FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Sledování stavu vozidla prostřednictvím mobilní aplikace
Student:	Bc. Tomáš Zimmerhakl
Vedoucí:	Ing. Filip Štěpánek
Studijní program:	Informatika
Studijní obor:	Návrh a programování vestavných systémů
Katedra:	Katedra číslicového návrhu
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Moderní automobily disponují OBD-II portem, který zpřístupňuje interní komunikační sběrnici vozidla (CAN). Ten slouží k vyčítání a zápisu (diagnostických) údajů. Cílem práce je navrhnout aplikaci pro mobilní telefon, která bude komunikovat s touto interní sběrnicí.

Práci rozložte na následující kroky.
- Analyzujte protokol OBD-II a seznamte se s fungováním CAN podle standardu ISO 15765-4.
- Nalezněte příkazy podle standardu ISO 15031-5. Využijte metody reverzního inženýrství či běžných metod známých z oblasti útoků na interní komunikaci vozidel (tzv. fuzzing). Zaměřte se na čtení informací o stavu vozidla a na zápis (ovládání) nekritických součástí vozidla.
- Navrhněte aplikaci pro mobilní telefon (operační systém Android), která bude pomocí OBD-II portu komunikovat s vozidlem pomocí příkazů nalezených v předchozím bodě.
- Aplikaci naimplementujte a otestujte na reálném vozidle.
- Výslednou aplikaci porovnejte s existujícími volně dostupnými řešeními.

Seznam odborné literatury

Dodá vedoucí práce.

doc. Ing. Hana Kubátová, CSc.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 21. prosince 2017

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Sledování stavu vozidla prostřednictvím mobilní aplikace

Bc. Tomáš Zimmerhakl

Katedra číslicového návrhu
Vedoucí práce: Ing. Filip Štěpánek

30. dubna 2018

Poděkování

Děkuji společnosti Digiteq Automotive s. r. o. za poskytnutí odborných znalostí a studijních materiálů. Děkuji také za přístup k vybavení a měřící technice a za půjčené náčiní použité při testování a měření. Nejvíce mi při odhalování skryté komunikace ve vozidle pomohli Ing. Slavomír Mikulecký, Radek Šťastný a Ing. Miroslav Čermák. Za milý přístup a ochotu k osobním setkáním při řešení děkuji také Ing. Martinu Dočekalovi, Bc. Luboši Šoganovi, Bc. Petru Pintrovi, Bc. Adéle Kramolišové a Ing. Lucii Hradecké.

Adaptér zapůjčil a včasný start prací umožnil Marek Bureš. Při řešení problémů s Android aplikací poskytl pomoc a cenné rady Ing. Tomáš Krabač. K úspěšné opravě vadného adaptéru přispěla svojí manuální zručností Kateřina Zimmerhaklová. Testovací vozy zapůjčili Ing. Pavel Zimmerhakl a Lenka Zimmerhaklová. S citací odborných pojmů pomohla Marie Zimmerhaklová. Děkuji vám.

Na závěrečné korekci práce spolupracovali Ing. Pavel Zimmerhakl a Bc. Martin Frumar. Oběma děkuji za všechny chyby a nesrovnalosti, které v textu nalezli a přispěli tím k jejich odstranění.

Děkuji vedoucímu práce Ing. Filipu Štěpánkovi za množství věnovaného času, námět, poznámky k práci a odborné rady.

Děkuji svojí rodině za podporu a servis, který mi umožnil věnovat se diplomové práci naplno.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, avšak pouze k nevýdělečným účelům. Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 30. dubna 2018 .....

Odkaz na tuto práci

Zimmerhakl, Tomáš.: *Sledování stavu vozidla prostřednictvím mobilní aplikace.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Tato diplomová práce se zabývá vytvořením mobilní aplikace, která je schopná komunikovat s osobním automobilem přes port OBD-II. Na základě analýzy výhod a nevýhod již existujících aplikací je vypracován návrh vlastní aplikace pro mobilní operační systém Android. Podle návrhu je vytvořena aplikace, která má dva režimy. V základním režimu uživatel sleduje aktuální data z vozidla (např. rychlost, teplota chladící kapaliny, tlak v sacím potrubí), která jsou periodicky obnovována. V pokročilém režimu uživatel odesílá do vozidla libovolné příkazy a čte reakce na ně. Ke zprostředkování komunikace je použit adaptér založený na mikrokontroléru ELM327. S vozidlem komunikuje prostřednictvím diagnostického portu OBD-II a s mobilním telefonem pomocí bezdrátového standardu bluetooth. Součástí práce je výzkum, který se zaobírá možnostmi ovládání nekritických součástí vozidla z mobilní aplikace.

Klíčová slova čtení stavu vozidla, rozhraní pro ovládání vozidla, komunikace jednotek ve vozidle, CAN sběrnice, OBD, UDS, ISO-TP, ELM327, Android aplikace

Abstract

This thesis deals with the development of mobile application capable of communicating with personal vehicle via the OBD-II interface. The design of this application is elaborated based on the analysis of advantages and disadvantages of existing applications. The application supports two modes of operation. In the first (basic) mode the current data on the operation of the vehicle (e.g., speed, coolant temperature, intake manifold pressure, etc.) can be monitored. In the second (advanced) mode commands can be sent to vehicle via terminal. Bluetooth OBD-II adapter based on microchip ELM327 serves as the communication interface between the mobile application and the vehicle. Also, there is included research on the possibilities of controlling non-critical parts of vehicle via mobile application.

Keywords reading state of the vehicle, car control interface, ECU communication in vehicle, CAN bus, OBD, UDS, ISO-TP, ELM327, Android application

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Uvnitř moderních automobilů se dnes nachází rozsáhlá komunikační síť a v rámci ní velké množství jednotek, které spolu komunikují. Moderní systémy vyžadují data z mnoha senzorů, jež se musí dostat během krátké chvíle na správné místo. Po zpracování dat řídící jednotka vyšle příkazy k akčním členům (brzdy, světla, atp.), které na aktuální situaci zareagují. Veškerý tento informační provoz probíhá na sběrnicích, pro něž platí pravidla zajišťující včasné a bezpečné doručení zpráv. Soubor takových pravidel se nazývá protokol a jedním z nejrozšířenějších je CAN.

Znalost protokolu CAN umožňuje vývojářům zpřístupnit mnoho informací z útrob vozu jeho uživatelům. K zobrazení dat slouží displeje na palubní desce, které informují posádku například o stavu paliva, počtu ujetých kilometrů a například i o nejbližší čerpací stanici. Mnohé automobily také samostatně dokážou diagnostikovat závady a zobrazit je řidiči. Pakliže takto vybavené vozidlo není k dispozici, přichází ke slovu mobilní aplikace jakožto užitečný nástroj nahrazující vestavěné zobrazovací zařízení a funkce. Je-li aplikace přímo od výrobce vozu, stačí obvykle propojit telefon s automobilem prostřednictvím bluetooth. Stále však existuje velké množství vozů, pro které aplikace neexistují. Zde vyplňují prostor univerzální aplikace třetích stran využívající faktu, že každé vozidlo v Evropské unii vyrobené po roce 2003 musí být vybaveno diagnostickým portem OBD-II a musí podporovat příkazy podle standardu ISO 15765-4. Ten definuje posílání diagnostických zpráv prostřednictvím protokolu CAN.

Tato diplomová práce se zabývá návrhem a implementací mobilní aplikace pro systém Android, která dokáže číst aktuální data z vozidla a zároveň bude schopná vozidlu odesílat příkazy, na něž vozidlo zareaguje. Aplikaci jsem pojmenoval *OBD Robot*.

Práce se skládá z 5 kapitol – *Analýza*, *Analýza aplikací zobrazujících data o vozidle v reálném čase*, *Návrh*, *Realizace* a *Testování*. První kapitola uvádí, jakým způsobem jsou komunikující zařízení ve vozidle uspořádána a postupně odhaluje detaily komunikace na CAN sběrnici uvnitř vozidla. Kapitola vy-

světluje pojmy sběrnice, protokol, CAN, OBD, TP-ISO a UDS a objasňuje, jak spolu tyto pojmy souvisí. Analýza dekomponuje komunikaci ve vozidle na dílčí vrstvy a detailně se jimi zabývá od nejnižší až po nejvyšší úroveň. To je důležité pro následný návrh aplikace.

Ještě před samotným návrhem je zařazena kapitola *Analýza aplikací zobrazujících data o vozidle v reálném čase*. Jejím cílem je porovnat existující aplikace pro systém Android, které umožňují zobrazování údajů o jízdě a stavu vozidla v reálném čase. Analýza hodnotí aplikace nejen podle množství údajů, které daná aplikace umí zobrazit, ale i podle uživatelské přívětivosti aplikace – tedy způsobu ovládání a zobrazení měřených údajů.

Kapitola *Návrh* definuje na základě provedených analýz požadavky a budoucí strukturu nové aplikace. Zároveň dekomponuje problém na více menších částí. Práce není jen o aplikaci pro Android, ale také o vytvoření funkčního celku, kde je aplikace jen jednou z mnoha potřebných komponent. Důležitou komponentou je vhodný adaptér, který zprostředkovává komunikaci mezi vozidlem a mobilním telefonem. V kapitole návrh je pojednáno nejen o výběru adaptéru, ale také o jeho nastavení a použití. Následuje krátká sekce věnovaná metodám čtení informací z vozidla a závěr kapitoly patří metodám ovládání vozidla. Je představeno více způsobů, jak získávat chráněné informace z vozidla a konkrétní ukázky odhalování skryté komunikace.

Kapitola *Realizace* podrobně popisuje pořízení adaptéru a vytváření mobilní aplikace. Detailně rozebírá problémy, se kterými jsem se při práci potýkal, a jejich řešení. Pochopení problémů usnadňují dokumentační obrázky, nákresy a schémata. Kapitola je vedena chronologicky jako deník, od vytyčení cíle až po jeho splnění.

Závěrečná kapitola testování dokládá funkčnost aplikace *OBD Robot*. Ta byla vyvíjena v souladu s návrhem a otestována na několika vozech značky Škoda. Průběh i výsledky testů dokládají nejen textové záznamy komunikace, ale především bohatá obrazová dokumentace. Součástí práce je také manuál k aplikaci *OBD Robot* v příloze D.

Kapitola 1

Analýza

Kapitola analýza uvádí, která zařízení jsou v moderních vozech schopná komunikace, jakým způsobem jsou zařízení ve vozidle uspořádána a postupně odhaluje, jak probíhá komunikace uvnitř vozidla prostřednictvím CAN sběrnice. Kapitola vysvětluje pojmy sběrnice a protokol, zaměřuje se především na CAN, OBD, TP-ISO a UDS a objasňuje, jak spolu tyto pojmy souvisí. Analýza dekomponuje komunikaci ve vozidle na dílčí vrstvy a detailně se jimi zabývá od fyzické až po aplikační úroveň. To je důležité pro pochopení komunikace uvnitř vozidla a následný návrh aplikace, která bude schopná tuto komunikaci nejen číst, ale také se na ní aktivně podílet.

1.1 Komunikace elektronických jednotek ve vozidle

Moderní automobily dnes obsahují velké množství elektronických zařízení, jejichž úkolem už dávno není jen zajištění správného pohonu vozidla. Ve vozidle je mnoho různých systémů, které se starají o pohodlí řidiče i posádky. Rádio, klimatizace, stěrače, tempomat, desítky asistenčních systémů od hlídání mrtvého úhlu přes udržování vozidla v jízdním pruhu až po zcela automatické parkování a to dokonce i s přívěsem[1] [1]. Každá z těchto vymožeností potřebuje pro svoji korektní funkci senzory, řídící jednotku a aktuátory[2]. Jak je vidět na obrázku 1.1[3], v moderním automobilu se mohou nacházet desítky řídících jednotek, stovky senzorů a stovky metrů komunikačních sběrnic. To vše tvoří velmi komplikovanou strukturu, která zůstává očím řidiče skryta.

[1] Od roku 2014 jsou některá vozidla koncernu Volkswagen vybavena příplatkovou funkcí *Trailer Assist*. Je-li auto vybaveno touto funkcí, dokáže samo zacouvat s připojeným přívěsem do zatáčky. Ovládání volantu je plně v režii vozidla, řidič řídí otočným knoflíkem, který plní funkci joysticku.

[2] Aktuátor je opakem senzoru. Převádí digitální pokyny na mechanickou činnost. Příkladem je elektromotorek ovládající stěrače.

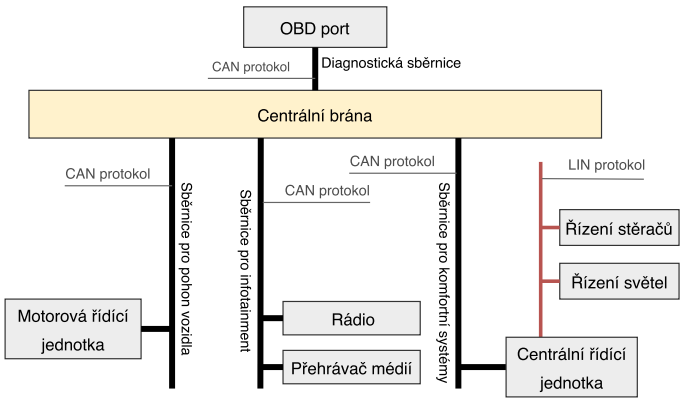
[3] Obrázek 1.1 je převzatý z [2].

Obrázek 1.1: Ilustrační schéma vedení komunikačních kanálů, senzorů a řídících jednotek ve vozidle Audi A8 2018

Aby byl do této struktury vnesen řád, stala se páteří veškeré komunikace takzvaná *sběrnice*. Sběrnici je možné přirovnat k železnici, po které jezdí vlaky. Každý vlak představuje zprávu, každé město na trati představuje řídící jednotku a každá zastávka je nějaký senzor nebo aktuátor. Stejně jako v reálném světě není možné, aby nejrychlejší vlaky obsluhovaly i ty nejmenší stanice, pak i ve světě sběrnic funguje hierarchické uspořádání. Jednotky v automobilu jsou pomocí sběrnic seskupeny do oblastí podle svého významu. Sběrnice se pak dělí na větší a významnější a menší a méně významné. Ústředním bodem, kde se všechny sběrnice potkávají, je centrální brána *(gateway)* [3]. Tu si lze představit jako pražské hlavní nádraží, které odbavuje jak malé regionální vlaky, tak mezinárodní expresy.

Hierarchické uspořádání a fyzické oddělení sběrnic má mnoho praktických důvodů. Představte si, že by byly všechny zprávy uvnitř automobilu na jediné sběrnici a v jednu chvíli by se potkal příkaz k brždění se zprávou o staženém okénku. Je zjevné, že stahování okének, aktuální hlasitost rádia nebo aktuální teplota klimatizace prostě nemohou mít takovou prioritu jako brždění, a právě proto jsou sběrnice oddělené. Tímto zapojením se snadno předchází situacím, kdy řídící jednotky čtou informace, které jsou pro ně zcela nerelevantní. Přesto i tak mohou nastat kolize mezi zprávami stejného významu. Ty se řeší pomocí priorit zpráv, o čemž detailně hovořím v podkapitole 1.4.

Samotná sběrnice jako kus drátu ovšem ke komunikaci nestačí. Stejně jako se železniční doprava řídí pravidly, aby se vlaky nesrazily, a aby byli všichni cestující spokojeni, má i komunikace na sběrnici svůj soubor pravidel, kterému se říká *protokol*. Protokol přesně udává, jaká je struktura zprávy vyslané na sběrnici, jak rychle je možné zprávy na sběrnici posílat nebo kolik vodičů a jakým způsobem bude ke komunikaci využito. Mezi protokoly patří například CAN (ISO 15765) [4] nebo KWP (ISO 14230) [5].



Obrázek 1.2: Propojení komunikačních jednotek ve vozidle

Centrální brána je styčným bodem mezi sběrnicemi a přeposílá jen ty informace, u nichž to má smysl. Celý systém je obvykle hierarchicky uspořádán, tedy pod centrální bránou mohou být zapojené další brány a až k nim mohou být připojeny sběrnice s jednotkami. Na menších sběrnicích je pak zcela běžné použití jiných protokolů než na těch větších za účelem zjednodušení komunikace a úspory peněz. Konkrétně ve vozidlech Škoda Auto se mezi obsluhou stěračů a centrální řídící jednotkou používá komunikační protokol LIN (levnější, jednodušší), na globální úrovni se mezi řídícími jednotkami používá protokol CAN (dražší, složitější) [3]. Schéma zapojení komunikačních jednotek ve vozidle je vidět na obrázku 1.2.

1.2 Způsoby propojení vozidla a mobilního telefonu

Mým cílem je navodit komunikaci mezi chytrým mobilním telefonem a automobilem. Je tedy nejprve nutné si ujasnit, jak tyto dva světy propojit. Možností je více a každá z nich má svá úskalí. Stručný přehled těchto možností popisuje tabulka 1.1.

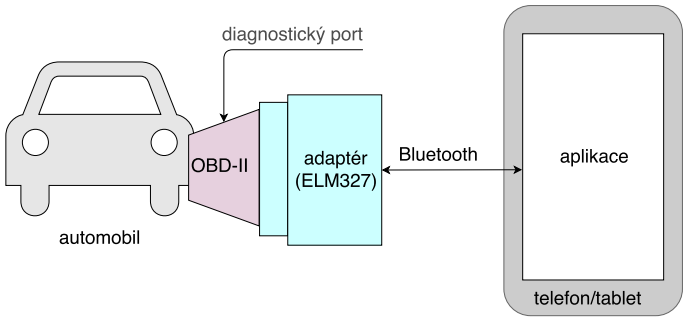
Mojí snahou bylo použít přístup, který je co nejméně invazivní a propojení nebude vyžadovat demontáž dílů vozidla. To tedy vyloučilo přímé připojení na sběrnici. Dále jsem chtěl využít způsob aplikovatelný na co největší množství automobilů. Proto nebylo vhodné jako komunikační kanál zvolit internet, ke kterému je zatím připojeno mizivé množství automobilů. Rovněž USB port je prozatím rozšířen pouze v nových vozidlech, a proto ani ten jsem pro komunikaci s vozidlem nepoužil. Z přehledu v tabulce 1.1 zbývá už jen diagnostický port a ten se stal místem, z něhož jsem čerpal interní informace z vozidla. Schéma propojení je vidět na obrázku 1.3.

Tabulka 1.1: Způsoby propojení vozidla a mobilního telefonu

Způsob propojení	*Poznámka*
diagnostický port ve vozidle	Diagnostický port je v každém vozidle a pro připojení je nutné být fyzicky přítomen v automobilu. Na trhu existuje mnoho adaptérů, které je možné k portu připojit. Nevýhodou může být nedostatek dat na diagnostické sběrnici. Připojení k portu je rychlé a lze ho snadno opakovat v různých vozidlech.
přímé připojení na sběrnici	Vyžaduje odstranění krytů a izolace na správném místě ve vozidle, rozpoznání sběrnice a odhalení vodičů. Následně je možné se připojit k adaptéru prostřednictvím elektrikářských krokosvorek. Výhodou je možnost dostat se téměř k jakékoliv interní komunikaci, bez dokumentace ke konkrétnímu vozidlu je však velmi obtížné a zdlouhavé nalézt správné místo, kde se připojit. Také při neopatrné manipulaci hrozí poškození elektrického vedení ve vozidle.
připojení přes USB port	U vozidel vybavených tímto portem je možné propojit chytrý telefon a automobil prostřednictvím standardního USB kabelu. Výhodou je přítomnost tohoto náčiní v každé domácnosti, problémem však může být nepřítomnost USB portu ve starších vozidlech.
připojení přes internet	Nejmodernější vozidla jsou připojena k internetu prostřednictvím datové SIM karty ve vozidle. K takto vybavených vozidlům je možné se vzdáleně připojit pomocí TCP/IP protokolu a terminálové služby telnet, což předvedli například Charlie Miller a Chris Valasek na vozidle Jeep Cherokee [6]. Je-li komunikační zařízení připojeno k internetu, může se bezdrátově připojit k vozidlu odkudkoliv na světě. Aktuálně je ale k internetu připojeno jen mizivé procento vozidel.

1.3 Diagnostický systém OBD-II

Od roku 2001 musí být podle [7] všechna motorová vozidla s benzínovým motorem prodávaná v Evropské unii vybavena diagnostickým systémem OBD-II *(On-board diagnostics)*. Stejné nařízení se týká také vozidel s dieselovým motorem, pro něž je závazné od 1.1.2003. Totožným systémem musí být vybaveny také všechny automobily prodávané ve Spojených státech amerických. Díky tomu dnes existuje jednotná diagnostika závad a měření emisí, která umožňuje snadnou testovatelnost každého auta v libovolném servisu. Kromě diagnostic-



Obrázek 1.3: Propojení vozidla a mobilního telefonu

kých informací definuje protokol OBD-II také metody, jak získat aktuální data o jízdě (rychlost, otáčky motoru, apod.) a data o vozidle (například VIN[4]).

