EKONOMICKÁ FAKULTA TUL

Bakalářská práce

Návrh a realizace DoCAN zařízení s mikroprocesorem a rozšířenými schopnostmi

Studijní program: B0688P140002 Informační management
Autor práce: **Alexandr Adam**
Vedoucí práce: Ing. Marián Lamr, Ph.D.
Ústav mechatroniky a technické informatiky

Liberec 2025

EKONOMICKÁ FAKULTA TUL

Zadání bakalářské práce

Návrh a realizace DoCAN zařízení s mikroprocesorem a rozšířenými schopnostmi

Jméno a příjmení:	**Alexandr Adam**
Osobní číslo:	E22000288
Studijní program:	B0688P140002 Informační management
Zadávající katedra:	Katedra informatiky
Akademický rok:	2024/2025

Zásady pro vypracování:

1. Standardy a protokoly OBD2 a CAN sběrnice
2. Mikroprocesory a jejich uplatnění v automotive
3. Návrh firmwaru pro mikrokontrolér ESP32
4. Implementace bezdrátové komunikace mezi donglem a vytvoření uživatelského rozhraní
5. Zhodnocení a doporučení

Rozsah grafických prací:
Rozsah pracovní zprávy: 30 normostran
Forma zpracování práce: tištěná/elektronická
Jazyk práce: čeština

Seznam odborné literatury:

- BABIUCH, Marek; Petr FOLTYNEK a Pavel SMUTNY, 2019. Using the ESP32 Microcontroller for Data Processing. online. In: Kot, A a A Nawrocka. *2019 20TH Interrnational carpathian control conference* (ICCC), 2019. Dostupné z: https://doi.org/10.1109/carpathiancc.2019.8765944.
- BAYILMIS, Cuneyt a Zuhal YILDIRIM, 2016. CANWEB: A Web-Based Educational Interface for Controller Area Networks. online. *Journal of polytechnic-politeknik Dergesi*, vol. 19, no. 4, s. 455–460. ISSN 1302-0900. Dostupné z: https://doi.org/10.2339/2016.19.4.455-460.
- INTERNATIONAL ORGANISATION FOR STANDARTISATION, 2021. ISO 15765-4:2021, *Road vehicles — Diagnostic communication over Controller Area Network (DoCAN)*.
- JAMES, Mike a Harry FAIRHEAD, 2023. *Programming the ESP32 in MicroPython*. Leyburn, N. Yorkshire: I/O Press. ISBN 978-1-871962-82-6.
- LEE, Jeong-Hwan; Jaekyung LEE a Jihun CHA, 2018. How to build Controller Area Network Communication Test Environment using NVIDIA TX2 for Unmanned Aerial Vehicle. In: Perkovic, T; Z Milanovic; K Vukojevic; JJPC Rodrigues; S Nizetic et al. 2018 3RD *International conference on smart and sustainable techlonlogies (SPLITECH),* 2018. ISBN 978-953-290-083-5.

Konzultant: Jiří Traegner, Manager automobilové diagnostiky T-mobile

Vedoucí práce: Ing. Marián Lamr, Ph.D.
Ústav mechatroniky a technické informatiky

Datum zadání práce: 1. listopadu 2024
Předpokládaný termín odevzdání: 31. srpna 2026

L.S.

doc. RNDr. Miroslav Brzezina, CSc., dr. h. c.
děkan

Mgr. Tereza Semerádová, Ph.D.
garant studijního programu

V Liberci dne 1. listopadu 2024

Prohlášení

Prohlašuji, že svou bakalářskou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědom toho, 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 nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

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

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

12. května 2025 Alexandr Adam

Poděkování

Chtěl bych vyjádřit své upřímné poděkování vedoucímu práce, Ing. Bc. Mariánu Lamrovi, Ph.D., za jeho odborné vedení, trpělivost a cenné připomínky při zpracování této bakalářské práce.

Dále děkuji Ing. Lukášovi Krčmářovi za jeho odbornou pomoc při realizaci projektu a návrhu kabelového systému.

Na závěr patří mé poděkování přítelkyni, Mgr. Dianě Kyselové, a mé rodině za jejich neustálou podporu a motivaci, které mi pomohly tuto práci úspěšně dokončit.

Abstrakt

Tato bakalářská práce se zabývá návrhem a realizací diagnostického zařízení DoCAN, jehož cílem je zjednodušení a zefektivnění procesu testování a diagnostiky vozidel v rámci automobilového vývoje. Zařízení je založeno na mikrokontroléru ESP32, doplněném o převodník CAN sběrnice a step-down měnič pro úpravu napájení. Výsledné řešení umožňuje bezdrátovou komunikaci a nabízí intuitivní uživatelské rozhraní, což vede ke značné úspoře času při testovacích procesech.

Klíčová slova: Mikrokontroler, ESP32, CAN, CAN-FD, Sběrnice, OBD 2, Protokol

Abstract

This bachelor thesis deals with the design and implementation of the DoCAN diagnostic device, which aims to simplify and streamline the process of vehicle testing and diagnostics in automotive development. The device is based on an ESP32 microcontroller, complemented by a CAN bus converter and a step-down converter for power conditioning. The resulting solution enables wireless communication and offers an intuitive user interface, leading to significant time savings in testing processes.

Keywords*:* Microcontroller, ESP32, CAN, CAN-FD, Bus, OBD 2, Protocol

Obsah

Seznam obrázků **6**

Seznam tabulek **7**

Seznam zdrojových kódů **8**

Seznam zkratek **9**

Úvod **10**

1 Protokoly OBD2 a CAN **11**

1.1 OBD2 11

1.2 OBD2 PIDs 13

1.3 CAN 15

1.4 CAN FD 15

1.5 Topologie CAN BUS 16

1.6 Princip diferenciální komunikace 17

1.7 Formát rámců CAN BUS 19

1.8 CAN BUS Request 19

1.9 CAN BUS Response 20

1.10 Ukázka komunikace na CAN BUS 20

2 Mikroprocesory v automotive a jejich sběrnice **22**

2.1 Mikroprocesor a jeho využití 22

2.2 Nároky na mikroprocesory v automotive 22

2.3 Mikroprocesor ESP32 23

2.4 Hlavní rodiny mikroprocesorů v automotive 24

2.5 Oblasti využití procesorů Snapdragon v automotive 25

2.6 Sběrnice I²C 27

2.7 Sběrnice I2S 30

2.8 Sběrnice SPI 33

2.9 Sběrnice UART 36

2.10 Porovnání sběrnic 39

3 Elektrická zařízení motorových vozidel 41

3.1 Primární zdroj energie 41

3.2 Změna napětí – Step-down měnič 41

3.3 Změna napětí – Boost konvertor 41

4 Nástroje pro vývoj 43

4.1 Programovací jazyky 43

4.2 Vývojové prostředí 43

5 Návrh řešení 45

5.1 Návrh Hardwarového řešení 48

5.2 Návrh firmwaru pro ESP32 51

5.3 Návrh Aplikace pro ovládací zařízení 55

5.4 Realizace – prvotní prototyp 58

5.5 Realizace – finální řešení 61

Závěr 69

Zdroje 70

Seznam obrázků

Obrázek 1: OBD2 Port – Volkswagen 12
Obrázek 2: Topologie sběrnice CAN 19
Obrázek 3: Principiální schéma 47
Obrázek 4: ESP32-WROOM-32 48
Obrázek 5: LCD 16x2 SPI 48
Obrázek 6: MCP2515 49
Obrázek 7: LM317 49
Obrázek 8: 3D Návrh obalové skořepiny 50
Obrázek 9: Schéma zapojení 50
Obrázek 10: První verze aplikace 58
Obrázek 11: Prvotní zapojení 60
Obrázek 12: Finální verze zařízení 63
Obrázek 13: Ověření funce Launcheru 64
Obrázek 14: Nové GUI 67
Obrázek 15: Editor maker 68

Seznam tabulek

Tabulka 1: Popis pinů ODB2 – Volkswagen 12
Tabulka 2: OBD2 PIDs 13
Tabulka 3: Ukázka PID 15
Tabulka 4: Formátování CAN Request 20
Tabulka 5: Formátování CAN Response 20
Tabulka 6: Dílčí porovnání sběrnic 39

Seznam zdrojových kódů

Zdrojový kód 1: Parsování ID:DATA 53
Zdrojový kód 2: Zpracování přijaté CAN zprávy 54
Zdrojový kód 3: Příprava na běh apliakce 55
Zdrojový kód 4: Čtení dat 56
Zdrojový kód 5: Asynchronní engine 57
Zdrojový kód 6: použití TKinter 57
Zdrojový kód 7: Launcher 65

Seznam zkratek

CAN	Controller Area Network
CAN FD	CAN Flexible Data-rate
DEC	Decimální
DoCAN	Diagnostic over CAN
GUI	Graphical User Interface
HEX	Hexadecimální
OBD	On-Board Diagnostics
PID	Parameter ID

Úvod

Automobilový průmysl dnes čelí rostoucím nárokům na rychlost vývoje, testování a nasazení nových technologií, což klade vysoké požadavky na efektivitu a přesnost diagnostických a testovacích procesů. Tato bakalářská práce se proto zaměřuje na návrh a realizaci diagnostického zařízení s prvky DoCAN (Diagnostic Communication over Controller Area Network), které umožňuje efektivní komunikaci mezi vozidlem a diagnostickým softwarem bez nutnosti využití klasického diagnostického rozhraní.

Zařízení, které je předmětem této bakalářské práce, bylo navrženo s využitím mikrokontroleru ESP32, který disponuje dostatečným výpočetním výkonem, integrovanou Wi-Fi a Bluetooth komunikací a zároveň umožňuje snadnou a rychlou implementaci potřebného firmwaru. Součástí zařízení je také CAN převodník umožňující komunikaci s CAN sběrnicí vozidla a step-down měnič pro napájení přímo z OBD2 konektoru.

Kromě samotného návrhu a realizace hardware byla součástí práce také tvorba firmwaru, který umožňuje odesílání vlastních příkazů a diagnostických dotazů do vozidla. Dále byla vytvořena aplikace s responzivním uživatelským rozhraním pro počítače, která komunikuje bezdrátově se zařízením DoCAN a umožňuje jednoduchou správu a monitorování diagnostických procesů v reálném čase.

Výsledkem je unikátní nástroj s vysokou mírou customizovatelnosti, který umožňuje efektivně zrychlit a zjednodušit testovací procesy, například při flashování řídicích jednotek, čímž přináší významnou časovou úsporu a snižuje nároky na pracovníky. Bakalářská práce tak reaguje na reálné potřeby automobilového průmyslu, konkrétně vývojového prostředí automobilky Škoda, kde bylo zařízení úspěšně testováno a ověřeno.

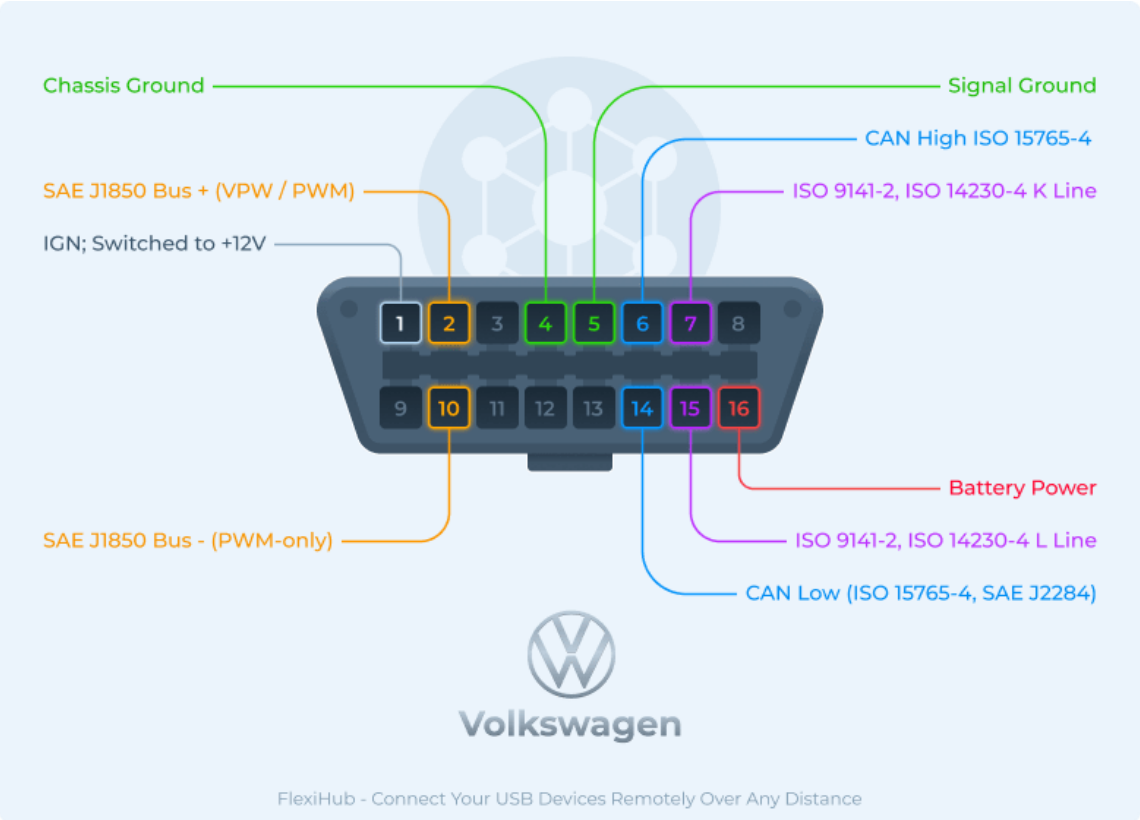
1 Protokoly OBD2 a CAN

1.1 OBD2

OBD, neboli *On-Board Diagnostics*, je systém diagnostiky palubních systémů vozidel, který umožňuje sledovat a analyzovat stav různých komponent automobilu. Původně byl vyvinut pro účely emisní kontroly a diagnostiky motorových systémů, ale dnes se používá k monitorování mnoha systémů vozidla. Tento standard poskytuje informace o chybových kódech, které mohou signalizovat závady v motoru, převodovce, brzdách, emisních systémech a dalších částech.

V tomto systému chyběla jednotná standardizace napříč jednotlivými značkami vozidel, což způsobovalo různé komplikace. Pro každé auto bylo nutné mít speciální čtečku závad, která nešla použít pro jiná vozidla. Každý výrobce měl odlišné typy konektorů, různé diagnostické protokoly a také různá umístění konektorů v autě.

Zásadním pokrokem bylo zavedení OBD2 případně OBD II v roce 1996, které již přineslo tolik nutnou standardizaci a rozšíření schopností. I přes sjednocení standardů využívá OBD II několik typů hardwarových protokolů, které vyžadují příslušný adaptér. Většina evropských a asijských automobilů používá protokoly dle norem ISO 9141-2 a ISO 14230, známé jako KWP 2000. Americká vozidla fungují na protokolech SAE J1850VPW a SAE J1850PWM. Novější automobily přecházejí na protokol podle normy ISO 15765 (CAN BUS), který je v USA povinný pro všechna nová vozidla od roku 1998. Diagnostická zásuvka je standardizována normou SAE J1962 a formát chybových kódů stanovuje norma SAE J2012. (Vlk, 2006) Port skrze který tento systém komunikuje je znázorněni na obrázku 1.



Obrázek 1: OBD2 Port – Volkswagen

Zdroj: *flexihub.com*, 2025a.

Tabulka 1: Popis pinů ODB2 – Volkswagen

Pin Number	Description
1	IGN; Switched to +12V when the ignition is on
2	SAE J1850 Bus + (VPW / PWM)
3	Manufacturer Discretionary
4	Chassis Ground
5	Signal Ground
6	CAN High (ISO 15765-4 and SAE J2284)
7	ISO 9141-2 / ISO 14230-4 K Line
8	Manufacturer Discretionary
9	Manufacturer Discretionary
10	SAE J1850 Bus - (PWM-only)
11	Manufacturer Discretionary
12	Manufacturer Discretionary
13	Manufacturer Discretionary
14	CAN Low (ISO 15765-4 and SAE J2284)
15	ISO 9141-2 / ISO 14230-4 L Line (Optional)
16	Vehicle Battery Power: Type "A" 12V/4A, Type "B" 24V/2A

Zdroj: *flexihub.com*, 2025b.

1.2 OBD2 PIDs

OBD-II PIDs (*identifikátory parametrů palubní diagnostiky*) jsou kódy používané pro vyžádání dat z automobilu, které slouží jako nástroj k diagnostice vozidla. Tyto identifikátory byly definovány ve standartu SAE J1979 Znázorněny jsou v Tabulce 2.

Podle standardu SAE J1979 existuje mnoho různých OBD-II PIDs. Všechna vozidla a nákladní automobily na silnicích prodávané v Severní Americe a Evropě musí podporovat určitou část těchto kódů, což je vyžadováno hlavně kvůli emisním kontrolám. Kromě toho výrobci často přidávají další PIDs specifické pro své modely. I když to není povinné, řada motocyklů podporuje také OBD-II PIDs

Některé těžké nákladní vozy v Severní Americe, například značky Mack a Volvo, využívají pro diagnostiku OBD-II konektor SAE J1962 běžný pro osobní vozy, ale pracují s 29bitovými CAN identifikátory místo běžnějších 11bitových.

Standart SAEJ1979 dále definuje takzvané Services nebo Módy. Tyto služby jsou očíslovány pomocí Hexadecimálního systému (Anon, 2025).

Tabulka 2: OBD2 PIDs

Service / Mode (hex)	**Description**
01	Show current data
02	Show freeze frame data
03	Show stored Diagnostic Trouble Codes
04	Clear Diagnostic Trouble Codes and stored values
05	Test results, oxygen sensor monitoring (non CAN only)
06	Test results, other component/system monitoring (Test results, oxygen sensor monitoring for CAN only)
07	Show pending Diagnostic Trouble Codes (detected during current or last driving cycle)
08	Control operation of on-board component/system
09	Request vehicle information
0A	Permanent Diagnostic Trouble Codes (DTCs) (Cleared DTCs)

Zdroj: Zdroj: *wikipedia.org*, 2025a.

