děkuji Ing. Pavlu Jandurovi, Ph.D., vedoucímu mé bakalářské práce, za věcné připomínky a odbornou pomoc.

Poděkování patří také společnosti ČEZ, za ochotné jednání a poskytnutí finančních prostředků na nákup komponent pro tuto práci.

Abstrakt

Cílem této bakalářské práce je návrh řídicích algoritmů jednotky osvětlení experimentálního elektromobilu eŠus. Experimentální elektromobil je společným projektem Fakulty mechatroniky, informatiky a mezioborových studií a Fakulty strojní Technické univerzity v Liberci. Elektromobil slouží k výzkumným a výukovým účelům.

V teoretické části je v jejím úvodu popsán účel čelních světel automobilu. Uvedeny jsou rozdíly mezi různými konvenčními zdroji světla, halogenovými a xenonovými žárovkami. Práce se dále zabývá vysvětlením a popisem LED zdrojů světla, jejich klady i zápory a vývojem, kterým v minulosti tyto zdroje prošly.

Následuje popis aktuálních trendů v automobilové osvětlovací technice. Jsou zde vysvětleny současné moderní funkce, které se v automobilovém průmyslu začaly objevovat. Je objasněn princip adaptabilního systému předních světel Adaptive Frontlighting Systém (AFS), který zajišťuje, že se světlomety přizpůsobí aktuální situaci, tedy počasí či prostředí, ve kterém se vozidlo pohybuje.

V praktické části se práce zabývá návrhem algoritmů. Pro návrh řídicích algoritmů je zvolena open source platforma Arduino. V práci jsou popsány důvody pro výběr právě této platformy, její výhody i nevýhody a porovnání s dalšími možnými variantami. Dále jsou tu i jednotlivé funkce programu pro automatické řízení světel. Hlavní funkce jsou podrobně rozebrány a uvedeny části kódu, psané pro platformu Arduino i s komentáři přímo ve zdrojovém kódu. Dále jsou popsány průběhy měření proudů, intenzity osvětlení a teplot jednotlivých zdrojů. Výsledky měření jsou pro přehlednost vloženy do tabulky. Práce také popisuje jednotlivé kroky k výběru zdrojů osvětlení a důvody, které vedly k výběru konkrétních modelů.

Klíčová slova: osvětlení, řídicí jednotka, algoritmus, Arduino, CAN bus

Abstract

The aim of this thesis is to design controlling algorithm of lighting unit for experimental electric car eŠus. This electric car is common project of Faculty of Mechatronics, Informatics and Interdisciplinary Studies and the Faculty of Engineering of the Technical University in Liberec. Electric car is used for research and educational purposes.

In this thesis are described purpose of the lights in front of cars in the introduction. There are described differences between conventional light sources, halogen's lamps and xenon's lamps. The thesis also deals with the explanation and description of LED light sources, their advantages and disadvantages and development of this sources which have gone through in the past.

This thesis follow with a description of actual trends in automotive lighting technologies. There are explained modern contemporary features that began emerge in the automotive industry. Frontlighting Adaptive System (AFS) which ensure adapting headlamps of the current situation depend of the weather and the environment in which is the vehicle moves is there explained.

Practical part of thesis is focused on a designing of algorithm. For designing of control algorithms was chosen open source Arduino platform. There are described the reasons for choosing this platform, its advantages and disadvantages in comparison with other possibilities. Next the thesis describes the various functions of the program for automatic lighting control. The main functions are described in detail and include parts of code written for the Arduino platform and comments in the source code. There is described the process of measuring currents, light intensity and temperature each of sources. The measurement results were for better clarity inserted in the table. Thesis describes the single steps for the light sources selection and reasons that led to the choice of concrete models too.

Keywords: lighting, control unit, algorithm, Arduino, CAN bus

Obsah

Úvod

Tato práce se zabývá čelním osvětlením experimentálního elektromobilu eŠus. Jedná se o dvoumístné, dvoustopé vozidlo, které slouží k výzkumným a výukovým účelům. Je společným projektem Fakulty mechatroniky, informatiky a mezioborových studií a Fakulty strojní Technické univerzity v Liberci. Jeho samonosný rám je zhotoven z extrudovaných hliníkových stavebnicových profilů, přičemž konstrukce vozidla je taková, že řidič a spolujezdec sedí vedle sebe.

Práce dále popisuje typy zdrojů světla v automobilech. Vzhledem k tomu, že se jedná o elektromobil, u nějž je zdrojem energie baterie s omezenou kapacitou, je práce zaměřena hlavně na moderní zdroje světla, jako jsou LED, které v poslední době prošly velkým vývojem. Za posledních několik let došlo k výraznému zlepšení všech klíčových parametrů, jakými jsou například svítivost nebo účinnost a tudíž již dokáží plně nahradit konvenční zdroje a v mnoha ohledech je dokonce i předčí.

Práce popisuje funkce světel v současných moderních automobilech. Dále se zabývá také současnými trendy v automobilové osvětlovací technice a zmiňuje adaptabilní systém předních světel Adaptive Frontlighting Systém (AFS), který zajišťuje, že se světlomety přizpůsobí prostředí, ve kterém se vozidlo nachází.

Dalším cílem je výběr vhodné platformy pro ovládání světel. Práce popisuje důvody pro výběr právě platformy Arduino, její historii a výhody i nevýhody v porovnání s dalšími platformami. Jsou zde uvedeny příklady kódů programu s vysvětlením jeho jednotlivých částí.

V závěru praktické části je popsán návrh funkčního prototypu pro vozidlo eŠus. Je zde popsán proces výběru jednotlivých zdrojů pro světla automobilu. V práci jsou porovnávány konvenční a LED zdroje světla. Následně jsou popsány průběhy měření různých veličin, tedy proudů, intenzity osvětlení a teplot použitých zdrojů.

1 Čelní osvětlení

Mezi nejdůležitější součásti automobilů patří čelní světlomety. Jsou důležitým vybavením každého vozidla a bez nich je nelze provozovat na pozemních komunikacích. V provozu plní nezastupitelnou funkci, umožňují nám lépe vidět a také zajišťují, abychom i my byli viděni. Jsou tedy součástí aktivní i pasivní bezpečnosti. V posledních několika letech, ostatně jako v mnoha dalších oblastech automobilového průmyslu, pokračoval vývoj osvětlovací techniky mílovými kroky. Tento vývoj značně usnadnil řidičům řízení vozidel v běžném provozu ale i za zhoršených světelných podmínek.

1.1 Typy osvětlení v automobilovém průmyslu

V současné době existuje několik směrů, kterými se vývoj osvětlení ubírá. Tyto směry se snaží uspokojit výrobce a následně i uživatele automobilů v mnoha klíčových oblastech. Mezi nejvýznamnější lze zařadit zvýšení bezpečnosti, zvýšení spolehlivosti, důraz na ekologii, snahu o snížení výrobních nákladů a v neposlední řadě vzhledově se odlišit od konkurence. [1]

1.1.1 Adaptive Frontlighting System

Adaptive Frontlighting System (AFS), je adaptabilní systém předních světel. Tento systém zajišťuje, že se světlomety přizpůsobí aktuální situaci, tedy počasí, směru jízdy, či prostředí, ve kterém se vozidlo pohybuje. To nejvíce oceníme při jízdě v mlze, hustém sněžení nebo za deště. Světlomety jsou tedy konstruovány tak, aby co nejméně omezovaly všechny účastníky provozu, například oslněním nebo odlesky. Nabízejí různé režimy osvětlení, například pro město, dálnici či jízdu v dešti. [2]

Poprvé s úpravami světel přišla v 60. letech automobilka Citroën. Technické řešení bylo založeno čistě na mechanickém principu. V současné době je využíváno složité programové vybavení a moderní řídící elektronika. Systém AFS se snaží řidiči poskytnout co možná nejvíce světla v každém okamžiku a za každé situace. Nejvíce je jeho potenciál samozřejmě využit v noci, ale své uplatnění najde, i pokud se řidič s vozidlem pohybuje v úzkých tmavých uličkách měst nebo za šera na polních cestách mezi vesnicemi. Systém AFS světla automaticky nastavuje a přizpůsobuje podle rychlosti jízdy, směru vozovky a natočení volantu.

1.1.2 Halogenové žárovky

Halogenové žárovky jsou základním a stále hojně využívaným zdrojem světla. Mají v porovnání s konvenčními žárovkami delší životnost, vyšší účinnost a také svítivost. U automobilových halogenových žárovek se podařilo překročit hranici 1000 lm u tlumených a 1600 lm u dálkových světlometů. Skleněná baňka z křemičitého skla je u halogenové žárovky naplněna směsí plynů s příměsemi halových prvků, např. bromem a metylbromidem. Barevná teplota halogenových žárovek se pohybuje okolo 3200 K. Světelný výkon halogenových žárovek je však dosti závislý na velikosti napětí, s jeho poklesem se svítivost významně snižuje. Při vyšším napětí se zase razantně snižuje jejich životnost. Vliv na životnost má i dlouhodobé působení otřesů, což lze při použití v automobilech pouze obtížně eliminovat.

1.1.3 Xenonové výbojky

Zdrojem světla je v xenonových světlometech výbojka, která zaručuje vysokou svítivost a dlouhou životnost. Oproti halogenovým světlometům jsou výrazně dražší, proto se používají u vyšších tříd automobilů.

Světlo ve výbojce vydává elektrický oblouk, který vzniká mezi dvěma elektrodami, umístěnými ve skleněné baňce, naplněné inertním plynem – Xenonem. Pro vznik výboje mezi elektrodami je zapotřebí vysokonapěťový impulz o velikosti několika tisíc voltů, který je generován zapalovacím modulem.

Největší výhodou je bezesporu barevná teplota světla, vydávaná výbojem v xenonovém prostředí. Ta se pohybuje okolo 4 000 K, takže je bližší barevné teplotě denního světla (cca 5 000 K) než halogenová žárovka, která má 3 200 K.

Výbojky se používají z důvodu vyšší účinnosti. Při stejném příkonu dokáží emitovat přibližně dvaapůlkrát více světla v porovnání s halogenovou žárovkou, tudíž mají i vyšší dosvit. Jejich životnost je zhruba šestinásobná.

Xenonové světlomety jsou vyspělým moderním zařízením se spoustou podpůrných systémů, které řídí jejich start, hlídají teplotu, proudy a automaticky nastavují sklon světlometů podle aktuální zátěže vozu. Podle předpisů musí být vybaveny ostřikovači.

Standardní xenonové světlomety v sobě kombinují halogenové žárovky, které jsou použity pro dálková světla a výbojky pro světla potkávací. V novějších bi-xenonových světlometech jsou používány výbojky pro oba typy světel.

1.2 LED technologie

Zavedení LED technologie v oblasti osvětlení automobilů bylo významným posunem v jeho kvalitě a v této oblasti to znamenalo malou revoluci. V porovnání s tradičními světelnými zdroji spotřebovávají diody třetinu elektrické energie. To je jejich hlavní výhodou.

Na počátku roku 2011 uvedla společnost OSRAM ve své tiskové zprávě, že LED světla mohou snížit spotřebu v automobilech, obzvlášť pokud jsou alternátory a generátory uzpůsobeny pro vyšší efektivitu. Výhody LED světel lze názorně předvést právě na elektromobilech. Podle údajů v uvedené tiskové zprávě je možné zvýšit dojezdovou vzdálenost údajně až o 9,5 km. V současné době je kladen důraz na co nejvyšší efektivitu, která souvisí se zmenšováním motorů, snižováním hmotnosti a celkové spotřeby. LED zdroje světla jsou tedy jedním z dalších logických kroků, které pomáhají plnit náročné emisní normy. Běžné halogenové žárovky ve vozidlech mají totiž příkon 240 W, zatímco OSRAM JOULE LED světla pouze 56 W. Problémem však i nadále zůstává výrazně vyšší cena. [3]

Snahou výrobců aut a zejména pak elektromobilů, je snížení energetické náročnosti doplňkových spotřebičů, tedy i osvětlení, které je nutné pro dodržení všech legislativních požadavků a norem pro provoz na pozemních komunikacích. Tím se zvyšuje následný dojezd a prodloužení životnosti hlavních akumulátorů z hlediska dobíjení, energetického vytížení apod. [4]

Již v roce 2003 americké ministerstvo energetiky zpracovalo rozbor, ze kterého vyplynulo, v jaké výši by byla možná úspora nákladů, pokud by byly všechny světelné zdroje v osobních automobilech v USA vyměněny za LED. Potencionální úspora činí až 308 100 litrů paliva. V následující tabulce, Tabulka 1 – porovnání spotřeby energie, je uvedeno porovnání spotřeby energie konvenčních a LED zdrojů.