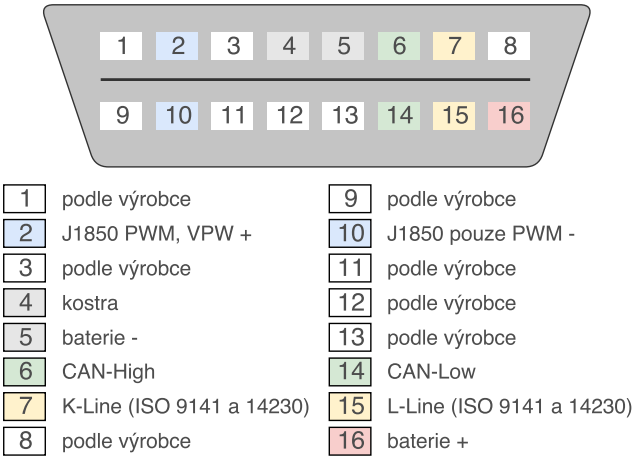
Systém OBD-II se skládá ze 2 klíčových částí:

- diagnostický port – konektor J1962
- 5 protokolů určených ke komunikaci

Vozidlo, které je vybaveno diagnostickým portem OBD-II, má tento port zpravidla ukrytý ze spodní strany přístrojové desky u řidiče přibližně na úrovni jeho kolen. Obvykle je vnořen uvnitř přístrojového panelu, takže není na první pohled viditelný. Někdy také bývá skrytý za plastovým krytem, který se musí

[4]VIN (Vehicle identification number) je kód složený ze 17 znaků (čísla, písmena), který jednoznačně identifikuje každé motorové vozidlo.

Obrázek 1.4: Pozice OBD-II konektoru ve vozidle Škoda Karoq 2018



Obrázek 1.5: OBD-II port samice a význam jeho pinů

nejprve odklopit. Pokud není port vidět, doporučuji posvítit si baterkou do prostoru pedálů a důkladně prozkoumat spodní stranu panelu, na které je připevněn volant. Typická pozice OBD-II konektoru je vidět na obrázku 1.4, nicméně přesná poloha není normou nijak specifikována a každý výrobce si může port umístit podle vlastní volby. Konektor má tvar, který je vyobrazen na obrázku 1.5 a má růžovou barvu.

Obrázek 1.5 znázorňuje vnitřní strukturu OBD-II diagnostického portu (konektor J1962, samice). Skládá se ze 16 pinů, jejichž význam je též objasněn. Pokud najdete ve vozidle tento konektor, s jistotou v něm budou aktivní piny 4, 5 a 16. Z ostatních budou aktivní pouze piny odpovídající používanému komunikačnímu protokolu v prozkoumávaném vozidle.

Každé vozidlo vybavené systémem OBD-II musí komunikovat s diagnostickým zařízením prostřednictvím jednoho z následujících protokolů. Není nutná podpora všech, stačí jeden z nich.

- SAE J1850 PWM
- SAE J1850 VPW
- ISO 9141-2 K-Line
- ISO 14230-4 KWP
- ISO 15765-4 CAN

Popisy protokolů jsou uvedeny níže, informace jsem čerpal především z [8]. Dnes nejrozšířenější protokol CAN(podle standardu ISO 15765-4) v tomto seznamu záměrně vynechávám, protože se jím detailně zabývám v sekci 1.4.

SAE J1850

Jedná se o starší a levnější alternativu k CAN protokolu. Nalezneme ho v některých automobilech General Motors a Chrysler a dělí se na 2 typy

(PWM a VPW). PWM (Pulse width modulation) používá piny 2 a 10, logická jednička je představována napětím 5 V a logická nula napětím 0 V. Protokol VPW (Variable pulse width) používá pouze jeden vodič, který je vyveden v ODB-II konektoru na pin 2. Logická jednička je představována napětím 7 V, logická nula stejně jako u PWM napětím 0 V. Tento protokol používá časově závislé signálování, což znamená, že bit je považován za 1/0 pouze pokud zůstává nízké/vysoké napětí neměnné po určitý počet taktů.

Keyword protocol (KWP) ISO 14230
Protokol KWP (také znám pod označením KWP200) je v OBD-II konektoru na pozici 7 a používá se především v amerických vozidlech vyrobených po roce 2003. Zprávy mohou obsahovat až 255 bytů. Protokol má dvě varianty, které se liší inicializací.

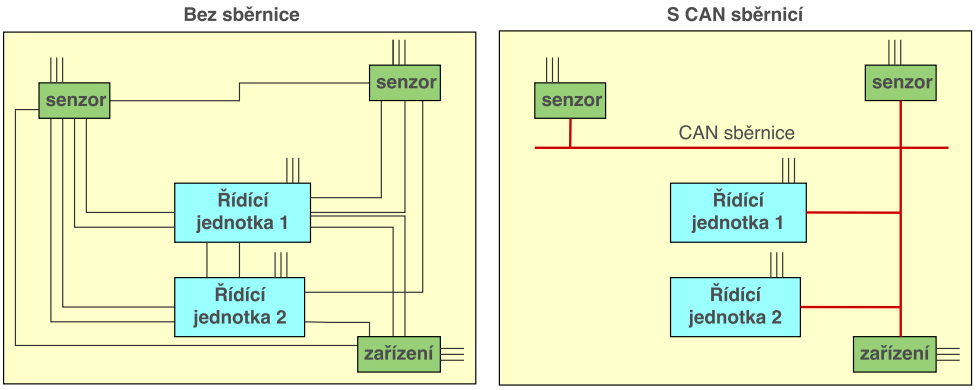
ISO 9141-2 K-Line
Tento protokol je odvozen od KWP a používá se především v evropských autech. V OBD-II konektoru obývá pin 7 a volitelně může použít také pin 15. Pakety se skládají z priority, zdrojové adresy, cílové adresy, dat (maximálně 7 B) a CRC[5].

Poslední protokol z rodiny OBD-II je CAN podle standardu ISO 15765-4. Ten je povinně podporován všemi vozidly prodávanými v USA po roce 2008. Využívá se hojně například ve vozidlech koncernu Volkswagen a já se na něj zaměřím v sekci 1.4. Standard ISO 15765-4 definuje posílání diagnostických zpráv prostřednictvím protokolu CAN, nikoliv samotný protokol. CAN sběrnice se skládá ze dvou vodičů *CAN high* a *CAN low*, které jsou v OBD-II konektoru připojeny na piny 6 a 14. Detailní výklad k vodičům CAN sběrnice se nachází v sekci 1.4.2.

1.4 CAN sběrnice a CAN protokol

CAN *(Controller Area Network)* je sériový komunikační protokol vyvinutý firmou Bosch v roce 1986. Původně byl protokol navržen pro nasazení v automobilech, avšak díky nízké ceně, spolehlivosti, poměrně vysoké přenosové rychlosti a snadné rozšiřitelnosti je tento protokol stále více využíván také v jiných průmyslových aplikacích [9]. Dnes je možné tento protokol nalézt také v tramvajích, vlacích nebo metru. Uplatnění nalezl protokol i v letectví. Výrobci zdravotnického vybavení používají CAN uvnitř svých výrobků a v některých nemocnicích je možné nalézt CAN v moderních operačních sálech, kde zprostředkovává komunikaci mezi centrálním řízením a jednotlivými

[5]CRC (Cyclic redundancy check) je krátký kód vypočítaný z dat, která tímto zabezpečuje. Před přenosem se vypočítá a přidá na konec přenášených dat. Po přenosu se vypočítá znovu a porovná s obdrženým CRC. Jsou-li oba shodné, víme, že data nebyla přenosem nijak změněna.



Obrázek 1.6: Srovnání systému bez CAN sběrnice (nalevo) a systému s CAN sběrnicí (napravo)

komponentami v sále jako například světly, polohovatelnými stoly, rentgeny, apod. [10].

Pojem CAN *(Controller Area Network)* zapouzdřuje mnoho vrstev síťového modelu – od fyzické sběrnice (tedy vodiče, po kterých se přenášejí data) až po protokol (soubor pravidel pro přenos dat). Jedno bez druhého nemůže fungovat, je však nutné mít na paměti rozdíl mezi CAN sběrnicí a CAN protokolem. Protokol CAN je definovaný normou ISO 11898. Ta byla později rozdělena na 3 menší části 11898-1, 11898-2 a 11898-3 [11], [12], [13]. Dokumenty obsahují jak specifikaci komunikačního protokolu, tak požadavky na fyzickou a linkovou vrstvu[6].

1.4.1 Vlastnosti CAN sběrnice

Sběrnice poskytuje všem jednotkám ve vozidle jednotné a levné rozhraní, které umožňuje komunikaci každého s každým. Výhodnou je, že řídící jednotce postačuje přítomnost CAN rozhraní a nemusí disponovat mnoha různými digitálními a analogovými vstupy, aby mohla komunikovat s každým zařízením ve vozidle. Díky použití CAN sběrnice dochází k dramatickému snížení množství použitých vodičů. To nejen zlevňuje datovou infrastrukturu ve vozidle, ale navíc to napomáhá ke zvýšení spolehlivosti. Čím méně vedení ve vozidle je, tím méně vodičů se může porouchat a tím méně chyb může vzniknout. Rozdíl mezi systémem s a bez CAN sběrnice ilustruje obrázek 1.6.

V následujících bodech shrnuji vlastnosti CAN. Mluvím-li o uzlu, myslím tím jakékoliv zařízení, které je na sběrnici připojené. V automobilovém

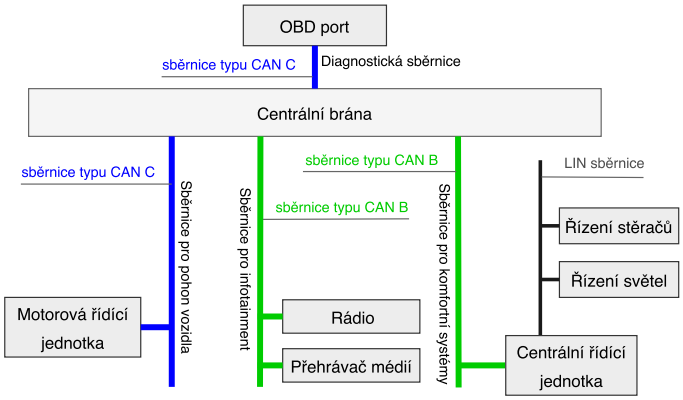
[6] Jedná se o dvě nejnižší vrstvy ISO/OSI modelu. Fyzická vrstva definuje elektrické vlastnosti rozhraní a určuje, jaké napětí je považováno za logickou jedničku a jaké za logickou nulu. Umožňuje přenos bitů kanálem a obsahuje A/D a D/A převodníky.
Linková vrstva zajišťuje detekci a případnou korekci chyb, formátuje data do rámců a zajišťuje přístup k lince[14].

kontextu tedy přichází v úvahu buď řídící jednotka, senzor nebo aktuátor. Vycházím přitom především z [8], [9] a [10].

- Sběrnice je sériová, což znamená, že data jsou přenášena nikoliv paralelně přes více kanálů, ale po jediném kanálu a to za sebou v pořadí, v jakém byly ze zdroje odeslány.
- Sběrnice je typu *multi-master*, což znamená, že na sběrnici neexistuje centrální autorita, ale každý z uzlů pracuje nezávisle na ostatních. Každý uzel může řídit chování jiného uzlu. V případě poruchy jednoho uzlu může sběrnice fungovat dále.
- Na sběrnici je možné pozorovat komunikaci mezi dvěma uzly pomocí zpráv. Signalizace chyb a pozastavení komunikace jsou indikovány pomocí dvou speciálních zpráv.
- CAN zprávy se šíří metodou *broadcast*, což znamená, že každá zpráva je viditelná všem uzlům na sběrnici a každý uzel může zprávu přijmout a reagovat na ni. Z toho vyplývá, že každé zařízení, které chce operovat na CAN sběrnici musí obsahovat mikroprocesor.
- Díky výše zmíněným vlastnostem je velice snadné a levné sběrnici rozšířit o další uzly. Jednoduše se ke sběrnici připojí, žádná další konfigurace není třeba.
- Každá zpráva na sběrnici má svoji prioritu. Pokud se dva uzly pokusí odvysílat dvě zprávy ve stejný čas, odvysílá se jen zpráva s vyšší prioritou. Druhá zpráva se odloží a uzel ji odvysílá později.
- CAN protokol obsahuje kontrolu chyb pomocí CRC kódu. Uzel při příjmu zprávu zkontroluje a je-li poškozená, ignoruje ji. Dále je odeslána chybová zpráva, po jejímž příjmu si uzel, který vyslal poškozenou zprávu, zvýší počet chyb o jedna. Při překročení určité kvóty může uzel přestat zprávy vysílat.

1.4.2 Fyzická vrstva

Standard definuje tři typy CAN sběrnice, které se liší počtem vodičů, maximální rychlostí, úrovněmi napětí, způsobem vysílání a samozřejmě také využitím. Uplatnění následujících typů sběrnic ilustruje obrázek 1.7.



Obrázek 1.7: Využití vysokorychlostní a nízkorychlostní CAN sběrnice ve vozidle

High speed CAN

Vysokorychlostní CAN je nejrychlejším a nejpoužívanějším typem sběrnice. Komunikační rychlost se pohybuje mezi 500 – 1000 kb/s. Sběrnice se skládá ze dvou vodičů a někdy bývá také označována jako CAN C nebo ISO 11898-2 [12]. Pro svoji rychlost je využívána tam, kde je zásadní rychlost přenášené informace – tedy na sběrnici vyhrazené pro pohon vozidla, kde se vyskytují informace od brzd, motoru nebo z emisního systému.

Low speed CAN

Nízkorychlostní CAN je rovněž sestavena ze dvou drátů, ale dosahuje rychlostí pouze do 125 kb/s. Vžily se pro ní také názvy CAN B nebo ISO 11898-3 a používá se typicky tam, kde není rychlost přenosu dat důležitá [13]. V tomto režimu fungují sběrnice pro infotainment[7] i sběrnice pro komfortní systémy, na nichž se vyskytují zprávy užitečné pro rádio, navigaci nebo telefon. Nízkorychlostní CAN nabízí vyšší odolnost proti poruchám než vysokorychlostní, a proto se používá například tam, kde dráty procházejí skrz kloub do dveří a kde je vysoká míra namáhání spoje.

[7] Infotainment je systém umístěný v přední centrální části automobilu mezi sedadly řidiče a spolujezdce, který se skládá z velkého displeje a slotů pro externí zařízení. Jeho účelem je umožnit posádce vozu ovládat z jednoho místa většinu komfortních funkcí vozu a zároveň dostávat informace o aktuální jízdě, stavu a nastavení vozidla.

Single wire CAN

Jednodrátová CAN komunikuje maximální rychlostí 33 kb/s a nepoužívá se tak často, jako 2 výše uvedené typy. Příkladem využití může být sběrnice, přes kterou probíhá nastavování sedadel nebo zrcátek. Pro tento typ se rovněž používají názvy CAN A, SAE-J2411 a GMLAN.

Vzhledem k nízkému využití typu CAN A hovoří následující odstavce o *Low speed CAN* a *High speed CAN*. Kvůli vysoké rychlosti a odolnosti proti rušení se skládá sběrnice ze dvou drátů, které jsou izolované a vzájemně zakroucené. Jeden z drátů se nazývá *CAN high* a druhý *CAN low*, což je vidět na obrázku 1.8. Oba vodiče jsou na každém konci vysokorychlostní CAN sběrnice propojeny přes odpor o velikosti 120 ohmů. Pokud experimentátor narazí v autě na kroucenou dvoulinku a pomocí multimetru naměří mezi vodiči odpor o velikosti 60 ohmů[8], může si být téměř jistý, že objevil CAN sběrnici. U nízkorychlostní sběrnice se ukončovací rezistory nenachází mezi dvěma vodiči sběrnice, ale na každém uzlu. Hodnota odporů se počítá pro každý systém zvlášť pomocí vzorečků, které přesahují rámec této práce. Zájemci se mohou detaily dočíst v [15].

Obrázek 1.8: Kroucená dvojlinka CAN sběrnice

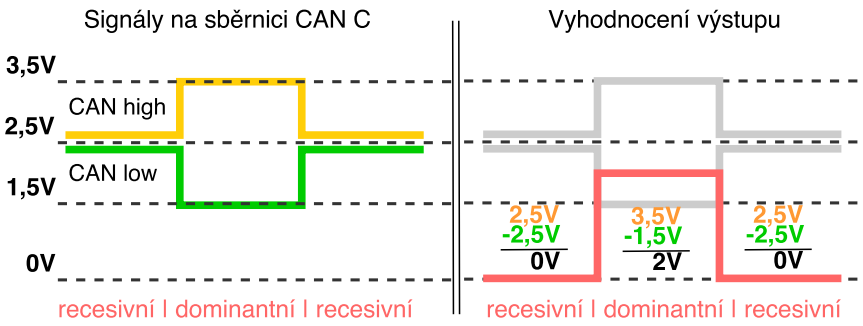
Na každém vodiči může být jedna ze dvou komplementárních[9] hodnot – *dominantní* nebo *recesivní*. Dominantní hodnota má v digitálním světě význam logické nuly a recesivní hodnota má význam logické jedničky.

Na drátech probíhá přenos dat prostřednictvím takzvané diferenciální signalizace (*differential signaling*). To znamená, že na obou vodičích je ten samý elektrický signál, ale na jednom vodiči má opačnou polaritu. Probíhá-li vysílání, obvody přijímače nevyhodnocují napětí na vodičích vůči zemi, ale napětí vůči sobě navzájem. Přijímač odečte hodnotu napětí na *CAN low* od hodnoty napětí na *CAN high* a podle výsledku určí, zda je právě přenášená hodnota recesivní nebo dominantní.

V případě sběrnice typu CAN C (tedy té nejrychlejší) je na obou vodičích v klidovém stavu napětí 2,5 V, což zároveň představuje recesivní hodnotu. Při přenosu dominantní hodnoty musí napětí na drátě *CAN high* stoupnout na 3,5 V a napětí na *CAN low* musí klesnout na 1,5 V. Přijímač vyhodnocuje napětí na drátech tak, že odečítá menší od většího a je-li rozdíl nulový, interpretuje hodnotu jako recesivní (logická jednička). Pokud je rozdíl napětí

[8] Z Ohmova zákona plyne, že mám-li mezi dvěma uzly dva rezistory o odporech R_1 a R_2, mohu je nahradit jedním o odporu $1/(\frac{1}{R_1} + \frac{1}{R_2})$.

[9] komplementární = vzájemně opačný

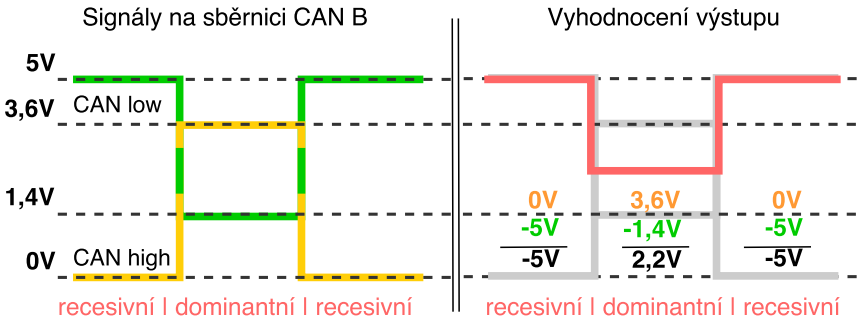


Obrázek 1.9: Význam napěťových hladin a vyhodnocení výstupu na sběrnici typu High speed CAN

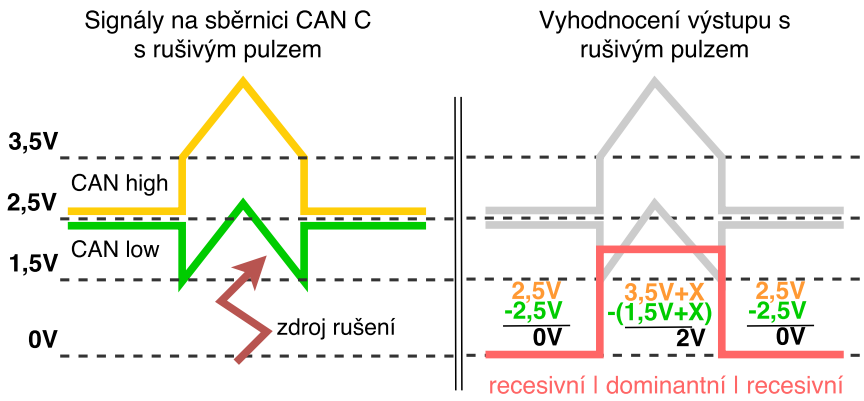
2 V, interpretuje příjemce hodnotu jako dominantní (logická nula). Celý proces vyhodnocování napětí na kroucené dvojlince sběrnice typu CAN C je na obrázku 1.9.