Každá Service obsahuje množství PID, které pak provádí požadované činnosti definované skrz kombinaci Service + PID. Např. Service 01 (Show current data) obsahuje 200 PIDs z rozsahu 00 – C8.

Každé toto PID, má nadefinovanou, jakou syntaxi má mít Request, Response a požadované hodnoty.

Např.

Tabulka 3: Ukázka PID

PID (hex)	PID (dec)	Data bytes returned	Popis	Min value	Max Value	Units	Formula
04	4	1	Vypočtená zátěž motoru	0	100	%	$\frac{100}{255}A$

Zdroj: *wikipedia.org*, 2025b.

1.3 CAN

CAN, je zkratka pro *Control Area Network*, je komunikační protokol určený pro výměnu informací mezi mikrokontroléry a zařízeními v takzvané CAN sběrnici. Poprvé ho představila firma Robert Bosch GmbH v roce 1986, přičemž v roce 1991 byl uveden ve verzi CAN 2.0. Mezinárodní organizace pro normalizaci (ISO) navíc zveřejnila specifikaci CAN jako standard ISO 19898. CAN funguje jako systém s rozšířeným vysíláním, kde datové rámce nejsou předávány přímo mezi odesílatelem a příjemcem, ale rozesílají se do celé sítě, aby je mohly přijímat všechny koncové body. Tento způsob realizace je levný, spolehlivý, efektivní a flexibilní. V rámci komunikačního modelu ISO/OSI pokrývá CAN architektura fyzickou a datovou vrstvu. Ve vozidlech se CAN sběrnice používá k výměně dat mezi řídicími jednotkami nebo senzory, které jsou v nich instalovány (Bosh, 1991).

1.4 CAN FD

CAN FD (Controller Area Network Flexible Data-rate) je rozšířená verze standardního protokolu CAN, která byla navržena s cílem zlepšit přenosové rychlosti a efektivitu datové komunikace. Tento protokol, jehož specifikace byly zveřejněny organizací ISO (International Organization for Standardization) pod názvem ISO 11898-1:2015, byl vyvinut v roce 2012.

Hlavní charakteristikou CAN FD je flexibilní datová rychlost. Na rozdíl od standardního CAN, který omezuje velikost datového rámce na 8 bytů, CAN FD umožňuje přenášet

datové rámce o velikosti až 64 bytů. Tato schopnost umožňuje posílat více informací v jednom datovém paketu, čímž se zvyšuje efektivita přenosu.

Další výhodou CAN FD je možnost vyšších přenosových rychlostí. Zatímco standardní CAN obvykle funguje při maximální rychlosti 1 Mb/s, CAN FD podporuje rychlosti až 8 Mb/s. Tato vlastnost je obzvláště užitečná v moderních vozidlech, kde je třeba rychle přenášet velké objemy dat, například v pokročilých asistenčních systémech pro řidiče (ADAS) a infotainment systémech.

CAN FD je také navržen tak, aby byl zpětně kompatibilní se standardním protokolem CAN. To znamená, že zařízení a systémy, které používají standardní CAN, mohou být integrovány do novějších systémů CAN FD, což usnadňuje přechod na tuto novou technologii.

Rámcová struktura CAN FD zachovává základní prvky standardního CAN, avšak rozšiřuje je o další pole, která umožňují přenos většího množství dat. Nový rámec rovněž obsahuje mechanismy pro detekci chyb a řízení toku dat, což zajišťuje spolehlivost komunikace.

Celkově lze říci, že CAN FD představuje významný pokrok v oblasti automobilové komunikace, neboť kombinuje výhody tradičního CAN s rostoucími požadavky moderních aplikací. Jeho schopnost rychle a efektivně přenášet velké objemy dat z něj činí ideální volbu pro budoucnost automobilové elektroniky a dalších průmyslových aplikací (Bosh, 2023).

1.5 Topologie CAN BUS

Topologie CAN představuje uspořádání, které popisuje způsob, jakým jsou zařízení, jako jsou řídicí jednotky a senzory, připojena k síti CAN. Tato architektura hraje klíčovou roli v efektivní komunikaci mezi jednotlivými komponentami v automobilovém průmyslu a v dalších oblastech.

Základní struktura CAN BUS je založena na dvouvodičové lince, která přenáší data mezi zařízeními, přičemž tato linie se označuje jako "CAN sběrnice". Všechna zařízení v síti jsou k této sběrnici připojena paralelně, což umožňuje decentralizovanou komunikaci, kde každé zařízení může vysílat a přijímat zprávy.

Mezi typy topologií patří lineární a stromová. Lineární topologie je nejčastější uspořádání, kdy jsou zařízení připojena k jedné hlavní sběrnici. Signál se šíří oběma

směry, což zajišťuje jednoduchou a efektivní komunikaci mezi jednotlivými zařízeními. Stromová topologie, na druhé straně, zahrnuje rozdělení hlavní sběrnice na větve, které vedou k různým skupinám zařízení, což umožňuje lepší organizaci a usnadňuje rozšiřování systému.

Na každém konci CAN sběrnice se nachází terminátor, což je rezistor, který zabraňuje odrazům signálu a zajišťuje tak kvalitní přenos dat. Obvykle se používají rezistory o hodnotě 120 ohmů. CAN BUS podporuje různé rychlosti přenosu dat, které se obvykle pohybují v rozmezí od 10 kb/s do 1 Mb/s, přičemž konkrétní rychlost závisí na délce sběrnice a počtu připojených zařízení.

Mezi hlavní výhody CAN BUS patří robustnost, neboť je odolná vůči rušení a dokáže fungovat v náročných podmínkách, což je v automobilovém prostředí klíčové. Dále vysoká efektivita umožňuje sdílení sběrnice mezi více zařízeními, což zvyšuje efektivitu komunikace a snižuje náklady na kabeláž (Bosh, 2023).

1.6 Princip diferenciální komunikace

CAN sběrnice využívá rozdílový přenos signálu, což znamená, že informace je zakódována do napěťového rozdílu mezi vodiči CAN_H a CAN_L (také označovanými jako CAN_High a CAN_Low). Tato metoda přenosu přináší dvě klíčové výhody. Odolnost vůči rušení spočívá v tom, že případný rušivý signál ovlivňuje oba vodiče (CAN_H a CAN_L) shodně, což umožňuje rozdílovému přijímači eliminovat rušení sledováním pouze rozdílu napětí mezi vodiči, nikoli jejich absolutních hodnot. Tato vlastnost zajišťuje spolehlivý přenos dat i v prostředí s vysokým elektromagnetickým rušením. Nízká spotřeba energie pak vyplývá z principu diferenciálního přenosu, který snižuje energetické nároky na komunikaci. To je zvláště výhodné v automobilových aplikacích, kde je efektivní využití energie klíčové. Signály v CAN sběrnici mají dvě základní úrovně:

1) Dominantní úroveň: Napěťový rozdíl mezi CAN_H a CAN_L je přibližně 2 V (CAN_H ≈ 3,5 V, CAN_L ≈ 1,5 V). Tato úroveň má vyšší prioritu a „přebíjí“ ostatní zprávy.
2) Recesivní úroveň: Napětí na CAN_H a CAN_L je téměř shodné (oba vodiče ≈ 2,5 V). Tato úroveň představuje pasivní stav sběrnice.

Komunikace v CAN sběrnici je řízena pevně stanovenými fázemi, které zajišťují spolehlivý přenos dat mezi zařízeními v síti. Každá fáze má jasně definovanou funkci a přispívá k efektivitě a robustnosti komunikace.

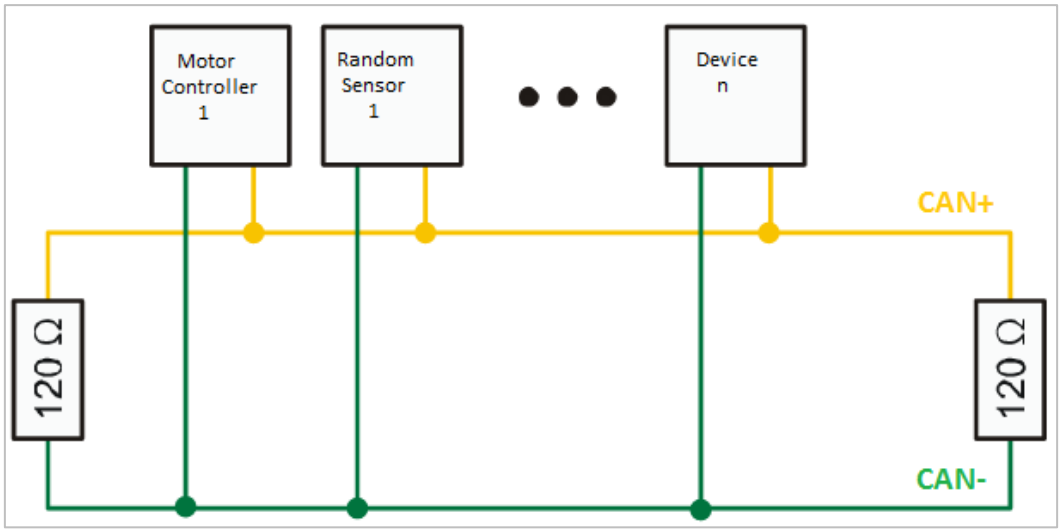
Start: Přenos začíná dominantní úrovní, která signalizuje začátek rámce. Dominantní úroveň na sběrnici (rozdílové napětí mezi CAN_H a CAN_L) je interpretována jako logická 0. Tento signál slouží jako synchronizační bod pro všechna zařízení připojená k sběrnici, která se připraví na příjem dat.

Přenos bitů: Data jsou přenášena jako posloupnost dominantních a recesivních úrovní. Dominantní úroveň představuje logickou 0, zatímco recesivní úroveň, při které je napětí na obou vodičích téměř shodné, značí logickou 1. Tato sekvence umožňuje efektivní přenos dat, kde každé zařízení na sběrnici dokáže interpretovat jednotlivé bity a rekonstruovat původní informaci.

Arbitráž: Při situaci, kdy více zařízení začne vysílat současně, vstupuje do hry mechanismus arbitráže. Priorita zpráv je určena jejich identifikátory – nižší hodnota identifikátoru má vyšší prioritu. Zařízení s dominantním bitem pokračuje v přenosu, zatímco zařízení s recesivním bitem přenos přeruší. Tento mechanismus zajišťuje, že zprávy s vyšší prioritou jsou přenášeny bez zpoždění, a zároveň eliminuje kolize dat.

Zakončení: Přenos je ukončen speciální sekvencí zahrnující kontrolní a potvrzovací mechanismy. Kontrolní sekvence (CRC) ověřuje integritu přenesených dat a detekuje případné chyby, které mohly vzniknout během přenosu. Potvrzovací sekvence (ACK) pak slouží k signalizaci správného přijetí dat. Přijímací zařízení přidá na sběrnici dominantní úroveň, pokud byla data přijata bez chyb, čímž informuje vysílající zařízení o úspěšném dokončení přenosu.

Tento postup fázování zajišťuje, že komunikace v CAN sběrnici je synchronizovaná, efektivní a odolná vůči chybám, což je klíčové pro její využití v automobilovém průmyslu i dalších aplikacích (Bosh, 1991).



Obrázek 2: Topologie sběrnice CAN

Zdroj: robotsforroboticists.com, 2025.

1.7 Formát rámců CAN BUS

Podle definice v normě ISO 15765-4 používají protokoly pro emise (včetně OBD-II, EOBD, UDS atd.) transportní vrstvu ISO-TP (ISO 15765-2). Všechny CAN rámce posílané prostřednictvím ISO-TP mají délku dat 8 bytů. Je doporučeno vyplnit nevyužité datové byty hodnotou 0xCC.

Dotazování na PIDs a následné odpovědi probíhá na CAN sběrnici vozidla. Standardní OBD dotazy a odpovědi využívají funkční adresy. Diagnostická čtečka začíná dotaz pomocí CAN ID 7DF, které slouží jako adresní vysílání, a přijímá odpovědi od jakéhokoli ID v rozsahu 7E8 až 7EF. Řídicí jednotky (ECU), které mohou odpovídat na OBD dotazy, poslouchají jak na funkční vysílací ID 7DF, tak na své přidělené ID v rozmezí 7E0 až 7E7. Odpověď má ID odpovídající jejich přidělenému ID plus 8, tedy od 7E8 do 7EF (International organization for standardization, 2021).

1.8 CAN BUS Request

Dotazy jsou formátovány způsobem uveden v této tabulce.

Tabulka 4: Formátování CAN Request

	Byte							
PID Type	0	1	2	3	4	5	6	7
SAE	Počet dalších bytů - 2	Služba: 01, 02...	PID code Např. 0D	Nepoužívá se, Norma ISO15765-2 doporučuje vyplnit pomocí AA				
VS	Počet dalších bytů - 3	Služba: 01, 02...	PID code Např. 4980		Nepoužívá se, Norma ISO15765-2 doporučuje vyplnit pomocí AA			

SAE – Standardizovaný PID

VS – Specifický PID pro dané vozidlo

Zdroj: (Bosh, 2023).

1.9 CAN BUS Response

Odpovědi na Requesty jsou formátovány dle následující tabulky.

Tabulka 5: Formátování CAN Response

	Byte							
CAN Adresa	0	1	2	3	4	5	6	7
SAE Např. 7E8	Počet dalších bytů 3 - 6	Služba + 40 01+40=41	PID code Např. 0D	Hodnota parametru, byte 0	Hodnota, byte 1 (nepovinné)	Hodnota, byte 2 (nepovinné)	Hodnota, byte 3 (nepovinné)	Nepoužívá se (může být 00 nebo 55)
VS Např: 7E8 nebo 8 + Fyzické ID	Počet dalších bytů 4 - 7	Služba + 40 01+40=41	PID code Např. 4980		Hodnota parametru, byte 0	Hodnota, byte 1 (nepovinné)	Hodnota, byte 2 (nepovinné)	Hodnota, byte 3 (nepovinné)
VS	Počet dalších bytů 3	7F *	Vlastní služba: (např.: 22 = rozšířené diagnostické údaje podle PID, 21 = rozšířené údaje podle posunu)		31	Nepoužívá se (může být 00)		

7F – obecná odpověď obvykle znamená, že modul požadavek nerozpoznal.

SAE – Standardizovaný

VS – Specifický pro dané vozidlo

Zdroj: (Bosh, 2023).

1.10 Ukázka komunikace na CAN BUS

Například takto by vypadala komunikace v případě, že se dotazujeme na rychlost vozidla:

Náš požadavek by byl směřován na PID 0D, které je k tomuto určené.
Request: 7DF 02 01 0D AA AA AA AA AA

- 7DF označuje tzv. broadcast ID – zpráva je určena všem dostupným jednotkám na sběrnici.
- 02 definuje délku datové části požadavku (2 byty).
- 01 označuje servisní režim požadavku (v tomto případě je to dotaz na aktuální data vozidla).
- 0D je právě zmíněný PID, tedy parametr rychlosti vozidla.
- AA AA AA AA AA jsou doplňkové hodnoty, které zde fungují jako „výplňová data“, aby byla zpráva kompletní.

Response: 7E8 03 41 0D 12 AA AA AA AA

- 7E8 je identifikátor řídicí jednotky motoru (ECU), která odpověděla.
- 03 představuje délku datové odpovědi (3 byty).
- 41 znamená kladnou odpověď (pozitivní potvrzení požadavku).
- 0D potvrzuje, že se jedná o odpověď na náš původní dotaz na PID 0D (rychlost vozidla).
- 12 je hexadecimální údaj, který je třeba převést na dekadické číslo, aby bylo možné zjistit rychlost. Číslo 0x12 v šestnáctkové soustavě odpovídá hodnotě 18 v desítkové soustavě.
- AA AA AA AA jsou opět pouze výplňové hodnoty bez dalšího významu.

V tento moment diagnostický nástroj dekóduje odpověď, a my vidíme na obrazovce, že se vozidlo právě pohybuje rychlostí 18 km/h.

Tímto způsobem probíhá veškerá CAN komunikace ve vozidle – každá akce, od rychlosti přes otáčky motoru až po teplotu chladicí kapaliny, má svůj vlastní PID a specifický formát zprávy. Díky této standardizaci může každý diagnostický přístroj snadno a rychle získávat potřebná data.

2 Mikroprocesory v automotive a jejich sběrnice

2.1 Mikroprocesor a jeho využití

Mikroprocesory hrají klíčovou roli v automobilovém průmyslu a jsou základem většiny moderních automobilových systémů. Vzhledem k rostoucímu zaměření na elektroniku a automatizaci se jejich význam neustále zvyšuje a tím se i zvyšuje jejich počet.

Zde je krátký seznam činností, které mikroprocesory v automobilu zajišťují.

- Řízení motoru – ECU Engine control unite, optimalizace výkonu podle vnějších i vnitřních parametrů
- Asistenční systémy řidiče – ADAS, Tyto systémy využívají mikroprocesory k analýze dat z různých senzorů a kamer pro zajištění bezpečnosti a usnadnění řízení. Příkladem jsou adaptivní tempomaty, systémy varování před kolizí a autonomní parkovací asistenty.
- Infotainment a konektivita – Mikroprocesory ovládají multimediální systémy, navigaci a funkce konektivity, jako je Bluetooth a Wi-Fi.
- Bezpečnostní systémy – Mikroprocesory jsou klíčové pro správu bezpečnostních funkcí, jako jsou airbagy, ABS (antiblokovací brzdový systém) a elektronická stabilizace vozidla.

V dnešní době, kdy architektury vozu vedou k vysokému množství řídících jednotek (ECUs), je každá z těchto jednotek vybavena logickým obvodem, stěžejním pro její chod. Tento obvod se nazývá procesor, případně mikroprocesor. Rozeznání těchto dvou se dá učinit takto: Mikroprocesor je konkrétní fyzická realizace procesoru na čipu, zatímco procesor je obecný termín pro jakoukoli jednotku, která provádí zpracování dat.