Tabulka 1 – porovnání spotřeby energie [5]

Automobilová světla podle funkce	Roční doba využití (h/rok)	Příkon konvenčního osvětlení (W)	Roční spotřeba konvenčního osvětlení (Wh)	Příkon osvětlení s LED (W)	Roční spotřeba osvětlení s LED (Wh)
přední potkávací	200	86 až 110	21 600	90 až 130	22 000
přední dálková	30	130 až 240	3 900	90 až 130	3 300
brzdová	200	51,20	10 240	6,0	1 200
koncová	220	15,10	3 322	0,5	110
parkovací	220	14,20	3 115	2,0	440
směrovky	220	6,92	3 045	1,2	264
centrální brzdové	200	25,10	5 016	2,2	440
couvací	25	35,80	895	6,0	150
osvětlení RZ	220	9,48	2 086	2,5	550
denní (20 %)	280	51,20	2 867	3,5	392
celkem			**59 926**		**29 596**

1.3 Dimenzování průřezů vodičů

Dimenzování průřezů vodičů je obecně možné navrhnout s pomocí vzorce:

$$S = \frac{2 \cdot I[A] \cdot l[m]}{\rho\left[\frac{mS}{m}\right] \cdot U[V]}\,[mm^2]$$

Tento vzorec lze však použít pouze v případě, že vodiče budou využívány pro vyšší napětí v řádech desítek voltů. Elektrická soustava elektromobilu má palubní napětí 12 V, což je pro aplikaci vzorce nízké a po dosazení hodnot nám vyjdou velmi zkreslené a nepřesné výsledky. Pro určení průřezů přívodních vodičů k jednotlivým zdrojům světla lze použít tabulku v příloze 1, Tabulka 10 – přehled doporučených průřezů vodiče, která se využívá pro učení průřezů vodičů například pro konstrukci přívěsných karavanů s palubním napětím 12 V.

1.4 Spínací prvky pro osvětlení vozidla

Pod pojmem spínací prvky si lze představit oblast elektrotechnických zařízení, sloužících ke spojení nebo rozpojení elektrického obvodu. Existují 2 varianty: buď elektromechanická, která jsou však dnes již na ústupu, nebo polovodičová relé. Ta jsou určena pro spínání a zesilování digitálních signálů v automatizační technice, stejně jako pro galvanické oddělení mezi řídícím obvodem a obvodem zátěže.

1.4.1 Automobilová spínací relé

Průmyslová relé, používaná v automobilech, jsou navržena speciálně pro všestranné použití v široké řadě náročných aplikací. Velký důraz je kladen na provozní spolehlivost, což minimalizuje náklady na opravy a výměny.

1.4.2 Tranzistor MOS-FET

Z mnoha typů polem řízených tranzistorů se ve výkonové elektronice nejvíce používá IGFET (MOSFET) obohacovacího typu s kanálem N nebo P. Oproti bipolárnímu tranzistoru má dynamické parametry podstatně lepší, rovněž řídicí výkony tohoto tranzistoru jsou menší. Podstatný vliv na dynamické parametry má (díky vstupní kapacitě) rezistor zařazený do obvodu řídicí elektrody. Tranzistor umožňuje rovněž paralelní řazení.

Zkratka MOS-FET (Metal Oxide Semiconductor Field Effect Transistor) označuje typ tranzistoru, řízený elektrickým napětím. Jedná se o elektronickou součástku, obvykle se třemi vývody, která v tomto případě byla použita jako spínač. Tyto vývody mají označení G (Gate – hradlo), D (Drain – kolektor), S (Source – emitor), případně ještě B (Base – substrát), který většinou není vyveden ven z pouzdra, ale uvnitř spojen s vývodem emitoru (S). [6]

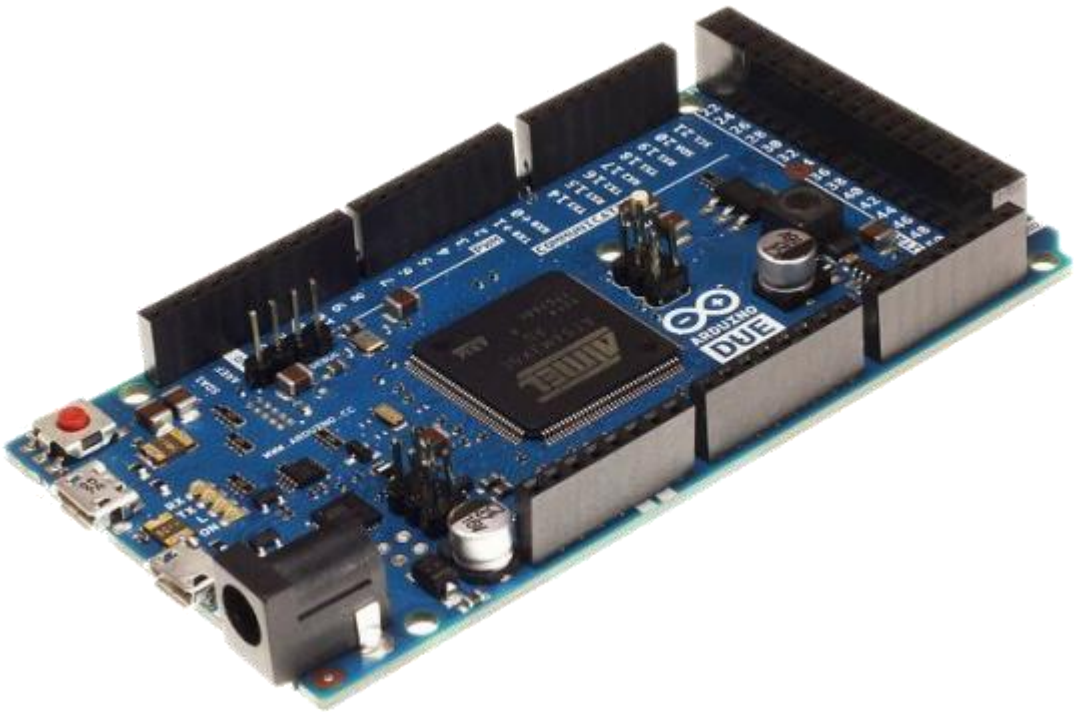
Unipolární tranzistory, mezi které patří i tranzistor MOS-FET, využívají ke své funkci jednoho typu nosičů náboje – elektronů (N MOS-FET) nebo děr (P MOS-FET). Pomocí napětí, přiloženého na kovové elektrodě (hradle) lze ovládat zahnutí energetických pásů v povrchové vrstvě polovodiče a tím i vodivost této vrstvy mezi emitorem a kolektorem. Hradlo je přitom od polovodiče elektricky izolováno. [7]

2 Řídicí jednotka osvětlení vozidla

Řídicí jednotka osvětlení je založena na platformě Arduino. Celý projekt Arduino byl již od svého počátku koncipován jako tzv. opensource. Díky tomu jsou uživatelům k dispozici všechny zdrojové soubory. Je založen na mikro kontrolérech ATMEGA8 a ATMEGA168 od Atmelu. Návrhy těchto modulů jsou vydávány pod licencí Creative Commons. Software Arduino mohou zkušení programátoři snadno rozšířit. Techničtí uživatelé mají možnost přejít do programovacího jazyka AVR C, na kterém je založen. [8]

2.1 Vývojový kit Arduino DUE

Hlavním důvodem pro výběr tohoto modelu byla jeho hardwarová podpora CAN sběrnice a dále to, že poskytuje 54 digitálních vstupně/výstupních pinů, z nichž 16 lze použít jako PWM výstupy. Právě větší počet těchto pinů byl pro projekt důležitý, jelikož bylo nutné fyzicky připojit všechna světla, spínače, senzory a přesto musely zůstat nějaké piny volné pro připojení kontrolek, případně dalších periferií. Model DUE dále obsahuje 12 analogových vstupů s 12bitovým rozlišením, 4 UART (hardwarové sériové porty) a dva DAC výstupy (převaděč digitálního na analogový), 84 MHz krystalový oscilátor, dvě USB spojení, napájecí zdířku, ICSP header, JTAG header a tlačítko reset. Maximální napětí, které mohou I/O piny poskytovat nebo vydržet, je 3,3 V. Při přivedení vyššího napětí (např. 5 V)) do vstupního pinu, může být poškozena deska. Arduino Due, Obr. 1 – deska Arduino DUE, je první deskou, postavenou na 32 bitovém ARM jádrovém procesoru Atmel SA M3X8E ARM Cortex-M3 MCU, který zlepšuje všechny standardy funkcí Arduino a přidává mnoho nových. [9]



Obr. 1 – deska Arduino DUE [13]

Model Due má dva USB konektory. Jeden s mikro USB typ B konektorem a druhý, který je schopen chovat se jako USB host. Může být tedy připojen ke kompatibilní externí USB periferii, k další vývojové desce, myši, klávesnici, nebo chytrému telefonu. Druhý USB port je určen pro programování. Model Due dále obsahuje dva piny označené CANTX a CANRX, které jsou připraveny pro CAN sběrnici.

2.1.1 PWM výstup

Arduino nemá žádný analogový výstup. Místo něj je však k dispozici 16 digitálních výstupů, u kterých lze použít PWM (Pulse Width Modulation), tedy pulzně šířkovou modulaci. PWM modulace se nejčastěji využívá k řízení výkonu různých strojů, především motorů a světel. Střídu výstupního signálu lze nastavit v 255 úrovních od 0 do 100 %. Frekvence výstupního signálu je přibližně 490 Hz. Signál má obdélníkový průběh se střídou D (anglicky označována jako duty cycle) — to je podíl délky kladného pulzu τ a délky jedné periody T. Výkon, a tedy i průměrnou velikost výstupního napětí, lze měnit právě střídou.

2.2 Vývojové prostředí (IDE)

Samotné prostředí je složeno z několika nástrojů. Tím nejdůležitějším, který zabírá největší plochu, je samozřejmě editor pro psaní kódu. Pod ním se nachází okno pro zobrazení zpráv ze sériové linky. V menu nalezneme několik různých nabídek, například pro výběr typu desky, která je připojena, nebo pro výběr sériového portu. Informace o aktuální vybrané desce a sériovém portu, na který je připojena, nalezneme v pravém dolním rohu vývojového prostředí. Vývojové prostředí bylo přeloženo do více než třiceti různých jazyků, včetně češtiny. Defaultně se IDE nahrává v jazyce, který je nastavený v operačním systému. K výchozímu nastavení se lze vrátit výběrem Systém Default v rozevíracím seznamu. Změna se projeví až po restartu Arduino IDE. Pokud změníme nastavení v operačním systému, změna se opět projeví až po restartu softwaru. [10]

3 Program pro řízení osvětlení

Program byl psán v textovém editoru vývojového prostředí Arduino, které obsahuje několik knihoven pro mapování pinů jednotlivých typů desek. Díky tomu je možné používat universální kód pro různé platformy. Vývojové prostředí zároveň za uživatele definuje nastavení vnitřních registrů procesoru, v případě potřeby je však samozřejmě možné registry uživatelem předefinovat. Kódy, které se do textového editoru vývojového prostředí píší, tzv. sketch, mají koncovku *.ino. Na konzoli se pak zobrazují zprávy, obsahující informace o činnosti a běhu programu. Najdeme zde také informace o průběhu nahrávání programu do Arduina nebo chybové hlášky. Pro komunikaci s vývojovou deskou lze použít sériový port. Okno pro zadávání příkazů a zobrazení zpráv od vývojové desky lze vyvolat pomocí rozevíracího menu.

Osobně se textový editor jeví jako jednoduchý, přehledný, optimální pro menší projekty. Pro psaní obsáhlejších programů tu v porovnání s jinými, mnohem sofistikovanějšími editory, chybí funkce našeptávače, která by výrazně urychlila práci při psaní programů. Funkce našeptávače by zároveň značně eliminovala riziko chyb, vzniklých překlepy, či záměnou proměnných, popř. uvedení špatného datového typu v argumentu funkce.