Sběrnice typu CAN B (druhá nejrychlejší) se chová velmi odlišně. V klidovém stavu (recesivní hodnota) je na drátě *CAN high* napětí 0 V, na *CAN low* je napětí 5 V. V okamžiku vysílání dominantní hodnoty stoupá napětí na *CAN high* na 3,6 V, ve stejnou chvíli je na druhém drátě napětí pouze 1,4 V. Vyhodnocování na straně přijímače probíhá obdobně jako v případě sběrnice typu CAN C. Rozdíl mezi napětími okolo -5 V znamená recesivní hodnotu, rozdíl 2,2 V znamená dominantní hodnotu [16]. I zde je pro lepší porozumění vše vyobrazeno na obrázku 1.10.

Vodiče CAN sběrnice jsou vedeny motorovým prostorem, kde mohou přijít do kontaktu s různými druhy rušení. Zkrat, přetížení nebo statický výboj mohou vytvořit na vedení CAN sběrnice krátký pulz, který zcela změní hodnoty napětí na jejích vodičích. Avšak díky diferenciálnímu signálování je sběrnice vůči těmto vlivům velice odolná. Na příkladu sběrnice typu CAN C ukážu, jak



Obrázek 1.10: Význam napěťových hladin a vyhodnocení výstupu na sběrnici typu Low speed CAN



Obrázek 1.11: Vyhodnocení výstupu na sběrnici typu High speed CAN během rušení

se sběrnice vyrovná s náhlým pulzem. Na obrázku 1.11 je v levé části vidět náhlý pulz, který změnil napěťovou hladinu obou drátů CAN sběrnice během vysílání dominantní hodnoty. Při vyhodnocování v momentě rušení odečítá přijímač jednu napěťovou hodnotu od druhé. Je vidět, že i přes odlišné vstupní hodnoty vyšel správný výsledek odpovídající dominantní hodnotě – tedy 2V. Jsou-li oba vodiče rušeny stejnou měrou, princip diferenciálního signálování pulzy zcela odfiltruje.

Počet připojených zařízení ke sběrnici není normou nijak limitován a ani technicky není problém připojit libovolný počet zařízení. V praxi ale existují důvody, proč se to nedělá a maximální počet připojených uzlů k jedné sběrnici se omezuje na 64 [9]. Jednak se zvyšujícím se počtem uzlů na sběrnici klesá přenosová rychlost a roste riziko kolizí. Za druhé s rostoucím množstvím připojených zařízení na sběrnici roste délka sběrnice, což je nežádoucí. Původně byla CAN sběrnice určena jen pro malé vzdálenosti v automobilech, kde jen stěží překoná délku 40 m. Od této hranice její přenosová rychlost prudce klesá a při délce 1200 m činí přibližně 70 kb/s (zatímco teoretická maximální je 1 Mb/s) [9].

1.4.3 Typy CAN zpráv a jejich struktura

Nyní se vrátím k analogii s železniční přepravou z kapitoly 1.1. Uvedl jsem, že si lze zprávu běžící po sběrnici představit jako vlak. Na první pohled vypadají zvenku všechny vlaky stejně, mají lokomotivu a standardizované vagóny, jen cílová stanice a lidé uvnitř se mění. S CAN zprávami je to stejné – norma ISO 11898 předepisuje, jak má každá zpráva vypadat [12]. To, co se mění, jsou už jen samotná přenášená data.

Původní verze normy z roku 1986 (dnes nazývaná *CAN 2.0A*) zavedla *standardní* formát zprávy. V roce 1991 vydala firma Bosch novou specifikaci,

která přidává takzvaný *rozšířený* formát zprávy [9], [17]. Systém, který používá rozšířený formát zprávy se označuje *CAN 2.0B*. Jediný zásadní rozdíl mezi oběma formáty je v délce identifikátoru zprávy. Specifikaci 2.0A je možné chápat jako elektrické vlaky a specifikaci 2.0B jako dieselové vlaky. S výjimkou lokomotivy se v podstatě neliší a oba systémy mohou spolu fungovat na stejných kolejích. To o společném fungování obou systémů platí i pro CAN 2.0A a 2.0B, pokud to jednotky na sběrnici podporují.

Norma uvádí pro zprávu v originálním anglickém znění pojem *frame*. Ten se používá hojně mezi techniky i v češtině, já ale pro lepší porozumění textu dále používám slovo *zpráva*. Bez ohledu na formát (standardní/rozšířený) existují tyto 4 typy zpráv:

Datová zpráva *(data frame)*
: Datové zprávy jsou to hlavní, jejich množství na sběrnici je nejvyšší a je možné si je představit jako vlaky pro osobní přepravu. Maximální kapacita takové zprávy je 8 bytů.

Žádost o data *(remote frame)*
: Žádost o data je velmi „okleštěná“ zpráva a její význam je jasný už z názvu. Zpráva neobsahuje žádný prostor pro data. Lze ji přirovnat k samotné lokomotivě.

Zpráva o chybě *(error frame)*
: Tato zpráva slouží k signalizování chyb na sběrnici. Jakmile nějaký uzel na sběrnici detekuje poškozenou zprávu, vytvoří zprávu o chybě a vyšle ji na sběrnici. Ani tato zpráva nepřenáší žádná data, jedná se spíše o servisní akci, na železnici bych ji přirovnal k drezíně.

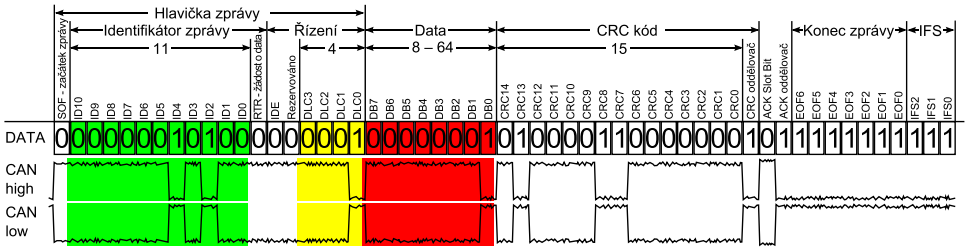
Zpráva o přetížení *(overload frame)*
: Tuto zprávu generují zařízení, která už nejsou schopná zpracovávat další požadavky. Vysláním této zprávy se zařízení snaží oddálit přísun dalších datových zpráv, které jsou na ně zacíleny. I tato zpráva by se na železnici dala přirovnat k drezíně, protože ovlivňuje řízení toku dat na sběrnici.

V následujících odstavcích popíšu strukturu datové zprávy, vycházím především ze zdrojů [8], [9], [10] a [17]. Jak standardní tak rozšířený formát datové zprávy se skládají z částí, jejichž zkratky a význam je uveden v tabulce 1.2.

Datová zpráva se skládá ze tří hlavních částí – z hlavičky, dat a CRC kódu *(Cyclic redundancy check)*. Hlavička se skládá především z identifikátoru zprávy a DLC kódu *(Data length code)* a na obrázku 1.12 je vyznačena zleva nejprve zelenou, pak bílou a žlutou barvou. Datová část je na obrázku 1.12 vyznačena červenou barvou. Celá zpráva ve standardním formátu je se všemi svými částmi přehledně vyobrazena na obrázku 1.12, kde je ve spodní části vidět i napětí na vodičích *CAN high* a *CAN low*. Zatímco v případě standardního formátu CAN 2.0A je identifikátor zprávy dlouhý 11 bitů, ve specifikaci 2.0B je délka identifikátoru 29 bitů.

Tabulka 1.2: Části datové CAN zprávy

Anglická zkratka	*pojmenování*	*počet bitů*	*význam*
SOF	začátek zprávy (Start of frame)	1	Dominantní hodnota indikuje začátek vysílání zprávy.
ID	identifikátor (identifier)	11/29	Identifikátor zprávy jednoznačně určuje zařízení, které zprávu odeslalo. Slouží k řízení přístupu na sběrnici v případě kolizí. Pro standardní formát je identifikátor 11 bitů dlouhý, rozšířený formát zprávy má identifikátor dlouhý 29 bitů.
SRR	(substitute remote request)	1	Vyskytuje se jen v rozšířeném formátu zprávy a má vždy recesivní hodnotu.
RTR	žádost o data (remote request)	1	Pokud je dominantní, jedná se o datovou zprávu, v opačném případě se jedná o zprávu typu *žádost o data*.
IDE	rozšíření identifikátoru (identifier extension)	1	Dominantní hodnota indikuje standardní formát, recesivní hodnota indikuje rozšířený formát zprávy.
DLC	délka dat (data length code)	4	Tyto 4 bity určují délku přenášených dat, která je v rozmezí 0 – 8 bytů.
CRC	zabezpečovací kód (cyclic redundancy check)	15	Kontrolní kód se vypočítá podle určitých pravidel z dat a je umístěn na konec zprávy. Jednotka, která přijme zprávu, vypočítá kód znovu a porovná ho s přijatým kódem. Pokud jsou obě hodnoty totožné, je ověřeno, že zpráva nebyla při přenosu porušena (zachování integrity dat).
ACK	potvrzení (acknowledgment)	2	Odesílatel zprávy nastaví první bit na recesivní hodnotu. Pokud přijímač obdrží platnou zprávu (zkontrolovanou pomocí CRC), odešle ji znovu s dominantní hodnotou prvního ACK bitu. Tímto způsobem se odesílatel zprávy dozví, že byla zpráva doručena. Druhý bit má roli oddělovače a jeho hodnota je vždy recesivní.
EOF	konec zprávy (end of frame)	7	Konec zprávy se signalizuje vždy sedmi za sebou jdoucími bity s recesivní hodnotou.
IFS	mezera mezi zprávami (interframe space)	3	Mezi 2 zprávami je mezera o délce 3 bity, všechny mají recesivní hodnotu.

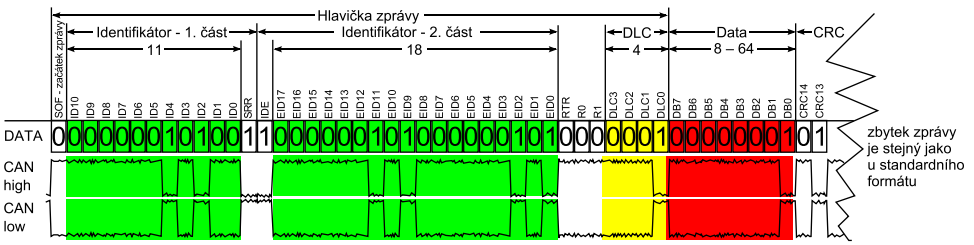


Obrázek 1.12: Struktura standardní datové zprávy na sběrnici High speed CAN

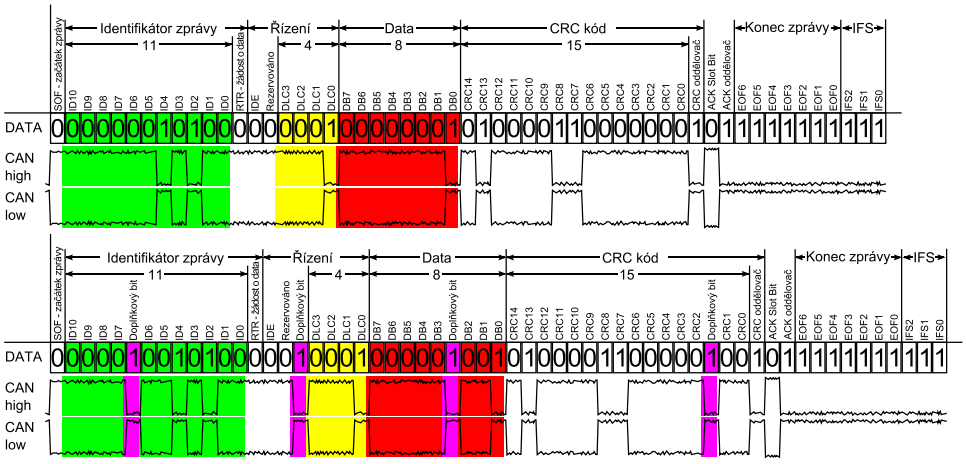
Struktura rozšířené zprávy je trochu pozměněna a je vidět na obrázku 1.13. Identifikátor je kvůli zpětné kompatibilitě rozdělen na 11 a 18 bitů. Ihned za jedenáctibitovou částí identifikátoru následuje bit SRR *(Substitute remote request)*, který má vždy recesivní hodnotu. Díky tomu při vzájemné kolizi zpráv s 11 a 29bitovým identifikátorem dostane přednost formát s 11bitovým identifikátorem. Dále následuje IDE bit nastaven na recesivní hodnotu, poté 18bitová část identifikátoru a dále již následují stejné položky jako u standardního formátu.

Nyní už je zřejmé, z čeho se zprávy skládají a jak jsou části zprávy za sebou uspořádány. Prakticky to ale přesně takto nefunguje. Kdyby si člověk zkusil spočítat délku nějaké odposlechnuté zprávy přímo na sběrnici, zjistil by, že zpráva je ve skutečnosti o dost delší. Do každé zprávy se vkládají takzvané „doplňkové bity“ (anglicky se proces nazývá *bit stuffing*) [9]. Proces vkládání doplňových bitů do CAN zprávy je velmi prostý. Uzel vysílá zprávu na sběrnici a přitom počítá, kolik za sebou stejných bitů právě vyslal. Pokud vyslal 5 stejných bitů za sebou, vloží za ně jeden bit opačné hodnoty. Poté pokračuje vysíláním zbývajících bitů. Takto se chová uzel během odesílání celé zprávy, výjimku pochopitelně tvoří závěrečná sekvence EOF[10]a IFS[10] skládající se ze 7 a 3 za sebou jdoucích recesivních bitů bez přerušení. Na obrázku 1.14 je v horní části zpráva bez doplňkových bitů a dole je ta samá zpráva s doplňkovými bity označenými fialovou barvou.

[10]viz tabulku 1.2: Části datové CAN zprávy



Obrázek 1.13: Struktura rozšířené datové zprávy na sběrnici High speed CAN



Obrázek 1.14: Vkládání doplňkových bitů do CAN zprávy

Důvody pro používání doplňkových bitů jsou dva. Zaprvé časté střídání hodnot je dobré pro udržení synchronního vysílání. Zadruhé to zvyšuje spolehlivý přenos a velmi to usnadňuje detekci chyb. S výjimkou výše uvedených případů je 6 za sebou jdoucích stejných bitů považováno za chybu.

1.4.4 Posílání zpráv a řízení priorit

Zprávy na sběrnici odesílají a přijímají uzly. Jak jsem již uváděl dříve, uzel je jakékoliv zařízení, které je schopné se připojit a komunikovat na CAN sběrnici. Může to být řídící jednotka, senzor, sdružený přístrojový panel a další podobná zařízení. V železničním modelu by uzlem byly nádraží a zastávky. Každý uzel na sběrnici se mimo jiné skládá z procesoru (mikrokontroléru), jehož součástí je CAN řadič schopný přijímat nebo odesílat CAN zprávy. Tento řadič obsahuje rozhraní pro příjem – `Rx` a rozhraní pro vysílání – `Tx`. Obě tato rozhraní jsou připojená ke CAN sběrnici. Některé uzly umí vysílat i přijímat zprávy souběžně, norma to ale nevyžaduje.

Každý uzel na svém `Rx` portu monitoruje, co se na sběrnici právě děje. Pokud je na sběrnici dostatečně dlouhou dobu recesivní hodnota, uzel ví, že sběrnice je volná. Kdyby právě probíhalo vysílání zprávy, musela by kvůli principu doplňkových bitů nejpozději po 5 recesivních hodnotách přijít alespoň jedna dominantní a potom by uzel věděl, že sběrnice volná není.

Pokud chce uzel vyslat zprávu, počká až bude sběrnice volná. Jakmile se tak stane, započne uzel vysílat na svém `Tx` portu. Nejprve vyšle startovací dominantní bit a pak bit po bitu vysílá identifikátor zprávy přesně podle obrázku 1.14. Během vysílání uzel zároveň poslouchá na `Rx` portu, jestli jsou vysílané hodnoty skutečně přítomny na sběrnici. Pokud přítomny jsou, uzel pokračuje ve vysílání. Může ovšem nastat situace, kdy uzel právě vysílá na `Tx`

portu hodnotu recesivní, ale na `Rx` portu je přesto v tu samou chvíli hodnota dominantní. To znamená, že došlo ke kolizi a dominantní hodnotu na sběrnici vyslal nějaký jiný uzel. V tabulce 1.3 je vidět, která z hodnot na sběrnici přetrvá, dojde-li ke kolizi. Jak už název napovídá, potkají-li se ve stejnou chvíli recesivní a dominantní hodnota, vždy „zvítězí" ta dominantní. Z toho vyplývá, že fyzické přenosové médium musí realizovat funkci logického součinu. Jestliže zaměníte slovo „dominantní" za logickou nulu a slovo „recesivní" za logickou jedničku, dostanete pravdivostní tabulku konjunkce[11].

Tabulka 1.3: Určení priorit při kolizi dvou hodnot na sběrnici

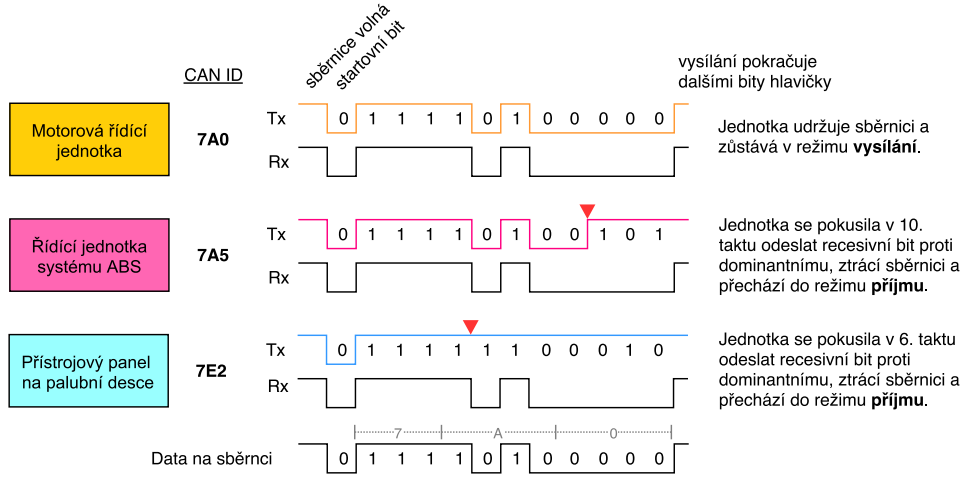
vstup 1	vstup 2	vítězná hodnota na sběrnici
dominantní	dominantní	dominantní
dominantní	recesivní	dominantní
recesivní	dominantní	dominantní
recesivní	recesivní	recesivní

Jakmile uzel zjistí, že na sběrnici je jiná hodnota, než kterou tam vyslal, musí se vzdát sběrnice, přestat vysílat a počkat, než bude sběrnice znovu volná. Teprve pak může svoje vysílání zopakovat. Pozornému čtenáři jistě neušlo, že to první, co se na sběrnici ze zprávy objeví, je identifikátor. Ten rozhoduje o tom, která ze zpráv bude mít v případě kolize přednost. A protože dominantní hodnota představuje v digitálním světě nulu, budou mít zprávy s nižšími hodnotami identifikátorů vyšší prioritu před těmi s vyššími hodnotami identifikátorů.

Nyní na konkrétním příkladu ukážu, jak funguje arbitráž[12] sběrnice. Sledujte přitom obrázek 1.15. Na stejné sběrnici jsou umístěny motorová řídící jednotka, řídící jednotka systému ABS a přístrojový panel na palubní desce, jejichž CAN identifikátory jsou po řadě `7A0`, `7A5` a `7E2`. Všechny tři jednotky vyšlou startovací bit a první čtyři bity svých identifikátorů. Dochází sice ke kolizi, ale protože všechny tři jednotky vysílají stejné hodnoty, žádná z nich o tom zatím neví. V šestém taktu vysílá přístrojový panel na palubní desce na `Tx` portu recesivní hodnotu, ale na `Rx` portu je přítomna dominantní hodnota (viz 6. bit na posledním řádku na obrázku 1.15). Přístrojový panel tedy zjistil, že na sběrnici vysílá ještě někdo jiný s vyšší prioritou, a proto se musí sběrnice ihned vzdát a přechází do režimu příjmu. Ostatní dvě jednotky zůstávají v režimu vysílání ještě 4 takty. V 10. taktu odesílá řídící jednotka systému ABS recesivní hodnotu, ale na sběrnici je v tu chvíli přítomna dominantní hodnota.

[11] Konjunkce je matematicko-logická operace dvou a více binárních proměnných. Výsledkem konjunkce je hodnota `1` pouze v případě, že všechny složky mají také hodnotu `1` [18].

[12] Arbitráž (rozhodčí řízení) je proces při kterém se řeší konflikt mezi dvěma nebo více zainteresovanými stranami [19]. Zde se jedná o konflikt jednotek, které chtějí ve stejnou chvíli vysílat zprávu na sběrnici.