Ve snaze zjednodušit výrobu a vývoj těchto jednotek, je velká snaha přesunout jejich „inteligenci“ z jednotek samotných do Softwaru je ovládajícího. Z logiky samotné se proto jednotky dělají výkonnější, než musí být, aby byly futureproof a jednoduše updatovatelné.

2.2 Nároky na mikroprocesory v automotive

Oproti komerčně dostupným mikroprocesorům, které známe například z obchodů s elektronikou. Se tyto komponenty instalované do vozidel podrobují mnohem

náročnějším testům a musí splňovat větší množství certifikací. Jedním z požadavků je zvýšená odolnost proti vibracím, zvýšeným teplotám a elektromagnetickému rušení. Tyto podmínky nejsou nijak běžné v domácnostech, ale pro běžný provoz vozidla nijak neojedinělé.

Pro ukázku lze zvolit standardy jako:

- ISO 26262: Tento standard se zaměřuje na funkční bezpečnost automobilových systémů a zajišťuje, že systémy jsou navrženy a implementovány s minimálním rizikem selhání.
- AEC-Q100: Tento standard určuje kvalifikaci integrovaných obvodů pro použití v automobilech, zajišťující vysokou spolehlivost.

Díky tomu, že zařízení, které je vyvíjeno v rámci této bakalářské práce bude použito při diagnostice vozu za statických podmínek. Není potřeba aby mikroprocesor v něm použitý tyto standardy splňoval.

2.3 Mikroprocesor ESP32

V tomto projektu bude využit mikroprocesor ESP32-WROOM-32.

Jedná se o univerzální modul MCU s Wi-Fi + Bluetooth® + Bluetooth LE, který cílí na širokou škálu aplikací, od nízkoenergetických senzorových sítí po nejnáročnější úkoly, jako je kódování hlasu, streamování hudby a dekódování MP3.

V jádru tohoto modulu se nachází čip ESP32-D0WDQ6*. Tento integrovaný čip je navržen tak, aby byl škálovatelný a adaptabilní. Obsahuje dvě CPU jádra, která lze řídit jednotlivě, a frekvence hodinového signálu CPU je nastavitelná od 80 MHz do 240 MHz. Čip také disponuje nízkoenergetickým koprocesorem, který může být použit místo CPU, aby se šetřila energie při plnění úkolů, které nevyžadují velký výpočetní výkon, jako je monitorování periferií. ESP32 integruje širokou škálu periferií, včetně kapacitních dotykových senzorů, rozhraní pro SD karty, Ethernet, vysokorychlostní SPI, UART, I2S a I2C.

Integrace Bluetooth, Bluetooth LE a Wi-Fi zajišťuje, že modul může cílit na široké spektrum aplikací a je univerzální: použití Wi-Fi umožňuje velký fyzický dosah a přímé připojení k internetu prostřednictvím Wi-Fi routeru, zatímco použití Bluetooth umožňuje uživateli pohodlně se připojit k telefonu nebo vysílat nízkoenergetické majáky pro jeho detekci. Sleep cournet čipu ESP32 je nižší než 5 µA, což jej činí

vhodným pro aplikace napájené z baterií a nositelnou elektroniku. Modul podporuje datovou rychlost až 150 Mbps a výstupní výkon 20 dBm na anténě (Espressif Systems, 2019).

Další nespornou výhodou kromě velikého Výběru I/O pinů, je i nízká pořizovací cena.

2.4 Hlavní rodiny mikroprocesorů v automotive

Moderní automobilový průmysl využívá různé rodiny procesorů, které jsou navrženy tak, aby splňovaly specifické požadavky na výkon, spolehlivost a bezpečnost. Tyto procesory podporují funkce od základního řízení motoru po pokročilé systémy autonomního řízení a konektivity.

Mezi nejvýznamnější rodiny procesorů patří například produkty společnosti Renesas Electronics, které zahrnují řady R-Car a RH850. Řada R-Car je využívána především v systémech autonomního řízení, infotainmentu a pokročilých asistenčních systémech (ADAS), protože zvládají v reálném čase zpracovávat data ze senzorů, jako jsou kamery, radary a lidary. Procesory RH850 jsou zase specializované na kritické automobilové aplikace, například řízení motoru, ABS či airbagy, díky své výjimečné odolnosti proti elektromagnetickému rušení.

NXP Semiconductors nabízí několik významných řad, například i.MX pro digitální kokpity, infotainment a Human Machine Interface (HMI), které vynikají zejména grafickým výkonem. Další řada S32 je zaměřena na systémy ADAS a řízení pohonných jednotek, a to díky integrované komunikaci CAN FD a Ethernetu. Řada MPC5xxx splňuje normu ISO 26262 a je určena hlavně pro řízení motorů a bezpečnostní systémy.

Texas Instruments vyrábí například řadu Jacinto, která je určena pro infotainment a ADAS aplikace s důrazem na energetickou úspornost a vysoký výkon. Jejich mikroprocesory C2000 jsou hojně využívané v hybridních a elektrických vozidlech pro přesné řízení motoru.

Infineon Technologies nabízí řady jako AURIX TC4x, které splňují přísné bezpečnostní standardy pro aplikace jako ABS, ESP a řízení motoru, a řadu Traveo II určenou pro digitální kokpity a pokročilé infotainment aplikace díky svým grafickým schopnostem a podpoře ethernetových sítí.

NVIDIA poskytuje procesory platformy Drive pro autonomní řízení, která díky umělé inteligenci a hlubokému učení zvládá náročné analýzy dat. Řada Tegra je určena pro digitální kokpity a infotainment s vysokým grafickým výkonem a senzorickou podporou.

Společnost Intel v automotive využívá především procesory Atom pro aplikace navigace a digitálních kokpitů s důrazem na nízkou spotřebu energie. Xeon procesory se používají tam, kde je nutné zpracovávat velké množství dat v reálném čase, jako jsou autonomní vozidla a cloudové aplikace.

Qualcomm dominuje v automobilové konektivitě a pokročilých technologiích se svou rodinou Snapdragon Automotive, která podporuje infotainment, ADAS, autonomní řízení a díky integraci 5G technologií zajišťuje špičkový uživatelský zážitek.

Procesory ARM, například řady Cortex-R a Cortex-A, jsou známé svou univerzálností. Cortex-R se používá pro bezpečnostní systémy a řízení pohonných jednotek, zatímco Cortex-A nachází uplatnění v aplikacích infotainmentu a ADAS díky své efektivitě a vysokému výkonu.

STMicroelectronics nabízí řady SPC5, zaměřené na řízení motoru a bezpečnostní aplikace, a Accordo pro infotainment a HMI aplikace s nízkou spotřebou energie a vysokou spolehlivostí.

MediaTek pak se svou platformou Dimensity Automotive spojuje 5G konektivitu s pokročilou umělou inteligencí pro moderní digitální kokpity a infotainment aplikace.

Z hlediska našeho použití je nejblíže rodina Snapdragon společnosti Qualcomm, díky svým schopnostem v oblasti vysokého výkonu, energetické účinnosti a robustní konektivity (Bluetooth, Wi-Fi, 5G), což ji činí vhodnou pro širokou škálu automobilových systémů od infotainmentu až po pokročilé asistenční systémy řidiče (ADAS).

2.5 Oblasti využití procesorů Snapdragon v automotive

Pro bližší rozebrání záběru bylo zde použito příkladu rodiny Snapdragon. Tato eodina je rozsáhlá jak v mobilních zařízeních a je v širším povědomí lidí zajímajících se o elektroniku.

- Infotainment a konektivita Procesory Snapdragon podporují pokročilé funkce v oblasti zábavních a informačních systémů vozidel. Mezi hlavní vlastnosti patří:
    - Podpora vysoce kvalitního obrazu a zvuku pro multimediální aplikace.
    - Možnosti streamování obsahu, včetně přímé konektivity s mobilními zařízeními přes Wi-Fi a Bluetooth.
    - Integrace navigačních systémů s real-time informacemi o dopravě a povětrnostních podmínkách.
- Pokročilé asistenční systémy pro řidiče (ADAS) Snapdragon procesory slouží jako základ pro mnoho pokročilých asistenčních funkcí, včetně:
    - Analyzání dat z kamer a senzorů pro detekci překážek, rozpoznávání dopravních značek a sledování jízdních pruhů.
    - Podpora autonomního řízení prostřednictvím vysoce výkonných algoritmů strojového učení a umělé inteligence.
    - Zpracování dat vícekanálové komunikace mezi různými systémy vozidla.

- Konektivita a komunikace Snapdragon zajišťuje spolehlivou a rychlou komunikaci mezi vozidlem a okolním světem, a to prostřednictvím:
    - Technologie 5G pro vysokorychlostní přenos dat, což umožňuje real-time komunikaci s cloudovými službami a dalšími zařízeními.
    - Podpora standardů V2X (Vehicle-to-Everything) pro komunikaci s dalšími vozidly, infrastrukturou a pěšími uživateli.
- Energetická účinnost Snapdragon procesory byly navrženy s důrazem na optimalizaci spotřeby energie. To je klíčové pro zajištění dlouhé životnosti baterií v elektrických a hybridních vozidlech (Canalys, 2025).

Specifické modely Snapdragon pro automotive

Qualcomm vyvinul řadu čipů přímo určených pro automobilový průmysl, jako je Snapdragon Automotive Cockpit Platform a Snapdragon Ride Platform:

- Snapdragon Automotive Cockpit Platform Tento procesor podporuje pokročilé infotainmentové funkce a přístrojovou desku s integrovanými funkcemi AI. Je navržen tak, aby zvládal souběžné zpracování dat z různých aplikací bez ztráty výkonu.
- Snapdragon Ride Platform Tato platforma je zaměřena na autonomní řízení a ADAS systémy. Podporuje strojové učení, zpracování obrazu a komplexní analýzu dat z senzorů a kamer.

Výhody Snapdragon procesorů v automotive

- Modularita: Možnost snadné integrace do různých architektur vozidel.
- Vysoký výkon: Zpracování velkého množství dat v reálném čase.
- Energetická účinnost: Minimalizace spotřeby energie bez kompromisů výkonu.
- Bezpečnost: Implementace pokročilých bezpečnostních standardů pro ochranu dat a systémů vozidla.

Procesory Snapdragon představují klíčový prvek v transformaci automobilového průměru směrem k digitalizaci, konektivitě a autonomii. Jejich schopnost poskytovat vysoký výkon a současně udržet energetickou efektivitu je činí ideální volbou pro budoucnost mobility (Qualcomm, 2024).

2.6 Sběrnice I²C

Sběrnice I²C (Inter-Integrated Circuit), často označovaná také jako I2C, je sériová komunikační sběrnice určená pro připojení různých periferií k mikrokontroléru nebo

mikroprocesoru. Byla vyvinuta firmou Philips Semiconductor (nyní NXP Semiconductors) v roce 1982. Hlavní výhodou této sběrnice je její jednoduchost a možnost připojit více zařízení pomocí pouhých dvou vodičů.

Princip funkce

I²C sběrnice využívá dva signálové vodiče:

- SDA (Serial Data Line) – datový vodič pro přenos dat.
- SCL (Serial Clock Line) – hodinový vodič určující rychlost přenosu dat.

Zařízení na sběrnici jsou rozdělena na dva typy:

- Master (řídící zařízení): Generuje hodinový signál a iniciuje komunikaci s jinými zařízeními.
- Slave (podřízená zařízení): Reagují na výzvy master zařízení, přijímají nebo odesílají data dle instrukcí.

Přenos dat probíhá synchronně, což znamená, že data jsou přenášena v závislosti na hodinovém signálu generovaném zařízením typu master.

Charakteristiky sběrnice I²C

Sběrnice I²C je sériová synchronní sběrnice, která komunikuje pomocí dvou vodičů označovaných jako SDA a SCL. Díky své multi-master a multi-slave architektuře umožňuje současné připojení a komunikaci více zařízení, přičemž každé zařízení disponuje unikátní adresou. Kapacita adresování sběrnice závisí na použité adresaci; standardní 7bitová adresace umožňuje připojit až 128 zařízení, zatímco rozšířená 10bitová adresace umožňuje připojení až 1024 zařízení. Typické přenosové rychlosti sběrnice jsou 100 kbps ve standardním režimu, 400 kbps v režimu Fast mode, 1 Mbps v režimu Fast mode plus a až 3,4 Mbps v režimu High-speed mode (NXP, 2021).

Vrstvy a jejich funkce

I²C sběrnice je tvořena dvěma základními vrstvami dle modelu ISO/OSI:

1. Fyzická vrstva

Fyzická vrstva sběrnice I²C definuje elektrické parametry signálu, tedy napětí, proudy a časování signálů. Každý vodič sběrnice používá otevřený kolektor (open-drain/open-collector), který je připojen přes pull-up rezistor k napájecímu napětí. Tento princip umožňuje jednoduchou detekci konfliktů (bus arbitration) a zároveň

zajišťuje, že více zařízení může bezpečně sdílet stejnou sběrnici bez rizika elektrických konfliktů.

2. Datová vrstva

Datová vrstva sběrnice I²C stanovuje formát datových rámců, způsob adresování zařízení a řízení samotné komunikace. Každá komunikace začíná tzv. start podmínkou, při níž signál SDA přejde z vysoké do nízké úrovně při vysokém stavu signálu SCL. Následuje vyslání adresy zařízení spolu se směrem komunikace (čtení nebo zápis). Poté dochází k samotnému přenosu datových bitů, přičemž po každých osmi přenesených bitech dat je vyslán potvrzovací bit ACK (acknowledge), který slouží jako potvrzení úspěšného přijetí dat přijímacím zařízením. Komunikace je ukončena stop podmínkou, kdy SDA přechází z nízké do vysoké úrovně při vysokém stavu signálu SCL.

Výhody a omezení

Mezi hlavní výhody sběrnice I²C patří její jednoduchost a nízký počet potřebných vodičů, což výrazně usnadňuje návrh plošného spoje a snižuje náklady na výrobu. Navíc je vhodná pro aplikace s nízkou spotřebou energie.

Na druhou stranu omezení spočívají v omezené rychlosti přenosu dat a citlivosti na rušení při delších vzdálenostech kabeláže. Doporučená délka sběrnice je obvykle do několika metrů v závislosti na přenosové rychlosti.

Využití v automotive sektoru

V automobilovém průmyslu je sběrnice I²C často používána pro komunikaci mezi mikrokontrolérem a senzory, displeji nebo jinými periferiemi s nižšími rychlostními nároky, například pro ovládání klimatizace, infotainment systémů, čtení senzorů tlaku, teploty, nebo řízení osvětlení interiéru.

Sběrnice I²C je díky své flexibilitě a jednoduchosti oblíbenou volbou také pro integraci s OBD2 diagnostickými systémy, kde může sloužit k propojení různých interních komponentů diagnostického zařízení, jako jsou senzory nebo displeje (NXP, 2021).

2.7 Sběrnice I2S

I²S (Inter-Integrated Sound) je sériová komunikační sběrnice navržená speciálně pro digitální přenos audio dat mezi integrovanými obvody. Byla vyvinuta společností Philips Semiconductor (nyní NXP Semiconductors) a stala se standardem pro přenos

digitálního zvuku v mnoha aplikacích, jako jsou multimediální zařízení, automobilová audio technika, telekomunikace a profesionální audio zařízení.

Princip fungování

Sběrnice I²S umožňuje jednosměrnou, sériovou komunikaci digitálních audio dat mezi zařízeními. Typická I²S komunikace probíhá mezi jedním vysílačem (např. digitálním audio převodníkem) a jedním nebo více přijímači (např. digitálně-analogovým převodníkem – DAC).

Charakteristika sběrnice I²S

Sběrnice I²S se vyznačuje synchronní sériovou komunikací, která umožňuje přenos audio dat s oddělenými signály pro data a hodiny. Díky této separaci dosahuje vysoké přesnosti a zároveň nízké citlivosti na elektromagnetické rušení. Sběrnice podporuje stereo přenos, kdy jsou audio data rozdělena na levý a pravý kanál. Dále umožňuje flexibilní konfiguraci vzorkovacích frekvencí podle specifických potřeb dané aplikace (Philips Semiconductors, 1996).

Signály sběrnice I²S

I²S využívá tři základní signály: SCLK (Serial Clock), což je sériový hodinový signál synchronizující přenos bitů dat; WS (Word Select), který označuje levý a pravý audio kanál – typicky je v nízké úrovni pro levý kanál a v úrovni vysoké pro pravý kanál; a SD (Serial Data), což je datová linka, na které jsou sériově vysílána audio data. Některé implementace mohou zahrnovat i další signály jako Master Clock (MCLK), který poskytuje základní časovou referenci pro všechny ostatní signály.

Datový formát

Data v rámci I²S jsou přenášena jako sériová posloupnost bitů, obvykle 16, 24 nebo 32 bitů na vzorek. Datový tok je časově rozdělen mezi levý a pravý audio kanál, přičemž WS signál indikuje, který kanál je aktuálně přenášen. Data jsou obvykle přenášena v dvoukomplementovém formátu.

Vrstvy I²S

I²S sběrnice operuje zejména na dvou vrstvách ISO/OSI modelu. Fyzická vrstva definuje elektrické charakteristiky přenosových linek, specifikuje typy vodičů, úrovně napětí a metody diferenciálního přenosu a zabývá se synchronizací pomocí hodinového signálu SCLK. Datová vrstva pak definuje rámcový formát audio dat, zahrnuje metody identifikace levého a pravého kanálu prostřednictvím WS signálu a zajišťuje správnou synchronizaci a časování datových rámců.

Funkce jednotlivých vrstev

Fyzická vrstva sběrnice I²S určuje elektrické charakteristiky signálů, jako jsou napěťové úrovně a časování signálů. Typická napěťová úroveň je kompatibilní s TTL nebo CMOS logikou (např. 3,3 V nebo 5 V). Hodinový signál SCLK je zodpovědný za synchronizaci jednotlivých bitů, zatímco WS signál označuje přenos jednotlivých audio vzorků. Na datové vrstvě se realizuje samotné formátování audio dat do sériových rámců. Každý rámec obsahuje data pro levý a pravý kanál, přičemž kanály se střídají na základě WS signálu. Data se typicky přenáší nejprve nejvýznamnějším bitem (MSB first).