3.1 Knihovny

Knihovny pro platformu Arduino jsou balíčky souborů s kódy, jenž programátorovi zpřístupňují nové funkce. Ty lze po přidání knihovny do vývojového prostředí využívat při psaní programů. Ve vývojovém prostření (Arduino IDE) jsou již některé knihovny obsaženy přímo po instalaci a je možné je ihned začít využívat. Knihovny je většinou potřeba přidat pro práci s rozšiřujícími deskami, které obsahují různé senzory, displeje atd. Vlastní knihovna jsou ve skutečnosti 2 soubory s příponou .cpp a .h, přičemž první obsahuje vlastní kód knihovny a druhý pak předpisy jednotlivých tříd a funkcí. Pokud je potřeba v programu nějakou knihovnu, která je do vývojového prostředí přidána, použít, stačí ji vložit pomocí klíčového slova *#include* a do špičatých závorek zadat jméno souboru. Tím se knihovna automaticky načte a je možné při psaní programu používat její funkce. Do jednoho programu je možné vložit téměř libovolné množství knihoven. Jediné omezení plyne z velikosti vnitřní paměti Arduina, jelikož každá přidaná knihovna výsledný program zvětší.

3.2 Přiřazení jednotlivých pinů

V úvodu programu jsou přiřazeny jednotlivé piny vývojové desky k zástupným názvům. Na začátku je použita direktiva překladače „const“, která zajistí, že se při překladu programu nahradí všechny použité zástupné názvy konkrétní adresou pinu. Pro všechny názvy, nejen proměnných v celém programu, je použito české pojmenování a tzv. „CamelCase“, tedy psaní víceslovných frází způsobem, kdy jednotlivá slova nejsou oddělena mezerami, ale každé z nich začíná velkým písmenem, s výjimkou prvního.

```
//přiřazení na piny Arduino
const byte senzorNatoceniVolantu = A0;
const byte senzorOtacekMotoru = A1;
const byte senzorOsvetleni = A2;

const byte mlhovkaLeva = 2;
const byte mlhovkaPrava = 3;
const byte blinkrLevy = 4;
const byte blinkrPravy = 5;
const byte denniSviceniLeve = 6;
const byte denniSviceniPrave = 7;
const byte tlumenaSvetla = 8;
const byte dalkovaSvetla = 9;

const byte spinacMlhovky = 31;
const byte spinacBlinkrLevy = 33;
const byte spinacBlinkrPravy = 35;
const byte spinacBlinkryVystrazne = 37;
const byte spinacObrysovaSvetla = 39;
const byte spinacTlumenaSvetla = 41;
const byte spinacDalkovaSvetla = 43;

const byte cornerUhelNatoceniVolantu = 10; //úhel
potřebný pro zapnutí funkce Corner, (0-128)
const byte cornerMaxRychlost = 45; //maximální rychlost,
ve které funguje funkce Corner

const byte rychlostZmenyJasu = 15; // 0-255
const int  intervalBlikani = 660; //1000 ms = 1 sec
```

Kód 1 – přiřazení na piny

3.3 Deklarace a inicializace proměnných

Proměnné nám umožňují pojmenování a uložení hodnot k dalšímu užití v programu. Hodnoty v proměnných se na rozdíl od konstant mohou průběžně měnit. Proměnné je nutné deklarovat před jejich prvním použitím v programu. Při deklaraci je také vhodné

přiřadit jim hodnoty, tedy je inicializovat. Není to však nutné, jelikož po deklaraci se do proměnné automaticky zapíše defaultní hodnota, podle datového typu to u čísel bývá hodnota 0 nebo u logického typu boolean hodnota false. Je však dobré je na začátku programu inicializovat ručně. Programátor má tak nad nimi větší kontrolu. Navíc, pokud programátor předtím v prostředí Arduino nepracoval, je lepší na defaultní inicializaci nespoléhat.

Proměnným bychom měli pro lepší orientaci v programovém kódu dávat popisné názvy. Názvy proměnných, které přesně vystihují, co obsahují. Jak již bylo zmíněno, v celém programu jsou použity české, výstižné, názvy. Ty programátorovi i komukoli jinému pomáhají při čtení kódu snadněji rozpoznat, co proměnná znamená. Proměnná může být pojmenována libovolným jménem, které nepatří mezi klíčová slova v jazyce Arduino.

```
//inicializace proměnných
int   rychlost = 0;
int   otackyMotoru = 0;
int   natoceniVolantu = 128;
int   autoBlinkr = 3;

byte  intenzitaMlhovky = 0; //0 - 255
byte  intenzitaDenniSviceni = 128; //0 - 255

unsigned long pocetMilisekund; //Počet milisekund od
začátku běhu programu
unsigned long blikacPredchoziPocetMilisekund = 0;     //
bude ukládat poslední případ aktualizace LED
boolean blikac = false;
```

Kód 2 – inicializace proměnných

Proměnných je v programu požito velké množství, proto je zde pro ukázku uveden pouze začátek bloku, kde se proměnné definují.

3.4 Datové typy

Každá proměnná musí mít definovaný datový typ. Datový typ určuje, jaká data můžeme v proměnné najít. Jedná se o číselné hodnoty, znaky, nebo o logickou hodnotu (pravda, nepravda). Základní datové typy používané v prostředí Arduino jsou uvedeny v následujících kapitolách.

3.4.1 Číselné datové typy

Číselné datové typy se využívají nejčastěji. Vždy je třeba řádně zvážit, jak se bude v programu s proměnnou pracovat a jakých maximálních hodnot bude proměnná nabývat. Může se zdát, že je jedno, jakého typu proměnná bude. Zvolí se ten s největším rozsahem a dále se není potřeba se o nic starat. Částečně je to pravda, ale jen do doby, než se bude pracovat na projektu, ve kterém bude potřeba použít velké množství proměnných. Proměnná každého datového typu totiž zabírá v paměti jiné místo. Podle toho se tedy vybere ten nejvhodnější datový typ, který bude zabírat přijatelně velké místo v paměti, ale nebude program omezovat svým rozsahem. Číselné datové typy lze rozdělit na celočíselné a neceločíselné, jak je uvedeno v následující tabulce Tabulka 2 – celočíselné datové typy.

Tabulka 2 – celočíselné datové typy

Typ	Velikost	Signed	Unsigned
byte	8 b	—	0 až 255
int	16 b	−32768 až 32767	0 až 65535
long	32 b	−2147483648 až 2147483647	0 až 4294967295

V tabulce jsou uvedeny názvy datových typů, jejich velikost, kterou zabírají v paměti a minimální a maximální hodnoty, kterých mohou nabývat. Všechny tyto datové typy existují ve dvou variantách. Se znaménkem „mínus“, ve kterých lze ukládat i záporné hodnoty (signed) a bez znaménka (unsigned). Pokud se při deklaraci v programu vynechá klíčové slovo „signed“ nebo „unsigned“ je tato proměnná defaultně brána jako „signed“, tedy znaménková.

Tabulka 3 – neceločíselné datové typy

Typ	Velikost	Rozsah
float	32 b	$-3{,}40 \times 10^{38}$ až $3{,}40 \times 10^{38}$
double	64 b	$-1{,}7 \times 10^{38}$ až $1{,}7 \times 10^{38}$

Do proměnných, označených jako neceločíselné, uvedených v tabulce Tabulka 3 – neceločíselné datové typy, lze zapisovat i desetinná čísla s přesností na 6 – 7 desetinných míst. Používání neceločíselných proměnných má však jistá úskalí. V paměti zabírají více místa a aritmetické operace trvají daleko déle. Operace s nimi nejsou díky zaokrouhlování zcela přesné. Je tedy lepší, pokud je to možné, použít některý z celočíselných datových typů.

3.4.2 Další datové typy

Mezi další, často používané, datové typy můžeme zařadit ještě typ boolean a char, Tabulka 4 – další datové typy. Boolean v sobě může uchovávat pouze dvě hodnoty, buďto true (pravda), nebo false (nepravda). Datový typ char slouží k uchování jednoho znaku textu. Znak je zde uchován, jako jeho číselná hodnota v ASCII tabulce znaků. Písmena, slova i věty se píší v uvozovkách.

Tabulka 4 – další datové typy

Typ	Velikost	Signed	Unsigned
boolean	8 b	true nebo false	—
char	8 b	−128 až 127	0 až 255

3.5 Hlavní část programu

Hlavní část všech programů pro Arduino se skládá ze dvou nezbytných složek – funkcí. Ty jsou povinné a obě musí být v programu použity vždy. Obě funkce jsou bez návratových hodnot i vstupních proměnných. To znamená, že jsou datového typu void a v závorkách nejsou žádné argumenty funkce.

3.5.1 Funkce setup()

Funkce setup() se provede jen jednou při spuštění programu, následně se cyklicky začne provádět funkce loop(). Nejprve se tedy v programu provedou deklarace veškerých proměnných a poté se teprve může začít vykonávat funkce setup(). V té se provádějí většinou jednorázové operace. Používá se tedy k nastavení jednotlivých pinů jako vstupních, výstupních, nebo k nastavení komunikace Arduina po sériovém portu. Obě dvě nastavení jsou provedena i v tomto programu. Piny, na které jsou připojena jednotlivá světla, se pomocí klíčového slova OUTPUT nastaví jako výstupní, a piny, ke kterým máme připojeny kontakty od spínačů, jsou nastaveny jako vstupní.

```
void setup() {
  pinMode (mlhovkaLeva, OUTPUT);
  pinMode (mlhovkaPrava, OUTPUT);
  pinMode (blinkrLevy, OUTPUT);
  pinMode (blinkrPravy, OUTPUT);
  pinMode (denniSviceniLeve, OUTPUT);
  pinMode (denniSviceniPrave, OUTPUT);
  pinMode (tlumenaSvetla, OUTPUT);
  pinMode (dalkovaSvetla, OUTPUT);

  pinMode (spinacMlhovky, INPUT);
  pinMode (spinacBlinkrLevy, INPUT);
  pinMode (spinacBlinkrPravy, INPUT);
  pinMode (spinacBlinkryVystrazne, INPUT);
  pinMode (spinacTlumenaSvetla, INPUT);
  pinMode (spinacDalkovaSvetla, INPUT);

  Serial.begin(9600);
}
```

Kód 3 – funkce setup()

3.5.2 Funkce loop()

Funkce loop() následuje ihned po funkci setup(). V těle této funkce je nejobsáhlejší část programu. Kód v těle této funkce je opakovaně prováděn v nekonečné smyčce. V této části se vykonávají nejdůležitější části programu, jako například čtení vstupů, nastavování výstupů atp.

3.5.2.1 Ovládání směrových světel

Směrová světla je možné provozovat v několika režimech. Základním je blikání v režimu výstražných světel. Výstražná světla je možné zapnout vždy, bez ohledu zda je vozidlo v pohybu či je zapnuté zapalování. V režimu výstražných světel blikají obě strany, tedy levý i pravý blinkr zároveň. Směrové ukazatele je dále možno trvale zapnout. V jednom okamžiku svítí pouze jedna strana a to přerušovaným světlem se zvolenou střídou. Směrová světla jsou doplněna ještě o funkci tzv. komfortního blikání, kdy stačí sepnout spínač pouze na krátkou chvíli a komfortní blikání zajistí, že se směrový ukazatel rozsvítí a zhasne několikrát za sebou. Počet bliknutí je možné nastavit hodnotou v proměnné komfortniBlikani.

```
if (zapnutoBlinkryVystrazne) {
    digitalWrite(blinkrLevy, blikej(intervalBlikani));
    digitalWrite(blinkrPravy, blikej(intervalBlikani));
  } else {
    if (zapnutoBlinkrLevy || (komfortniBlikaniLevy > 0))
    {
      digitalWrite(blinkrLevy, blikej(intervalBlikani));
      zapnutoBlinkrPravy = false;
      komfortniBlikaniPravy = 0;
    }
    else
    {
      digitalWrite(blinkrLevy, LOW);
    }
    if (zapnutoBlinkrPravy || (komfortniBlikaniPravy >
0))
    {
      digitalWrite(blinkrPravy,
blikej(intervalBlikani));
      zapnutoBlinkrLevy = false;
      komfortniBlikaniLevy = 0;
    }
    else
    {
      digitalWrite(blinkrPravy, LOW);
    }
  }
```

Kód 4 – ovládání směrových světel

3.5.2.2 Adaptivní systém čelního osvětlení

Adaptivní systém čelního osvětlení ovládá intenzitu LED pásků, které mohou být provozovány ve dvou režimech. Při dostatečném okolním osvětlení plní funkci denního svícení. Při poklesu intenzity okolního osvětlení se automaticky sníží jejich jas a slouží jako světla obrysová.

```
if (zapnutoDenniSviceni)
  {
    if (sero)
    {
      analogWrite(denniSviceniLeve, 128);
      analogWrite(denniSviceniPrave, 128);
      zapnuto
    }
    else
    {
      analogWrite(denniSviceniLeve, 255);
      analogWrite(denniSviceniPrave, 255);
    }
  }
  else
  {
    analogWrite(denniSviceniLeve, 0);
    analogWrite(denniSviceniPrave, 0);
  }
```

Kód 5 – funkce adaptivního ovládání

3.5.3 Uživatelsky definované funkce

Programátor si může kromě běžně dostupných vestavěných funkcí napsat i svou vlastní, uživatelsky definovanou. Každá funkce má nějaké parametry, z nichž některé jsou povinné a musí být uvedeny vždy a některé jsou nepovinné a záleží na programátorovi, zda je definuje či nikoliv.