Obrázek 1.15: Proces získání CAN sběrnice při souběžném vysílání více jednotek

Proto se i ona musí vzdát sběrnice, přestat vysílat a přejít do režimu příjmu. Motorová řídící jednotka pokračuje ve vysílání až do konce, protože všechny ostatní jednotky už přestaly vysílat.

1.5 Transportní protokol na CAN sběrnici

V kapitole 1.4 jsem se zabýval fyzickou a linkovou vrstvou, které zajišťují fyzický transport a zabezpečení dat. Nyní se budu naopak zabývat dalšími dvěma vrstvami síťového modelu. Mám data, jejichž délka je 10 bytů a chci je poslat nějaké jednotce na CAN sběrnici. Jak to provést? Tento problém řeší transportní vrstva. Ta nezkoumá, k čemu data jsou a jaký je jejich význam, prostě je vezme a podle určitých pravidel je přenese přes CAN sběrnici tak, aby si je příjemce mohl poskládat tak, jak byla původně odeslána. Transportní protokol ovlivňuje pouze to, co se posílá v červeném sektoru „DATA“ na obrázku 1.12 v sekci 1.4.3.

Pro posílání dat existuje několik protokolů[13], já se zde zabývám pouze protokolem ISO-TP, který používají vozy koncernu Volkswagen [20]. Na těchto vozech jsem svoji aplikaci testoval a protože jsem potřeboval rozumět komunikaci na CAN sběrnici, musel jsem pochopit fungování protokolu ISO-TP.

[13]Například CANopen protokol, VW TP1.6, VW TP2.0. Další protokoly doporučuji vyhledat v [8].

1.5.1 Protokol ISO-TP

Protokol ISO-TP je definován normou ISO 15765-2 [21] a rozšiřuje limit pro posílání zpráv přes CAN sběrnici z 8 na 4096 bytů. Využívá k tomu zřetězení zpráv a přidává do zpráv takzvané *PCI byty* (Protocol Control Information). Tato metadata[14] jsou 1, 2 nebo 3 byty dlouhá a jejich význam objasňuje tabulka 1.4 [21], [8]. Umístění PCI bytů v CAN zprávě demaskuje obrázek 1.16.

Jak je patrné z tabulky 1.4, ISO-TP definuje 4 typy rámců na CAN sběrnici. Připomínám, že *frame* znamená *rámec* a jedná se jednu CAN zprávu – tedy maximálně 8 bytů dat plus hlavička a další povinné náležitosti detailně vysvětlené v sekci 1.4.3. Typy rámců podle ISO-TP jsou tyto:

SF Single frame
: Jednoduchý rámec s daty, který může přenášet maximálně 7 bytů dat.

FF First frame
: První datový rámec zprávy, která je delší než 7 bytů a tudíž musí být rozdělena na více rámců.

CF Consecutive frame
: Druhý a každý další datový rámec zprávy, která je delší než 7 bytů.

FC Flow control frame
: Nejedná se o datový rámec. Používá ho příjemce zprávy pro řízení datového toku na straně odesílatele. Je to krátká zpráva pro odesílatele o tom, že příjemce je nebo není připraven zprávu přijmout.

Abych ozřejmil, jak se PCI byty používají, uvedu příklad. Mám zprávu `01123456789ABCDEF0` a chci ji odeslat po CAN sběrnici. Zpráva má 9 bytů a proto se v žádném případě nevejde do jednoho CAN rámce. Ten dokáže přenést maximálně 8 bytů dat a uvážíme-li, že protokol ISO-TP z nich ještě spotřebuje minimálně 1 byte, zbývá 7 na data. Musím tedy zprávu rozdělit do 2 rámců. První rámec bude typu `FF`, proto první dva PCI byty budou `10 09`. Následuje prvních 6 bytů dat, která chci odeslat. Ukazuje to obrázek 1.16 ve své horní polovině. Zbylé 3 byty dat umístím do druhého rámce. Jeho první byte je `21`. První 4 byty mají hodnotu `0010`, za nimi následuje číslo segmentu, jehož hodnota je v tomto případě `0001`. Následují zbylé 3 byty zprávy. Druhý rámec je vidět v dolní polovině obrázku 1.16.

1.6 Aplikační protokol na CAN sběrnici

Úkolem aplikačního protokolu je definovat správnou posloupnost datových bytů a interpretovat jejich význam. Tedy mám syrová data a zajímá mě jejich význam. Jedním z aplikačních protokolů je UDS (Unified Diagnostic Services)

[14] Metadata jsou data o datech.

Tabulka 1.4: Čtyři typy rámců podle ISO-TP a jejich PCI byty

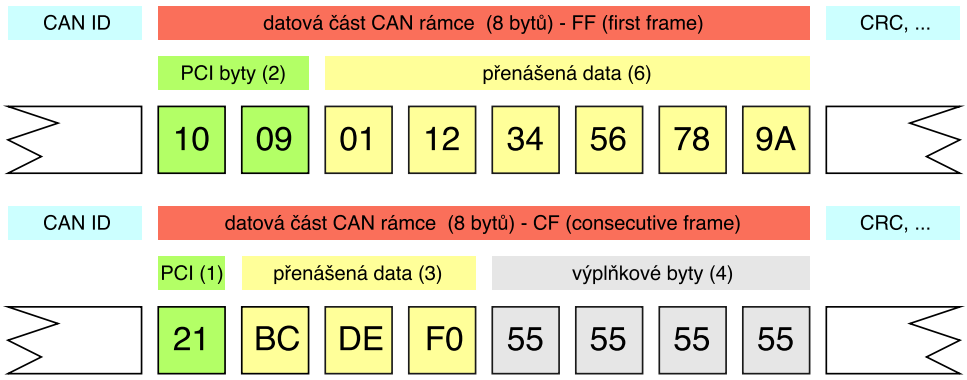
typ rámce	*PCI byty* 1. byte bity 7 – 4	1. byte bity 3 – 0	2. byte	3. byte	4. – 8. byte
SF	0000	SF_DL	data		
FF	0001	FF_DL		data	
CF	0010	SN	data		
FC	0011	FS	BS	STmin	FB

SF_DL: délka datového pole paketu (1 – 7)
FF_DL: délka datového pole paketu (8 – 4095)
SN: *segment number* – číslo segmentu
FF se bere jako nultý, tedy první CF má SN = 1 a každý další frame inkrementuje SN o 1. Po 15 následuje hodnota 0.
FS: *flow status* – stav toku dat
0 přenos může pokračovat
1 přenos pozastaven
2 přenos zrušen (lze poslat jen po 1. rámci)
BS: *block size* – velikost bloku
0 poslední FS rámec, odesílatel smí poslat, co zbývá
1 – 255 počet CF rámců do dalšího FC rámce
STmin: minimální pauza mezi CF rámci
$00 - 7F_{hex}$ 0 – 127 ms
$F1 - F9_{hex}$ 0,1 – 0,9 ms
ostatní hodnoty jsou rezervovány normou ISO 15765-2
FB: *filler bytes* – výplňkové byty
slouží jako výplň do povinných 8 bytů, zpravidla mají hodnotu 00, AA nebo 55

používaný koncernem Volkswagen. Já se jím zabývám, protože právě on je oknem do útrob vozidla, který umožňuje do jisté míry ovládat vozidlo zvenku pomocí diagnostického aparátu.

1.6.1 Unified Diagnostic Services

Služby UDS jsou navrženy tak, aby poskytly mechanikům v servisu jednotný přístup ke zjištění, co se s vozidlem děje, aniž by museli platit velké částky za licence k proprietárním formátům komunikace každému z výrobců. UDS jsou standardizovány v ISO 15765-3 a ISO 14229 [22], [23], [23], [23]. Jedná se sice o standard a CAN zprávy mají jednotnou formu, obsah už je ale rozdílný



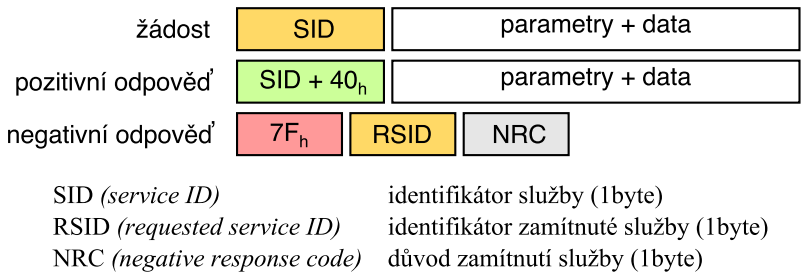
Obrázek 1.16: Odeslání dlouhé zprávy pomocí protokolu ISO-TP

nejen mezi výrobci, ale dokonce i mezi modely stejného výrobce podle data výroby. V této sekci čerpám data z výše uvedených standardů a [8].

UDS zprávy se dělí na 3 typy - žádost, pozitivní odpověď a negativní odpověď. Každá z nich má pevnou strukturu, která je popsána na obrázku 1.17. Žádost se skládá z identifikátoru služby a dat. Pozitivní odpověď má v prvním bytu identifikátor žádosti, jehož první byte je zvýšený o 40_{hex}, pak následují data odpovědi. Negativní odpověď má v prvním bytu $7F_{hex}$, ve druhém identifikátor zamítnuté služby a na konci je jeden byte, jehož hodnota udává důvod zamítnutí služby. Některé důvody zamítnutí služby a jejich kódy udává tabulka 1.7.

UDS systém poskytuje operátorovi diagnostiky mnoho akcí, jako například čtení dat, čtení z paměti, zápis do paměti nebo reset jednotky. Všechny služby jsou uvedeny v tabulce 1.5. Teoreticky je možné prostřednictvím UDS služby odeslat žádost k odemčení dveří. Pokud jednotka takovou službu podporuje, odešle na základě požadavku na CAN sběrnici zprávu, která zajišťuje odemčení dveří.

Aby mohl pracovník diagnostiky pomocí UDS něco zjistit, musí daná řídící jednotka konkrétní službu podporovat a mít ji od výrobce implementova-



Obrázek 1.17: Struktura UDS zpráv

Tabulka 1.5: Identifikátory a význam UDS služeb

skupina	*ID služby (SID)*	*název služby*
diagnostika a komunikační management	10_h	přepínání diagnostické relace
	11_h	reset řídící jednotky
	27_h	bezpečnostní přístup
	28_h	řízení komunikace
	$3E_h$	tester present
	83_h	parametry časování přístupu
	84_h	přenos zabezpečených dat
	85_h	řízení DTC nastavení
	86_h	odpověď na událost
	87_h	*link control*
přenos dat	22_h	čti data podle identifikátoru
	23_h	čti z adresy v paměti
	24_h	čti *scaling data* podle identifikátoru
	$2A_h$	čti data periodicky podle identifikátoru
	$2C_h$	dynamicky definuj datový identifikátor
	$2E_h$	zapiš data podle identifikátoru
	$3D_h$	zapiš na adresu v paměti
přenos uložených dat	14_h	vymaž uložené diagnostické chybové kódy
	19_h	čti uložené diagnostické chybové kódy
	$2F_h$	test akčních členů
	31_h	vzdálená aktivace rutiny
nahrávání a stahování	34_h	žádost o nahrání softwaru do jednotky
	35_h	žádost o stažení softwaru z jednotky
	36_h	přenos dat
	37_h	žádost o ukončení přenosu
	38_h	žádost o přenos souboru

nou. Navíc musí být tester přepnutý do správné diagnostické relace. Existují 4 standardní relace a jejich přehled udává tabulka 1.6. Některé služby jsou přístupné v každé relaci, některé pouze v jedné relaci. Kromě standardních si každý výrobce definuje i další vlastní typy relací. Koncern Volkswagen používá například *development session* – tedy vývojovou relaci, ve které má uživatel ta nejvyšší oprávnění.

Pro přechod mezi diagnostickými relacemi existuje služba s identifikátorem `10`. Pokud chci přejít z výchozí relace do rozšířené diagnostické relace, musím použít následující příkazy:

```
diagnostik      001.1 > 10 03              (žádost o~přepnutí do relace 03)
řídící jednotka 001.2 > 50 03 00 32 01 F4  (kladná odpověď + dodatečná data)
diagnostik      002.0 > 3E                 (příkaz tester present)
diagnostik      004.0 > 3E                 (příkaz tester present)
diagnostik      006.0 > 3E                 (příkaz tester present)
```

Nachází-li se diagnostické zařízení v jakékoliv jiné než výchozí relaci, musí v periodě 2 – 4 sekund vysílat `TP` *(tester present)* příkaz, který potvrzuje smysl aktivní relace. V případě, že diagnostické zařízení neodešle každých 5 sekund příkaz `TP`, bude relace násilně ukončena a diagnostické zařízení se vrátí zpět do výchozí relace s omezenými právy.

Tabulka 1.6: UDS relace

kód relace	*význam*
01_h	výchozí relace
02_h	programovací relace
03_h	rozšířená diagnostická relace
04_h	diagnostická relace bezpečnostních systémů (airbagy)

Tabulka 1.7: Některé důvody zamítnutí služby a jejich kódy

NRC kód	*význam*
11_h	služba není podporována
13_h	nesprávná délka zprávy nebo její formát
14_h	odpověď je příliš dlouhá
22_h	nesprávné podmínky
25_h	nepřišla odpověď od jednotky v podsíti
26_h	závada zabraňuje spuštění požadované akce
31_h	požadavek mimo rozsah
33_h	bezpečnostní přístup zamítnut
36_h	vyčerpán počet pokusů
76_h	požadavek dorazil v pořádku, čeká se na odpověď
$7E_h$	dílčí služba není podporována v aktuální relaci
$3F_h$	služba není podporována v aktuální relaci

1.7 Síťové vrstvy na CAN sběrnici – shrnutí

Čtenář už má nyní kompletní přehled o tom, jak probíhá komunikace na všech vrstvách CAN sběrnice. Je zřejmé, že pro přehlednost a snazší implementaci je třeba používat síťové vrstvy. Dekompozici sběrnice na vrstvy předepisuje ISO/OSI model, který definuje celkem 7 vrstev. Každá řeší nějaký problém, přitom využívá služeb vrstvy nižší a poskytuje služby vrstvě vyšší [14]. Tabulky 1.8 a 1.9 zobrazují, jaké protokoly se využívají ve které vrstvě v případě zákonem dané OBD diagnostiky respektive jaké protokoly využívají výrobci vozů pro svoji rozšířenou (interní, neveřejnou) diagnostiku. Tabulky 1.8 a 1.9 neobsahují všechny standardy, jen ty, které se vztahují k CAN sběrnici a ty, na které jsem při své práci narazil [4], [11], [12], [13], [22], [21], [23], [24], [25], [26].

Tabulka 1.8: Přehled síťových vrstev na CAN sběrnici a jejich protokolů pro OBD diagnostiku

číslo a název vrstvy	*zákonem daná OBD diagnostika protokol zajišťující vrstvu*
7 – aplikační	ISO 15031-5 [26]
6 – prezentační	—
5 – relační	—
4 – transportní	ISO 15031-5 [26], ISO 15765-2 [21], ISO 15765-4 [4]
3 – síťová	ISO 15031-5 [26], ISO 15765-2 [21], ISO 15765-4 [4]
2 – linková	CAN (ISO 11898-1 [11], ISO 15765-4 [4])
1 – fyzická	CAN (ISO 11898-2 [12], ISO 11898-3 [13], ISO 15765-4 [4])

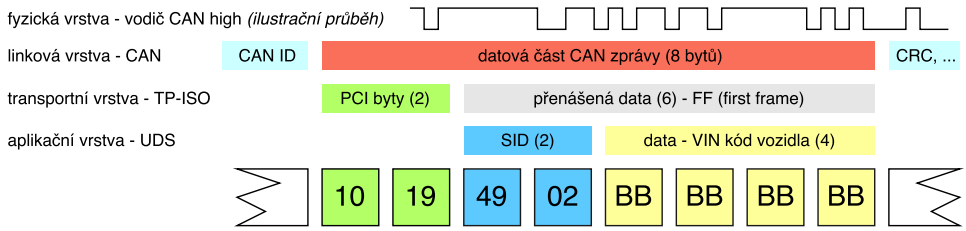
Tabulka 1.9: Přehled síťových vrstev na CAN sběrnici a jejich protokolů pro rozšířenou diagnostiku

číslo a název vrstvy	*rozšířená diagnostika výrobců vozidel protokol zajišťující vrstvu*
7 – aplikační	UDS (ISO 15765-3 [22], ISO 14229-1 [23], ISO 14229-3 [25])
6 – prezentační	—
5 – relační	UDS (ISO 15765-3 [22], ISO 14229-2 [24])
4 – transportní	ISO-TP (ISO 15765-2 [21])
3 – síťová	ISO-TP (ISO 15765-2 [21])
2 – linková	CAN (ISO 11898-1 [11])
1 – fyzická	CAN (ISO 11898-2 [12], ISO 11898-3 [13])

Aby byla problematika vrstev zcela jasná, předvedu nyní na příkladu, které protokoly se kde využívají. Budu postupovat od nejvyšších vrstev po nejnižší.

V prvním odstavci popíšu proces získání VIN kódu z vozidla a ve druhém odstavci popíšu jeho paralelu – školní výlet do Krkonoš.

Cílem je získat VIN kód vozila. Odešlu žádost a zodpovědná řídící jednotka mi pošle VIN kód vozidla ve tvaru `4902 BBBB...`. Aplikační protokol udává, že první dva byty odpovědi pro VIN kód vozidla mají tvar `4902` a pak následuje 17 bytů, které představují samotný VIN kód. Relační vrstva ověří, zda má tazatel oprávnění tento údaj zjistit. Délka zprávy je dva plus sedmnáct bytů, proto se nevejde do jednoho CAN rámce. Transportní protokol tedy sestaví jeden úvodní rámec a dva následné rámce ve tvaru `10 19 49 02 BB BB BB BB`, `21 BB BB BB BB BB BB BB` a `22 BB BB BB BB BB BB AA`. Poslední rámec by měl pouze 7 bytů, a tak se přidá výplňkový byte ve tvaru `AA`. Linková vrstva sestaví z rámců 3 CAN zprávy, přidá tedy potřebné náležitosti jako CAN ID, CRC kód atp. Zároveň zajistí, že v případě nedoručení se konkrétní zpráva vyšle znovu. Fyzická vrstva zajistí doručení CAN zpráv po vodičích k cíli – tedy diagnostickému zařízení. Obrázek 1.18 ukazuje odeslání prvního rámce odpovědi na všech vrstvách ISO/OSI modelu.



Obrázek 1.18: CAN zpráva – odpověď na žádost o VIN kód – role všech vrstev

Cílem je dostat 17 tuctů maďarských žáků na školní výlet do Krkonoš. Aplikační protokol udává, že žáci nemohou jet sami, ale každý tucet žáků musí doplnit jeden učitel. Navíc musí jet i zdravotníci a školní psycholog. Je tedy celkem třeba odeslat 17 tuctů žáků a dva tucty zaměstnanců školy. Relační protokol ověří u všech cestujících, že mají platný cestovní doklad, aby mohli přejet státní hranice. Jeden vlak uveze maximálně 8 tuctů lidí, transportní protokol maďarských drah proto rozdělí školní výpravu do 3 vlaků. Do prvního vlaku přidělí dráhy 1 tucet navigátorů, 1 tucet průvodčích a 6 tuctů cestujících. Do každého dalšího vlaku pak 1 tucet průvodčích a 7 tuctů cestujících. Díky navigátorům první vlak nezabloudí, díky průvodčím se po dojetí vlaků do cíle cestující opět spojí do jedné velké školní skupiny. Linková vrstva připraví vlak, naloží všechny lidi do vagónů, přidá lokomotivu a natankuje palivo. Zároveň zkontroluje, že všechny dveře jsou za cestujícími bezpečně zavřené. Fyzická vrstva představuje koleje, po který dojede vlak do Krkonoš.

Kapitola 2

Analýza aplikací zobrazujících data o vozidle v reálném čase

V kapitole 1 jsem shrnul znalosti o komunikaci uvnitř vozidla. Uvedl jsem, jaká se používá sběrnice a jaké protokoly určují pravidla přenosu zpráv. Nejvýznamnějším z nich je protokol CAN, který jsem detailně popsal. Naznačil jsem také, jakými způsoby je možné komunikovat s vozidlem zvenku a z analýzy vyplynulo, že budu informace z vozidla čerpat prostřednictvím diagnostického portu OBD-II. Než se pustím do návrhu vlastní aplikace, budu se věnovat těm, které již existují.

V této analýze zacílím svoji pozornost o úroveň výše. Od interní komunikace a jejích pravidel uvnitř vozidla se přesouvám k aplikacím do mobilního telefonu, které tuto interní komunikaci dokážou zachytit a interpretovat člověku v lidsky srozumitelném formátu. Cílem této analýzy je porovnat aplikace, které umožňují zobrazování údajů o jízdě a stavu vozidla v reálném čase. Zaměřuji se na aplikace pro mobilní operační systém Android, protože pro něj budu navrhovat svoji aplikaci. Díky tomu je možné měřené údaje zobrazit na libovolném zařízení, tedy nejčastěji na telefonu nebo tabletu. Analýza hodnotí aplikace především podle množství údajů, které daná aplikace umí zobrazit, stejnou měrou se ale do hodnocení promítá i uživatelská přívětivost aplikace – tedy způsob ovládání a zobrazení měřených údajů. Aplikace se dokáže připojit k vozidlu prostřednictvím nástroje, který se připojí k vozidlu do diagnostického konektoru OBD-II.