Výhody a nevýhody I²S

Mezi hlavní výhody sběrnice I²S patří jednoduchá a efektivní komunikace audio dat, oddělení hodinového signálu od dat, které umožňuje nízké rušení a vysokou přesnost, a podpora vysoké kvality přenosu zvuku bez nutnosti složitých protokolů. Nevýhodou je omezená vzdálenost přenosu kvůli citlivosti na rušení a časování a také skutečnost, že se jedná o jednosměrnou komunikaci, která vyžaduje samostatné linky pro vysílání a příjem.

Použití sběrnice I²S

Sběrnice I²S je široce využívána v digitálních audio zařízeních, jako jsou přehrávače a rekordéry, audio subsystémy mobilních telefonů a tabletů, automobilové audio a infotainment systémy, a také v profesionálních audio rozhraních a studiových zařízeních (Philips Semiconductors, 1996).

2.8 Sběrnice SPI

SPI (Serial Peripheral Interface) je synchronní sériová komunikační sběrnice, která umožňuje rychlou a efektivní komunikaci mezi mikrokontroléry a periferními zařízeními. Vyvinuta byla společností Motorola v roce 1980. Jedná se o plně duplexní komunikační sběrnici, což znamená, že data mohou být současně vysílána i přijímána.

Základní princip fungování

SPI využívá architekturu typu master-slave, kde jedno zařízení funguje jako hlavní (master) a řídí komunikaci, zatímco ostatní zařízení fungují jako podřízená (slave). Master zařízení generuje hodinový signál (CLK), který synchronizuje přenos dat,

a zároveň určuje, se kterým slave zařízením komunikuje pomocí zvláštního signálu výběru (Slave Select – SS).

SPI sběrnice využívá čtyři vodiče:

- MOSI (Master Output Slave Input) – výstup z master zařízení do slave zařízení
- MISO (Master Input Slave Output) – výstup ze slave zařízení do master zařízení
- SCLK (Serial Clock) – hodinový signál, který řídí rychlost přenosu
- SS (Slave Select) – výběr aktivního slave zařízení

Každé slave zařízení má svůj samostatný SS vodič. Pokud je vodič SS v nízké logické úrovni, znamená to, že zařízení je vybráno a připraveno k přenosu dat (Stoicescu, 2018).

Charakteristika sběrnice SPI

SPI se vyznačuje vysokou rychlostí přenosu, která může dosahovat i několika desítek MHz. Díky své jednoduchosti a rychlosti je ideální pro komunikaci s různými periferiemi, jako jsou senzory, paměťové čipy, displeje, AD/DA převodníky nebo komunikační moduly.

Důležité charakteristiky SPI:

- Plně duplexní komunikace – současný příjem a vysílání dat.
- Vysoká rychlost – běžně až do 50 MHz a více.
- Jednoduchá hardwarová realizace – díky minimálnímu počtu vodičů.
- Snadná rozšiřitelnost – možnost připojení více slave zařízení.
- Nízká spotřeba energie – ideální pro aplikace s omezeným zdrojem energie.

Vrstvy a jejich funkce

Komunikace SPI je tvořena několika vrstvami, které zajišťují spolehlivý přenos dat:

1. Fyzická vrstva

Tato vrstva definuje elektrické vlastnosti signálů a typy vodičů. Sběrnice využívá digitální logické úrovně (typicky 0 V pro logickou „0“ a 3.3 V nebo 5 V pro logickou „1“).

2. Linková vrstva

Linková vrstva řídí způsob, jakým se data přenášejí mezi zařízeními. Hlavním úkolem této vrstvy je generování a synchronizace hodinového signálu (CLK), který udává

rytmus, kdy přesně jsou data odesílána a přijímána. Dále má na starosti řízení signálu Slave Select (SS), který rozhoduje, se kterým periferním zařízením bude hlavní zařízení (master) komunikovat, a přenos dat přes vodiče MOSI a MISO.

Linková vrstva pracuje s dvěma důležitými parametry, které jsou zásadní pro určení charakteristiky přenosu:

- Polarita hodin (CPOL) – tento parametr určuje výchozí stav hodinového signálu, když nedochází k přenosu dat. Pokud je CPOL nastaveno na hodnotu 0, hodinový signál začíná v nízké úrovni, naopak při CPOL nastaveném na hodnotu 1 hodinový signál začíná ve vysoké úrovni.
- Fáze hodin (CPHA) – tento parametr určuje okamžik, kdy dochází ke vzorkování (čtení) nebo zápisu dat vzhledem k hodinovému signálu. Pokud je CPHA nastaveno na 0, data jsou vzorkována na náběžné hraně hodinového signálu. Při CPHA nastaveném na 1 jsou data vzorkována na sestupné hraně hodinového signálu.

Díky kombinaci těchto dvou parametrů vznikají čtyři možné komunikační režimy:

- Režim 0 (CPOL=0, CPHA=0) – hodinový signál začíná v nízké úrovni, data jsou vzorkována na náběžné hraně hodinového signálu. Tento režim je nejčastěji používaný a podporovaný většinou zařízení.
- Režim 1 (CPOL=0, CPHA=1) – hodinový signál je výchozí v nízké úrovni, avšak data jsou vzorkována na sestupné hraně hodinového signálu. Využívá se především pro zařízení, která vyžadují specifické načasování čtení dat.
- Režim 2 (CPOL=1, CPHA=0) – hodinový signál začíná ve vysoké úrovni a data jsou čtena na náběžné hraně hodinového signálu. Tento režim se využívá méně často, obvykle v aplikacích s velmi specifickými požadavky.
- Režim 3 (CPOL=1, CPHA=1) – hodinový signál je ve vysoké úrovni, data jsou vzorkována na sestupné hraně hodinového signálu. Tento režim je rovněž méně častý, avšak vhodný pro některá zařízení, která vyžadují specifickou synchronizaci přenosu.

3. Aplikační vrstva

Aplikační vrstva definuje konkrétní význam přenášených dat. Na této úrovni jsou data interpretována a využita pro specifické funkce periferních zařízení, jako například ukládání dat do paměti, zobrazování údajů na displeji, nebo převod analogového signálu na digitální.

Výhody a nevýhody sběrnice SPI

Výhody:

- Jednoduchá implementace.
- Rychlá a spolehlivá komunikace.
- Plně duplexní přenos dat.
- Snadná rozšiřitelnost.
- Nízká spotřeba energie.

Nevýhody:

- Vyšší počet vodičů při větším počtu slave zařízení.
- Omezená délka vedení kvůli citlivosti na rušení.
- Žádná vestavěná detekce chyb.

Použití sběrnice SPI v automotive

V automobilovém průmyslu je sběrnice SPI široce využívána díky své rychlosti a spolehlivosti. Typickými aplikacemi jsou řídicí jednotky, senzory, digitální displeje, paměťové moduly a různé komunikační rozhraní. SPI umožňuje rychlou a efektivní komunikaci mezi mikrokontroléry a periferními komponentami ve vozidlech, čímž přispívá k lepší funkčnosti a vyšší bezpečnosti automobilových systémů (Stoicescu, 2018).

2.9 Sběrnice UART

UART (Universal Asynchronous Receiver-Transmitter) představuje sériový komunikační protokol, který umožňuje přenos dat mezi zařízeními bez nutnosti synchronizačního hodinového signálu. Jde o jeden z nejčastěji používaných komunikačních standardů v oblasti elektroniky, zejména pro komunikaci mezi mikrokontroléry a periferními zařízeními, jako jsou GPS moduly, Bluetooth moduly, senzory nebo počítače.

Princip funkce

UART umožňuje asynchronní komunikaci, což znamená, že vysílač a přijímač pracují nezávisle, každý se svým vlastním generovaným hodinovým signálem. Komunikace probíhá po dvou vodičích, označovaných jako TX (transmit) a RX (receive). Data jsou

odesílána sériově, bit po bitu, přičemž každý bit je vyslán v přesně definovaném časovém intervalu.

Každý rámec (frame) dat začíná startovacím bitem (start bit), který indikuje začátek přenosu. Poté následuje 5 až 9 datových bitů, volitelně paritní bit (pro kontrolu chyb) a rámec končí jedním nebo dvěma stop bity (stop bit), které signalizují konec datového přenosu. Tento rámec umožňuje snadnou detekci a synchronizaci jednotlivých přenosů bez společného hodinového signálu (Texas Instrument, 2021).

Charakteristika sběrnice UART

- Asynchronní komunikace: Bez sdíleného hodinového signálu mezi přijímačem a vysílačem.
- Duplexní komunikace: Plný duplex je možný při použití dvou samostatných vodičů TX a RX.
- Jednoduchost: Snadná implementace a integrace s mikrokontroléry.
- Rychlost komunikace: Typicky od 1200 bps (bitů za sekundu) do několika Mbps, obvykle 9600 až 115200 bps.
- Parita a kontrola chyb: Možnost volby paritního bitu pro jednoduchou detekci chyb.

Vrstvy a jejich funkce

Sběrnice UART odpovídá fyzické a částečně také datové vrstvě referenčního modelu ISO/OSI.

1. Fyzická vrstva (Physical layer)

Tato vrstva se zabývá přímým elektrickým propojením mezi zařízeními. Používají se většinou napěťové úrovně podle standardu TTL (Transistor-Transistor Logic), tedy 5 V nebo 3,3 V úrovně. Při delších vzdálenostech a rušivém prostředí lze využít převodníky standardu RS-232 nebo RS-485, které zajistí větší odolnost a delší dosah komunikace.

- TX (transmit line) – vodič, po kterém zařízení odesílá data.
- RX (receive line) – vodič, po kterém zařízení přijímá data.

2. Datová vrstva (Data link layer)

Datová vrstva UART je relativně jednoduchá. Jejím úkolem je zajistit správný formát rámců dat pro úspěšnou detekci začátku a konce přenosu.

- Start bit – vždy logická nula, signalizuje začátek přenosu dat.
- Datové bity – obvykle 8 bitů dat, ale počet může být mezi 5 až 9 bity.
- Paritní bit (volitelný) – používá se pro základní kontrolu integrity přenášených dat. Parita může být lichá, sudá nebo se nemusí vůbec používat.
- Stop bity – logická jednička, indikuje konec rámce, obvykle jeden nebo dva bity.

Použití UART v automotive

V automobilovém průmyslu se UART využívá pro komunikaci mikrokontrolerů s periferními zařízeními, jako jsou GPS moduly, Bluetooth rozhraní, senzory tlaku v pneumatikách, nebo pro propojení s diagnostickými moduly

Výhody a nevýhody UART

Výhody:

- Snadná implementace a jednoduchý hardware
- Minimální počet vodičů (dvoulinka TX a RX)
- Asynchronní komunikace, není potřeba synchronizační signál

Nevýhody:

- Omezená přenosová rychlost
- Náchylnost k chybám při dlouhých vzdálenostech bez dalšího zesílení signálu nebo převodníků
- Nepodporuje komunikaci více zařízení na jedné sběrnici bez použití speciálních metod (např. RS-485)

2.10 Porovnání sběrnic

Tabulka 6: Dílčí porovnání sběrnic

Vlastnost	I2C	SPI	UART	I2S
Typ přenosu	Sériový, synchronní	Sériový, synchronní	Sériový, asynchronní	Sériový, synchronní (audio)
Počet vodičů	2 (SDA, SCL)	3–4 (MOSI, MISO, SCK, SS)	2 (TX, RX)	3–5 (WS, SD, SCK, volitelně MCLK)
Rychlost	až 3,4 Mbit/s (High-Speed Mode)	až desítky Mbit/s	typicky 115.2 kbit/s – 1 Mbit/s	závisí na vzorkovací frekvenci (např. 3.072 Mbit/s při 48 kHz/16bit/2ch)
Adresování	Ano (7bit nebo 10bit adresy)	Ne – přímé připojení zařízení	Ne – point-to-point	Ne – určeno pouze pro audio
Full Duplex	Ne	Ano	Ne	Ano
Komplexnost	Nízká až střední	Nízká	Velmi nízká	Střední (specifická pro audio aplikace)
Spotřeba pinu	Nízká (2 piny)	Vyšší (3–4 piny)	Nízká (2 piny)	Vyšší (3+ piny)
Chybová detekce	ACK/NACK	Žádná (nutné řešit softwarově)	Parita (volitelně)	Žádná – přenos závisí na přesném časování
Typ komunikace	Master-Slave	Master-Slave	Peer-to-peer	Master-Slave (audio datový tok)
Použití	Senzory, paměti, RTC, jednoduchá zařízení	Displaye, paměti, rychlé periferie	Moduly, GPS, Bluetooth, PC komunikace	Audio kodeky, digitální mikrofony, DAC

Zdroj: vlastní pracování

Shrnutí a doporučení

- I2C: Ideální pro komunikaci s více nízkorychlostními zařízeními na krátkou vzdálenost (např. senzory). Výhodou je jednoduché zapojení s minimem vodičů.
- SPI: Vhodné tam, kde je požadována vyšší přenosová rychlost nebo potřeba full-duplex. Má nízkou latenci, ale větší nároky na počet vodičů.
- UART: Nejjednodušší a nejlevnější způsob komunikace mezi dvěma zařízeními, často využívaný pro komunikaci s PC nebo moduly jako Bluetooth.
- I2S: Specializovaná sběrnice pro přenos digitálního zvuku. Pokud zařízení nepracuje s audio signálem, není I2S vhodná.

Ekonomické hledisko

Z hlediska nákladů nejsou tyto sběrnice spojeny s přímými finančními náklady (jsou podporovány ve většině mikrokontrolérů jako ESP32), ale liší se podle požadavků na počet vodičů, kabeláž, rozložení PCB a případné komplexity firmware:

- UART a I2C mají nízké nároky a jsou levné na implementaci.
- SPI vyžaduje více pinů a složitější zapojení, ale poskytuje lepší výkon.
- I2S je specializovaná sběrnice – náklady rostou pouze v případě potřeby audio zpracování.

3 Elektrická zařízení motorových vozidel

3.1 Primární zdroj energie

V motorových vozidlech poháněných fosilními palivy nebo elektřinou je primárním zdrojem elektrické energie pro potřeby elektroniky vozu akumulátorová autobaterie. Tato baterie má standartně napětí 12 V a slouží jako nutná zásoba energie pro udržení chodu některých elektronických zařízení v době kdy není aktivovaná hlavní pohonná jednotka. Zároveň u vozidel, která jsou poháněna motory na fosilní paliva slouží tato baterie k nastartování daného motoru. V moment, kdy je aktivovaná pohonná jednotka je tato autobaterie dobíjena pomocí generátoru (dynamo nebo alternátor). U vozidel, která jsou elektrická dochází k dobíjení této baterie skrze trakční baterii.

3.2 Změna napětí – Step-down měnič

Pro potřeby řídících jednotek vozidla (ECU) je napětí 12 V často vysoké. Z tohoto důvodu je nutné ho snížit na hodnotu se kterou bude dané zařízení schopné pracovat. V tento moment nastupuje tzv. Step-down měnič. Toto zařízení nám umožňuje dosažení správných hodnot napětí. Výhodou tohoto zařízení je jeho pracovní rozsah a možnost přesného nastavení. V případě této práce bylo použit stabilizovaný zdroj s LM317. Tento zdroj umožňuje nastavení výstupního napětí od 1.25 do 37 V při proudu až 1.5 A. Tohoto výsledku se však podařilo až dosáhnout po přidání dvou 100nF kondenzátoru pro stabilizaci napětí.

Samotný step-down měnič funguje na principu spínacího prvku, který se periodicky spíná a vypíná. Během sepnutého stavu se ukládá energie ve formě magnetického pole generovaného induktorem. Následně dojde p přepnutí do vypnuté fáze. Zde se induktor pokouší udržet plynulý proud proudu. Zde se projevují námi přidané kondenzátory, které pomáhají vyladit napěťové špičky. Regulace výstupního napětí se provádí změnou času (duty cycle), kdy je spínač sepnutý. Čím delší doba sepnutí tím vyšší je průměrné výstupní napětí. To je však stále nižší než napětí vstupní.

3.3 Změna napětí – Boost konvertor

V některých aplikacích v automobilové technice je nezbytné naopak zvýšit elektrické napětí oproti standardnímu palubnímu napájení. Typickým příkladem takového

požadavku jsou aktivní stabilizační systémy u vozidel značky BMW. Tyto systémy využívají napětí 48 V, které je čtyřnásobně vyšší než běžné napětí 12 V palubní sítě, napájené klasickou olověnou akumulátorovou baterií. Aby bylo možné tohoto vyššího napětí dosáhnout, je do obvodu zařazen tzv. Step-up měnič, známý také pod názvem boost konvertor.

Boost konvertor je spínaný napěťový měnič, jehož základní princip spočívá ve zvýšení vstupního napětí na vyšší úroveň. Funkce tohoto měniče je založena na řízeném cyklu nabíjení a vybíjení indukčního prvku – induktoru. V první fázi provozního cyklu, kdy je spínací prvek (obvykle výkonový tranzistor typu MOSFET) sepnutý, proud protéká induktorem. V tomto okamžiku se elektrická energie akumuluje ve formě magnetického pole uvnitř induktoru. V této fázi je výstupní dioda polarizována závěrně a brání průchodu proudu směrem ke kondenzátoru, čímž se energie do výstupu nepřenáší.

Ve druhé fázi, po rozepnutí spínače, dochází k náhlému přerušení toku proudu. Induktor se podle zákona elektromagnetické indukce snaží zachovat plynulý tok energie a začne se vybíjet. V důsledku kolapsu magnetického pole induktor generuje napětí opačné polarity, které se přičítá k původnímu vstupnímu napětí. Toto součtové napětí překoná úroveň výstupního napětí a otevře výstupní diodu. Následně se uložená energie přenáší do výstupního kondenzátoru, kde dochází k její akumulaci. Tímto procesem dochází ke zvýšení výstupního napětí.

Klíčovým parametrem pro regulaci výstupního napětí je tzv. střída (duty cycle), což je poměr doby, po kterou je spínač sepnutý, k celkové periodě spínacího cyklu. Zvyšováním této doby se v induktoru uloží více energie, což vede k vyššímu výstupnímu napětí. V moderních systémech je tento parametr řízen mikroprocesorem nebo integrovaným řídicím obvodem, který dynamicky přizpůsobuje střídu aktuálním provozním podmínkám.