Každá funkce má svůj datový typ. Datový typ se volí podle druhu dat, která chceme, aby funkce vracela. Datový typ se musí určit vždy, i když funkce žádná data nevrací. Musí se definovat jako „void“. To je speciální datový typ právě pro funkce bez návratových hodnot.

V uživatelsky definované funkci je možno deklarovat lokální proměnné, které platí pouze v těle této funkce. Samozřejmě je možné pracovat i s globálními proměnnými. Definice

samotné funkce se musí provést mimo tělo ostatních funkcí. Nesmí být tedy definována ani ve funkci setup() a loop(), ale mimo ně. Na konkrétním umístění, zda před, za, nebo mezi nimi nezáleží, ale pro přehlednost kódu se obecně doporučuje je umisťovat na konec programu.

Při definování vlastní funkce je tedy nejprve nutné vědět, zda bude vracet nějakou hodnotu a podle toho zvolit void, či některý standardní datový typ. Poté následuje název funkce, který může být libovolný, ovšem jednoslovný bez mezer a nesmí obsahovat názvy, které patří mezi klíčová slova v jazyce Arduino.

```
//čtení zmáčknutí tlačítka
boolean ctiTlacitko(byte tlacitko,  boolean
predchoziStavTlacitka, boolean zapnuto) {
  boolean zapni = zapnuto;
  soucasnyStavTlacitka = digitalRead(tlacitko);
  delay(5);

  if (soucasnyStavTlacitka == LOW &&
predchoziStavTlacitka == HIGH && zapnuto == HIGH) {
    zapni = false;
  }
  if (soucasnyStavTlacitka == LOW &&
predchoziStavTlacitka == HIGH && zapnuto == LOW) {
    zapni = true;
  }
  return zapni;
}
```

Kód 6 – funkce čtení stisku tlačítka

Jednou z uživatelsky definovaných funkcí je čtení zmáčknutí tlačítka. Je datového typu boolean a pojmenována ctiTlacitko. Má tři vstupní parametry. Číslo vstupního pinu, předchozí stav tlačítka a stav příslušného světla. Dále je zde příkaz digitalRead() pro čtení stisku a funkce delay(), která zajistí krátké zpoždění, aby se odstranily případné zákmity tlačítka. Funkce zajišťuje, aby program reagoval na náběžnou hranu při stisku tlačítka a změnil stav booleovské proměnné „zapni“ na opačný. Tuto proměnnou pak vrací jako návratovou hodnotu funkce.

```
//Blikač střídající stavy se střídou dle zadaného
intervalu
boolean blikej(int interval) {

  if (pocetMilisekund - blikacPredchoziPocetMilisekund >
interval) {
    blikacPredchoziPocetMilisekund = pocetMilisekund;

    //změna stavu
    if (blikac == LOW)
    {
      blikac = HIGH;

    }
    else
    {
      blikac = LOW;
      if (komfortniBlikaniLevy > 0)
      {
        komfortniBlikaniLevy = komfortniBlikaniLevy - 1;
      }
      if (komfortniBlikaniPravy > 0)
      {
        komfortniBlikaniPravy = komfortniBlikaniPravy -
1;
      }
    }
  }
  return blikac;
}
```

Kód 7 – funkce blikač

Dále je tu funkce tzv. blikače. Ta zajišťuje pravidelné střídání dvou stavů v zadaném intervalu. Blikání není možné pomocí funkce delay(), jelikož během čekání se zastaví vykonávání dalších operací, respektive procesor provádí pouze NOP instrukce. To má za následek, že nezpracovává další kód a není schopen například reagovat na stisky tlačítek, nebo rozsvěcet další světla. Základní myšlenkou funkce blikej() je nechat cyklus funkce co nejrychlejší a brzdit jej pouze vykonáváním akcí, pokud je to potřeba. Kód je sice o něco složitější, je potřeba proměnné na to, aby si pamatovala časy, ale běh je potom plynulý, bez zastavování. Musíme použít funkci millis(), pomocí které se dá zjistit hodnota, uložená ve vnitřním časovači procesoru. Zde je uchována informace o délce běhu programu od jeho spuštění. Tato funkce tedy nepotřebuje žádný parametr a vrací počet milisekund od začátku programu. Tento počet však není nekonečný. Po překročení dojde k takzvanému přetečení časovače, který poté znovu začne počítat od nuly. K tomu

dojde zhruba jednou za 50 dnů nepřetržitého běhu, Jelikož maximální vrácená hodnota je 4 294 967 295 ms. Funkce millis() se využívá právě tam, kde je třeba čekat, ale není žádoucí, aby byl přerušen chod programu, jako je tomu v tomto případě.

Dále je zde implementována funkce komfortního blikání, tedy že blinkr, bez ohledu na to, jak dlouho je sepnut spínač, blikne minimálně 3x. Počet bliknutí se dá nastavit v konstantě na začátku programu.

3.5.3.1 Funkce Corner

Funkce Corner slouží pro přisvětlování do zatáček mlhovými světlomety. Této funkce se využívá především v městském provozu, proto je její spuštění omezeno maximální rychlostí. Pro zapnutí světla je dále nutné výraznější vybočení z přímého směru. Její spuštění je tedy podmíněno natočením volantu. Obě tyto hodnoty, jak rychlost, tak míru natočení volantu lze nastavit příslušnými konstantami na začátku programu.

```
if (zapnutoMlhovky) {
    analogWrite(mlhovkaLeva, 255);
    analogWrite(mlhovkaPrava, 255);
  }
  else {
    analogWrite(mlhovkaLeva, intenzitaMlhovkaLeva);
    analogWrite(mlhovkaPrava, intenzitaMlhovkaPrava);
    if (rychlost < cornerMaxRychlost) {
      if (natoceniVolantu > 512 +
cornerUhelNatoceniVolantu) {
        if (intenzitaMlhovkaPrava < 255) {
          intenzitaMlhovkaPrava = intenzitaMlhovkaPrava
+ abs(rychlostZmenyJasu);
        }
      }
      else {
        if (intenzitaMlhovkaPrava > 0) {
          intenzitaMlhovkaPrava = intenzitaMlhovkaPrava
- abs(rychlostZmenyJasu);
        }
      }
      if (natoceniVolantu < 512 -
cornerUhelNatoceniVolantu) {
        if (intenzitaMlhovkaLeva < 255) {
          intenzitaMlhovkaLeva = intenzitaMlhovkaLeva +
abs(rychlostZmenyJasu);
        }
      }
      else {
        if (intenzitaMlhovkaLeva > 0) {
          intenzitaMlhovkaLeva = intenzitaMlhovkaLeva -
abs(rychlostZmenyJasu);
        }
      }
    }
    else {
      intenzitaMlhovkaLeva = 0;
      intenzitaMlhovkaPrava = 0;
    }
  }
```

Kód 8 – funkce Corner

Mlhové světlomety jsou připojeny na výstupy Arduina s pulzně šířkovou modulací. Aby rozsvícení mlhového světla nebylo náhlé a nedalo se zaměnit například se světlem blinkru, používá se u funkce corner postupného zvyšování intenzity. Aby bylo možné považovat rozsvěcení za plynulé, je potřeba použít frekvenci nad časovou rozlišovací schopností lidského oka. Ta je dána především rychlostí vedení vzruchů a jejich zpracování (tzv. setrvačností oka). Kmitočet splynutí, tedy frekvence jednotlivých

záblesků, které oko ještě rozliší jako blikání a nad níž není blikání pro dané podmínky již vnímatelné, závisí především na maximálním jasu, trvání osvětlení, tvaru náběhu světla a úhlu pozorování. Obvykle se pohybuje kolem 50 Hz. Pokud tedy bude frekvence vyšší, bude platit Talbotův zákon, který říká, že vjem nad kmitočtem splynutí je stejný jako při působení trvalého podnětu se stejnou průměrnou energií. Z těchto důvodů je konstanta rychlosti změny jasu nastavena na hodnotu 5, což zhruba odpovídá mezní frekvenci 50 Hz a i dle subjektivního názoru je změna jasu opravdu plynulá.

3.6 Simulace posílání dat od řídicí jednotky

V době dokončení práce nebyla k dispozici řídicí jednotka vozidla. Tudíž bylo nutné komunikaci nasimulovat. Pro simulaci posloužila druhá deska Arduino Due, k níž byly připojeny spínače a proměnné odpory, simulující např. zasunutí klíčku v zapalování, snímač polohy volantu nebo senzor intenzity osvětlení. Data, získaná z těchto senzorů jsou vyhodnocována a posílána desce, řídící osvětlení.

```
void setup() {
  pinMode(13, OUTPUT);

  pinMode(klicek, INPUT);
  Serial.begin(9600);
  Serial1.begin(9600);
}
void loop() {
  zapnutyKlicek = digitalRead(klicek);
  //otackyMotoru = analogRead(senzorOtacekMotoru);
  otackyMotoru = 20;
  natoceniVolantu = analogRead(senzorNatoceniVolantu);
//0-1023
  intenzitaOsvetleni = analogRead(senzorOsvetleni);

  Serial1.println(zapnutyKlicek);
  Serial1.println(otackyMotoru);
  Serial1.println(natoceniVolantu);
  Serial1.println(intenzitaOsvetleni);
  delay(100);
}
```

Kód 9 – kód simulující řídicí jednotku

Po dokončení řídicí jednotky vozidla by komunikace měla probíhat po CAN sběrnici. Arduino DUE má její hardwarovou podporu. Obsahuje celkem dvě sběrnice, CAN0 a CAN1. Pro sběrnici CAN0 jsou přiřazeny piny CANTX, CANRX a pro sběrnici CAN1 piny DAC0 a 53. Příkaz pro odesílání dat je CAN.sendFrame(), jehož jediným parametrem je proměnná frame datového typu struct. Ta obsahuje informace jako ID, zda se jedná o rozšířený rámec, nebo velikost dat, která se budou posílat. Vlastní data můžeme zapisovat po bytech příkazem frame.data.Byte[], nebo po horních a dolních čtyřech bajtech příkazem frame.data.HIGH případně frame.data.LOW. Přenosové rychlosti sběrnice se nastavují příkazem CAN. begin().

4 CAN sběrnice

V nedávné minulosti se rapidně zvýšily požadavky automobilového průmyslu na bezpečnost, ekologii, nízkou spotřebu a emise, ale i komfortní prvky. To s sebou nese mimo jiné i používání různých elektronických systémů, které mezi sebou potřebují komunikovat, předávat si informace o rychlosti jízdy, otáčkách motoru, teplotě a mnoha dalších údajích. Pro komunikaci mezi jednotlivými elektronickými systémy bylo důležité je spojit univerzální sběrnicí, která bude moci předávat všechny potřebné informace.