2.1 Vybrané aplikace

Pro testování byly vybrané pouze univerzální aplikace, které uvádí, že by měly fungovat pro všechna vozidla a nejsou uzpůsobené pro jednu značku. Stejně tak byl pro komunikaci mezi vozidlem a aplikací vybrán adaptér, který udává, že je pro všechna vozidla a není přizpůsoben pro nějakou značku.

Do testování byly zařazeny následující aplikace:

Car Scanner

MotorData OBD


OBDLink


OBD Car Doctor

OBDmax

OliviaDrive

Torque

Aplikace jsem vybíral na základě jejich popisu v oficiálním obchodě s aplikacemi Google Play. Není cílem analýzy porovnat všechny aplikace, které existují, ale pouze náhodně vybranou část těch, které splňují následující kritéria:

- Aplikace má od uživatelů hodnocení vyšší než 3 (na škále od 0 do 5, kde nejlepší je 5).
- Aplikace není hodnocena negativními recenzemi typu: „Aplikace se vůbec nedokáže připojit apod.“
- Aplikace podporuje komunikaci s vozidlem pomocí adaptéru, který:
  - komunikuje s telefonem/tabletem pomocí bluetooth
  - komunikuje s vozidlem skrz diagnostické rozhraní OBD-II pomocí protokolu CAN
- Aplikace je zdarma nebo má neplacenou verzi.
- Aplikace dokáže zobrazit alespoň 5 různých údajů.

2.1.1 Podmínky a průběh testování

datum:	22. 10. 2017
vozidlo:	Škoda Octavia II. generace, rok výroby 2011
adaptér:	ELM327 OBD2 CAN-BUS Interface scanner (Jedná se o levný čínský klon v ceně do 150 Kč.)
jazyk aplikace:	není-li uvedeno jinak, je aplikace v anglickém jazyce
tester:	Tomáš Zimmerhakl

Při testování byly přítomny vždy minimálně 2 osoby – řidič a tester. Před jízdou byl do diagnostického portu vozidla zapojen adaptér ELM327, který je na obrázku 2.1. Aplikace byla propojena s vozidlem podle schématu na obrázku 1.3 (obrázek je v kapitole 1.2). Testování probíhalo v reálném provozu při nenulové rychlosti. Tester procházel postupně všechny funkce každé aplikace a zkoumal, zda nabízené veličiny zobrazují reálné hodnoty. Pokud měl tester u nějaké veličiny pochybnost o správnosti zobrazovaných údajů, přezkoumal hodnotu veličiny na palubním počítači vozidla nebo požádal řidiče o manévr, který měl správnost zobrazovaných údajů potvrdit/vyvrátit.

Obrázek 2.1: Adaptér ELM327

Po testování jsem pro každou z aplikací vytvořil stručný přehled, který obsahuje snímek aplikace při běhu, její výhody, nevýhody, slovní shrnutí a moje subjektivní hodnocení na stupnici od 0 do 100 %. Přehledy pro všechny testované aplikace se nachází v následujících sekcích této kapitoly.

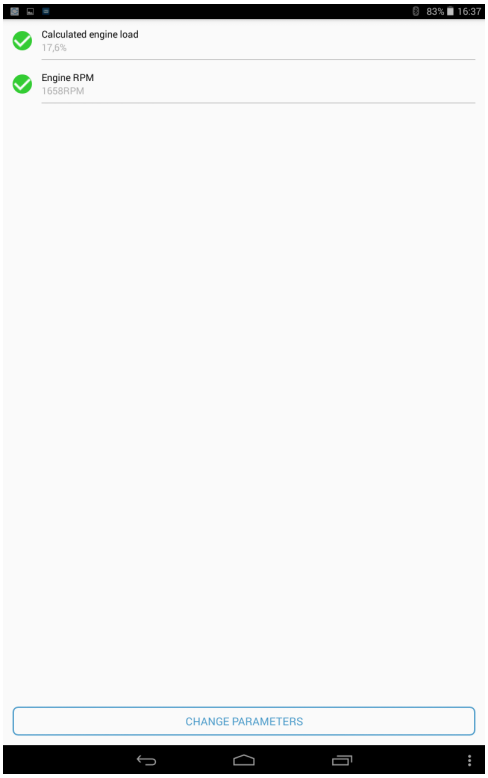
2.2 Aplikace CarScanner

Výhody

- jednoduché a přehledné uživatelské rozhraní, aplikace se nesnaží o žádnou přehnanou grafickou reprezentaci hodnot
- umí odfiltrovat hodnoty, které nejsou pro dané auto k dispozici

Nevýhody

- ve verzi zdarma podporuje pouze 2 současně zobrazené parametry (placená verze zobrazí současně 5 parametrů)
- neumí zobrazit sledované hodnoty do grafů



Obrázek 2.2: Aplikace CarScanner

Přestože je množství zobrazovaných údajů poměrně malé, aplikace působí příjemným dojmem a uživateli rychle a přehledně nabídne několik základních údajů ve vizuálně přívětivé podobě.

Celkové hodnocení: 80 %

2.3 Aplikace MotorData OBD

Výhody

- přehlednost, jednoduchý design
- velké množství údajů na jedné obrazovce, které se neustále aktualizují
- rychlé spojení s ELM327 adaptérem
- aplikace navíc obsahuje seznam DTC[15] kódů s jejich popisem
- ze všech testovaných aplikací zobrazuje největší množství údajů

Nevýhody

- obnovovací frekvence údajů na společné stránce je cca 2 s, což je příliš dlouhá doba
- při zobrazení údajů v seznamu pod sebou se u hodnot nezobrazují fyzikální jednotky
- není možné nechat si zobrazit na stránce pouze jeden údaj
- nepodporuje grafy

Obrázek 2.3: Aplikace MotorData OBD

Vítěz testu. Jako jediná neklade tato aplikace žádné omezení na množství zobrazených údajů na jedné stránce a uživatel má tedy na první pohled k dispozici maximální množství dat. Není to na úkor přehlednosti, údaje jsou zobrazeny velmi přehledně a čitelně buď v mřížce, nebo jednoduchém seznamu. Důvodem proč aplikace nedostala 100 % hodnocení je absence fyzikálních jednotek při zobrazení údajů v seznamu.

Celkové hodnocení: 90 %

[15] *Diagnostic Trouble Codes* = diagnostické chybové kódy

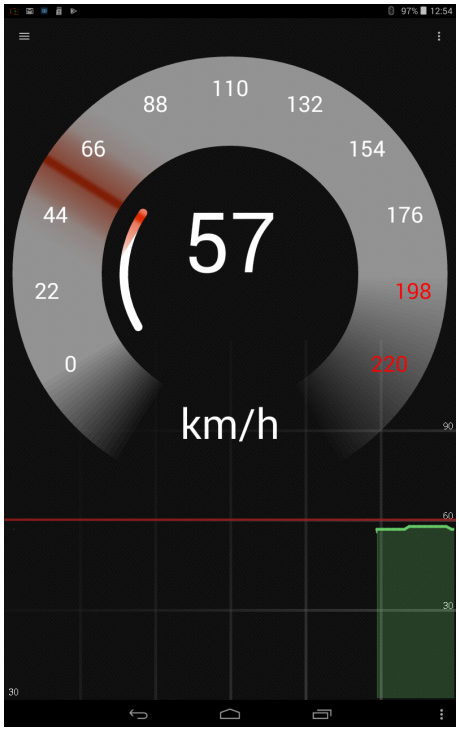
2.4 Aplikace OBD Car Doctor

Výhody

- možnost ukládání statistik
- perfektní grafické zpracování detailního zobrazení údaje
- podpora jak ciferníků, tak grafů
- největší množství podporovaných údajů z testovaných aplikací
- po klepnutí na údaj zobrazí jeho PID[16] množství údajů
- podporuje zobrazení HUD[17]
- grafické zpracování a možnost grafické personalizace je na nejvyšší úrovni ze všech testovaných aplikací

Nevýhody

- nelze zobrazit naráz více než jeden údaj
- složitější inicializační fáze – při prvním spuštění trvá poměrně dlouho, než se aplikace připojí k vozidlu



Obrázek 2.4: Aplikace OBD Car Doctor

V zobrazení jednoho údaje je tato aplikace nejlepší ze všech testovaných. Avšak neumožňuje zobrazit ani 2 údaje naráz, což považuji za poměrně velký handicap a proto dávám 70 %. To lze vyřešit zakoupením placené varianty aplikace za necelých 90 Kč.

Celkové hodnocení: 70 %

[17] *Parameter identifier* je kód parametru, který diagnostické zařízení využívá k získání jeho hodnoty z vozidla.

[17] *Head-up display* promítá v noci údaje na sklo vozidla do zorného pole řidiče. Aplikace zrcadlově převrátí všechna data na displeji a telefon nebo tablet se umístí pod čelní sklo vozidla.

2.5 Aplikace OBDmax

Výhody

- umí zobrazit naráz 4 hodnoty nebo jednu
- přehledné a široké nastavení
- rychlé spojení s ELM327 adaptérem
- aplikace navíc obsahuje seznam DTC kódů s jejich velmi podrobným popisem
- aplikace navíc obsahuje seznam ikon, které se mohou rozsvítit na přístrojové desce s vysvětlením jejich významu

Obrázek 2.5: Aplikace OBDmax

Nevýhody

- k některým zobrazovaným hodnotám chybí fyzikální jednotky
- problémy s automatickým připojením, připojení s adaptérem bylo nutné sestavovat ručně
- množství podporovaných údajů není tak velké jako u konkurenčních aplikací
- popisky k hodnotám a fyzikální jednotky jsou příliš daleko od čísel a v zobrazení 4 hodnot snadno může dojít k chybné interpretaci hodnot

V porovnání s ostatními aplikacemi podporovala tato méně údajů. Měl jsem dojem, že aplikace není úplně odladěná a že si občas „dělá, co chce“. Slabší dojem aplikace dohání manuálem DTC kódů a ikonek palubní desky. Kvůli grafickému zpracování dávám pouze 50 %.

Celkové hodnocení: 50 %

2.6 Aplikace OliviaDrive

Výhody

- aplikace zobrazuje naráz 12 údajů
- všechny zobrazované údaje jsou neustále obnovovány
- přehledné zobrazení údajů v mřížce 3x4
- podporuje HUD *(head-up display)*

Nevýhody

- nelze ovlivnit, které údaje se budou zobrazovat, aplikace tedy podporuje pouze 12 údajů
- není možné jiné zobrazení než v mřížce 3x4
- absence fyzikálních jednotek
- aplikace je tak špatně přeložena, že to znemožňuje porozumění uvedeným údajům
- aplikaci nelze přepnout do jiného jazyka, výchozím jazykem je mix češtiny a angličtiny, kterému někdy není možné porozumět

Obrázek 2.6: Aplikace OliviaDrive

Překlad do češtiny je katastrofální a díky absenci fyzikálních jednotek je porozumění některým zobrazovaným hodnotám zcela nemožné. Například jsem nezjistil, co znamenají údaje „heslo“, „vyhořelé“ nebo „křižování“. Z celkem 12 zobrazených údajů jsem porozuměl pouze čtyřem. Je to velká škoda, protože aplikace je po designové stránce zpracována výborně, je navíc možné měnit barvu textu i pozadí a podporuje HUD v režimu více zobrazených hodnot na jedné obrazovce.

Celkové hodnocení: 30 %

2.7 Aplikace Torque

Výhody

- možnost přidat do monitorování vlastní PID, tedy veličinu, kterou aplikace ve výchozím hodnocení nepodporuje
- možnost vytvořit si vlastní plochu, na které bude libovolný počet údajů, je možné vytvořit si celkem 7 různých ploch
- veškeré údaje na dané ploše se aktualizují v reálném čase
- každý údaj může být zobrazen buď ve formě ciferníku, grafu nebo jako číslo
- uživatel má k dispozici mnoho vzhledů/motivů
- aplikace je v českém jazyce

Obrázek 2.7: Aplikace Torque (1)

Nevýhody

- přidávání údajů na plochu není intuitivní, stejně tak posuv nebo odstranění je zbytečně pracné
- ovládání není snadné, trvá dlouho, než uživatel pochopí, jak aplikaci ovládat;
- design aplikace je zastaralý
- při zobrazení veličiny v grafu chybí fyzikální jednotky
- každý údaj je nutné přidat na plochu zvlášť, není možné přidat jich více naráz
- aplikace vizuálně nijak neoddělí veličiny, které vozidlo podporuje, od těch nepodporovaných
- překlad do češtiny je občas na škodu při porozumění veličinám

Při prvním použití jsem měl z aplikace velmi špatný dojem. Ovládání není intuitivní, design aplikace je zastaralý. Naštěstí je možné všechno individuálně přizpůsobit. Počínaje pozadím a vzhledem ciferníků až po jejich množství a umístění na jednu z mnoha dostupných ploch. Pokud tedy uživatel vynaloží přibližně hodinu času na to, aby si aplikaci přizpůsobil svým potřebám, může docela dobře plnit svoji funkci. Bohužel při zobrazení údajů ve formě grafu chybí fyzikální jednotky, což spolu s dalšími nedostatky snižuje hodnocení na 50 %.

Obrázek 2.8: Aplikace Torque (2)

Celkové hodnocení: 50 %

2.8 Aplikace OBD Link

Tato aplikace funguje výhradně s diagnostickými adaptéry OBDLink a s adaptérem ELM327 se ji vůbec nepodařilo spárovat. Testování tedy vůbec neproběhlo, proto jsem aplikaci nezařadil ani do závěrečného přehledu v tabulce 2.1.

Celkové hodnocení: nehodnoceno

2.9 Porovnání schopností vybraných aplikací

Tabulka 2.1 zobrazuje, která aplikace (v záhlaví sloupců) podporuje zobrazení kterých údajů (záhlaví řádků). Pokud je v buňce uvedeno písmeno P, znamená to, že aplikace daný údaj podporuje. Písmeno C značí, že údaj je podporován, ale jeho hodnota byla při měření prokazatelně chybná. Je-li buňka prázdná, aplikace daný parametr nepodporuje. Použité zkratky jsou vysvětleny v příloze A na straně 113.

Tabulka 2.1: Porovnání schopností vybraných aplikací

			aplikace						
parametr - anglický výraz	*parametr - český překlad*	*jednotky*	*CarScanner*	*MotorData OBD*	*OBD Car Doctor*	*OBDmax*	*OliviaDrive*	*Torque*
acceleration	zrychlení	$m \cdot s^{-2}$						P
acceleration 0 – 60 mph	zrychlení 0 – 60 mph	s						P
ambient temperature	teplota okolního vzduchu	°C	P	P	P	P		P
average consumption	průměrná spotřeba	l/100 km			P			P
average consumption	průměrná spotřeba	km/l			P			P
barometric pressure	venkovní tlak vzduchu	kPa	P	P	P	P		P
battery voltage	napětí akumulátoru	V	P	P	P		P	P
calibration ID (CID)	CID - ID softwaru řídící jednotky			P				
calibration verification numbers	CVN - kalibrační verifikační číslo			P				
commanded EGR	míra recirkulace spalin	%	P	P	P			
coolant temperature	teplota chladící kapaliny	°C	P	P	P	P		P
current consumption	aktuální spotřeba	l/100 km			P	C	P	C
current consumption	aktuální spotřeba	km/l			P			P
distance traveled since DTCs cleared	vzdálenost ujetá od vymazání DTC	km	P	P	P			
EGR error	EGR error	%	P	P	P			
engine load	zatížení motoru	%	P	P	P	P	P	P
engine run time since start	doba běhu motoru od startu	min:s	P	P	P			
engine speed	otáčky motoru	rpm	P	P	P	P	P	P
fuel rail pressure	tlak v sacím potrubí	kPa	P	P	P	P		
intake air temperature	teplota nasávaného vzduchu	°C	P	P	P	P		P
intake manifold abs. pressure (MAP)	tlak v sacím potrubí	kPa	P	P	P	P		P
MAF air flow rate	množství nasávaného vzduchu	g/s	P	P	P	P		P
O2S1 WR lambda equivalence ratio	lambda poměr		P	P	P			P
oxygen sensor - voltage	napětí generované O2 senzorem	V	P	P				P
protocol used	použitý protokol pro komunikaci		P	P	P	P		
speed	rychlost	km/h	P	P	P	P	P	P
supported PIDs	podporované identif. parametrů				P			
throttle position	pozice plynového pedálu	%	P	P	P	P		
turbo boost	zesílení turba	psi						P
vehicle identification number (VIN)	VIN – identifikační číslo vozidla		P	P	P			
warm-ups since codes cleared	počet zahřátí od vynulování DTC		P	P	P			

2.10 Závěrečné shrnutí analýzy aplikací

Analýza ukázala, že pro monitorování údajů vozidla v reálném čase existuje poměrně široká nabídka volně dostupných aplikací. Ty se však vzájemně velmi liší jak množstvím poskytovaných údajů, tak svým grafickým zpracováním a ovladatelností. Stanovil jsem 3 odlišné skupiny uživatelů, podle způsobu použití aplikace. Pro každou skupinu je nejvhodnější jiná aplikace, což je patrné z tabulky 2.2.

Ukázalo se, že kvalitní řešení existují a každý si může vybrat podle toho, co hledá. Dokonalá aplikace neexistuje, přesto prostor k vylepšení už není na straně podporovaných údajů, ale v uživatelském rozhraní. Ideální aplikace by měla podporovat jak více údajů na jedné obrazovce, tak zobrazení pouze jediného údaje. Nejblíže tomu je aplikace OBD Car Doctor, která je svým grafickým zpracováním o třídu lepší než ostatní aplikace a ve své placené verzi se tomuto ideálu velmi blíží.

Tabulka 2.2: Nejlepší aplikace pro sledování dat o vozidle

Uživatel	*Vítězná aplikace*
uživatel, který chce monitorovat jeden údaj	OBD Car Doctor
nenáročný uživatel, který chce monitorovat mnoho údajů současně	MotorData OBD
náročný uživatel, který chce monitorovat mnoho údajů současně a je ochoten věnovat přibližně hodinu času k přizpůsobení a nastavení aplikace	Torque

Mnoho aplikací má kromě zobrazení aktuálních dat také doplňkové funkce. Mezi ty nejužitečnější řadím zobrazení diagnostických chybových kódů, ukládání naměřených dat. Aby se nová aplikace v tomto segmentu uchytila, musí podporovat zobrazení více údajů na jedné obrazovce, musí umět přečíst z vozidla velké množství údajů, musí je přehledně a vizuálně přitažlivě zobrazit a zároveň musí být co nejjednodušší, aby ji nový uživatel mohl ihned začít používat bez nutnosti číst návod.

Kapitola 3

Návrh řešení

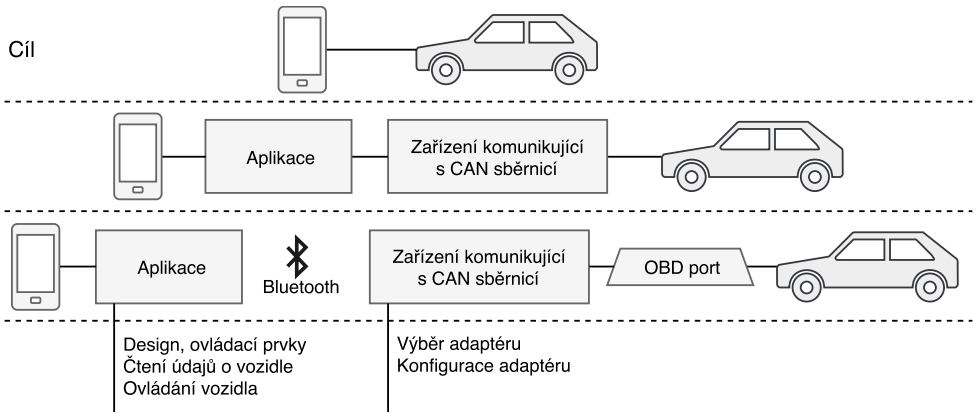
Úkolem této práce je vytvořit aplikaci pro systém Android, která bude komunikovat s CAN sběrnicí ve vozidle. Aplikace bude na jedné straně číst informace o stavu vozidla, na straně druhé bude možné prostřednictvím aplikace iniciovat akce, které bude možné přímo pozorovat (například vypnutí rádia, klimatizace, zapnutí stěračů, rozsvícení světel). Řešení první části úkolu je dobře známé a díky normě ISO 15031-5 [26] existuje standardizované diagnostické rozhraní, prostřednictvím něhož je možné číst informace o vozidle. Jak ukázala Analýza aplikací zobrazujících data o vozidle v reálném čase v kapitole 2, pro čtení těchto údajů existuje poměrně velké množství volně dostupných aplikací.