Boost konvertory jsou velmi efektivní zařízení, jejichž účinnost se v závislosti na konstrukci, kvalitě komponent a provozním rozsahu může pohybovat v rozmezí 80 až 95 %. V automobilovém průmyslu je jejich využití zvláště výhodné v aplikacích, kde je potřeba napájet výkonnější aktuátory či jiné komponenty vyšším napětím, aniž by bylo nutné zavádět samostatné napájecí větve nebo zvyšovat rozměry a hmotnost akumulátoru.

4 Nástroje pro vývoj

4.1 Programovací jazyky

V rámci této bakalářské práce byly použity 2 programovací jazyky. Pro vývoj firmwaru pro zařízení na bázi ESP32 byl použit jazyk C/C++. Pro vývoj ovládací aplikace se využilo jazyka Python. Obě tyto volby byly vybrány na základě zhodnocení náročnosti, dostupných zdrojů, komunity a znalostí se kterými bylo pracováno.

C/C++

Jazyk C byl vytvořen v roce 1972 Dennisem Ritchiem v laboratořích společnosti Bell Labs jako jazyk pro tvorbu systémového softwaru, zejména operačního systému UNIX. Jedná se o kompilovaný, procedurální jazyk, který poskytuje nízkoúrovňový přístup k paměti a umožňuje efektivní řízení hardwaru. Je známý svou jednoduchostí, přenositelností a výkonností.

C++ vznikl jako rozšíření jazyka C v 80. letech 20. století (autorem je Bjarne Stroustrup) a přidal podporu objektově orientovaného programování. Umožňuje programátorům využívat jak nízkoúrovňové funkce (jako C), tak i složitější struktury (třídy, šablony, dědičnost apod.) (Nemonet, 2025).

Python

Python je vysokoúrovňový interpretovaný jazyk, který vytvořil Guido van Rossum a poprvé byl vydán v roce 1991. Klade důraz na čitelnost kódu, jednoduchou syntaxi a univerzálnost. Python se rychle stal jedním z nejpoužívanějších jazyků v oblastech jako je automatizace, datová analýza, webový vývoj, vědecké výpočty nebo strojové učení (Python Software Foundation, 2025).

4.2 Vývojové prostředí

Arduino IDE

Arduino Integrated Development Environment (IDE) je jednoduché a přehledné vývojové prostředí navržené pro programování mikrokontrolérů, zejména těch, které používají platformu Arduino. První verze byla uvedena v roce 2005 a od té doby si získala popularitu díky své uživatelské přívětivosti, otevřenému přístupu a rozsáhlé komunitní podpoře.

Arduino IDE umožňuje psaní, kompilaci a nahrávání kódu do mikrokontrolérů prostřednictvím několika kliknutí. Používá zjednodušený jazyk založený na C/C++, což usnadňuje přechod začátečníkům, ale zároveň ponechává prostor pro pokročilé funkce. Výhodou je i velké množství dostupných knihoven, které usnadňují práci s různými senzory, moduly a komunikačními protokoly (Arduino, 2025).

Visual Studio Code

Visual Studio Code (VS Code) je moderní, multiplatformní textový editor a vývojové prostředí od společnosti Microsoft, představené v roce 2015. Díky své modularitě, nízké náročnosti a rozsáhlé podpoře rozšíření (extensions) se rychle stal jedním z nejpopulárnějších editorů pro vývoj v různých programovacích jazycích.

Pro vývoj v jazyce Python byl v rámci této práce využit právě VS Code. Pomocí rozšíření jako „Python“, „Pylance“ a „Jupyter“ bylo možné efektivně psát, ladit a testovat skripty, které komunikovaly s OBD2 donglem a analyzovaly získaná data. Díky vestavěné podpoře pro Git, ladění a integrovaný terminál nabízí VS Code výborné zázemí i pro rozsáhlejší projekty (Microsoft, 2025).

5 Návrh řešení

Při vývoji univerzálního diagnostického nástroje pro automobilové aplikace byl kladen důraz na jeho snadnou nastavitelnost, přizpůsobitelnost různým diagnostickým úlohám a kompatibilitu s běžně používanými systémy vozidel. Z toho důvodu bylo nezbytné vyřešit několik klíčových konstrukčních aspektů, které mají zásadní dopad na celkovou funkčnost, spolehlivost a uživatelskou přívětivost zařízení.

Napájení zařízení

Zařízení je napájeno z palubní sítě vozidla prostřednictvím standardizovaného diagnostického konektoru OBD-II, který poskytuje napětí 12 V. Toto napětí však není přímo použitelné pro napájení nízkonapěťových logických obvodů mikrokontroléru a dalších citlivých komponent. Z tohoto důvodu byl integrován Step-down DC/DC měnič (tzv. buck konvertor), jehož úkolem je konvertovat vstupní napětí na stabilizovanou hodnotu 5 V. Tato výstupní úroveň je vhodná pro napájení řídicích obvodů, komunikačních rozhraní a periferií. Měnič byl zvolen s ohledem na vysokou účinnost, kompaktní rozměry a schopnost pracovat v prostředí s proměnlivým zatížením.

Komunikace s CAN sběrnicí

Zajištění komunikace s řídicí jednotkou vozidla prostřednictvím CAN (Controller Area Network) sběrnice je jedním ze základních požadavků. Pro tento účel byl zvolen specializovaný komunikační modul MCP2515, který slouží jako rozhraní mezi CAN sběrnicí a mikrokontrolérem. Tento převodník umožňuje přenos dat prostřednictvím SPI (Serial Peripheral Interface), které je nativně podporováno zvoleným mikrokontrolérem. Díky tomu je možné dosáhnout rychlé a stabilní komunikace při minimální zátěži procesoru, což je klíčové pro reálný časový přenos diagnostických dat.

Komunikace s ovládacím zařízením

Pro ovládání zařízení a přenos diagnostických informací k uživateli bylo zvoleno Bluetooth rozhraní. Ačkoliv použitý mikrokontrolér ESP32 disponuje i Wi-Fi konektivitou, bylo její použití zavrženo z důvodu potenciálních problémů s rušením a přetížením bezdrátových sítí v prostředí autoservisů. Bluetooth technologie nabízí nižší energetickou náročnost, dostatečný komunikační dosah pro práci v okolí vozidla a zároveň dostatečnou datovou propustnost pro přenos výsledků diagnostiky. Výběr

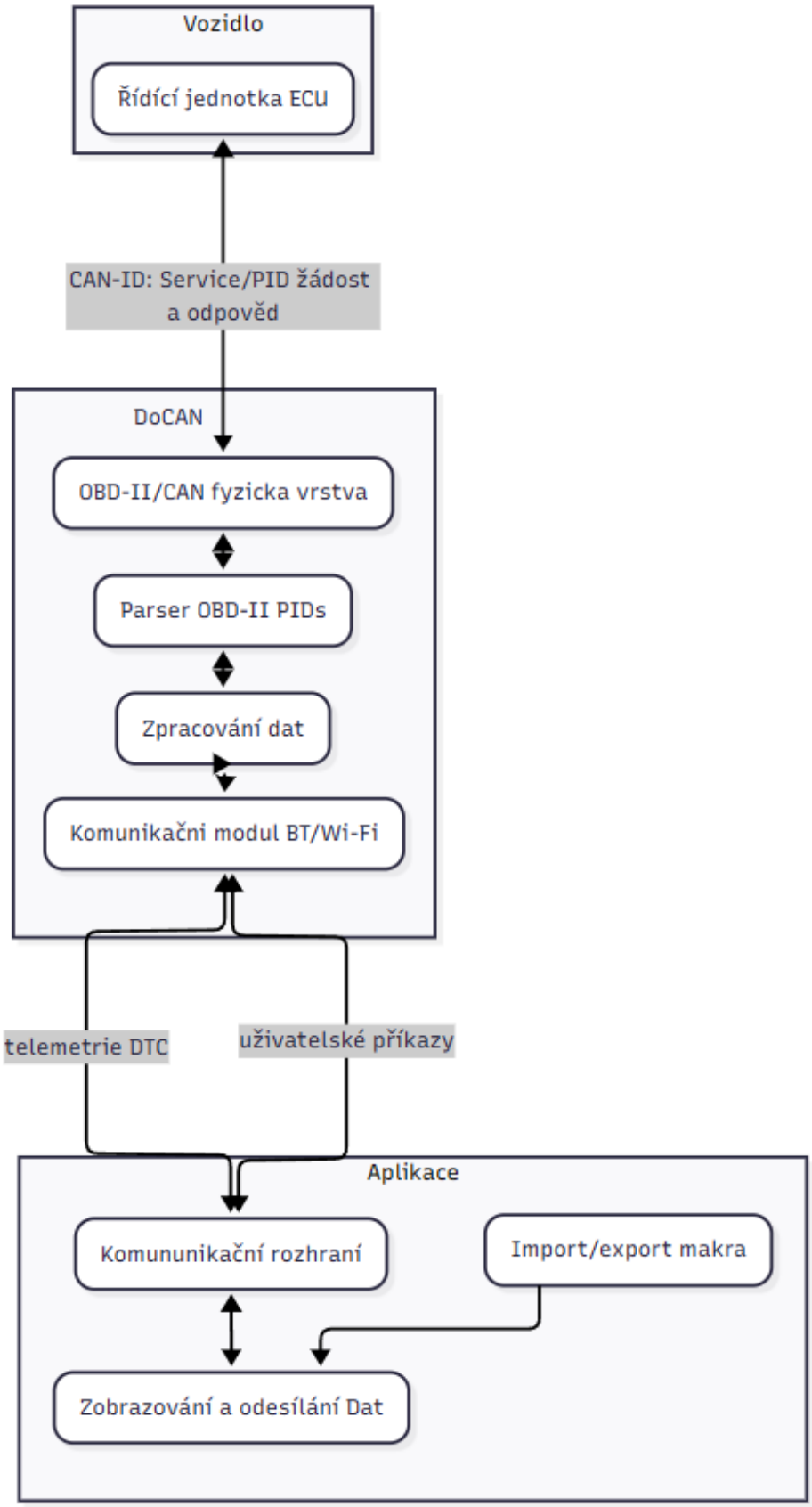
tohoto řešení dále pomáhá minimalizovat proudové zatížení napájecího měniče, který není navržen pro trvalé dodávání vysokých proudů.

Indikační prvky

Pro zvýšení uživatelské přehlednosti a snadnou orientaci během diagnostického procesu byl do návrhu zařízení integrován LCD displej. Tento prvek slouží k zobrazování provozních stavů, chybových hlášení nebo aktuálních hodnot diagnostických parametrů. Vizuální zpětná vazba výrazně přispívá ke snížení chybovosti a zrychlení práce servisního technika.

Ovládací aplikace

Významným prvkem celého systému je návrh uživatelského softwarového rozhraní pro platformu Windows, které umožňuje pohodlné ovládání diagnostického nástroje. Aplikace poskytuje intuitivní prostředí pro zadávání příkazů, zobrazení výsledků měření a interpretaci diagnostických kódů. Díky propojení s Bluetooth modulem je možné zařízení ovládat bezdrátově a zcela nezávisle na fyzickém připojení. Tento přístup výrazně zvyšuje komfort obsluhy a snižuje celkovou časovou náročnost servisních úkonů.

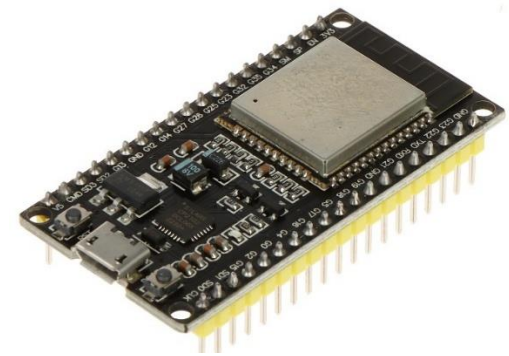


Obrázek 3: Principiální schéma

Zdroj: Vlastní diagram

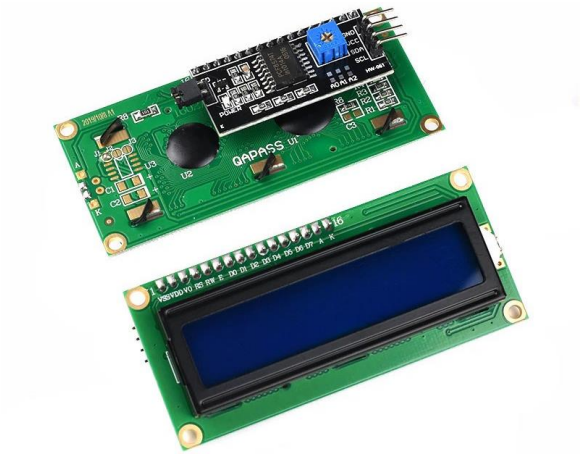
5.1 Návrh Hardwarového řešení

Celé zařízení je sestaveno z několika klíčových komponent. Centrálním prvkem je mikrokontrolérový modul ESP-WROOM-32, který v tomto návrhu využívá také integrovaný převodník CP2102. Tento čip výrazně usnadňuje proces vývoje a nahrávání firmwaru, jelikož zajišťuje spolehlivou komunikaci s vývojovým prostředím prostřednictvím rozhraní USB.



Obrázek 4: ESP32-WROOM-32
Zdroj: hurt.zabezpieczeniapoznan.pl, 2025.

Součástí systému je také znakový LCD displej s rozlišením 16×2 (16 znaků na dvou řádcích), který je osazen převodníkem umožňujícím komunikaci po sběrnici I2C, čímž se snižuje počet potřebných vodičů a zjednodušuje propojení.



Obrázek 5: LCD 16x2 SPI
Zdroj: Tecmikro.com, 2025.

Jednou z nejdůležitějších komponent vedle samotného mikrokontroléru je převodník MCP2515, jenž zajišťuje propojení mezi CAN sběrnicí vozidla a SPI rozhraním mikrokontroléru. Tento obvod splňuje normu CAN 2.0B, což umožňuje kompatibilitu i s novějšími modely vozidel a zajišťuje dlouhodobou využitelnost zařízení.

Obrázek 6: MCP2515
Zdroj: Laskakit.cz, 2025a.

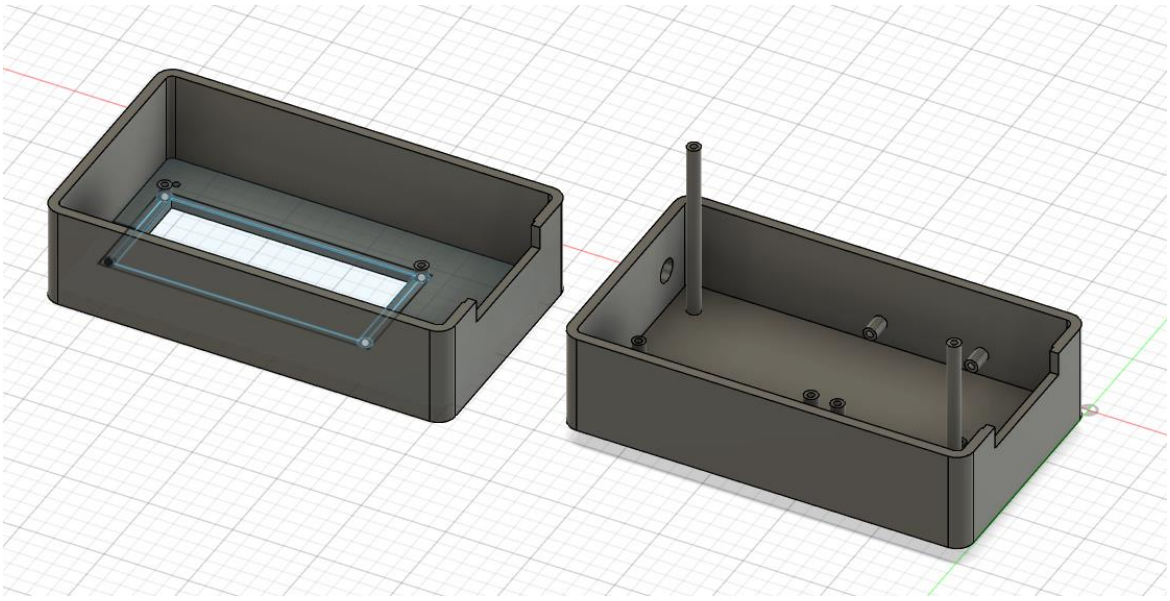
Napájení celého systému je realizováno prostřednictvím diagnostického OBD2 portu vozidla, který poskytuje napětí 12 V. Pro zajištění provozu citlivé elektroniky je nezbytná stabilizace napětí na úroveň 5 V, což je realizováno pomocí lineárního regulátoru LM317. Tento regulátor poskytuje dostatečný výkon a stabilní výstupní napětí pro napájení všech komponent zařízení.

Obrázek 7: LM317
Zdroj: Laskakit.cz, 2025b.

Veškeré použité moduly byly voleny s ohledem na kompatibilitu s napájením 5 V, čímž bylo eliminováno riziko nesouladu s napětím 3,3 V, které vyžaduje samostatný napájecí obvod. Tímto přístupem dochází ke zjednodušení konstrukce a minimalizaci možných komplikací při návrhu i provozu zařízení.