CAN sběrnice byla vyvinuta v roce 1983 firmou Robert Bosch GmbH a byla koncipována pro použití v automobilovém průmyslu. Poprvé byla užita v sériové výrobě o tři roky později, kdy se při zabudování do vozu BMW 850 coupe, podařilo konstruktérům docílit úspory 2 km kabelů a hmotnost vozidla se snížila o 50 kg oproti případu, kdy by bylo použito propojení pomocí standardních vodičů. CAN sběrnice se brzy stala standardem, který je používán v celém automobilovém průmyslu, a jeho výhod využívají všechny současné automobilky. Sběrnice se postupem času rozšířila i do ostatních odvětví průmyslové automatizace. Je to způsobeno tím, že je celosvětově normalizovaná, jednoduše rozšiřitelná a pro své produkty ji používají různí výrobci.

4.1 Výhody CAN

Výhodou CAN sběrnice je jednoznačně významně nižší množství kabelů. To s sebou nese i snížení hmotnosti, nákladů i prostoru potřebného pro vodiče. Dalším pozitivem je snadná rozšiřitelnost, vyplívající z podstaty CAN sběrnice. Stala se již standardem a je podporována všemi výrobci automobilů i výrobci doplňků a příslušenství, jejichž implementace probíhá na softwarové úrovni. V neposlední řadě přináší úsporu času i servisním technikům, jelikož všechny chybové stavy systémů vozu, napojených na tuto sběrnici, lze diagnostikovat naráz. Není tedy třeba kontrolovat každé zařízení zvlášť a tím je možné práci výrazně zefektivnit.

4.2 Princip fungování

Sběrnice je fyzicky nejčastěji realizována dvěma vodiči, označenými CAN_L a CAN_H. Po sběrnici je možné přenášet dvě úrovně, dominantní a recesivní. Tyto dva stavy jsou reprezentovány rozdílovým napětím mezi vodiči. Nominální hodnoty jsou uvedeny

v normě ISO 11898. Ta udává pro recesivní úroveň velikost rozdílového napětí V_{diff} = 0 V. V dominantním stavu je na vodiči CAN_H napětí v rozsahu 3,5 V až 5 V, na vodiči CAN_L napětí v rozsahu 0 až 1,5 V, pro dominantní úroveň by rozdílové napětí V_{diff} mělo být 2 V.

4.2.1 Rušení a ochrana

Ve vozidlech se nachází mnoho součástí, které mohou působit jako zdroje rušení. Většinou se jedná o součásti spínající elektrický obvod nebo vydávající jiskry, ale také zdroje elektromagnetického vlnění, jako třeba mobilní telefony. Všechny tyto rušivé zdroje mohou být příčinou nečitelnosti nebo dokonce pozměnění přenášených dat. Aby se těmto nepříznivým vlivům co nejvíce zamezilo v ovlivňování přenášených dat, je vedení tvořeno dvěma kroucenými vodiči. Po těchto vodičích se přenáší diferenciální signál, takže se na vedení nachází vždy opačné napětí, kdy na jednom vodiči je 0 V a na druhém 5 V. Pokud se signál nepřenáší, je napětí na obou vodičích stejné a má hodnotu 2,5 V. Takto je možno zajistit, aby měl součet napětí v každém okamžiku stejnou hodnotu. Tím je zajištěna ochrana proti vnějším rušivým vlivům, jelikož elektromagnetická pole obou vodičů se vzájemně vyruší a vodič se vůči svému okolí chová neutrálně.

4.2.2 Součásti datové sběrnice CAN ve vozidle

CAN sběrnice je složena z několika částí: řadiče, vysílače, vodičů a zakončovacích odporů. Řadič sběrnice zpracovává informace, které mají být dále posílány. Řadič je připravuje, předává vysílači a současně od vysílače data také přijímá. Přijatá data zpracuje a předá mikropočítači v řídicí jednotce. Vysílač má za úkol vysílat a přijímat data, která ve formě elektrických impulzů prochází vodiči sběrnice. Impulzy na sběrnici jsou přijímány všemi zařízeními, která jsou na ni připojena. Na koncích sběrnice jsou umístěny 120Ω odpory, které zajišťují, aby se elektrické impulzy neodrážely zpět a nedocházelo k interferencím.

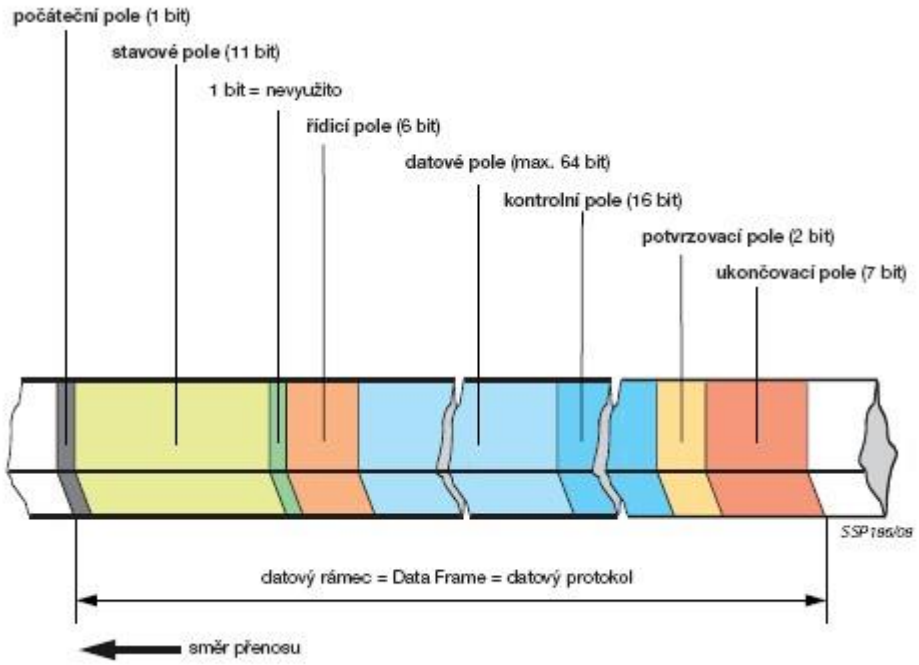
4.3 Datový přenos

Datový rámec se mezi řídicími jednotkami přenáší po CAN sběrnici v krátkých časových intervalech. Data se vysílají z řídicí jednotky, která předává údaje, jež mají být poslány, řadiči. Ten je dále předá vysílači, který je zpracuje na sériové elektrické impulzy a vyšle

po sběrnici. Všechny řídicí jednotky, připojené na sběrnici, vyslaná data přijmou a vyhodnotí, zda jsou pro ně určena. Pokud jsou data některou z jednotek vyhodnocena jako potřebná, tak je jednotka převezme a zpracuje. Pokud pro jednotku potřebná nejsou, tak na ně nereaguje.

4.3.1 Datový protokol

Datový protokol je složen z několika po sobě následujících bitů, jejichž počet je závislý na velikosti datového pole. Každý bit má vždy hodnotu buď 0, nebo 1. Pomocí číslic 0 a 1 lze v binární (dvojkové) číselné soustavě vyjádřit jakékoliv číslo. Datový protokol je tvořen vždy dle totožného datového rámce, data frame, Obr. 2 – datový rámec. Tento rámec obsahuje sedm datových polí.



Obr. 2 – datový rámec [17]

4.3.2 Popis polí

Počáteční pole (Start of Frame)

Počáteční pole označuje počátek datového protokolu.

Stavové pole (Arbitration Field)

Ve stavovém poli je stanovena priorita datového protokolu. Chtějí-li např. dvě řídicí jednotky odeslat svůj datový protokol současně, má ten, jehož priorita je vyšší, přednost. Zároveň je v něm označen obsah zprávy (např. otáčky motoru).

Řídicí pole (Control Field)

Řídicí pole obsahuje jako kód počet informací, které jsou obsaženy v datovém poli. Díky tomu může příjemce zkontrolovat, zda mu došly všechny.

Datové pole (Data Field)

V datovém poli jsou přenášeny informace, které jsou důležité pro ostatní řídicí jednotky. Obsahem informací 0 až 64 bit (0 až 8 byte) je polem

s největším množstvím informací. 8 bit = 1 byte.

Kontrolní pole (CRC Field) (Cyclical Redundancy Check)

Kontrolní pole slouží ke zjiš»ování chyb v přenosu. Jedná se o metodu založenou na cyklickém výpočtu kontrolního kódu dat před přenosem a po přenosu.

Potvrzovací pole (ACK Field) (Acknowledgement)

Potvrzení přijetí. Příjemce signalizuje objektu, který zprávu vyslal, že datový protokol byl správně přijat. Byla-li zjištěna chyba, je to vysílacímu objektu ihned sděleno a ihned dochází k opětovnému poslání zprávy.

Ukončovací pole (End of Frame)

V tomto poli kontroluje vysílač svůj datový protokol a potvrdí objektu, který zprávu vyslal, zda je v pořádku. Jestliže není, dojde okamžitě k přerušení a opakovanému zahájení přenosu. Tím je datový přenos protokolu ukončen. [11]

5 Konstrukce funkčního prototypu

V této části se práce zabývá jednotlivými zdroji, které se používají pro čelní osvětlení automobilů. Pro konstrukci prototypu byly vybrány na trhu dostupné paraboly pro motocyklová světla, u kterých byly nahrazeny konvenční halogenové žárovky kompatibilními LED zdroji. Dále byl prototyp doplněn o denní svícení a integrované směrové ukazatele, jak je vyobrazeno na Obr. 3 – funkční prototyp.

Obr. 3 – funkční prototyp

5.1 Výběr zdrojů světla

Výběr světelných zdrojů, žárovek, byl bohužel ovlivněn finančním limitem. Proto bylo přikročeno ke kompromisu a LED zdroje v hlavních světlometech nemají funkci tlumených světel, pouze dálkových. Všechny světelné zdroje, které byly použity, nemají homologaci, a tudíž je není možno používat na veřejných pozemních komunikacích. Vzhledem k tomu, že samotný elektromobil také nesplňuje příslušné předpisy, jelikož slouží pouze pro experimentální účely, není to na závadu.

5.2 Žárovky do hlavních světlometů

Na trhu neexistuje v současné době plnohodnotná homologovaná náhrada konvenčních žárovek. Automobilky, které do některých svých modelů vozů LED světla nabízejí, mají svá velmi sofistikovaná řešení, jejichž technologii si úzkostlivě střeží. Jedná se o technicky vyspělá a velice nákladná zařízení, často svou cenou přesahující i 60 000 Kč, tudíž pro naše potřeby nevyužitelná.

Jako optimální náhrada žárovky H4 se jevila LED žárovka s použitými čipy od firmy Cree a přepínáním tlumených/dálkových světel, vyobrazená na obrázku Obr. 4 – plnohodnotná náhrada žárovky H4. Obsahuje 4x LED čip CREE XM-L (svítivost čipu je až 1 040 lm, celková 3 000 lm) s barvou světla 6 500 K. Výkon 80 W (2 x 40 W), napájecí napětí 12/24 V, proudový odběr cca 1,65 A/12 V.

Obr. 4 – plnohodnotná náhrada žárovky H4

Ovšem tento typ, jenž je možno použít jako plnohodnotnou náhradu, byl z finančních důvodů zamítnut. Cena za jeden pár totiž přesahuje částku 3 500 Kč, jenž nebyla akceptovatelná. Bylo tedy nutné zvolit finančně dostupnější variantu. Levnější modely LED žárovek typu H4, které se běžně vyskytují na našem trhu, lze rozdělit do dvou kategorií podle použitého chlazení – s aktivním chlazením (většinou s demontovatelným malým ventilátorem) a s pasivním chlazením (čipy jsou umístěny na vhodném, tepelně vodivém nosiči). Žárovky s aktivním chlazením mají měděné, nebo hliníkové tělo, které slouží k odvodu tepla od čipu. Některé modely lze použít jako plnohodnotnou náhradu klasických žárovek typu H4, jelikož mají možnost přepínání režimu tlumená / dálková. Těch je ovšem na trhu minimum. Většinou se setkáváme pouze s modely použitelnými v režimu dálkových světel.

Problémem žárovek s aktivním chlazením je jejich prostorová náročnost. Kromě mohutného chladiče a vlastního ventilátoru je pro provoz nutná ještě řídicí elektronika, jenž je umístěna pro každou žárovku zvlášť v samostatném pouzdře. Pouzdro je pak se žárovkou propojeno speciálním konektorem, jenž není nijak unifikovaný. Každý výrobce používá většinou svůj vlastní typ. Výrobci však počítají s umístěním pouzdra s řídicí elektronikou mimo kryté prostory světlometu. Konektory jsou odolné vůči prachu a vlhkosti a mívají krytí IP 65, nebo vyšší.