Na druhou stranu ovládání součástí vozidla pomocí mobilní aplikace je univerzálně nemožné. Neexistuje žádné standardizované rozhraní, které by dovolilo uživateli ovládání vozu jinak než prostřednictvím přepínačů, tlačítek a infotainment systému přímo ve vozidle. Každý výrobce osobních automobilů požívá vlastní komunikační protokoly, které jsou v jeho vlastnictví a podléhají utajení. Zprávy, které na vnitřních sběrnicích vozidla iniciují nastavení vozidla, jsou neveřejné jednak kvůli bezpečnosti a také kvůli udržení náskoku před konkurencí. Proto ani není možné stáhnout na internetu program či aplikaci, která by dokázala ovládat vozidlo alespoň zčásti tak, jako ho může ovládat posádka přímo ve vozidle. Existují sice aplikace, které umožňují ovládání vozidla, ale jsou vždy úzce zaměřeny na konkrétního výrobce a podporují jen málo modelů dané značky, zpravidla ještě omezených rokem výroby. K mání je například doplněk do aplikace Torque pro vozidla značky Nissan. S ním a s použitím vhodného adaptéru do OBD portu je možné ovládat světla, klakson nebo zámek dveří. Celkem aplikace umožňuje ovládat 10 – 15 funkcí [27] a je svými schopnostmi unikátní. Podobnou aplikaci pro jiné značky se mi nalézt nepodařilo.

Absence hotových řešení ovšem neznamená, že uzavřu svoji práci bez výsledku. Na internetu i mimo něj je možné nalézt mnoho podrobných návodů,

jak *hackovat*[18] auta [6], [8], [28], [29], [30]. Existují přístupy, kterými lze dosáhnout i na údaje, které automobiloví výrobci nechávají v utajení. Na ně se v této kapitole zaměřím.

Nyní se nebudu pouštět do implementačních detailů. Naopak, dekomponuji[19] problém na více menších částí, kterými se budu podrobně zabývat v následujících podkapitolách. Proces dekompozice je zobrazen na obrázku 3.1.



Obrázek 3.1: Dekompozice návrhu

3.1 Spojení s OBD-II portem

V této sekci představím hardwarové nástroje, které je možné použít ke komunikaci s vozidlem. Nejprve představím profesionální drahé zařízení, které se používá k diagnostice v autoservisech a při testování ve vývoji. Poté naopak uvedu několik jednoduchých a poměrně levných variant. V anglické literatuře se používá pojem „device“ [8], což česky znamená „zařízení“. To je ale dle mého názoru velmi obecný pojem, z něhož by čtenář nemusel přesně pochopit, o čem hovořím. Budu se bavit o nástrojích, které zpracovávají signál z CAN sběrnice a nějakým způsobem ho v jiné formě posílají buď do počítače nebo mobilního telefonu. Pro taková zařízení budu požívat pojem *adaptér*, protože převádí (adaptuje) komunikaci ve vozidle do formy, která je zpracovatelná počítačem, telefonem, atp. V následujícím výčtu čerpám ze zdrojů [8], [29], [30]. Ceny adaptérů jsou platné v březnu 2018.

[18] Hackování je proces, při kterém se počítačový expert snaží dostat do systému, ke kterému nemá přístup. Systém je pro něj „černá bedna“, o které nic neví. Cílem je vytěžit ze systému maximum informací. Pojem *hackování* je v této práci vždy míněn v kladném smyslu a chápán jako činnost, která je prováděna se souhlasem všech účastníků a v prostředí s regulovaným provozem. Výsledky nemohou ovlivnit běžného účastníka silničního provozu, naopak jejich sdílení s výrobcem přispívá k vyšší bezpečnosti provozu jeho vozidel.

[19] Dekompozice je proces rozložení složitého problému na mnoho menších, jejichž pochopení je snazší a řešení může být přímočaré. [31]

Obrázek 3.2: Profesionální diagnostické zařízení VAS 6154

VAS 6154
Jedná se o profesionální zařízení v ceně přibližně 100 eur. Je určeno pro diagnostiku vozů koncernu Volkswagen (značky Škoda, Seat, Audi, atp.) a je schopné komunikovat jak přes WiFi, tak přes USB kabel. Používá se společně s programem ODIS (Offboard Diagnostic Information System) při vývoji a testování i při servisních úkonech. Je na obrázku 3.2.

CANtact *(http://linklayer.github.io/cantact)*
CANtact je open source rozhraní mezi CAN a USB. Je k dostání za 60 dolarů a má velikost asi 2 krabiček zápalek. K autu se připojuje přes konektor typu DB9. Je tedy třeba mít redukci z DB9 na J1962 (standardní konektor OBD2), který je k dostání za 5 – 10 dolarů.

Mikrokontrolér ELM327 *(https://www.elmelectronics.com/ic/elm327/)*
ELM327 je mikročip od firmy ELM Electronics, který posílá CAN komunikaci do počítače po sériové lince RS232. Zvládá interpretovat všechny diagnostické protokoly popsané v sekci 1.3. Ve velkých sériích se dá pořídit už od 15 dolarů a na stránkách výrobce jsou přímo uvedené návody, jak použít čip k sestrojení diagnostického nástroje. Asijští hackeři tento mikročip s oblibou kopírují a vytvářejí jeho klony, které jsou jádrem malých diagnostických adaptérů [32]. Přestože originální čip stojí minimálně 15 dolarů, čínští obchodníci nabízejí na e-shopech Ebay, Amazon, AliExpress a dalších celé adaptéry za cenu mezi 5 a 10 dolary. Tyto adaptéry obsahují J1962 konektor typu samec a k počítači je možné je připojit prostřednictvím bluetooth, WiFi nebo USB kabelu.

USB2CAN *(https://www.8devices.com/products/usb2can/)*
Za 65 dolarů je možné zakoupit adaptér s DB9 konektorem typu samec na jedné straně a USB konektorem na straně druhé. Pro připojení do vozidla je tedy nutné stejně jako u CANtact dokoupit redukční kabel z DB9 na J1962. USB2CAN dokáže monitorovat CAN sběrnici a je kompatibilní jak s operačním systémem Windows tak se systémem Li-

nux. Neumožňuje bezdrátové spojení a není tedy vhodný pro komunikaci s mobilním telefonem. Adaptér je na obrázku 3.3[20].

CAN232 a CANUSB *(https://www.can232.com/)*
Produkty CAN232 a CANUSB od švédské firmy Lawicel AB je možné zakoupit za cenu okolo 100 dolarů. Oba jsou malé adaptéry, které převádí komunikaci na CAN sběrnici na komunikaci po sériové lince RS232. Oba adaptéry mají na jedné straně DB9 konektor a pro spojení s automobilem je třeba redukční kabel z DB9 na J1962. CAN232 má na druhé straně standardní RS232 konektor, zatímco CANUSB má na druhé straně USB konektor. Posílání a příjem CAN zpráv se provádí z počítače prostřednictvím ASCII protokolu.

Obrázek 3.3: Adaptér USB2CAN

Adaptérů existuje velké množství, tento výčet však považuji za dostatečný. Obsahuje adaptéry, které jsou k dostání a bez nutnosti programovat k nim řadiče nebo ovladače je možné je po krátkém čtení návodu ihned používat. Daly by se pořídit i vývojové desky založené na platformě Arduino nebo Raspberry Pi s moduly pro CAN komunikaci, ty ale vyžadují programování vlastní obsluhy. Není nutné, abych se ve své práci zabýval přenosem CAN zpráv mezi telefonem a adaptérem. Místo toho rád využiju hotové řešení.

Rozhodl jsem se, že pro svoji práci zakoupím adaptér ELM327, který má na jedné straně OBD-II port (samec) a na druhé straně komunikuje pomocí bluetooth s libovolným zařízením. Důvody výběru jsou následující:

[20] Obrázek 3.3 je převzatý z [33].

- Bezdrátový adaptér vybavený bluetooth anténou je logicky nejkomfortnější, protože odpadá nutnost fyzického spojení s mobilním telefonem. Tomu odpovídá i nabídka na trhu – OBD-II adaptéry mají na straně komunikace s počítačem výhradně bluetooth, WiFi nebo USB-A rozhraní. Adaptér s konektorem typu microUSB (samec) je velmi obtížné sehnat.
- Adaptér ELM327 lze v čínských e-shopech pořídit velmi výhodně, cena se pohybuje v přepočtu od 100 do 200 Kč (podzim 2017).
- Mikrokontrolér ELM327 je velmi dobře zdokumentovaný [34] a je široce konfigurovatelný. Dokáže libovolně nastavit svoje CAN-ID a naopak umí libovolně filtrovat příchozí zprávy podle jejich CAN-ID.
- Adaptér komunikuje s mobilním zařízením prostřednictvím sérového protokolu. Díky tomu lze ke komunikaci využít hotové aplikace a programy, kterých je ke stažení zdarma celá řada. Pro počítač existují například *Putty* [35] nebo *Advanced Serial Port Terminal* [36], pro Android existuje aplikace *Serial Bluetooth terminal* [37].

3.2 Práce s adaptérem ELM327

ELM327 je mikročip od firmy ELM Electronics, který posílá CAN komunikaci do počítače po sériové lince RS232. Je vidět uprostřed na obrázku 3.4. Dodává se i s návodem, jak s využitím tohoto čipu sestavit adaptér do auta. Mnoho firem (především v Číně) to využívá a vyrábí různé druhy adaptérů založené na čipu ELM327. Mikročip odesílá data do počítače pomocí sériové komunikace, fyzické médium ale může být libovolné. Vyrábí se jak tradiční adaptéry s USB kabelem (na obrázku 3.4 vpravo), tak bezdrátové, komunikující buď přes bluetooth (na obrázku 3.4 vlevo) nebo WiFi.

Ke komunikaci s adaptérem ze strany počítače nebo chytrého mobilního zařízení je třeba obyčejný sériový terminál nebo libovolná aplikace, která ho dokáže emulovat[21]. Dále v textu budu pro zjednodušení používat slovo „počítač“, i když je možné ho libovolně nahradit pojmy „chytrý telefon“ nebo „tablet“. V této sekci čerpám informace z manuálu k čipu ELM327 [34].

ELM327 podporuje všechny protokoly definované standardem OBD-II (viz sekci 1.3), tedy i CAN protokol podle ISO 15765-4, a to ve 4 variantách:

- 11-bitový CAN identifikátor, 500 kb/s
- 29-bitový CAN identifikátor, 500 kb/s
- 11-bitový CAN identifikátor, 250 kb/s
- 29-bitový CAN identifikátor, 250 kb/s

[21] Emulátor je software, který umožňuje běh programů i na přístrojích, které k tomu nejsou přirozeně uzpůsobeny. Například mají nevhodný operační systém nebo nemají potřebné porty.

Obrázek 3.4: Uprostřed mikročip ELM327 a po stranách na něm založené adaptéry ELM327

Adaptéry ELM327 zpravidla nemají vypínač a zapínají se jednoduše tak, že se zasunou do OBD portu. Ve chvíli, kdy je na pinu 16 (viz obrázek 1.5) přítomné napětí 12 V, adaptér 4krát za sebou zabliká LED diodou a pošle do počítače úvodní zprávu:

```
ELM327v2.2>
```

Zde jsem uvedl uvítací zprávu adaptéru verze 2.2, adaptéry se postupně vyráběly od verze 1.0 až po současnou 2.2. S adaptérem je možné komunikovat ve dvou základních režimech:

AT příkazy
: Pokud přijatý příkaz začíná sekvencí znaků `AT`, ELM327 ho interpretuje jako AT příkaz. Ty slouží ke konfiguraci a nastavení ELM327 adaptéru. Při posílání AT příkazů uživatel nijak neinteraguje s vozidlem a není tedy ani nutné být přitom připojen na CAN sběrnici.

Příkazy pro vozidlo
: Pokud přijatý příkaz začíná znakem `0`–`9` nebo `A`–`F`, jedná se o příkaz pro automobil. Pro taková data sestaví ELM327 validní CAN zprávu a odešle ji do vozidla.

Pokud přijatý příkaz nesplňuje ani jednu z výše uvedených podmínek, ELM jednoduše odešle zpět do počítače `?>`, což znamená, že zadaný příkaz nebyl rozpoznán a nic se nestalo.

Když adaptér čte data ze sériové linky, čte je tak dlouho, dokud nenarazí na znak „CR“ (*carriage return*), což je hexadecimálně `0D`. V tu chvíli čtení zastaví a začne příkaz interpretovat. Z toho tedy vyplývá jednoduché pravidlo, že každý AT příkaz i příkaz pro vozidlo odeslaný počítačem musí být ukončen bytem `0D`. Při komunikaci z adaptéru do počítače zase platí, že každá zpráva je zakončená znakem `>`, což indikuje, že ELM327 je připraveno na další příkaz.

Několik důležitých informací pro komunikaci mezi ELM327 a počítačem už uvedu pouze heslovitě:

- ELM327 nerozlišuje na vstupu malá a velká písmena, řetězce `at i` a `AT I` jsou považovány za identické.
- ELM327 ignoruje na vstupu mezery a tabulátory, řetězce `AT PP SV` a `ATPPSV` jsou považovány za identické.
- ELM327 si pamatuje poslední zadaný příkaz, pokud ho chce uživatel zopakovat, stačí, když pošle jen ukončovací byte `OD`.

3.2.1 AT příkazy

AT příkazy slouží k nastavení ELM adaptéru. Je možné pomocí nich zjistit, jak je adaptér nastaven a také toto nastavení změnit. S jejich využitím lze ovlivnit použitý protokol, změnit formátování odesílaných zpráv jak do vozidla, tak do počítače, nebo ovlivnit dobu, po kterou bude adaptér čekat na odpověď z vozidla. Ty nejdůležitější příkazy, které budu používat pro svůj projekt, uvádím v tabulce 3.1. Ostatní příkazy si zájemci mohou dohledat v [34], kde je k nim i rozsáhlý popis. Manuál k ELM327 je volně ke stažení na webových stránkách společnosti ELM Electronics.

Níže uvádím příklad komunikace pomocí AT příkazů. Pokud AT příkaz něco nastavuje, ELM327 odpoví `OK>`. Tím dává uživateli najevo, že nastavení bylo úspěšně provedeno.

```
uživatel| AT I~(dotaz na verzi zařízení)
adaptér | ELM327 v1.5>  (odpověď)
uživatel| AT L1         (posílej za odpovědí oddělovač řádku)
adaptér | OK>           (adaptér potvrzuje nastavení)
uživatel| AT H1         (posílej hlavičky CAN zpráv)
adaptér | OK>           (adaptér potvrzuje nastavení)
uživatel| AT S1         (odděluj byty mezerami)
adaptér | OK>           (adaptér potvrzuje nastavení)
uživatel| AT AL         (povolí příjem zpráv delších než 7 bytů)
adaptér | OK>           (adaptér potvrzuje nastavení)
uživatel| AT DP         (zobraz použitý protokol)
adaptér | ISO 15765-4 (CAN 11/500)>
```

3.2.2 Příkazy pro vozidlo

Jestliže chci komunikovat prostřednictvím adaptéru s vozidlem, musím nejprve zasunout adaptér do OBD-II portu (J1962), pak musím zasunout klíč do zapalování a otočit do první polohy. Není nutné mít zapnutý motor, CAN sběrnice je aktivní i bez toho. Je ale zřejmé, že data z pohonné CAN sběrnice při vypnutém motoru nezískám.

Tabulka 3.1: Některé AT příkazy pro čip ELM327

Obecné příkazy – ovlivňují komunikaci adaptér ⟷ počítač

kód	*význam*
@1	ukáže popis adaptéru
D	resetuje všechna nastavení na výchozí
Ex	echo – opakuje před odpovědí zadaný příkaz, x=0 vypnuto, x=1 zapnuto
I	vypíše označení a verzi adaptéru
Lx	nový řádek na konci poslané zprávy, x=0 vypnuto, x=1 zapnuto
Sx	zobrazování mezer mezi byty odpovědi, x=0 vypnuto, x=1 zapnuto
WS	restart zařízení
Z	reset zařízení (vrátí adaptér do továrního nastavení)

OBD příkazy – ovlivňují komunikaci adaptér ⟷ vozidlo

kód	*význam*
AL	povolí příjem a odesílání zpráv delších než 7 bytů
CAFx	CAN auto formátování – automaticky přidává PCI byty do odeslané zprávy a odstraní je z přijaté, x=0 vypnuto, x=1 zapnuto, výchozí nastavení je CAF1
CF xxx	nastaví filtr – zobrazí uživateli jen zprávy s CAN ID, který odpovídá filtru
CM xxx	nastaví masku – když je bit masky 1, porovnává se bit v CAN ID na stejné pozici s filtrem, jinak ne
CRA xxx	poslouchá jen zprávy, jejichž CAN ID = xxx
DP	vypíše aktuálně používaný protokol
DPN	vypíše číslo aktuálně používaného protokolu
Hx	zobrazování CAN hlaviček v odpovědích, x=0 vypnuto, x=1 zapnuto
MA	spustí monitoring veškerého provozu na sběrnici
SH xxx	nastaví CAN ID zařízení na hodnotu xxx
SP x	nastaví komunikační protokol na x-tý (jsou uloženy pod čísly 1 – 10), 0 znamená, že se protokol vyhledá automaticky
ST xx	nastaví *timeout* na $xx \cdot 4$ ms, což je doba, po kterou čeká adaptér na odpověď od vozidla, výchozí nastavení je 200 ms

Také příkazy pro vozidlo jsou zakončeny povinným bytem `OD`. Ten se však do auta neodesílá, slouží jen jako signál pro adaptér a po přijetí je odebrán. Adaptér zkontroluje délku zprávy, protože odesílání zpráv delších než 7 bytů nepodporuje. Je-li zpráva kratší, spustí se následující proces:

1. Adaptér vytvoří z dat CAN zprávu podle norem (viz shrnutí v sekci 1.7) a vlastního nastavení a fyzicky ji odešle na vodiče *CAN high* a *CAN low*.

2. Potom adaptér po nastavenou dobu (maximálně 1 s) monitoruje CAN sběrnici a čeká na odpověď. Přitom kontroluje hlavičky zpráv podle vlastního filtru.

3. Pokud odpověď přišla, adaptér ji dekóduje a pošle podle nastavených pravidel po sériové lince do počítače.

4. Pokud odpověď nepřišla, adaptér pošle do počítače řetězec `NO DATA>`.

Ve výchozím nastavení adaptér odstraňuje všechny byty spojené s komunikačními protokoly a po sériové lince do počítače odesílá jen holá data. Je ale možné pomocí AT příkazů přimět adaptér, aby společně s daty posílal do počítače i CAN ID a PCI byty[22] transportního protokolu. Adaptér může uživateli odeslat i DLC kód, ale například CRC kód[23] odesílat nelze.

Níže uvádím příklad komunikace pomocí příkazů pro vozidlo:

```
uživatel| 01 0D       (dotaz na rychlost vozidla)
adaptér | 41 0D 09>   (odpověď - vozidlo jede 9 km/h)
uživatel| 01 0D       (dotaz na rychlost vozidla)
adaptér | 41 0D 10>   (odpověď - vozidlo jede 16 km/h)
uživatel| 01 FF       (nesmyslný dotaz)
adaptér | NO DATA>    (od CAN sběrnice nepřišla žádná odpověď)
```

3.3 Android Aplikace

V autě používám adaptér ELM327, který komunikuje s počítačem prostřednictvím bluetooth, uvnitř něhož se chová jako sériová linka RS232. Já nahrazuji počítač mobilním telefonem s operačním systémem Android (vyplývá ze zadání). Pro něj jsem naprogramoval aplikaci, která bude schopná pomocí sériového protokolu uvnitř bluetooth komunikovat s adaptérem. Pojmenoval jsem ji *OBD Robot*.

Bluetooth je standard pro bezdrátovou komunikaci, který definuje, jak technologie funguje. Profil definuje, jak je technologie využívána. U Bluetooth komunikace existuje podle [38] několik profilů:

SPP *Serial Port Profile*
: Jeho účelem je nahradit sériovou komunikaci (RS232, UART). Využívá se při komunikaci mikrokontrolérů nebo FPGA[24] zařízení, protože implementace sériového protokolu je v hardwaru poměrně snadná.

HID *Human Interface Device*
: Slouží zpravidla pro připojení drobných periferních zařízení k počítači (myš, klávesnice). Jeho účelem je nahradit USB kabel.

[22] vysvětleno v tabulce 1.4

[23] CAN ID, DLC kód a CRC kód jsou vysvětleny v tabulce 1.2

[24] *Field programmable gate array* – programovatelná hradlová pole jsou zařízení, do kterých si jejich funkcionalitu naprogramuje až uživatel. Funkcionalita může být kdykoliv změněna.

HFP a HSP *Hands-Free Profile a Headset Profile*
Používá se u jednoúčelové nositelné elektroniky jako jsou sluchátka nebo mikrofony. Obvykle implementuje i jednoduché akce, jako například zvednutí nebo zavěšení telefonního hovoru.