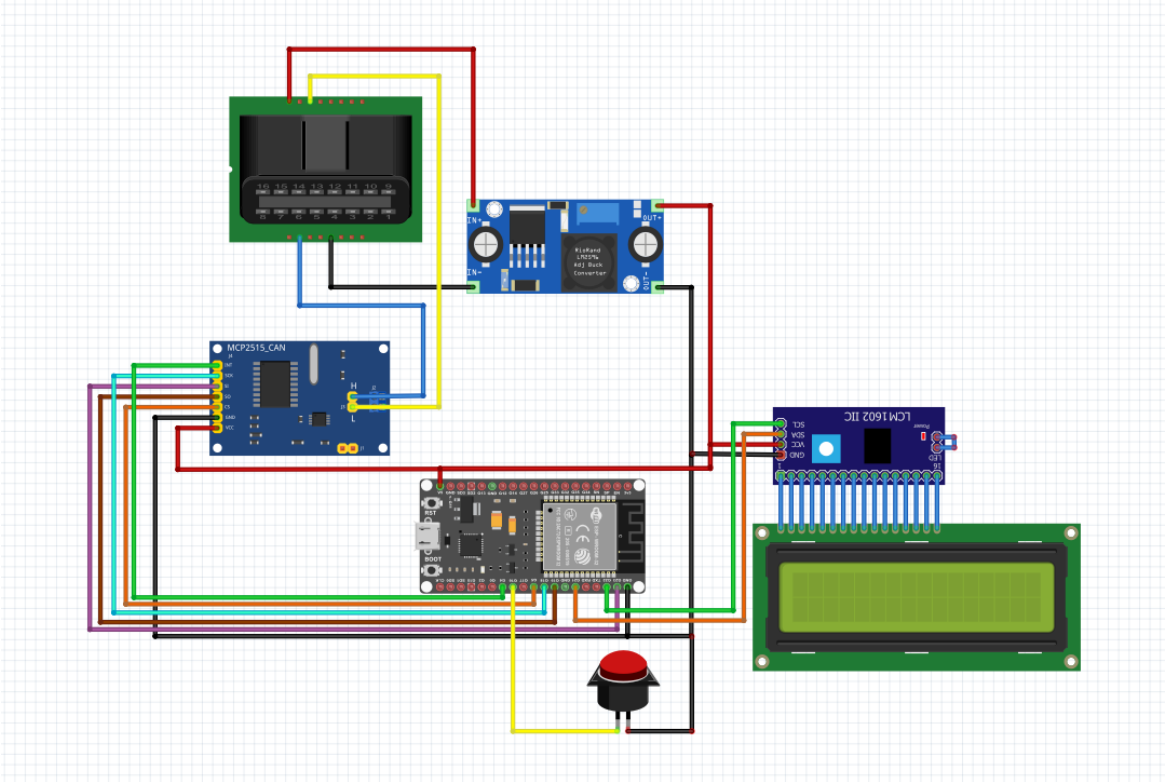
Celé zařízení následně bude uzavřené v obalu vytvořeném na 3D tiskárně. Díky tomu se docílí vytvoření dílů s přesnými rozměry a potřebami pro tento postup. Skořepina bude na 2 poloviny, kdy v jedné bude otvor pro display s vnitřní stranou připravenou pro uchycení pomocí šroubů. A druhá polovina bude uzpůsobena pro uchycení zbylých komponent. Do vymodelovaného otvoru na jedné straně bude pomocí drážky vložen OBD konektor, který bude zároveň sloužit jako uchycovací prvek a jako prvek strukturální pevnosti pro celou skořepinu. Obrázky níže pak popisují tvar obalové skořepiny a barevné schéma zapojení celého zařízení. Na schéma je v levém horním rohu OBD konektor, který přichází komunikace po CAN a napájení. Napájení následně

jde přes Step-down měnič (červený a černý vodič) a napájí všechny moduly. Komunikace jde skrze CAN převodník a následně do ESP32. Tato komunikace je pomocí 4 vodičů skrze sběrnici SPI. Z mikroprocesoru pak jsou 2 vodiče do převodníku u display, který nám zajišťuje jednodušší komunikaci a šetří místo a vodiče. Protože místo 16 jsou potřeba jen 4.



Obrázek 8: 3D Návrh obalové skořepiny

Zdroj: Vlastní příspěvek



Obrázek 9: Schéma zapojení

Zdroj: Vlastní příspěvek

5.2 Návrh firmwaru pro ESP32

Firmware vyvíjený pro mikrokontrolér ESP32 vytváří základní logiku zařízení určeného pro diagnostiku vozů prostřednictvím rozhraní OBD2 a CAN sběrnice. Tento firmware musí zajišťovat propojení mezi vozidlem a uživatelem pomocí tří hlavních komponent: CAN řadiče MCP2515, Bluetooth komunikace s PC a LCD displeje pro lokální vizualizaci dat.

Po spuštění zařízení bude docházet k inicializaci periferií:

LCD displej, který slouží pro zobrazení základních informací a přijatých zpráv z vozidla.

MCP2515, samostatný řadič sběrnice CAN, který komunikuje s ESP32. Řadič je inicializován s parametry, které jsou vhodné pro použití u vozů se kterými budeme pracovat. V našem případě Škoda:

- režim příjmu všech zpráv (*MCP_ANY*),
- přenosová rychlost 500 kbps,
- oscilátor 8 MHz.

Další modulem nutným inicializovat je Bluetooth SPP (Serial Port Profile), který umožňuje bezdrátovou komunikaci s počítačem.

V případě úspěšné inicializace bude uživatel informován prostřednictvím zprávy na LCD. V opačném případě se zařízení zastaví v chybovém stavu. Pokud celý proces proběhl korektně nastává další krok tzv. Hlavní smyčka (loop()).

Během běhu zařízení probíhají tyto činnost i:

- Čtení zpráv z CAN sběrnice
  Pokud řadič MCP2515 detekuje přijatou zprávu z vozidla, tato zpráva bude:
  - dekódována (identifikátor a datové bajty),
  - zobrazena na LCD (zkrácený výpis),
  - přeposlána ve formátu ID: 0xXXX Data: XX XX XX… přes Bluetooth do PC

- Příjem příkazů z ovládacího zařízení (Bluetooth)
  Firmware neustále naslouchá příkazům přicházejícím přes Bluetooth rozhraní. Příkazy jsou očekávány ve formátu „ID:DATA“.
  Například: 7DF: 02 01 0C
  Identifikátor zprávy – 7DF
  Datová část – 02 01 0C
- Firmware dále kontroluje stisknutí tlačítka. Tato činnost by vyvolala přednastavenou sekvenci, která v našem případě bude nastavena na spuštění definovaného makra / sady příkazů.

Zde je ukázka dekódování zprávy z druhé činnosti, kterou firmware zajišťuje. Funkce parseCommand přijímá vstupní řetězec cmd ve tvaru <ID>:<DATA> a ukazatele na proměnné, do kterých má uložit rozparsované CAN-ID, datové bajty a jejich počet. Nejprve pomocí indexOf(':') vyhledá pozici oddělovače mezi částí představující hexadecimální ID a částí s daty; pokud oddělovač chybí, okamžitě vrátí false. Poté vyjme podřetězec před dvojtečkou a pomocí strtoul ho převede z hexadecimálního zápisu na číselnou hodnotu, která se uloží do proměnné *canId. Dále inicializuje délku dat *len na nulu a vezme zbytkový podřetězec za dvojtečkou jako řetězec data. V následujícím cyklu pak až osmkrát vyhledává v data další mezerou oddělený blok znaků, každý z těchto bloků opět pomocí strtoul převede z hexu na jednobajtovou hodnotu a uloží ji do pole buf; současně zvyšuje čítač *len. Když dosáhne konce řetězce nebo maximálního počtu osmi bajtů, cyklus skončí a funkce vrátí true, čímž indikuje úspěšné rozparsování vstupu.

Zdrojový kód 1: Parsování ID:DATA

```
1. // Parsování „ID:DATA“
2. bool parseCommand(const String &cmd, unsigned long *canId,
   uint8_t *buf, uint8_t *len) {
3. int sep = cmd.indexOf(':');
4. if (sep < 0) return false;
5. *canId = strtoul(cmd.substring(0, sep).c_str(), NULL, 16);
6. *len = 0;
7. String data = cmd.substring(sep + 1);
8. int pos = 0;
9. while (pos < data.length() && *len < 8) {
10.         int nxt = data.indexOf(' ', pos);
11.         if (nxt == -1) nxt = data.length();
12.         buf[(*len)++] = (uint8_t)strtoul(data.substring(pos,
   nxt).c_str(), NULL, 16);
13.   pos = nxt + 1;
14.         }
15.         return true;
16.         }
```

A takto kod pracuje s přijatou zprávou po sběrnici CAN. Kydy nejprve se deklarují proměnné len (nastavené na 0), pole buf[8] pro uložení bajtů a 32bitové canId. Poté se pomocí CAN.checkReceive() ověří, zda je k dispozici nová CAN zpráva; pokud ano a volání CAN.readMsgBuf(&canId, &len, buf) vrátí CAN_OK, pokračuje zpracování. Nejprve se funkcí snprintf sestaví řetězec l1 s formátovaným CAN ID (tři hexadecimální číslice) a délkou zprávy, poté se do l2 vkládají až čtyři první datové bajty (pomocí min(len, 4)) jako dvoumístné hex hodnoty oddělené mezerami. Následně se vyčistí LCD, na první řádek se vypíše l1 a na druhý l2. Nakonec se sestaví objekt String msg začínající "ID:0x" plus hexadecimální ID a " Data:", ke kterému se v dalším cyklu připojí všechny bajty zprávy, a tento kompletní text se odesílá jak přes klasický sériový port Serial.println, tak přes Bluetooth SerialBT.println.

Zdrojový kód 2: Zpracování přijaté CAN zprávy

```
  // 2) Příchozí CAN zpráva
  uint8_t len = 0, buf[8];
  unsigned long canId;
  if (CAN_MSGAVAIL == CAN.checkReceive()) {
    if (CAN.readMsgBuf(&canId, &len, buf) == CAN_OK) {
      char l1[17], l2[17] = {0};
        snprintf(l1, sizeof(l1), "ID:%03X Len:%u", (unsigned)canId,
len);
      int toShow = min(len, (uint8_t)4), p = 0;
      for (int i = 0; i < toShow; i++)
        p += snprintf(l2 + p, sizeof(l2)-p, "%02X ", buf[i]);
      lcd.clear();
      lcd.setCursor(0,0); lcd.print(l1);
      lcd.setCursor(0,1); lcd.print(l2);

      String msg = "ID:0x" + String(canId, HEX) + " Data:";
      for (int i = 0; i < len; i++) msg += " " + String(buf[i], HEX);
      Serial.println(msg);
      SerialBT.println(msg);
    }
  }
```

První kód (parseCommand) nepracuje přímo s hardwarem – jako vstup bere čistě ASCII-řetězec ve formátu <ID>:<DATA> a jeho úkolem je tento řetězec rozparsovat na číselné CAN-ID a až osm bajtů dat uložených v poli, přičemž výsledky vrací volajícímu přes ukazatele a vyhodnocuje úspěch návratovou hodnotou true/false. Druhý blok kódu naopak čeká na skutečnou CAN-zprávu ze sběrnice (pomocí CAN.checkReceive() a CAN.readMsgBuf), čte jí přímo z hardware, formátuje její ID a data přes snprintf a poté je okamžitě zobrazí – první čtyři bajty na LCD, všechna data pak pošle přes sériový port i Bluetooth. Zatímco první řeší pouze převod textu na hodnoty, druhý obsluhuje komunikaci, formátování výstupu a jeho fyzické zobrazení či odeslání.

5.3 Návrh Aplikace pro ovládací zařízení

Navržená aplikace respektuje principy modulární architektury, což umožňuje snadné rozšiřování a údržbu jednotlivých částí systému. Základní architekturu tvoří tři nezávislé komponenty: vrstva pro komunikaci se zařízením, zpracování makrosekvencí a uživatelské grafické rozhraní.

Hlavní vstupní bod představuje třída BluetoothCANApp, která při inicializaci v metodě **init** vytváří všechny potřebné uživatelské prvky. Jedná se o rámce pro připojení k zařízení, logovací oblast, rozhraní pro odesílání dat, sekci pro makrosekvence, oblast pro nastavování filtrů a stavovou lištu. V této fázi také aplikace připravuje základní proměnné, které spravují sériové připojení, logování a běh asynchronního engine:

Zdrojový kód 3: Příprava na běh apliakce

```
self.serial_connection = None

self.running = False
self.log_file = os.path.join(os.path.expanduser('~'),
'bluetooth_can_log.txt')
self.current_macro = []
self.async_macro_engine = None
self.loop = None
self.loop_thread = None
```

Komunikační vrstva využívá knihovnu pyserial. Při spuštění aplikace dochází k automatické detekci dostupných portů pomocí serial.tools.list_ports.comports(). Po navázání spojení se zařízením probíhá komunikace pomocí metod connect_bluetooth() a disconnect_bluetooth(), které spravují instanci serial.Serial. Komunikace je kontinuálně sledována v samostatném vlákně, které čte příchozí data a rozhoduje o jejich dalším zpracování. Zde je příklad hlavního komunikačního vlákna:

Zdrojový kód 4: Čtení dat

```
    # --- Čtení dat ---

    def read_data(self):
        while self.running:
            try:
                if self.serial_connection and
self.serial_connection.in_waiting:
                    data =
self.serial_connection.readline().decode('utf-8',
errors='ignore').strip()
                    if data:
                        if self.async_macro_engine and
self.async_macro_engine.macro_active:
                            if not
self.async_macro_engine.stop_requested:
                                self.async_macro_engine.add_response(dat
a)
                        else:
                            self.safe_log_message(data)
            except Exception as e:
                self.safe_log_message(f"Error reading data: {e}")
            time.sleep(0.1)
```

Pro zpracování makrosekvencí slouží třída AsyncMacroEngine, která běží v samostatném vlákně s využitím asynchronního event-loopu z modulu asyncio. Díky tomu aplikace zajišťuje, že časově náročné operace, jako je odesílání příkazů a čekání na odpovědi, neblokují hlavní grafické rozhraní. Každý krok makra je zaznamenán, čeká na odpověď s nastaveným timeoutem a umožňuje okamžité zrušení uživatelem.

Zdrojový kód 5: Asynchronní engine

```
# --- Asynchroní makro engine ---

class AsyncMacroEngine:
    def __init__(self, serial_connection, log_function):
        self.serial_connection = serial_connection
        self.log = log_function
        self.response_queue = queue.Queue()
        self.macro_active = False
        self.stop_requested = False

    def add_response(self, message):
        if self.macro_active and not self.stop_requested:
            self.response_queue.put(message)
        else:
            self.log(message)

    async def wait_for_response(self):
        while True:
            try:
                response = self.response_queue.get_nowait()
                return response
            except queue.Empty:
                await asyncio.sleep(0.1)
```

Grafické rozhraní, realizované pomocí knihovny Tkinter, je strukturováno do přehledných sekcí pro snadné ovládání uživatelem. Rozhraní zahrnuje například dynamicky generované prvky pro výběr Bluetooth portů, logovací okno s možností filtrování zpráv a přehledný editor makrosekvencí. Například editor makrosekvencí využívá ttk.Treeview, který uživateli umožňuje pohodlně editovat jednotlivé kroky sekvence:

Zdrojový kód 6: použití TKinter

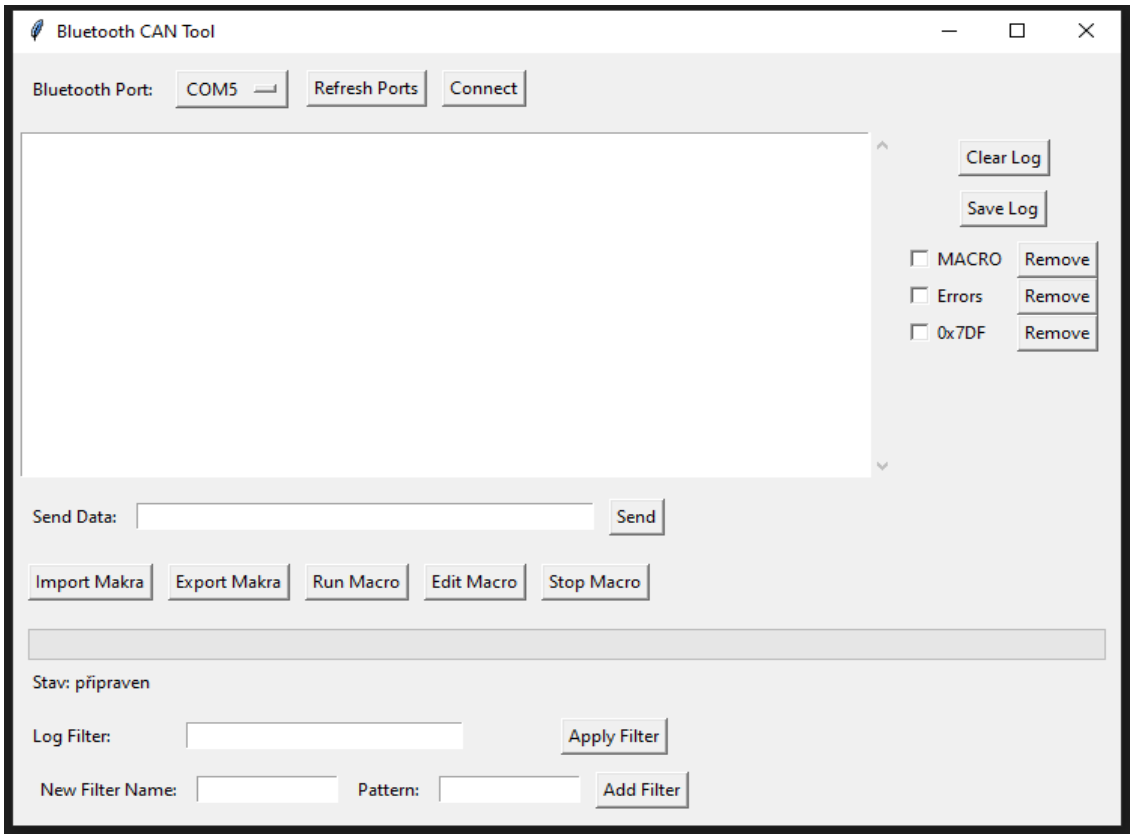
```
        self.tree = ttk.Treeview(self, columns=("send", "expected"),
show="headings", selectmode="browse")

        self.tree.heading("send", text="Odeslat")
        self.tree.heading("expected", text="Očekávaná odpověď")
        self.tree.pack(fill=tk.BOTH, expand=True, padx=10, pady=10)
```

Důležitým prvkem je také zabezpečení komunikace a robustnost aplikace. Chyby během sériové komunikace jsou zachyceny v blocích try/except, což zajišťuje vysokou spolehlivost a minimalizuje riziko pádu aplikace.