Na základě vyhodnocení dostupných parametrů byl vybrán model s pasivním chlazením, zobrazený na obrázku Obr. 5 – LED náhrada žárovky H4 bez přepínání. Nová generace LED autožárovek s paticí H4, bílá, 6x super svítivý LED čip CREE XBD 5W a rozptylová krycí čočka. Tato žárovka nemá přepínání dálková/potkávací světla, pracuje pouze v režimu dálkových světel.

Obr. 5 – LED náhrada žárovky H4 bez přepínání

5.3 Denní svícení se směrovými ukazateli

Byla použita sada 2 flexibilních LED denních světel s funkcí blinkru, využívající napětí 12 V. Jedná se o 2 pásky v délce 600 mm, Obr. 6 – flexibilní LED pásky, které je možno pomocí plastových držáků a malých šroubů připevnit na nosnou konstrukci světlometů. Pásky obsahují 2 barvy LED, oranžové a bílé. Díky jejich flexibilitě je možné je jednoduše tvarovat. Nelze je však ohýbat přes ostré hrany a rádius s poloměrem menším než 25 mm. Ohýbají se a tvarují pouze do oblého tvaru.

Obr. 6 – flexibilní LED pásky

5.4 Spínání světelných zdrojů

Pro spínání jednotlivých světelných zdrojů nelze použít výstupy ovládacího obvodu. Bylo proto nutné najít vhodný způsob, jak tyto zdroje spínat. Nejjednodušším způsobem bylo použití klasického relé, jako je na Obr. 7 – relé.

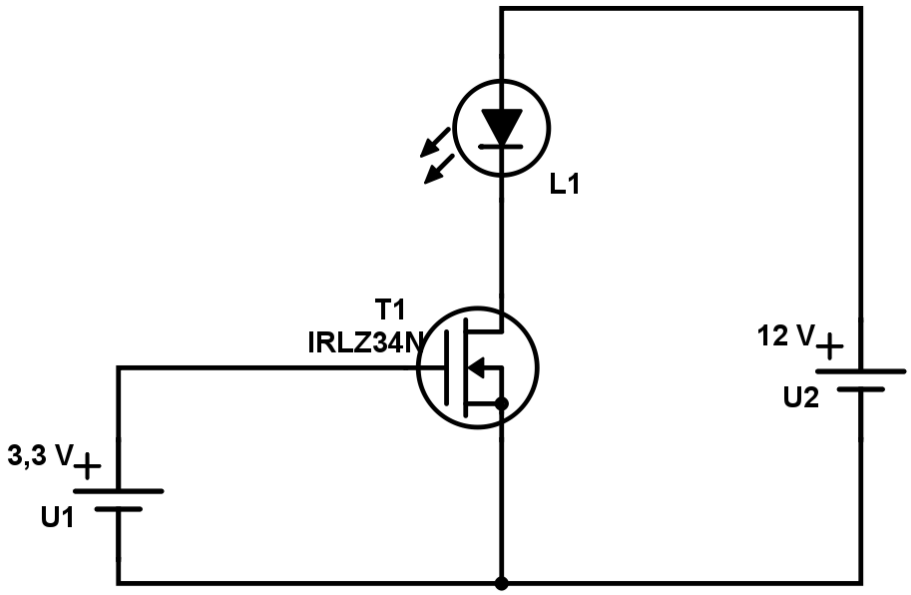
Obr. 7 – relé

To se však ukázalo jako nevhodné. Relé není uzpůsobeno k rychlému přepínání z jednoho stavu do druhého. Nelze tedy využít pulzně šířkovou modulaci pro ovlivnění intenzity

světla. Pokud by bylo použito relé bez paralelní ochranné diody před zpětným proudem, mohlo by dojít ke zničení procesoru, jelikož relé se při rozpínání chová jako indukční zátěž, která generuje zpětný proud. Navíc je většina relé konstruována na napětí 5 V a model Arduino Due pracuje s napětím 3,3 V. Z tohoto důvodu bylo přistoupeno k jinému řešení a tím je MOS-FET tranzistor.

5.4.1 MOS-FET tranzistor s indukovaným kanálem jako spínač

Tranzistor byl použit v jednoduchém zapojení jako spínač, jehož modelové schéma je na následujícím obrázku Obr. 8 – schéma zapojení tranzistoru.



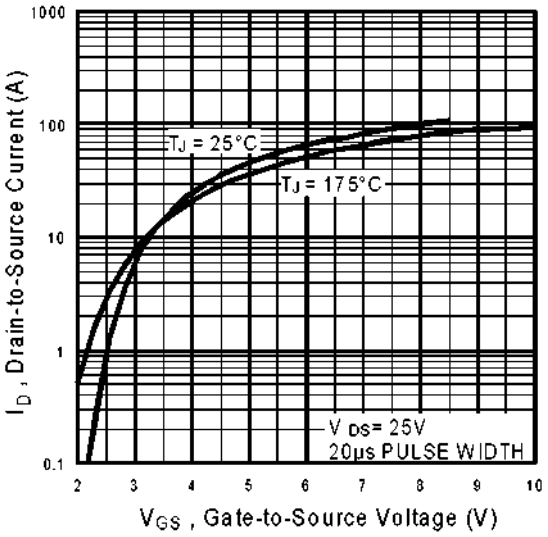
Obr. 8 – schéma zapojení tranzistoru

Pro ověření funkce řídicí jednotky osvětlení byl použit tranzistor MOS-FET s označením IRLZ34N. Jedná se o pátou generaci od International Rectifier, která využívá pokročilé technické zpracování pro dosažení nejnižšího možného odporu křemíkové oblasti. V kombinaci s vysokou rychlostí spínání, robustní konstrukcí v pouzdře TO220AB a relativně nízkou cenou, se jedná o efektivní zařízení pro použití v široké škále aplikací. Dle datového listu výrobce, Tabulka 5 – parametry tranzistoru IRLZ34N, má následující parametry.

Tabulka 5 – parametry tranzistoru IRLZ34N

Tranzitor IRLZ34N	
Provedení:	vývodové
Technologie:	N-MOSFET
Ochranná dioda:	ANO
I_{dss}:	30 A
U_{ds}:	55 V
U_{gs}:	16 V
R_{ds}:	0,035 Ohm
Pd:	68 W
Pouzdro:	TO220AB

Spíše než maximální hodnoty bylo důležité minimální napětí U_{GS}, potřebné pro otevření tranzistoru. Z následujícího grafu, Obr. 9 – závislost proudu I_D na napětí U_{GS}, lze vyčíst, že při napěti 3,3 V je proud mezi Drain a Source okolo 10 A, což je více než dostatečná hodnota.



Obr. 9 – závislost proudu I_D na napětí U_{GS}

6 Měření

Při praktické zkoušce světlometu i následném měření se projevily některé klady i zápory obou typů zdrojů.

6.1 Měření teplot

Pro měření teplot zdrojů světla byly vybrány žárovky s největším příkonem (vztaženo na konvenční zdroj), tedy typ H4 pro použití v dálkových světlech. Jak bylo zmíněno, použitá LED žárovka v režimu tlumených světel svítit nemůže. Měření teplot probíhalo při okolní teplotě 25 °C. Doba svícení byla 15 minut. Poté došlo ke změření teploty uvnitř světlometu na rozhraní přechodu paraboly světlometu a krycího skla, tedy 2,5 cm od vlastní žárovky.

Tabulka 6 – výsledky měření teplot

Typ	Konvenční žárovka	LED
tlumená světla	98 °C	—
dálková světla	132 °C	34 °C

Z měření, jehož výsledky jsou v tabulce Tabulka 6 – výsledky měření teplot, je zřetelně vidět, že rozdíly v teplotách jednotlivých zdrojů jsou značné. Proto se také u konvenčních zdrojů důrazně nedoporučuje ponechávat světla rozsvícená, pokud se vozidlo nepohybuje a nedochází k chlazení proudícím vzduchem. Dlouhodobé působení takto vysokých teplot má za následek degradaci paraboly světlometu i ostatních jeho částí. Světlomet, na kterém bylo měření prováděno, má parabolu z kovu a ta vyšší teploty snáší lépe než paraboly z plastů, které se používají u dnešních moderních vozidel. U plastových parabol může dojít k počátečním drobným deformacím dokonce i je-li vozidlo vystaveno dlouhodobému působení silného slunečního svitu, například v létě na nekrytém parkovišti.

LED žárovky se také zahřívají. Není to však tak výrazné, jako u konvenčních vláknových žárovek. Některé modely LED žárovek, především těch výkonnějších, po delším provozu dosahují pro čip nebezpečných teplot. To je dáno prostředím, ve kterém jsou provozovány a také účinností a konstrukcí jejich chladiče. To, jak kvalitně je čip chlazen, ovlivňuje životnost celého LED zdroje.

6.2 Měření proudů

Pro porovnání konvenčních a LED zdrojů v oblasti spotřeby proudů bylo provedeno několik měření. Výsledky jsou uvedeny v následující tabulce, Tabulka 7 – výsledky měření proudů, ze které je zřejmé, že vybrané LED zdroje přináší značné úspory.

Tabulka 7 – výsledky měření proudů

Typ	Konvenční žárovka	LED žárovka
Obrysová	1,24 A	0,570 A
Blinkr	0,70 A	0,540 A
Dálková	3,75 A	0,320 A
Tlumená	2,83 A	—

6.3 Měření intenzity osvětlení

Se světlometem bylo provedeno praktické měření z důvodu ověření, zdali je model světlometu funkční a bude možné jej na vozidle použít. Měření intenzity osvětlení bylo provedeno pro porovnání konvenčního druhu světla s LED technologií, která je použita v modelu světlometu.

K praktické zkoušce byla vybrána odlehlá, málo frekventovaná komunikace, jejíž vozovka byla v dobrém technickém stavu a to včetně vodorovného značení. Povrch vozovky nebyl osvětlen veřejným osvětlením a v blízkosti komunikace se nenacházely žádné významnější zdroje světelného znečištění, které by mohly mít na výsledky měření vliv.

Pro měření intenzity osvětlení byl použit kalibrovaný měřicí přístroj, Obr. 10 – měřicí přístroj Eurotest 61557, od firmy Metrel a sonda A1102 - Luxmetr, typ B; rozsah 0,01 lx ÷ 20 000 lx.

Obr. 10 – měřicí přístroj Eurotest 61557 [16]

Nejprve bylo provedeno měření s 2 ks konvenčních žárovek H4 v režimu dálkových světel. Poté byly vyměněny za 2 ks LED žárovek s čipy CREE XBD 5 W. Světlomety byly umístěny ve výšce 500 mm nad vozovkou, sonda luxmetru ve výšce 400 mm nad vozovkou a ve vzdálenosti 25 m od zdroje. Sklon světlometů byl postupně nastaven v rozmezí od 0,8 % do 1,5 %. Z výsledků, Tabulka 8 – naměřené intenzity osvětlení, je zřejmé, že LED zdroje mají menší svítivost.

Tabulka 8 – naměřené intenzity osvětlení

Sklon	Konvenční žárovka	LED žárovka
0,8 %	10,61 lx	6,20 lx
1,0 %	12,34 lx	6,56 lx
1,2 %	12,97 lx	6,52 lx
1,5 %	13,28 lx	6,59 lx

Svítivost použitých LED zdrojů je nižší, než u konvenčních žárovek. Bylo by tedy vhodné nahradit je jiným, výkonnějším a také dražším, modelem. Pro potřeby experimentálního automobilu jsou však tyto zdroje dostačující.

V průběhu měřená byly pořízeny fotografie, Obr. 11 – porovnání zdrojů. Ty byly zhotoveny digitálním fotoaparátem CANON EOS 550D s nasazeným objektivem Canon EF-S 17-55 mm f/2,8 IS USM. Fotografie byly pořízeny za konstantních podmínek expozice. Na fotoaparátu byly nastaveny následující parametry, uvedené v tabulce Tabulka 9 – parametry expozice.