A2DP *Advanced Audio Distribution Profile*
Stejně jako HSP a HFP také k přenosu dat mezi audiotechnikou, ale na rozdíl od nich je možné posílat data jen jedním směrem (například pouze z přehrávače do sluchátek). Díky tomu může být audio-kvalita daleko vyšší.

Ve své diplomové práci využívám sériovou komunikaci, a proto potřebuji implementovat SPP. Vzhledem k tomu, že tato práce není o tom, jak programovat přes bluetooth, využívám hotové řešení z webových stránek *Lemberg Solutions Blog* [39]. Zde je poměrně krátký a srozumitelný návod, podle kterého jsem postupoval. Z něho je zřejmé, že při implementaci bylo nutné dodržet následující sekvenci kroků:

1. Po spuštění aplikace zkontroluje zapnutý bluetooth.
2. Pokud není, vynutí si jeho zapnutí. Poté nabídne uživateli seznam spárovaných zařízení.
3. Uživatel klikem na název zařízení připojí zařízení přes Bluetooth.
4. Uživatel se dostane do režimu, kdy může komunikovat s adaptérem ELM327.

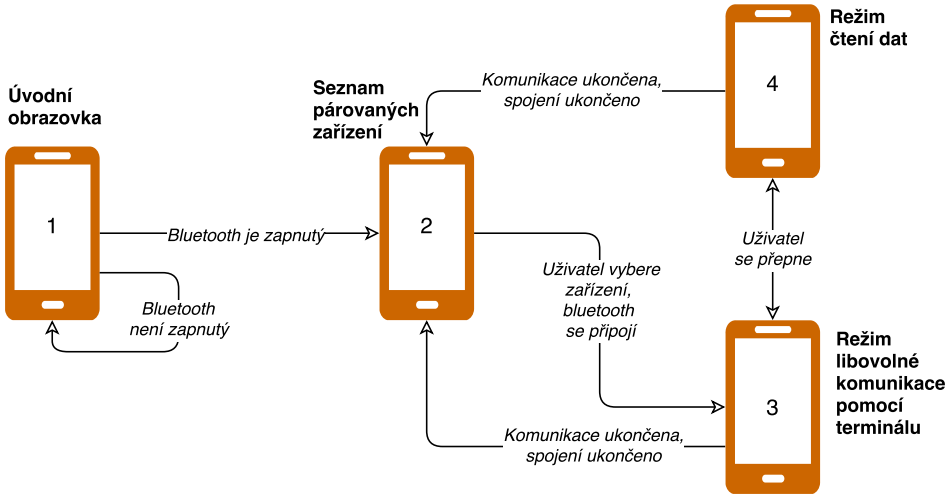
V momentě, kdy je uživatel schopen komunikovat s vozidlem, nabízí aplikace dva základní režimy:

1. režim čtení dat
2. režim libovolné komunikace prostřednictvím terminálu

V režimu čtení dat nabízí aplikace uživateli na jedné obrazovce nepřetržitý monitoring vybraných parametrů v jednoduché a střídmé formě. Klíčová je čitelnost a srozumitelnost, nikoliv designové zpracování. Proto v aplikaci nebudou žádné grafy ani obrázky, monitorované veličiny budou zobrazeny v seznamu pod sebou včetně fyzikálních jednotek a jejich hodnoty se budou periodicky obnovovat. Je to schopnost, kterou jsem nemohl ocenit ani u jedné z analyzovaných aplikací v kapitole 2.

V režimu libovolné komunikace prostřednictvím terminálu má uživatel k dispozici příkazový řádek, pomocí něhož bude moci zadávat jak příkazy pro vozidlo, tak AT příkazy pro konfiguraci adaptéru ELM327. Díky příkazům je uživatel schopen číst informace o vozidle, ale i částečně vozidlo ovládat.

Historie komunikace prostřednictvím příkazové řádky se zaznamenává na obrazovku a volitelně je možné ji uložit do textového souboru. Ani tuto schopnost nemá žádná ze sledovaných aplikací v kapitole 2. Základní funkcionalitu aplikace je dobře vidět na obrázku 3.5, kde je naznačen i průchod celou aplikací od spuštění až po ukončení.



Obrázek 3.5: Schéma fungování aplikace OBD Robot

3.4 Metody čtení informací z vozidla

Norma ISO 15031-5 [26] založená na původní normě SAE J1979 definuje diagnostické služby a zprávy, které musí být podporovány ve vozidlech vybavených OBD-II konektorem. Veškeré poskytované služby jsou uvedeny v tabulce 3.2. Jejich součástí jsou služby s identifikátorem začínajícím na `01`, které poskytují aktuální data z pohonného ústrojí vozu. A protože v sekci 1.3 uvádím, že OBD-II systém je přítomen dnes v každém moderním voze, není důvod tyto služby nevyužít. Identifikátory služeb *(ID)* jsou jednobytové hodnoty a v tabulce 3.2 jsou uvedené hexadecimálně.

Každá ze služeb uvedených v tabulce 3.2 poskytuje uživateli hodnoty mnoha parametrů. Aby diagnostik hodnotu parametru získal, musí poslat žádost obsahující identifikátor parametru (**PID** – *parameter ID*). Jedná se o jednobytový kód, který je snadno dohledatelný v [26]. Norma je ovšem zpoplatněna[26], proto doporučuji článek OBD-II PIDs na anglické Wikipedii [40]. Zde je možné všechny PID dohledat. Nejedná se o zdroj, ze kterého bych při psaní práce vycházel, [40] je spíše dobrou pomůckou pro ty, kteří by se o téma

[25] PB – počet získaných datových bytů v odpovědi, značím je zleva po řadě A, B, C,...
[26] Ke dni 3. 4. 2018 byla cena 200 CHF (švýcarských franků).

Tabulka 3.2: Přehled diagnostických služeb systému OBD-II

ID služby	*popis služby*
01	aktuální data z pohonného ústrojí
02	uložená data z pohonného ústrojí
03	přístup k uloženým DTC (diagnostickým chybovým kódům)
04	vymazání DTC
05	žádost o výsledky monitoringu senzoru kyslíku
06	žádost o výsledky monitoringu ostatních systémů
07	zobraz čekající DTC (zjištěny během aktuální nebo poslední jízdy)
08	řízení OBD systému, komponenty nebo testu
09	data o vozidle (trvalého charakteru)

chtěli více zajímat bez nutnosti investovat do nákupu normy. Ukázka tabulky s přehledem identifikátorů parametrů je v tabulce 3.3, hodnoty ve sloupci PID jsou hexadecimálně zapsané jednobytové hodnoty. Tabulka 3.3 také vysvětluje, jakým způsobem porozumět přijaté odpovědi.

Obsluhu dotazů na PID má na starosti zpravidla motorová řídící jednotka a ne každá podporuje všechny parametry. Je-li odeslán požadavek, který není motorovou řídící jednotkou podporován, jednoduše nepřijde žádná odpověď. Adaptér tedy po vypršení časového limitu odešle do počítače zprávu `NO DATA`. Aby se uživatel vyhnul zkoušení podporovaných parametrů, může je zjistit prostřednictvím parametrů `00`, `20`, `40`, `60` a `80` služby `01`.

Chci například zjistit, jaká aktuální data jsou ve vozidle podporována. Nelze to zjistit pro všechny naráz, ale vždy jen pro 20 za sebou jdoucích identifikátorů. Zjistím tedy, které z identifikátorů v rozsahu `01` – `20` automobil podporuje. (Všechny identifikátory parametrů uvádím hexadecimálně.)

Tabulka 3.3: OBD-II PIDs – Identifikátory parametrů – ukázka

PID	*PB*[25]	*popis*	*min. hodnota*	*max. hodnota*	*jednotky*	*výpočet*
Služba 01						
`00`	4	podporované PID 1 – 20	–	–	*vysvětleno v textu*	
`0C`	2	otáčky motoru	0	16 383,75	rpm	$\frac{256A+B}{4}$
`31`	2	doba běhu motoru	0	65 535	s	$256A + B$
`33`	1	tlak venkovního vzduchu	0	255	kPa	A
`5C`	1	teplota oleje	-40	210	°C	$A - 40$

```
uživatel pošle dotaz   | 01 00              (služba 01, PID 00)
řídící jednotka odpoví | 41 00 BE 1F A8 13
```

Obrázek 1.17 v sekci 1.6.1 uvádí, že pokud přijde první byte požadavku zvýšený o 40_{hex}, přišla kladná odpověď. To se také stalo. Následuje zopakovaný PID (00), pak následují 4 byty odpovědi, přesně jak udává tabulka 3.3. Jejich význam dekóduje obrázek 3.6.

Hexadecimálně	B				E				1				F				A				8				1				3			
Binárně	1	0	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	1	0	0	1	1
PID podporováno	✓	x	✓	✓	✓	✓	✓	x	x	x	x	✓	✓	✓	✓	✓	✓	x	✓	x	✓	x	x	x	x	x	x	✓	x	x	✓	✓
PID číslo (hexa)	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20

Obrázek 3.6: Dekódování odpovědi na dotaz o podporované identifikátory parametrů

Nyní mám jistotu, že PID 0C pro zjištění otáček motoru je podporovaný, a tak se zeptám na otáčky motoru:

```
uživatel pošle dotaz   | 01 0C              (služba 01, PID 0C)
řídící jednotka odpoví | 41 0C 0E F0
```

Obdržel jsem 2 datové byty odpovědi: $A = \texttt{0E}_{hex} = 14_{dec}$ a $B = \texttt{F0}_{hex} = 240_{dec}$. Podle vzorečku z posledního sloupce tabulky 3.3 zjistím hodnotu otáček motoru následovně:

$$otáčky = \frac{256A + B}{4} = \frac{256 \cdot 14 + 240}{4} = 956$$

Zjistil jsem, že hodnota otáček motoru je 956 otáček za minutu, což odpovídá stavu, kdy motor běží na volnoběh. Obdobným způsobem získám a vypočítám s pomocí [26] a [40] i ostatní parametry.

3.5 Metody ovládání vozidla

Jak už bylo řečeno v úvodu kapitoly 3, neexistuje univerzální způsob pro ovládání vozidel. Výrobci používají vlastní komunikační protokoly a ty podléhají utajení. Existuje pár aplikací, které ovládání vozidla umožňují, ale jsou vždy úzce zaměřeny na konkrétního výrobce a několik jeho modelů. I bez znalosti utajovaných protokolů však lze za určitých okolností odhalit některé příkazy pro ovládání vozidla. Metody, jak toho docílit, jsou k nalezení v [6], [8], [28], [29] a [30]. V této sekci popisuji, jak by měl člověk postupovat, aby odhalil skrytou komunikaci uvnitř vozidla. Jedná se v podstatě o *hackování* aut, ale v dobrém smyslu.

Pojem *hacker* je obvykle vnímán veřejností negativně, ale správné chápání tohoto pojmu by mělo být poněkud jiné. Hacker je někdo, kdo tvoří, zkoumá a

objevuje zákonitosti, jak věci fungují. Při své činnosti porozumí fungování automobilu a lépe tak identifikuje problémy. Hacker může objevit v autě funkce, o kterých dříve nevěděl, nebo si auto dovybavit vlastními. A neméně důležité je, že hacker je schopen posoudit bezpečnost a zranitelnost zkoumaného vozu z pohledu cizího vniknutí do něj [8].

Existují dvě základní metody – *fuzzing* [28] a *sniffing* [8]. O nich budou následující dvě podkapitoly.

3.5.1 Fuzzing

Překlad pojmu do českého jazyka není ustálen, proto budu používat anglický originál *fuzzing*. Jedná se o metodu, jejímž cílem je nalézt zranitelnosti systému. K tomu se využívá opakované posílání poškozených nebo záměrně neplatných dat do systému. Přitom se kontroluje chování systému a čeká se na odpovědi, které mohou poskytnout cennou zpětnou vazbu. Využívá se takzvaný *fuzzer*, což je zpravidla program nebo hardware, který buď zcela náhodně nebo na základě nějaké heuristiky[27] odesílá do systému požadavky a vyhodnocuje odezvu. Skládá se ze tří částí:

- generátor zpráv *(message generator)*
- odesílatel zpráv *(message publisher)*
- monitor cíle *(target monitor)*

Generátor zpráv je jádrem fuzzeru a kvalita fuzzingu závisí do velké míry právě na kvalitě generování zpráv. Generovat je možné zcela náhodně nebo používat platná data a na nich provádět náhodné změny. Třetí možnost zahrnuje vytvoření modelu, který definuje strukturu zprávy. Na základě modelu jsou následně tvořeny zprávy.

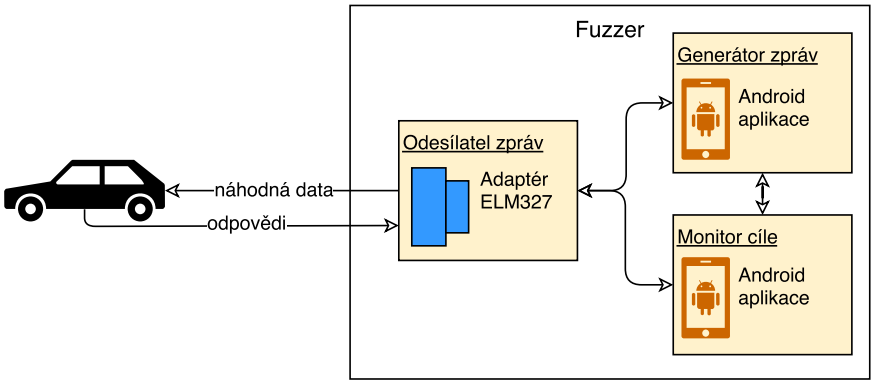
Zatímco generátor zpráv může být sdílený pro různé systémy, odesílatel zpráv je vždy vyvíjen přímo pro testované zařízení. Stará se o korektní odesílání zpráv prostřednictvím sítě, implementaci všech nutných protokolů nebo ukládání souborů do adresářů.

Monitor cíle sbírá odezvu ze systému, filtruje ji a posuzuje, jestli byl testovaný systém nějak generovaným vstupem zasažen. V ideálním případě se snaží detekovat, jakými vstupy byly vyvolány jaké výstupy a případným zopakováním vstupů svoji domněnku potvrdí.

V případě *hackování* automobilu bych jako generátor zpráv využil svoji aplikaci pro Android. Odesílal bych na CAN sběrnici náhodné zprávy a čekal bych na odpovědi ze systému. Jako odesílatel zpráv by sloužil adaptér ELM327, pro monitoring bych využil opět mobilní aplikaci. Vstupy bych dále modifikoval podle toho, na které z nich by přicházely odpovědi z vozidla. Ty

[27] Heuristika je řešení na základě kvalifikovaného odhadu. Používá se tam, kde není znám přesný algoritmus.

bych dále analyzoval a snažil bych se odhalit jejich význam. Na obrázku 3.7 je zjednodušené schéma fuzzingu na CAN sběrnici.



Obrázek 3.7: Fuzzing CAN sběrnice v automobilu – princip

3.5.2 Sniffing

Český překlad tohoto anglického termínu zní *čichání*, což je poměrně výstižné. V češtině se používá pojem *čmuchal* pro někoho, kdo slídí a zjišťuje informace [41]. To poměrně dobře vystihuje princip této metody. V případě CAN sběrnice je přeci jen přiléhavější pojem *odposlouchávání*, kterého se spolu s anglickým originálem budu dále držet.

Zatímco fuzzing aktivně generuje komunikaci, sniffing je přesný opak. Pouze odposlouchává veškerý provoz na sběrnici, ale nijak se na komunikaci aktivně nepodílí. *Sniffer* je softwarový nástroj, který provádí odposlech sběrnice. Existují nástroje jak pro příkazovou řádku, tak klasické programy pro Windows i Linux. Vyznačují se jednoduchým uživatelským rozhraním – uprostřed dominuje plocha, kam se po spuštění monitoringu vypisují zachycené CAN zprávy. Pokročilejší nástroje dovedou zachycené zprávy filtrovat, řadit nebo zobrazovat v režimu změn. To znamená, že nové zprávy se nevypisují pod ty původní chronologicky, ale nahrazují dřívější zprávy se stejným CAN ID (pro detaily viz tabulku 1.2). Na obrazovce pak uživatel vidí pouze tolik řádků, kolik druhů zpráv (rozlišených podle CAN ID) na sběrnici za sledované období projde. Nástroje vhodné pro sniffing sumarizuje tabulka 3.4.

Přístupů, které využívají sniffing jako hlavní zbraň při odhalování skryté komunikace, je mnoho. Níže uvedené tři mohou posloužit samy o sobě, ale může být výhodné je i vzájemně zkombinovat a vzít si z každého jen některou část.

Tabulka 3.4: Přehled softwarových nástrojů pro sniffing CAN sběrnice

jméno	*platforma*	*zpoplatněn*
CANiBUS	Linux	ne
CANoe	Windows	ano
cansniffer	Linux – příkazový řádek	ne
Caring Caribou	Linux – příkazový řádek	ne
Kayak	Linux, Windows	ne
Komodo	Linux, Windows	ano
Octane	Windows	ne
SavvyCAN	Linux, Windows	ne
Vehicle Spy	Windows	ano

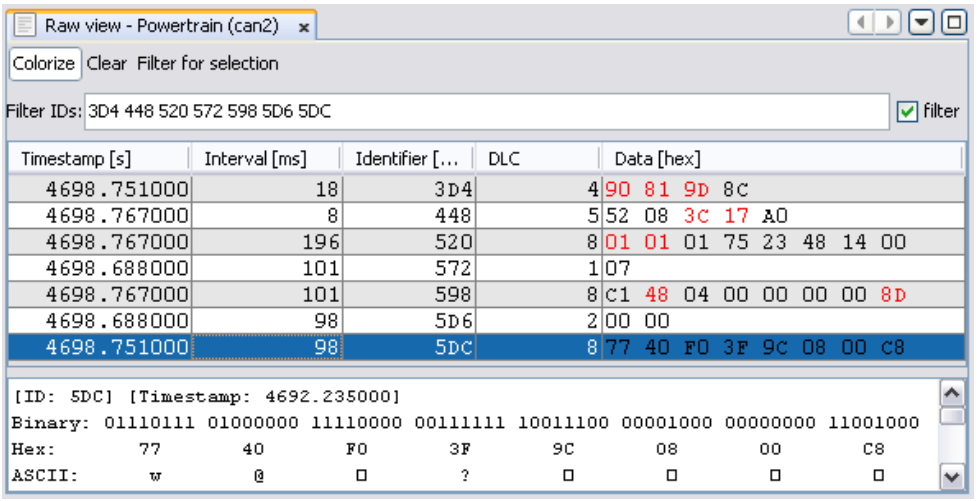
3.5.2.1 Monitoring sběrnice se zobrazením v režimu změn

Kenny Kuchera v [29] uvádí jeden z možných přístupů k odhalování komunikace v automobilu za použití sniffingu. Nejprve je nutné vygenerovat přirozeně akci, kterou chce později napodobit odesláním CAN zpráv. V textu se snaží o zachycení zprávy, která je zodpovědná za aktuální otáčky motoru. Spustil v počítači monitoring sběrnice v režimu změn. Nechal vozidlo v klidu se zapnutým motorem a počkal několik sekund. Během té doby sniffer odstranil zprávy, které se neměnily a ponechal na monitoru pouze ty, jejichž obsah se stále měnil. Následně šlápl na plyn a sledoval přitom, které ze zpráv se začaly měnit v závislosti na akci, kterou právě vyvolal. Postupně vylučoval jednu po druhé zprávy, které nijak se zvýšením otáček nesouvisely. Ve chvíli, kdy už nebylo co vyloučit, zůstalo několik kandidátů (zpráv). Tyto kandidáty postupně odesílal na sběrnici, dokud jedna z nich nezpůsobila rozpohybování ručičky na ciferníku s otáčkami motoru.