Celkový design aplikace klade velký důraz na uživatelskou přívětivost. Automatická instalace závislostí, možnost importu a exportu nastavení makrosekvencí ve formátu JSON a detailní systém filtrování logovacích zpráv přispívají k intuitivnímu ovládání i pro méně zkušené uživatele.

Výsledná aplikace je připravena pro efektivní použití v diagnostice vozidel Škoda přes OBD2 rozhraní, přičemž využívá výhod moderních technologií, jakými jsou například asynchronní zpracování a modulární architektura.



Obrázek 10: První verze aplikace

Zdroj: Vlastní příspěvek

5.4 Realizace – prvotní prototyp

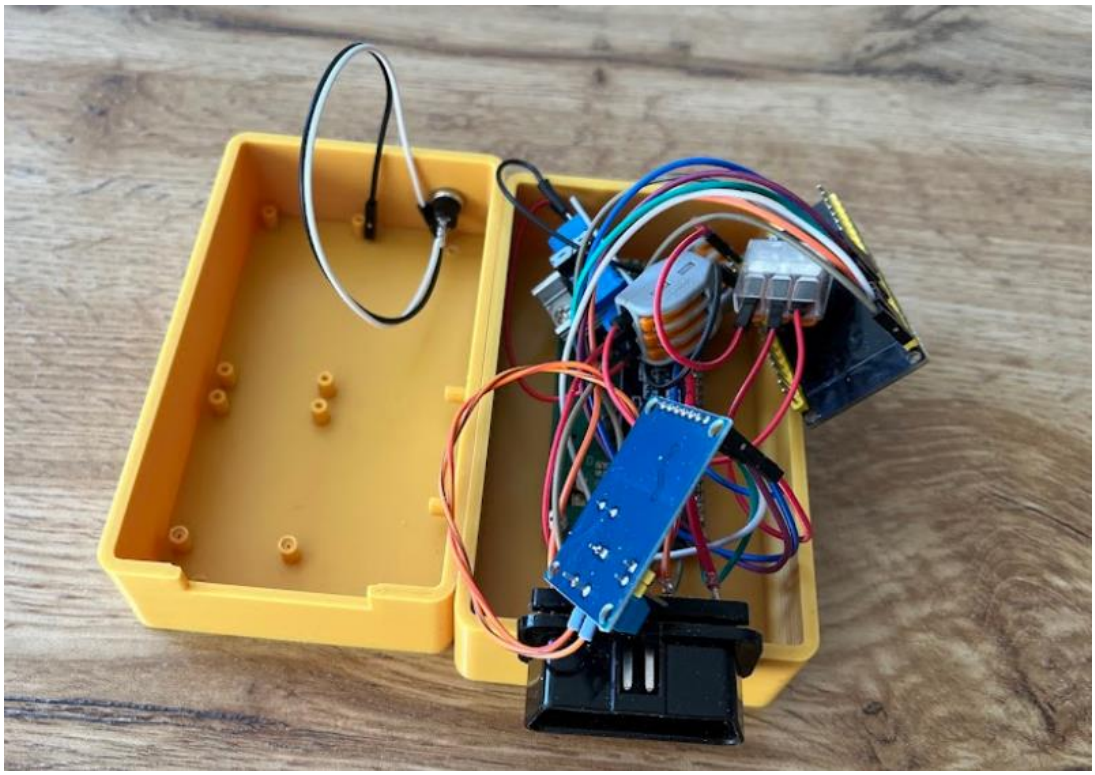
Po zajištění všech potřebných elektronických komponent, dokončení úvodní verze firmwaru a vývojové verze uživatelské aplikace bylo přistoupeno k prvnímu kroku praktické realizace prototypu. Klíčovým úkolem této fáze bylo připravit veškerou nezbytnou kabeláž, která slouží k propojení jednotlivých modulů a rozhraní zařízení za účelem prvotního testování funkcionality.

Byla provedena příprava napájecích vodičů a signálových kabelů, včetně jejich zakončení příslušnými konektory a přiřazení konkrétním pinům dle návrhového schématu. Při tomto procesu došlo také k ručnímu nastavení napájecího zdroje, aby bylo možné řídit napěťové úrovně jednotlivých napájecích větví podle potřeb zapojených komponent. Tato fáze představuje zásadní moment tzv. oživení prototypu, při kterém je poprvé ověřována funkčnost celého systému v reálném zapojení.

Vzhledem k charakteru této testovací etapy nebyl kladen důraz na optimalizaci uspořádání kabeláže (tzv. cable management). Primární cílem bylo ověřit správnou činnost hardwarových i softwarových částí, nikoliv finální mechanickou integraci.

Současně probíhal návrh obalové části zařízení, tedy tzv. ochranné skořepiny, která má za úkol chránit vnitřní elektroniku proti mechanickému poškození a usnadnit manipulaci se zařízením během provozu. K modelování bylo využito 3D CAD softwaru, ve kterém vznikla digitální konstrukce pouzdra s ohledem na rozměrové požadavky, umístění konektorů a vývodů. Po dokončení návrhu byl model exportován do tiskového formátu a následně vytištěn na 3D tiskárně metodou FDM, čímž vznikl první fyzický prototyp mechanické části.

Zásadním milníkem v této fázi vývoje bylo úspěšné navázání obousměrné komunikace s řídicí jednotkou vozidla (ECU) prostřednictvím CAN sběrnice. Během testování bylo ověřeno nejen správné vyčítání datových rámců z palubního systému vozidla, ale i schopnost zařízení aktivně odesílat zprávy zpět do sběrnice. Tento výsledek potvrdil správnou implementaci komunikační vrstvy a základní kompatibilitu s automobilovým protokolem, což představuje zásadní krok směrem k plně funkčnímu a robustnímu diagnostickému systému.



Obrázek 11: Prvotní zapojení

Zdroj: Vlastní příspěvek

Během vývoje prvního prototypu se vyskytlo několik konstrukčních a funkčních nedostatků.

- Dodané spojovací prvky neodpovídaly specifikaci uvedené v objednávce.
- Mechanické tolerance otvorů určených pro uchycení modulů byly nedostatečně dimenzovány, což znemožnilo upevnění šroubů do skořepiny.
- Délka některých kabelových svazků překračovala nezbytnou délku, a bude proto nutná jejich úprava pro optimalizaci prostoru a zajištění správného uspořádání
- Testovaný napájecí zdroj vykazoval známky výstupní nestability, což negativně ovlivňovalo spolehlivost celého systému.

V průběhu vývoje a implementace ovládací aplikace byl uplatněn iterativní přístup, jehož cílem bylo dosažení maximální funkčnosti a spolehlivosti v různorodém softwarovém prostředí cílových zařízení. Vzhledem k tomu, že nelze předpokládat jednotný stav systémového prostředí u všech uživatelů (např. rozdíly v operačním systému, dostupnosti knihoven, úrovni přístupových práv), bylo nezbytné klást důraz na maximální nezávislost aplikace na konkrétní konfiguraci hostitelského systému.

Jedním z důležitých kroků testovací fáze byla validace aplikace na tzv. Technologickém počítači, který byl záměrně nakonfigurován s přísnějšími bezpečnostními omezeními a uživatelskými restrikcemi. Tento scénář simuloval reálné provozní prostředí, kde může být spuštění neověřených aplikací omezeno např. politikou IT správy, nedostatečnými uživatelskými právy nebo absencí potřebných runtime komponent.

Aby bylo zajištěno bezproblémové spuštění aplikace i v takto omezených podmínkách, byla do aplikace integrována logika pro kontrolu a správu závislostí. Před samotným spuštěním hlavního modulu aplikace dochází k ověření přítomnosti nezbytných knihoven a komponent. V případě, že některé z nich chybí, aplikace si je automaticky stáhne a nainstaluje, případně provede potřebnou konfiguraci prostředí. Tímto způsobem bylo dosaženo vysoké míry autonomie aplikace bez nutnosti manuálních zásahů nebo administrátorských práv, což je zásadní pro nasazení v běžném uživatelském prostředí.

V rámci této fáze vývoje nebyl kladen důraz na estetiku nebo pokročilý návrh uživatelského rozhraní. Hlavním cílem bylo zajistit robustní a bezchybnou funkčnost jádra aplikace, která umožní spolehlivou komunikaci s diagnostickým zařízením a správné zpracování příkazů a výsledků. Ačkoliv grafické rozhraní aplikace může být v této fázi považováno za minimalistické či funkčně strohé, naplnilo záměr vývoje: aplikace je stabilní, spustitelná i ve specifických podmínkách a plně funkční.

Výsledkem tohoto vývojového a testovacího cyklu je aplikace, která prokázala schopnost adaptace na různé podmínky běhového prostředí a nabízí spolehlivé jádro pro další rozvoj. Prioritizace funkčnosti před vizuálním zpracováním se ukázala jako efektivní přístup v této fázi projektu, s tím, že uživatelská přívětivost bude řešena v následujících vývojových iteracích.

5.5 Realizace – finální řešení

Na základě výsledků testování a podrobného vyhodnocení chování prvotní verze prototypu byla identifikována řada konstrukčních i technických nedostatků. Tyto poznatky posloužily jako cenný základ pro cílené úpravy, které vedly nejen k odstranění konkrétních problémů, ale i k celkovému zvýšení kvality a robustnosti návrhu zařízení.

Mechanické úpravy montážních bodů a zlepšení montovatelnosti (řešení problémů č. 1 a 2)

Nesoulad mezi objednanými a skutečně dodanými šrouby, stejně jako příliš těsné tolerance otvorů pro uchycení modulů, vedly ke komplikacím při mechanické montáži komponent do obalové skořepiny. Tyto problémy byly řešeny revizí montážních bodů ve 3D modelu a následnou úpravou geometrie otvorů v krytu tak, aby odpovídaly reálně dostupným rozměrům spojovacích prvků. Byla také navržena větší tolerance otvorů a zvětšené dosedací plochy pro zajištění lepší kompatibility se standardizovanými šrouby a zvýšení opakovatelnosti montáže při výrobě dalších kusů.

Úprava kabeláže a optimalizace prostorového uspořádání (řešení problému č. 3)

Délky některých kabelových propojek byly při prvním sestavení nadměrné, což vedlo k nepřehlednému uspořádání vnitřního prostoru zařízení a potenciálně zvyšovalo riziko rušení a poškození vodičů. Tento problém byl řešen revidovaným návrhem kabeláže, při kterém došlo ke zkrácení jednotlivých vodičů na optimální délky. Zároveň byla provedena úprava tvaru a uspořádání skořepiny, která nyní umožňuje efektivnější vedení kabelů a přispívá k celkovému zmenšení rozměrů zařízení. Výsledkem je kompaktnější a prostorově efektivnější konstrukce s lepším mechanickým uspořádáním.

Zvýšení napěťové stability napájecí části (řešení problému č. 4)

V průběhu testování bylo zjištěno, že napájecí část vykazuje nestabilitu výstupního napětí, což se negativně projevovalo zejména při změnách zátěže. Tento problém byl analyzován a následně řešen přidáním stabilizačních keramických a elektrolytických kondenzátorů přímo na výstupní piny step-down měniče. Jejich úkolem je filtrovat vysokofrekvenční šum a zajišťovat stabilní napětí i při dynamických změnách zatěžovacích podmínek. Tato úprava vedla k výraznému snížení výkyvů napětí a zvýšení spolehlivosti napájení citlivých komponent, což se příznivě promítlo do celkové stability celého zařízení.

Obrázek 12: Finální verze zařízení
Zdroj: Vlastní foto

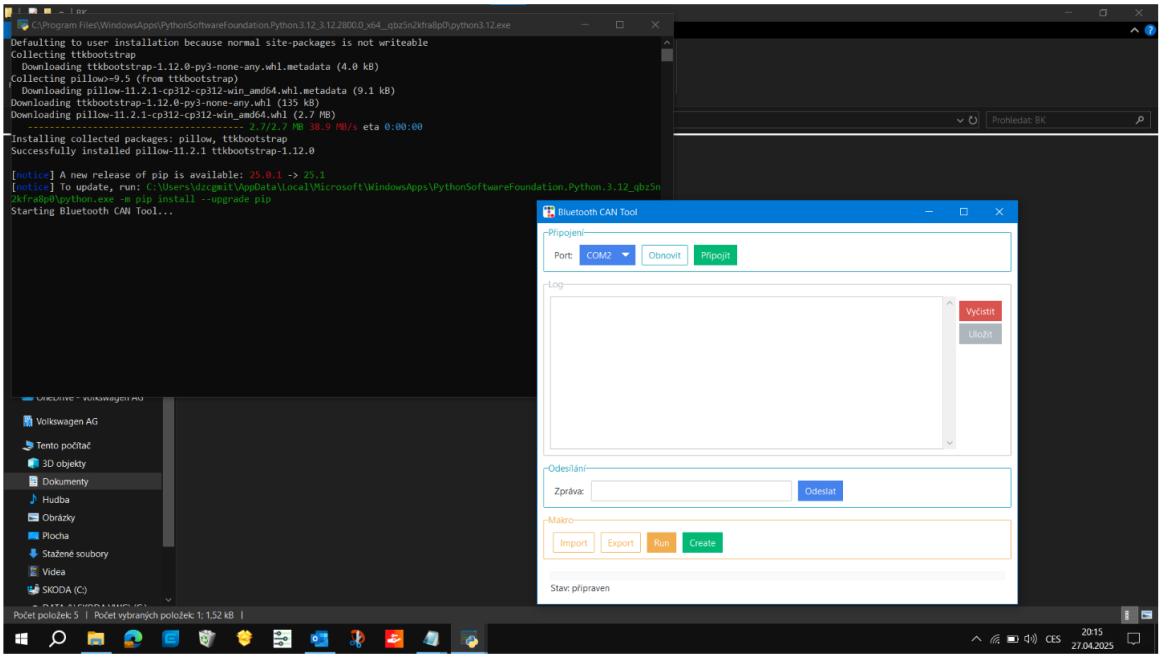
Změna v postupu spouštění aplikace – launcher

Linker představuje důležitý prvek celkového řešení, neboť zásadním způsobem zjednodušuje a zpřehledňuje práci s aplikací. Před načtením hlavního modulu provádí automatickou kontrolu všech potřebných knihoven (například *ttkbootstrap* nebo *pyserial*) a v případě jejich absence zajistí jejich instalaci. Proces může probíhat i přes definovaný proxy server, čímž se eliminuje potřeba ručního zadávání instalačních příkazů prostřednictvím *pip*. Tento přístup, implementovaný v rámci launcheru, je přehlednější než přímé zahrnutí správy závislostí do hlavního aplikačního kódu, kde by docházelo k nežádoucímu prolínání technické a funkční logiky.

Mezi další klíčové výhody patří centralizované zachytávání chyb. V případě selhání importu hlavního modulu nebo spuštění uživatelského rozhraní launcher automaticky zachytí vzniklou výjimku a zobrazí srozumitelné dialogové okno. Tím se zamezí výpisům nečitelných chyb v konzoli, a i méně zkušení uživatelé získají jasnou a přímou informaci o příčině problému, což jim umožní adekvátně reagovat.

Umístěním této logiky do samostatné vrstvy launcheru zůstává aplikační kód čistý, přehledný a zaměřený výhradně na hlavní funkcionalitu. Veškeré podpůrné činnosti –

jako je instalace knihoven, kontrola prostředí a chybová hlášení – jsou soustředěny do jediné, snadno spravovatelné komponenty. Tento přístup významně zvyšuje spolehlivost nasazení, usnadňuje údržbu a přispívá k celkové uživatelské přívětivosti.



Obrázek 13: Ověření funce Launcheru

Zdroj: Vlastní příspěvek

Na předchozím obrázku je zachyceno úspěšné otestování funkce launcheru. Tento mechanismus pomocí kontroly závislostí detekoval chybějící nebo zastaralou knihovnu, provedl její automatickou instalaci a následně připravil aplikaci k běhu. Po dokončení těchto kroků byla aplikace spuštěna, přičemž byl zajištěn její správný provoz. Testování bylo provedeno na technologickém počítači, čímž byl ověřen bezchybný chod aplikace i v prostředí se sníženou úrovní podpory.

Níže je uveden příklad, jakým způsobem launcher zajišťuje kontrolu závislostí:

Zdrojový kód 7: Launcher

```
# --- Ensure required packages are installed ---
def ensure_packages():
    packages = {
        'ttkbootstrap': 'ttkbootstrap',
        'serial': 'pyserial'
    }
    for module_name, package_name in packages.items():
        try:
            importlib.import_module(module_name)
        except ImportError:
            try:
                subprocess.check_call([sys.executable, '-m', 'pip', 'install', '--proxy', 'http://localhost:9001', package_name])
            except Exception as e:
                root = tk.Tk()
                root.withdraw()
                messagebox.showerror(
                    'Installation Error',
                    f"Failed to install package '{package_name}': {e}"
                )
                sys.exit(1)
```

Funkce ensure_packages() zajišťuje dostupnost dvou klíčových rozhraní nezbytných pro běh aplikace – grafických prvků z knihovny *ttkbootstrap* a sériové komunikace prostřednictvím *pyserial*. V úvodu jsou v proměnné packages namapována jména modulů (např. 'ttkbootstrap', 'serial') na odpovídající názvy balíčků používaných při instalaci přes *pip* (např. 'ttkbootstrap', 'pyserial').

Následuje cyklus, ve kterém se každý modul nejprve pokusí importovat prostřednictvím funkce importlib.import_module(). V případě, že import selže (vyvolá výjimku ImportError), je zahájena instalace balíčku pomocí volání subprocess.check_call() s parametry sys.executable -m pip install --proxy http://localhost:9001 <package_name>, přičemž se využívá nakonfigurovaný proxy server. Pokud dojde i při tomto kroku k chybě (např. z důvodu nefunkčního připojení

nebo nesprávného názvu balíčku), funkce vytvoří skryté okno pomocí tk.Tk() a root.withdraw(), v němž je zobrazeno chybové hlášení (messagebox.showerror()), a následně je aplikace ukončena voláním sys.exit(1).