Tabulka 9 – parametry expozice

Režim:	Manual
Ohnisko:	55 mm
ISO:	200
Čas:	8 s
Clona:	F/4
Barevná teplota:	5 200 K

Z provedených měření a následujících fotografií, Obr. 11 – porovnání zdrojů, vyplývá, že je zřetelný rozdíl mezi konvenčním typem žárovky a žárovkou s LED zdroji. Na první pohled je zřejmá odlišná barevná teplota.

Obr. 11 – porovnání zdrojů

Konvenční žárovky vydávají subjektivně nažloutlé světlo o teplotě okolo 3 000 K. To je i tento případ, kdy pomocí softwaru na zpracování fotografií, Adobe Photoshop Lightroom 5.2, byla z pořízených snímků získána následující data o barevné teplotě:

- Konvenční žárovka 2 800 K
- LED žárovka s čipy CREE XBD 5 W 5 500 K

Z informací, uvedených v kapitole vyplývá, že použití LED zdrojů světla o teplotě 5 500 K, které je velmi blízké teplotě denního světla (cca 6 000 – 6 500 K), má pozitivní vliv na bezpečnost silničního provozu. Mnoho lidí při snížené hladině serotoninu pociťuje ve zvýšené míře bolest, stres, depresi a úzkost. Lidské tělo je při vyzařování modré složky stimulováno k produkci serotoninu a dává tělu signál k omezení produkce melatoninu, který si tělo tvoří v noci. Serotonin pomáhá k vytváření a udržení pozitivních emocí a nálady, udržuje naši náladu v rovnováze, zmírňuje stavy úzkosti a ulevuje od deprese. [12] Všechny tyto projevy působení serotoninu jednoznačně přispívají k větší pohodě řidiče, pomáhají mu se více soustředit na řízení a činnosti s tím spojené a udržují jej v duševní rovnováze. To má jednoznačně pozitivní dopad na bezpečnost při řízení.

Závěr

V práci bylo popsáno osvětlení experimentálního elektromobilu eŠus, výzkumného elektromobilu.

Pro vlastní ovládání a řízení světel byla vybrána platforma Arduino, konkrétně model Arduino DUE. V práci byly popsány důvody pro výběr právě této platformy, historie jejího vzniku a výhody i nevýhody v porovnání s dalšími platformami dostupnými na našem současném trhu.

V práci bylo kromě představení vlastní platformy popsáno také vývojové prostředí (tzv. IDE), jenž bylo pro návrh programů použito a prostřednictvím kterého je možné na desku Arduino programy nahrávat. Byly popsány základy a obecné zvyklosti pro tvorbu programů ve vývojovém prostředí Arduino i práce s rozšiřujícími knihovnami, které je možné použít pro mapování pinů či komunikaci po CAN sběrnici.

Jazyk pro prostředí Arduino IDE byl v práci podrobně popsán, vysvětleny principy deklarace a inicializace proměnných. U jednotlivých typů proměnných byly popsány jejich vlastnosti, typy hodnot, jež mohou obsahovat, maximální hodnoty, jichž mohou nabývat i velikosti, které zabírají v paměti.

Dále jsou v práci ukázky vytvořených kódů a detailně popsána jejich funkce. Je zde vysvětleno rozdělení na hlavní části, setup() a loop() a jaký kód se ve které části nachází. Byly popsány funkce, které ovládají jednotlivá světla čelního osvětlení. Kromě základních jsou naprogramovány i pokročilé funkce, např. komfortní blikání, přisvětlování do zatáček, tzv. corner a další. Celý tento program nabízí do budoucna možnosti rozšíření, zejména o další pokročilé prvky ovládání světel, například naklápění v závislosti na zatížení vozidla nebo natáčení světlometů při zatáčení.

Pro funkční model byly zvoleny zdroje světla s ohledem na to, že se jedná o bateriový elektromobil s omezenou kapacitou. Byly vybrány zdroje s technologií LED, které v poslední době prošly velkým vývojem. Za posledních několik let došlo k výraznému zlepšení všech klíčových parametrů, jakými jsou například svítivost nebo účinnost a tudíž již dokáží plně nahradit konvenční zdroje a v mnoha ohledech je dokonce i předčí.

Z finančních důvodů nebylo bohužel možné pořídit plnohodnotné LED zdroje s funkcí přepínání mezi dálkovými a potkávacími světly.

Díky použití LED zdrojů bylo možné zvolit také barevnou teplotu vyzařovaného světla. V případě použití konvenčních žárovek, kde je jejich barevná teplota daná použitým plynem, nelze tuto hodnotu ovlivnit. U LED zdrojů však výrobci nabízejí různé hodnoty, od cca 3 000 K, což je teplota klasické žárovky, až po 8 000 K. Pro implementaci do funkčního prototypu byl vybrán zdroj, jehož barevná teplota je blízká té denní. Především pro jeho pozitivní účinky na lidský zrak i pozornost řidiče.

Na funkčním prototypu bylo provedeno několik měření, která porovnávala vybrané LED zdroje s jejich konvenčními protějšky. Z porovnání vyplynulo, že zvolené zdroje jsou vhodné a jsou adekvátní dostupnou náhradou.

Dále byl řešen problém se spínáním světel. Použití klasického relé se ukázalo jako nevhodné, jelikož zde dochází k indukci proudů, má omezený počet spínacích cyklů a není možné jej použít pro pulzně šířkovou modulaci. Proto byly místo relé použity tranzistory typu MOS-FET, které se ukázaly pro tuto aplikaci jako nejvhodnější. Na trhu bohužel neexistovalo mnoho těchto tranzistorů, které by byly schopné pracovat s napětím, které je použito u zvolené platformy. Po delším hledání byl vybrán model, který se ukázal jako vhodný a splňoval všechny potřebné požadavky.

Seznam zdrojů

[1] J. Smolík, „Současnost a budoucnost automobilového osvětlení,“ *Světlo, časopis pro světlo a osvětlování,* č. 5, pp. 4-6, 11 5 2011.

[2] Texas Instruments, „Automotive Adaptive Front-lighting System Reference Design,“ 2013. [Online]. Available: http://www.ti.com/lit/ug/spruhp3/spruhp3.pdf. [Přístup získán 13 2 2015].

[3] PR Newswire Association LLC, „OSRAM Forecasts Electric Vehicles to Benefit Most From New LED Lighting,“ 19 1 2011. [Online]. Available: http://www.prnewswire.com/news-releases/osram-forecasts-electric-vehicles-to-benefit-most-from-new-led-lighting-114220924.html. [Přístup získán 15 3 2015].

[4] J. Tomsa, *Elektrická zařízení elektromobilu eTUL,* Liberec: Fakulta mechatroniky, informatiky a mezioborových studií Technické univerzity v Liberci, 2014.

[5] Residit s.r.o., „Odborné časopisy - Časopis Světlo,“ 2014-2015. [Online]. Available: http://www.odbornecasopisy.cz/svetlo/casopis/tema/bile-led-svetlo-budoucnosti--16118. [Přístup získán 3 12 2014].

[6] M. Olejár, "elweb.cz :: elektronika na webu Martina Olejára ::.," OLEJAR technologies, 2016. [Online]. Available: http://www.elweb.cz/clanky.php?clanek=94. [Accessed 16 březen 2016].

[7] J. P. Stengl, Výkonové tranzistory MOSFET, Praha: BEN, 1999.

[8] Arduino, „Arduino - Introduction,“ 2016. [Online]. Available: https://www.arduino.cc/en/Guide/Introduction. [Přístup získán 19 listopad 2015].

[9] M. Banzi, Getting Started with Arduino, "O'Reilly Media, Inc., 2009.

[10] B. Evans, Beginning Arduino Programming, Apress, 2011.

[11] ŠkodaAuto a.s., [Online]. Available: http://sevcikpeta.wz.cz/downloads/24_CAN%20Bus.pdf. [Accessed 16 3 2016].

[12] I. Z. Rozehnal, "serotonin - přirozené způsoby jak posílit jeho produkci a tím i dobré zdraví a náladu :: Úspěšná léčba," 2015. [Online]. Available: http://www.uspesna-lecba.cz/co-se-vas-tyka/serotonin-prirozene-zpusoby-jak-posilit-jeho-produkci-a-tim-i-dobre-zdravi-a-naladu/. [Accessed 2 prosinec 2015].

[13] Caravanservis.cz, "Průřez vodiče pro použití v karavanu | Caravanservis.cz," [Online]. Available: http://www.caravanservis.cz/wp-content/uploads/2012/07/prurez_vodice.pdf. [Accessed 12. leden 2016].

[14] http://store-usa.arduino.cc/, Artist, *Vývojová deska Arduino DUE.* [Art].

[15] ILLKO, s.r.o., "EUROTEST 61557 Euro Set - detail produktu | MI 2086 EU - EUROTEST 61557 Euro Set | Multifunkční přístroje pro revize elektroinstalací | Produkty," 2015. [Online]. Available: http://www.illko.cz/eurotest-61557-euro-set-detail-produktu. [Accessed 11 listopad 2015].

[16] M. Doubravský, "CAN_BUS_prenos_dat," [Online]. Available: http://www.can-bus.wz.cz/prenos_dat.html. [Accessed 15 duben 2016].

Seznam obrázků

Seznam tabulek

Seznam kódů

Příloha 1 – průřezy vodičů

Tabulka 10 – přehled doporučených průřezů vodiče [13]

Napětí:		**12 V**				
Délka kabelu [m]		**10**	**5**	**3**	**2**	**1**
Proud [A];	**Výkon [W]**	**Průřez [mm^2]**				
1,0	12	1,0	0,5	0,5	0,5	0,5
1,5	18	1,0	0,5	0,5	0,5	0,5
2,0	24	1,5	1,0	0,5	0,5	0,5
2,5	30	2,0	1,0	0,5	0,5	0,5
3,0	36	2,0	1,0	1,0	0,5	0,5
3,5	42	2,5	1,5	1,0	0,5	0,5
4,0	48	3,0	1,5	1,0	1,0	0,5
4,5	54	3,0	1,5	1,0	1,0	0,5
5,0	60	3,5	2,0	1,0	1,0	0,5
5,5	66	4,0	2,0	1,5	1,0	0,5
6,0	72	4,0	2,0	1,5	1,0	0,5
6,5	78	5,0	2,5	1,5	1,0	0,5
7,0	84	5,0	2,5	1,5	1,0	0,5
7,5	90	5,0	2,5	1,5	1,0	0,5
8,0	96	6,0	3,0	2,0	1,5	1,0
8,5	102	6,0	3,0	2,0	1,5	1,0
9,0	108	6,0	3,0	2,0	1,5	1,0
9,5	114	10,0	3,5	2,0	1,5	1,0
10,0	120	10,0	3,5	2,0	1,5	1,0
15,0	180	10,0	5,0	3,0	2,0	1,0
20,0	240	15,0	10,0	4,0	3,0	1,5
25,0	300	20,0	10,0	5,0	3,5	2,0
30,0	360	20,0	10,0	6,0	4,0	2,0