Tímto postupem je zajištěno, že aplikace nebude pokračovat v provádění dalších částí kódu, dokud nejsou ověřeny a doplněny všechny nezbytné externí závislosti.

Změna GUI – graphical User Interface

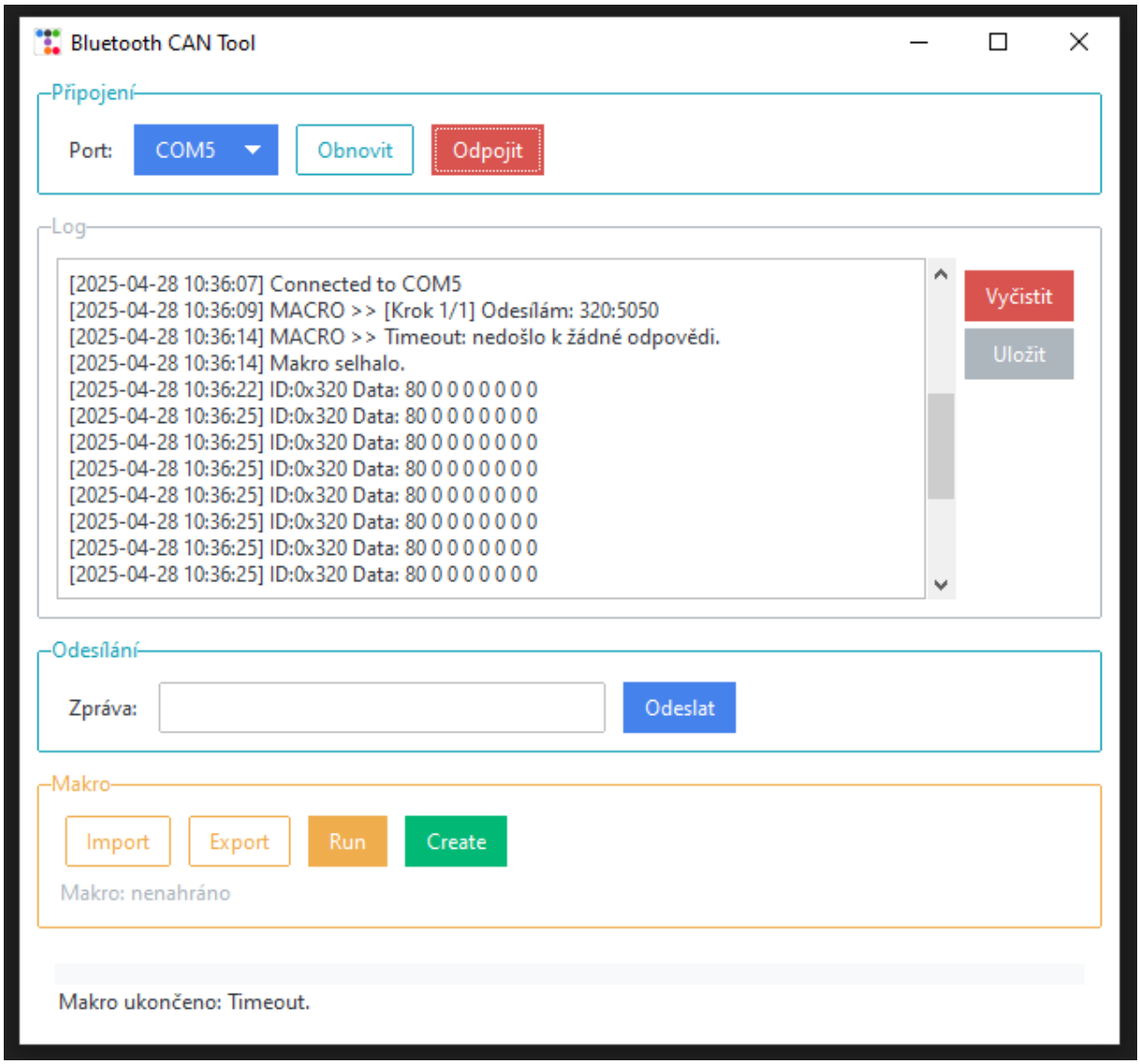
Grafické rozhraní představuje klíčový prvek celé aplikace, jelikož uživatelům poskytuje okamžitou a přehlednou vizualizaci všech stavů a funkcí v rámci jednoho okna. Díky využití knihovny *ttkbootstrap* je uživatelské rozhraní moderní, responzivní a konzistentně stylované napříč platformami. K dispozici jsou světlé i tmavé motivy, které lze měnit bez nutnosti zásahu do zdrojového kódu.

V horní části hlavního okna se nachází panel pro výběr sériového portu a připojení k donglu. Ve spodní části je umístěn přehledný monitor provozu (log) a stavové hlášky. Logovací panel běží ve vlastní komponentě *ScrolledText*, která díky automatickému posouvání zajišťuje zobrazení nejnovějších zpráv. Vedle tohoto panelu jsou tlačítka pro vymazání a uložení logu, což umožňuje snadné zálohování nebo reset výpisu bez nutnosti procházet menu.

Editor makra je dostupný v samostatném okně a využívá stromovou tabulku pro snadné přidávání, úpravu či export posloupností příkazů. Barevný indikátor informuje o stavu načteného makra a umožňuje rychlou orientaci v počtu připravených kroků. Běh makra je doprovázen dynamickým indikátorem průběhu a popiskem aktuálně prováděného kroku, čímž je zajištěna průběžná informovanost o procesu.

Díky oddělení uživatelské logiky do samostatných metod a tříd (*separation of concerns*) je zdrojový kód přehledný, dobře udržovatelný a snadno testovatelný. Asynchronní smyčka, která běží na pozadí v rámci makro enginu, neblokuje vykreslování grafického rozhraní, což přispívá k plynulému a příjemnému uživatelskému zážitku – bez ohledu na počet spuštěných kroků či složitost scénáře.

Celkově grafické rozhraní přispívá ke zvýšení efektivity práce, minimalizaci chyb způsobených ručním zadáváním příkazů a vytváří profesionální a stabilní dojem, který je zároveň přístupný i méně zkušeným uživatelům.



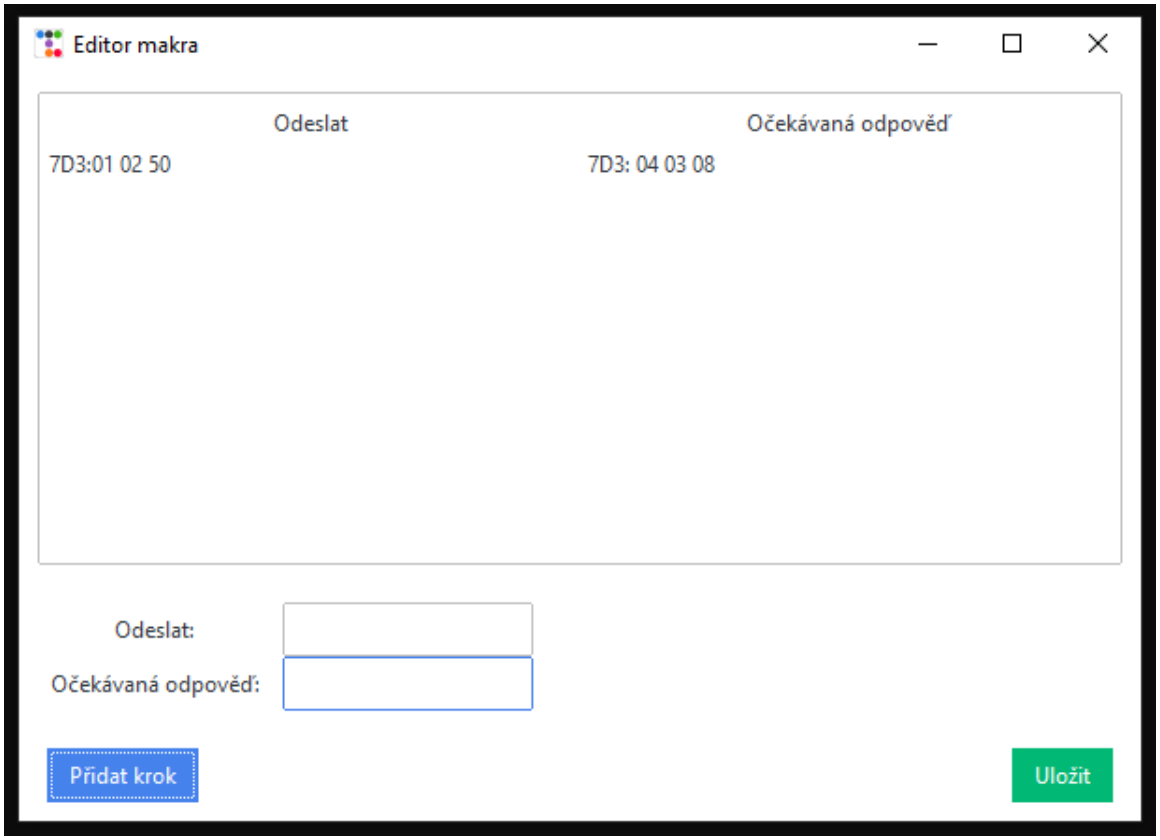
Obrázek 14: Nové GUI

Zdroj: Vlastní příspěvek

Tento obrázek znázorňuje nové uživatelské prostředí, které bylo upraveno za účelem zvýšení komfortu při práci s nástrojem. V dialogovém okně je zobrazeno úspěšné připojení ke komunikačnímu portu. Pokus o spuštění makra však selhal, jelikož zařízení bylo připojeno pouze k testovacímu stavu, a nikoliv přímo k vozidlu. Naopak je viditelný příjem simulovaných zpráv z adresy 0x320 s nákladem 80; zbývající byty jsou pro zjednodušené čtení reprezentovány nulou.

Realizovaná vylepšení přispěla nejen k odstranění identifikovaných nedostatků, ale také k celkovému zefektivnění návrhu z hlediska mechanického, elektrického i konstrukčního. Iterativní přístup, založený na analýze reálného chování zařízení, potvrdil význam prototypování a testování v raných fázích vývoje. Úspěšně provedené úpravy vytvořily pevný základ pro další pokračování vývoje směrem

k finální verzi produktu. V neposlední řadě je potřeba ukázat například i editor na makra, které je možné naprogramovat a následně tlačítek proces se vykoná.



Obrázek 15: Editor maker

Zdroj: Vlastní foto

Závěr

Ve své bakalářské práci jsem navrhl a implementoval přenosné diagnostické zařízení DoCAN, které se připojuje do OBD-II portu vozů od roku 2005. Cíle práce – od návrhu hardwarové architektury přes vývoj firmware až po mobilní aplikaci pro uživatelské rozhraní – byly nejen splněny. DoCAN zařízení spolehlivě komunikuje s řídicími jednotkami vozů, dokáže zasílat vlastní HEX příkazy a řídící aplikace poskytuje přehledné a responzivní prostředí pro diagnostiku i testování.

Během vývoje byla identifikována omezení v oblasti bezpečnostních pojistek proti odesílání nevalidních zpráv do vozidla. Toto rozhodnutí bylo záměrné, aby bylo možné simulovat celou škálu diagnostických i servisních scénářů, avšak vyžaduje, aby bylo zařízení používáno pouze odborníky se zkušenostmi v CAN a OBD protokolech. V praxi tak DoCAN významně urychluje proces testování flashování a ladění řídicích jednotek, zároveň však klade důraz na zodpovědné zacházení.

Realizace tohoto projektu prohloubila znalosti protokolů OBD-II a CAN, potvrdila nadčasovost těchto technologií a ukázala, že jejich robustnost a jednoduchost zajišťují spolehlivost i ve vysoce výkonných aplikacích. DoCAN představuje modulární platformu, kterou lze snadno rozšířit o další funkce – například pokročilou analýzu jízdních dat, vzdálenou aktualizaci firmware či integraci s cloudovými službami pro dlouhodobé sledování výkonu vozidla. Věřím, že navržené řešení může posloužit jako základ pro komerční diagnostické nástroje i pro další akademický výzkum v oblasti automobilové elektroniky.

Zdroje

ANON, 2025. OBD-II PIDs. Online. In: *Wikipedia: the free encyclopedia*. Dostupné z: https://en.wikipedia.org/wiki/OBD-II_PIDs. [citováno 2025-05-13].

ARDUINO, 2025. Arduino IDE 2.x. Program. Online. In: *Arduino*. Dostupné z: https://github.com/arduino/arduino-ide. [citováno 2025-02-14].

BOSH, Robert, 1991. CAN Specification. Version 2.0. PDF; Online. In: *CAN in Automation*. Dostupné z: http://esd.cs.ucr.edu/webres/can20.pdf. [citováno 2025-01-23].

BOSH, Robert, 2023. CAN FD (flexible data rate). Online. In: CAN in *Automation*. Dostupné z: https://www.can-cia.org/can-knowledge/can-fd-the-basic-idea. [citováno 2025-02-12].

CANALYS, 2025. Qualcomm's latest Snapdragon Elite automotive SoCs and strategic focus. Online. In: *Canalys*. Dostupné z: https://www.canalys.com/insights/qualcomm-snapdragon-elite-automotive-socs. [citováno 2025-03-06].

ESPRESSIF SYSTEMS, 2019. ESP32-WROOM-32. Online. In: *Laskakit.cz*. Dostupné z: https://www.laskakit.cz/user/related_files/esp32-wroom-32_datasheet_en.pdf. [citováno 2025-03-04].

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2021. ISO 15765-4:2021, Road vehicles — Diagnostic communication over Controller Area Network (DoCAN). Online. In: *ISO*. Dostupné z: https://www.iso.org/standard/78384.html. [citováno 2025-03-04].

MICROSOFT, 2025. Visual Studio Code. Online. Program. Microsoft. In: *Visual Studio Code*. Dostupné z: https://code.visualstudio.com/. Systémové požadavky: Windows. [citováno 2025-03-01].

NEMONET, 2025. C/C++ Programming for Beginners Basic to Advance. Online. In: *GitHub*. Dostupné z: https://github.com/The-Young-Programer/C-CPP-Programming. [citováno 2025-02-18].

NXP, 2021. I2C-bus specification and user manual. Online. In: *NXP Semiconductors*. Archivní kopie dostupná z: Internet Archive (distributor), https://web.archive.org/web/20221006073143/http://www.nxp.com/docs/en/user-guide/UM10204.pdf. [archivováno 2022-10-06]. [citováno 2025-03-10].

OBD2 Port – Volkswagen, nedatováno. Obrázek; online. In: *FlexiHub*. Dostupné z: https://www.flexihub.com/. [citováno 2025-03-05].

PHILIPS SEMICONDUCTORS, 1996. I2S bus specification. Online. In: *NXP Semiconductors*. Archivní kopie dostupná z: Internet Archive (distributor), https://web.archive.org/web/20070102004400/http://www.nxp.com/acrobat_download/various/I2SBUS.pdf. [archivováno 2007-01-02]. [citováno 2025-02-22].

PYTHON SOFTWARE FOUNDATION, 2025. Python. Online. *Program*. Verze neuvedena. Python Software Foundation. In: python.org. Dostupné z: https://www.python.org/downloads/. [citováno 2025-01-28].

QUALCOMM, 2024. Snapdragon 820 Automotive Platform. Online. In: *Qualcomm*. Dostupné z: https://www.qualcomm.com/products/automotive/snapdragon-820-automotive-platform. [citováno 2025-02-20].

STOICESCU, Alin, 2018. Getting Started with SPI. Online. In: *Microchip Technology*. Dostupné z: https://ww1.microchip.com/downloads/en/Appnotes/TB3215-Getting-Started-with-SPI-90003215A.pdf. [citováno 2025-03-03].

TEXAS INSTRUMENT, 2021. Universal asynchronous receiver/transmitter (UART). Online. In: *Texas Instruments*. Dostupné z: https://www.ti.com/lit/an/scea065b/scea065b.pdf?ts=1692931646648. [citováno 2025-01-29].

VLK, František, 2006. *Diagnostika motorových vozidel: diagnostické testery, motortestery, brzdové soustavy, geometrie řízení, tlumiče, kontrola podvozku, diagnostické linky*. 1. vyd. Brno: František Vlk. ISBN 978-80-239-7064-7.

[Tabulka 1: OBD2 pinout]. Tabulka; online. 2025b. In: *FlexiHub*. Dostupné z: https://www.flexihub.com/oobd2-pinout/. [citováno 2025-03-05].

[Tabulka 2: OBD-II PIDs]. Tabulka; online. 2025a. In: *Wikipedia: the free encyclopedia*. Dostupné z: https://en.wikipedia.org/wiki/OBD-II_PIDs. [citováno 2025-03-04].

[Tabulka 3: Ukázka PID]. Tabulka; online. 2025b. In: *Wikipedia: the free encyclopedia*. Dostupné z: https://en.wikipedia.org/wiki/OBD-II_PIDs. [citováno 2025-03-04].

Obrázek 2: Topologie sběrnice CAN. Obrázek; online. 2025. In: *Robots for Roboticists*. Dostupné z: https://www.robotsforroboticists.com/can-bus/. [citováno 2025-03-06].

Obrázek 4: ESP32-WROOM-32. Obrázek; online. 2025. In: *hurt.zabezpieczeniapoznan.pl.* Dostupné z: https://hurt.zabezpieczeniapoznan.pl/zestaw-deweloperski-esp32-esp-wroom-32-espressif-p-18631.html. [citováno 2025-03-06].

Obrázek 5: LCD 16x2 SPI. Obrázek; online. 2025. In: *Tecmikro.com*. Dostupné z: https://tecmikro.com/led-lcd/351-lcd-16x2-i2c.html. [citováno 2025-03-06].

Obrázek 6: MCP2515. Obrázek; online. 2025a. In: *Laskakit.cz*. Dostupné z: https://www.laskakit.cz/. [citováno 2025-03-06].

Obrázek 7: LM317. Obrázek; online. 2025b. In: *Laskakit.cz*. Dostupné z: https://www.laskakit.cz/. [citováno 2025-03-06].