Příloha 2 – kód programu

```
//přiřazení na piny Arduino
const byte senzorNatoceniVolantu = A0;
const byte senzorOtacekMotoru = A1;
const byte senzorOsvetleni = A2;

const byte mlhovkaLeva = 2;
const byte mlhovkaPrava = 3;
const byte blinkrLevy = 4;
const byte blinkrPravy = 5;
const byte denniSviceniLeve = 6;
const byte denniSviceniPrave = 7;
const byte tlumenaSvetla = 8;
const byte dalkovaSvetla = 9;

const byte spinacMlhovky = 31;
const byte spinacBlinkrLevy = 33;
const byte spinacBlinkrPravy = 35;
const byte spinacBlinkryVystrazne = 37;
const byte spinacObrysovaSvetla = 39;
const byte spinacTlumenaSvetla = 41;
const byte spinacDalkovaSvetla = 43;

const byte cornerUhelNatoceniVolantu = 20; //úhel
potřebný pro zapnutí funkce Corner, (0-128)
const byte cornerMaxRychlost = 45; //maximální rychlost,
ve které funguje funkce Corner
const int  intenzitaSera = 500; //0-1024 světlo 0, tma
1024

const byte rychlostZmenyJasu = 15; // 0-255 5,15,
const int  intervalBlikani = 660; //1000 ms = 1 sec
const byte komfortniBlikani = 3; //počet bliknutí při
komfortním blikání

const byte kontrolka = 13;



//=======================================================
================================  inicializace
proměnných
boolean zapnutyKlicek = false;
boolean sero = false;
int   rychlost = 0;
int   otackyMotoru = 0;
int   natoceniVolantu = 128;
int   intenzitaOsvetleni = 0;
```

```
int   pocetBliknuti = 3;


byte  komfortniBlikaniLevy = 0;
byte  komfortniBlikaniPravy = 0;
byte  intenzitaMlhovkaLeva = 0; //0 - 255
byte  intenzitaMlhovkaPrava = 0; //0 - 255
byte  intenzitaDenniSviceni = 128; //0 - 255

unsigned long pocetMilisekund; //Počet milisekund od
začátku běhu programu
unsigned long blikacPredchoziPocetMilisekund = 0;     //
bude ukládat poslední případ aktualizace LED
boolean blikac = false;


boolean zapnutoMlhovkaLeva = false;
boolean zapnutoMlhovkaPrava = false;
boolean zapnutoMlhovky = false;

boolean zapnutoBlinkrLevy = false;
boolean zapnutoBlinkrPravy = false;
boolean zapnutoBlinkryVystrazne = false;

boolean zapnutoDenniSviceniLeve = false;
boolean zapnutoDenniSviceniPrave = false;
boolean zapnutoDenniSviceni = false;

boolean zapnutoTlumenaSvetla = false;
boolean zapnutoObrysovaSvetla = false;
boolean zapnutoDalkovaSvetla = false;



boolean stavTlacitkaMlhovkaLeva = false;
boolean stavTlacitkaMlhovkaPrava = false;
boolean stavTlacitkaMlhovky = false;

boolean stavTlacitkaBlinkrLevy = false;
boolean stavTlacitkaBlinkrPravy = false;
boolean stavTlacitkaBlinkryVystrazne = false;

boolean stavTlacitkaDenniSviceniLeve = false;
boolean stavTlacitkaDenniSviceniPrave = false;
boolean stavTlacitkaDenniSviceni = false;

boolean stavTlacitkaTlumenaSvetla = false;
boolean stavTlacitkaObrysovaSvetla = false;
boolean stavTlacitkaDalkovaSvetla = false;


boolean soucasnyStavTlacitka;
```

```
//=========================================================
================================  Setup

void setup() {
  pinMode (mlhovkaLeva, OUTPUT);
  pinMode (mlhovkaPrava, OUTPUT);
  pinMode (blinkrLevy, OUTPUT);
  pinMode (blinkrPravy, OUTPUT);
  pinMode (denniSviceniLeve, OUTPUT);
  pinMode (denniSviceniPrave, OUTPUT);
  pinMode (tlumenaSvetla, OUTPUT);
  pinMode (dalkovaSvetla, OUTPUT);

  pinMode (kontrolka, OUTPUT);


  pinMode (spinacMlhovky, INPUT);
  pinMode (spinacBlinkrLevy, INPUT);
  pinMode (spinacBlinkrPravy, INPUT);
  pinMode (spinacBlinkryVystrazne, INPUT);
  pinMode (spinacTlumenaSvetla, INPUT);
  pinMode (spinacDalkovaSvetla, INPUT);

  Serial.begin(9600);
  Serial1.begin(9600);


}

//=========================================================
================================  Loop

void loop() {
  //otackyMotoru = 1000;
//analogRead(senzorOtacekMotoru); //0-1023

  // natoceniVolantu =
analogRead(senzorNatoceniVolantu); //0-1023
  // Serial.println(natoceniVolantu);

  pocetMilisekund = millis(); //millis() - počet
milisekund od začátku běhu programu, (po přetečení se
automaticky nuluje)


  //-------------------------------------------------
---------------------------------- čtení dat ze
sériové linky
  if (Serial1.available()) {
    if (Serial1.parseInt() == 1) {
```

```
      zapnutyKlicek = true;
    }
    else {
      zapnutyKlicek = false;
    }
    otackyMotoru = Serial1.parseInt();
    natoceniVolantu = Serial1.parseInt();
    intenzitaOsvetleni = Serial1.parseInt();
  }
  digitalWrite(kontrolka, zapnutoDenniSviceni);

  rychlost = otackyMotoru / 10; //pevný převod


  if (intenzitaOsvetleni > intenzitaSera)
  {
    sero = true;
  }
  else
  {
    sero = false;
  }



  //-------------------------------------------------------
---------------------------------- čtení tlačítek

  zapnutoBlinkrLevy = ctiTlacitko(spinacBlinkrLevy,
stavTlacitkaBlinkrLevy, zapnutoBlinkrLevy);
  stavTlacitkaBlinkrLevy = soucasnyStavTlacitka;
  if (zapnutoBlinkrLevy == false &&
stavTlacitkaBlinkrLevy == true)
  {
    komfortniBlikaniLevy = komfortniBlikani;
  }


  zapnutoBlinkrPravy = ctiTlacitko(spinacBlinkrPravy,
stavTlacitkaBlinkrPravy, zapnutoBlinkrPravy);
  stavTlacitkaBlinkrPravy = soucasnyStavTlacitka;
  if (zapnutoBlinkrPravy == false &&
stavTlacitkaBlinkrPravy == true)
  {
    komfortniBlikaniPravy = komfortniBlikani;
  }

  zapnutoBlinkryVystrazne =
ctiTlacitko(spinacBlinkryVystrazne,
stavTlacitkaBlinkryVystrazne, zapnutoBlinkryVystrazne);
  stavTlacitkaBlinkryVystrazne = soucasnyStavTlacitka;
```

```
  zapnutoMlhovky = ctiTlacitko(spinacMlhovky,
stavTlacitkaMlhovky, zapnutoMlhovky);
  stavTlacitkaMlhovky = soucasnyStavTlacitka;


  zapnutoTlumenaSvetla =
ctiTlacitko(spinacTlumenaSvetla,
stavTlacitkaTlumenaSvetla, zapnutoTlumenaSvetla);
  stavTlacitkaTlumenaSvetla = soucasnyStavTlacitka;

  zapnutoObrysovaSvetla =
ctiTlacitko(spinacObrysovaSvetla,
stavTlacitkaObrysovaSvetla, zapnutoObrysovaSvetla);
  stavTlacitkaObrysovaSvetla = soucasnyStavTlacitka;

  zapnutoDalkovaSvetla =
ctiTlacitko(spinacDalkovaSvetla,
stavTlacitkaDalkovaSvetla, zapnutoDalkovaSvetla);
  stavTlacitkaDalkovaSvetla = soucasnyStavTlacitka;

  //   zapnuto = ctiTlacitko(spinac, stavTlacitka,
zapnuto);
  // stavTlacitka = soucasnyStavTlacitka;

  zapnutoDenniSviceni = zapnutyKlicek;


  //-----------------------------------------------
ovládání světel




  if (zapnutoBlinkryVystrazne) {
    digitalWrite(blinkrLevy, blikej(intervalBlikani));
    digitalWrite(blinkrPravy, blikej(intervalBlikani));
  } else {
    if (zapnutoBlinkrLevy || (komfortniBlikaniLevy > 0))
    {
      digitalWrite(blinkrLevy, blikej(intervalBlikani));
      zapnutoBlinkrPravy = false;
      komfortniBlikaniPravy = 0;
    }
    else
    {
      digitalWrite(blinkrLevy, LOW);
    }
    if (zapnutoBlinkrPravy || (komfortniBlikaniPravy >
0))
    {
      digitalWrite(blinkrPravy,
blikej(intervalBlikani));
```

```
      zapnutoBlinkrLevy = false;
      komfortniBlikaniLevy = 0;
    }
    else
    {
      digitalWrite(blinkrPravy, LOW);
    }
  }


  if (zapnutoMlhovky) {
    analogWrite(mlhovkaLeva, 255);
    analogWrite(mlhovkaPrava, 255);
  }
  else {
    analogWrite(mlhovkaLeva, intenzitaMlhovkaLeva);
    analogWrite(mlhovkaPrava, intenzitaMlhovkaPrava);
    if (rychlost < cornerMaxRychlost) {
      if (natoceniVolantu > 512 +
cornerUhelNatoceniVolantu) {
        if (intenzitaMlhovkaPrava < 255) {
          intenzitaMlhovkaPrava = intenzitaMlhovkaPrava
+ abs(rychlostZmenyJasu);
        }
      }
      else {
        if (intenzitaMlhovkaPrava > 0) {
          intenzitaMlhovkaPrava = intenzitaMlhovkaPrava
- abs(rychlostZmenyJasu);
        }
      }
      if (natoceniVolantu < 512 -
cornerUhelNatoceniVolantu) {
        if (intenzitaMlhovkaLeva < 255) {
          intenzitaMlhovkaLeva = intenzitaMlhovkaLeva +
abs(rychlostZmenyJasu);
        }
      }
      else {
        if (intenzitaMlhovkaLeva > 0) {
          intenzitaMlhovkaLeva = intenzitaMlhovkaLeva -
abs(rychlostZmenyJasu);
        }
      }
    }
    else {
      intenzitaMlhovkaLeva = 0;
      intenzitaMlhovkaPrava = 0;
    }

  }
```

```
  if (zapnutoDenniSviceni)
  {
    if (sero)
    {
      analogWrite(denniSviceniLeve, 128);
      analogWrite(denniSviceniPrave, 128);
      zapnuto
    }
    else
    {
      analogWrite(denniSviceniLeve, 255);
      analogWrite(denniSviceniPrave, 255);
      Serial.println("255");
    }
  }
  else
  {
    analogWrite(denniSviceniLeve, 0);
    analogWrite(denniSviceniPrave, 0);
    Serial.println("0");
  }

  Serial.println(intenzitaOsvetleni);
  Serial.println(sero);
}

//------------------------------------------------------
---------------------------------------- UŽIVATELSKY
DEFINOVANÉ FUNKCE ------------------------------------
----------------------------
//čtení zmáčknutí tlačítka
boolean ctiTlacitko(byte tlacitko,  boolean
predchoziStavTlacitka, boolean zapnuto) {
  boolean zapni = zapnuto;
  soucasnyStavTlacitka = digitalRead(tlacitko);
  delay(5);

  if (soucasnyStavTlacitka == LOW &&
predchoziStavTlacitka == HIGH && zapnuto == HIGH) {
    zapni = false;
  }
  if (soucasnyStavTlacitka == LOW &&
predchoziStavTlacitka == HIGH && zapnuto == LOW) {
    zapni = true;
  }
  return zapni;
}


//Blikač střídající stavy se střídou dle zadaného
intervalu
boolean blikej(int interval) {
```

```
  if (pocetMilisekund - blikacPredchoziPocetMilisekund >
interval) {
    blikacPredchoziPocetMilisekund = pocetMilisekund;

    //změna stavu
    if (blikac == LOW)
    {
      blikac = HIGH;

    }
    else
    {
      blikac = LOW;
      if (komfortniBlikaniLevy > 0)
      {
        komfortniBlikaniLevy = komfortniBlikaniLevy - 1;
      }
      if (komfortniBlikaniPravy > 0)
      {
        komfortniBlikaniPravy = komfortniBlikaniPravy -
1;
      }
    }
  }
  return blikac;
}
```

```
//přiřazení na piny Arduino
const byte klicek = 52;
const byte senzorOtacekMotoru = A0;
const byte senzorNatoceniVolantu = A1;
const byte senzorOsvetleni = A2;

int zapnutyKlicek = 0;
int otackyMotoru = 0;
int natoceniVolantu = 512;
int intenzitaOsvetleni = 0;

void setup() {
  pinMode(13, OUTPUT);

  pinMode(klicek, INPUT);
  Serial.begin(9600);
  Serial1.begin(9600);
}

void loop() {
  zapnutyKlicek = digitalRead(klicek);
  //otackyMotoru = analogRead(senzorOtacekMotoru);
  otackyMotoru = 20;
  natoceniVolantu = analogRead(senzorNatoceniVolantu);
//0-1023
  intenzitaOsvetleni = analogRead(senzorOsvetleni);

  Serial1.println(zapnutyKlicek);
  Serial1.println(otackyMotoru);
  Serial1.println(natoceniVolantu);
  Serial1.println(intenzitaOsvetleni);

  delay(100);
}
```