3.5.2.2 Monitoring sběrnice a přehrávání záznamů

Velmi podobný přístup s použitím programu Kayak je uveden v [8]. Popisuje, jak z mnoha desítek zpráv určit tu správnou, která zodpovídá za námi zvolené chování automobilu. Ukázka monitoringu sběrnice v programu Kayak je na obrázku 3.8. Postup se dá shrnout do následujících bodů:

1. Spusť nahrávání.
2. Vykonej akci, například rozsviť světla.
3. Zastav nahrávání.



Obrázek 3.8: Sniffing – ukázka monitoringu sběrnice v programu Kayak

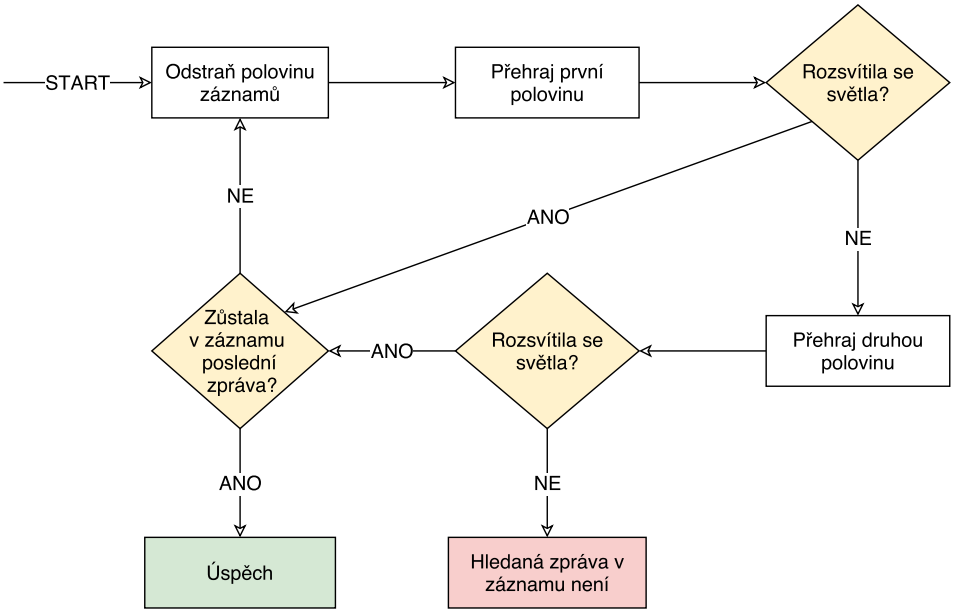
4. Stiskni tlačítko „playback“ (přehrát znovu).

5. Pozoruj, jestli se akce zopakovala, například světla se rozsvítila.

Jestliže stisk tlačítka „playback“ akci nespustí, může být na vině několik aspektů. Zaprvé, mohlo se stát, že nahrávání akci nezaznamenalo. Je tak určitě namístě nahrávání a provedení akce několikrát zopakovat. Zadruhé, zpráva sice proběhla, ale na jiném segmentu CAN sběrnice než je monitorovaný úsek. Jak ukazuje obrázek 1.2 v sekci 1.1, CAN sběrnice bývá rozdělena na více podsběrnic a některé zprávy probíhají jen na některých podsběrnicích.

Jakmile je vytvořen záznam, který spustí žádanou akci (například rozsvít světla), je možné pro odhalení konkrétní zprávy využít metodu půlení intervalu, která je naznačena v diagramu na obrázku 3.9. I 2 – 3 s krátký záznam se může skládat ze stovek CAN zpráv. Metoda půlení intervalu mezi nimi v rozumném čase nalezne tu jednu (nebo více), která je zodpovědná za vybranou akci. Jedná se o iterativní metodu[28], která v každém cyklu odstraní polovinu zpráv a vyzkouší, jestli je zbylý soubor zpráv stále schopen akci vyvolat. Pokud ano, v příštím kroku se opět polovina zpráv odstraní. Pokud ne, vrátí se stav o krok zpět a odstraní se místo toho druhá polovina zpráv. Opět se zkontroluje, jestli je zbytek zpráv schopen vyvolat akci. Když ano, pokračuje se do dalšího cyklu a tak dále, až dokud není odhalena minimální sada zpráv zodpovědná za vybranou akci.

[28]Iterativní metoda pracuje v cyklech. V každém cyklu provede úkony a zkontroluje, jestli se přiblížila řešení. Na konci každého cyklu je zkontrolována ukončovací podmínka, která garantuje ukončení procesu.



Obrázek 3.9: Sniffing – metoda půlení intervalu odhaluje zprávu zodpovědnou za rozsvícení světel

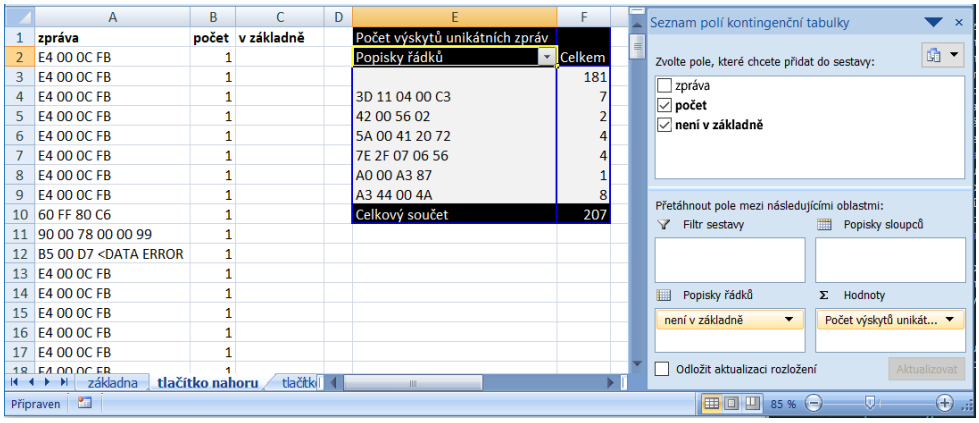
3.5.2.3 Monitoring sběrnice a rozdílová analýza

Kristoffer Smith v [30] odhaloval zprávy, které odesílají tlačítka na volantu. V jeho vozidle byl volant vybaven 5 tlačítky. Použil adaptér ELM327 a jako nástroj pro analýzu Microsoft Excel. Aplikoval následující postup:

1. Sedl si do auta, připravil měřící aparaturu a pustil motor.
2. Spustil sniffing sběrnice asi na 1 s a přitom nechal vozidlo v klidu. Tak získal „základnu“, což jsou zprávy které po sběrnici chodí pravidelně aniž by uživatel něco v automobilu prováděl.
3. Spustil sniffing sběrnice a přesně pětkrát stiskl na volantu první tlačítko.
4. Nahrávání ukončil a uložil do textového souboru.
5. Spustil odposlech sběrnice znovu a přesně pětkrát stiskl na volantu druhé tlačítko.
6. Nahrávání ukončil a uložil do textového souboru.
7. Třetí a čtvrtý krok zopakoval i pro 3., 4. a 5. tlačítko.
8. Textové soubory s nahrávkami importoval do programu Excel. Každou nahrávku do vlastního listu a každou zprávu na vlastní řádek.

9. Ke každé zprávě v záznamech tlačítek přidal sloupec „počet“ a vložil do něj hodnotu 1.

10. Ke každé zprávě v záznamech tlačítek přidal sloupec „v základně“. Do něj zkopíroval obsah zprávy pouze tehdy, pokud byl přítomen i v základně. Zprávy, které v základně nebyly, měly tento sloupec prázdný. Záznam i s přidanými sloupci je vidět na obrázku 3.10.



Obrázek 3.10: Sniffing – následné zpracování dat v programu MS Excel

11. Pro záznam z každého tlačítka vytvořil kontingenční tabulku[29], která obsahovala pouze zprávy, které nebyly v základně. Ty byly navíc seskupeny podle obsahu zprávy a ve sloupci „suma“ byla vypočtena četnost jejich výskytů. Kontingenční tabulka i s jejím nastavením je vidět na obrázku 3.10.

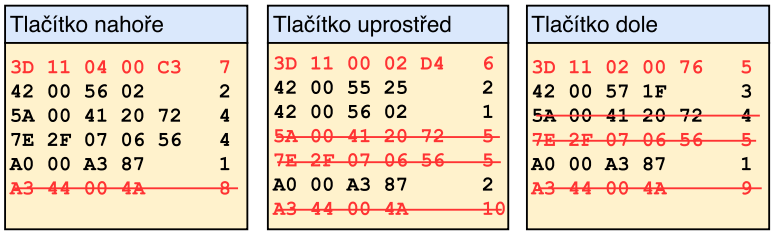
12. Na základě tohoto filtru zůstalo přibližně 5 – 10 zpráv. Z nich vyškrtl zprávy, které se vyskytly méně než 5krát. To jsou černé řádky na obrázku 3.11. Naopak zprávy, které se vyskytly častěji, mohou být hledané, protože při dlouhém stisku tlačítka se mohla odeslat více než jedna zpráva.

13. Nakonec porovnal tabulky pro každé tlačítko mezi sebou a hledal zbývající zprávy, které se nachází ve více tabulkách. I ty vyškrtl, protože stisk každého tlačítka musí generovat unikátní zprávu. Jedná se o přeškrtnuté červené zprávy na obrázku 3.11.

[29]Kontingenční tabulka je nástroj programu Microsoft Excel, který automaticky sumarizuje data s mnoha parametry. Automaticky vytváří z velkých a nepřehledných tabulek menší, které obsahují souhrny pro vybrané parametry.

14. Zbývající zprávy na obrázku 3.11 musí patřit ke stisknutým tlačítkům. To potvrdila nejen následná zkouška, ale i podobnost všech zpráv. Jsou tvořeny podle shodného vzoru, v závislosti na stisknutém tlačítku se mění pouze poslední 3 byty.

Velmi podrobný postup včetně konkrétních dat a mnoha obrázků je k nahlédnutí přímo v článku na webových stránkách [30].



Obrázek 3.11: Sniffing – rozdílová analýza odhaluje zprávu odeslanou po stisku tlačítka

O tom, jak jsem se zde vysvětlené pojmy a postupy pokoušel aplikovat, hovoří kapitola 4, sekce 4.3. Tam je také uvedeno, k jakým výsledkům jsem s využitím *fuzzingu* a *sniffingu* dospěl.

Kapitola 4

Realizace

Cílem této práce je podle zadání nalézt příkazy pro čtení aktuálních parametrů o vozidle a s jejich pomocí vytvořit vlastní aplikaci pro mobilní systém Android. Aplikace umí nejen číst data z vozidla, ale také odesílat do vozidla příkazy, na které vozidlo reaguje. V kapitole 1 jsem hovořil především o komunikačních protokolech ve vozidle. Bez jejich znalosti by nebylo možné navázat s vozidlem smysluplnou komunikaci. V kapitole 2 jsem prozkoumal, jaká řešení na trhu už existují. Zjistil jsem, že je mnoho aplikací pro čtení dat, ale mají svá omezení. Zároveň neexistuje ani jedna univerzální aplikace pro ovládání vozidla. V kapitole 3 jsem zdůvodnil, proč takové aplikace nejsou k dispozici a v sekci 3.5 jsem nastínil přístupy, jak ovládací příkazy odhalit. Zároveň jsem v kapitole *Návrh* definoval kroky, které jsou k realizaci a dosažení vytyčených cílů zapotřebí.

Vzhledem ke komplexnosti cíle jsem realizaci rozdělil do tří tématických celků:

1. pořízení adaptéru ELM327, jeho nastavení, práce s ním a čtení údajů o vozidle – sekce 4.1 od strany 62
2. vytvoření aplikace OBD Robot – sekce 4.2 od strany 72
3. odhalování interní komunikace ve vozidle – sekce 4.3 od strany 83

Vytvořil jsem si takový pracovní plán, aby práce na jednotlivých tématických celcích byla nezávislá (dále jen „celek“). Proto jsem v průběhu realizace diplomové práce mohl pracovat na výše uvedených celcích paralelně. Občas se práce na jednom celku protnula s činností na celku jiném, ale po většinu času nebylo pro jednotlivé celky zásadní, v jaké fázi řešení se ostatní nachází. Každý z celků má jasně definovaný cíl. Ty jsou uvedeny na začátku následujících podkapitol, které se zpracování tématických celků podrobně věnují. Cíle jednotlivých celků jsou stanoveny tak, aby při splnění všech tří byl zároveň splněn cíl celé práce.

4.1 Seznámení s adaptérem ELM327 a čtení dat

Cílem tohoto tématického celku je pořízení adaptéru ELM327. Dále zvládnutí jeho nastavení pomocí AT příkazů a schopnost využít ho pro čtení aktuálních údajů z vozidla.

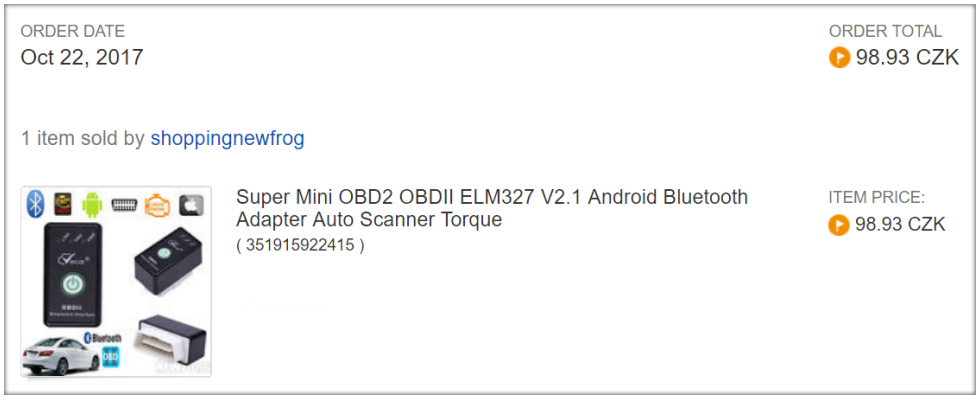
Den 1.

Známe se s Markem už 14 let. Od doby, kdy jsme spolu odmaturovali na mladoboleslavském gymnáziu se naše cesty vydaly různými směry, oba jsme však zamířili technickým směrem. Já ve službách Fakulty informačních technologií ČVUT a on pod křídly Matematicko-fyzikální fakulty Univerzity Karlovy. Naše diskuze před pátečními fotbalovými zápasy často zabrousí do oblasti aut a otázky typu „Tak za kolik jsi dneska jel...“ patří ke standardním úvodům konverzace, na které už ani není třeba odpovídat. Oba víme, že když není provoz, dá se jet ze Sobotky do Mladé Boleslavi pod 4,5 litru. Jeho Octavia z první generace je dříč a s pravidelnou péčí slouží více než 16 let.

A nezůstává jen u pravidelné péče, Marek je automobilový nadšenec a tento pátek mi předváděl svůj OBD-II adaptér ELM327. Je to malý modrý kvádřík o rozměrech 5 x 3 x 3 cm s průhledným krytem. S pomocí aplikace *Torque* v mobilním telefonu jsme sledovali otáčky motoru, teplotu, tlak, rosný bod ... No dobře, rosný bod to měřit neumí, ale i tak je to fantastická krabička, která dokáže dostat data z přístrojové kapličky do mobilu a navíc umí přidat i taková, která z budíku pod volantem rozhodně nevyčtete. Bavili jsme se s Markem o tom, že by chtěl s pomocí této krabičky číst komunikaci na CAN sběrnici v autě a zjistit z ní úhel natočení předních kol. Už ani přesně nevím, k čemu to potřeboval, ale bylo to takové snění, které mělo k realizaci daleko.

Já jsem v té době uváděl do chodu kolos jménem „diplomová práce“, zkoumal jsem různé možnosti a potřeboval jsem nutně pořídit nějaký adaptér. Věděl jsem, že chci krabičku, která bude dostupná pro každého blázna, jako jsem já. Mobil má dnes každý, auto také, tak proč nezkusit výzkum s tím, co máme po ruce? Když jsem zjistil, kde a za kolik Marek svůj adaptér pořídil, bylo jasno. Čínská kopie za 4 dolary z eBay[30]? To je přeci výzva! Profi servisy mají nástroje za desítky tisíc a já zkusím podobná kouzla s aparátem za pár dolarů. Kam s tím dojdu? Bude to vůbec fungovat jinak než s aplikacemi, které doporučuje prodejce? Hlavou se mi prohnalo hodně otázek a touha zjistit odpovědi mě přivedla až na webové stránky ebay.com, odkud jsem svůj první adaptér ELM327 objednal (obrázek 4.1). Standardní doba doručení z Číny se pohybuje okolo 6 týdnů. Abych nemusel tak dlouho čekat, půjčil mi navíc Marek svůj adaptér. Práce tedy mohla začít téměř okamžitě.

[30]eBay (www.ebay.com) je jeden z největších eshopů světa. Dá se tam sehnat téměř cokoliv a obchodníci z Číny odesílají velmi často zboží do Evropy zcela zdarma.



Obrázek 4.1: Objednávka adaptéru ELM327 značky Viecar

Den 10.

Práce na analýze aplikací (Výsledky analýzy uvádím v kapitole 2.) jsou v plném proudu. Předletová příprava spočívala v instalaci mnoha aplikací do tabletu, mnoho z nich se tvářilo jako černé skříňky a před spojením s adaptérem odmítaly vykázat jakoukoliv činnost. Do auta jsem se těšil jako malý kluk, když dostane svoji první plastovou motorku.

O víkendu jsem vytáhl rodinu na „výlet“ na hrad Houska a naše rodinná Octavia se po cestě proměnila v testovací vozidlo. Seděl jsem vpředu na sedadle spolujezdce s blokem a tužkou a když jsem zrovna nenadával nad tím, že se aplikace není schopná připojit k adaptéru, kabinou se rozléhaly pokyny „neutrál“, „plný plyn“, „Jakou máme rychlost?“. Načež se od volantu ozývaly máminy reakce „Vidíš tam něco?“, „Aby to to naše auto vůbec přežilo!“ a „57“.

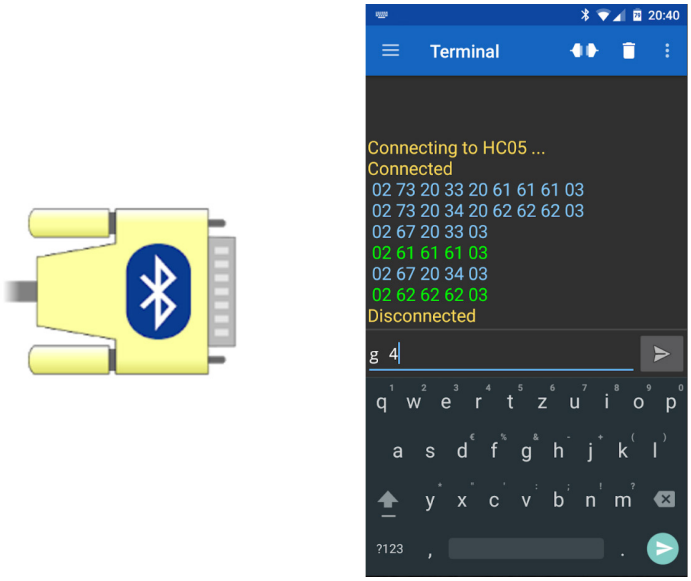
Připojování adaptéru k obslužným aplikacím se ukázalo jako velký kámen úrazu. Aplikace samozřejmě netuší, jaký komunikační protokol používá Octavia II. generace, a tak na začátku pošlou adaptéru příkaz `AT SP 0`, což je pro adaptér pokyn, aby si protokol našel sám. Jak jsem psal v sekci 1.3, diagnostický systém OBD-II podporuje mnoho komunikačních protokolů a některé navíc v různým nastaveních. Než adaptér projde všech deset možností a najde mezi nimi na indexu 6 tu správnou – ISO 15765-4 (CAN 11/500), trvá to někdy i desítky sekund. Naštěstí většina aplikací umožnila v nastavení protokol zvolit předem, a tak bylo napodruhé spojování přeci jen svižnější.

Dalším problémem byla kvalita spojení. OBD-II konektor je ve vozidle umístěn pod volantem nad pedály a je těžko přístupný. Po zasunutí do konektoru indikuje adaptér správnou činnost rozsvícením červené LED diody. Ta ale při testech vůbec nebyla vidět. Když jsem občas adaptér zasunul málo, dozvěděl jsem se o špatném spojení až po několika marných pokusech o spojení ze strany mobilní aplikace. Řešením by byl OBD prodlužovací kabel, ale kdesi

uvnitř mé hlavy se ozval strýček Skrblík a lakota zvítězila nad praktičností. Prozatím.

Den 37.

V obchodě Google Play s aplikacemi pro Android jsem narazil na perfektní aplikaci. Jmenuje se *Serial Bluetooth Terminal* a zprostředkovává komunikaci prostřednictvím sériové linky přes bluetooth. Implementuje profil SPP, přesně tak, jak jsem psal v sekci 3.3. Aplikace je velmi jednoduchá. V jedné obrazovce se nastavuje komunikační partner a na druhé obrazovce je terminál, kam se píšou zprávy. Do stejného terminálu se vypisují i příchozí zprávy, ale komunikace zůstává přehledná, protože příchozí a odchozí zprávy se rozlišují barvou. Na obrázku 4.2 je kromě loga aplikace také ukázka krátké komunikace, tento výřez z aplikace jsem stáhl přímo z náhledů v oficiálním obchodě s aplikacemi Google Play [37]. Protože ELM327 komunikuje s mobilními aplikacemi prostřednictvím sériového protokolu SPP, umožnila mi aplikace Serial Bluetooth Terminal spojit se s adaptérem přímo a komunikovat s ním jak na úrovni AT příkazů, tak na úrovni příkazů pro vozidlo (sekce 3.2.1 a 3.2.2).



Obrázek 4.2: Aplikace Serial Bluetooth Terminal – vlevo logo a vpravo ukázka komunikace

Den 38.

Před dvěma dny konečně dorazil z Číny objednaný adaptér značky Viecar. Na rozdíl od Markova adaptéru je černo-bílý a kryt má neprůhledný. Na něm je velký vypínač. Nahmatal jsem diagnostický konektor a zasunul jsem nový

adaptér. Spustil jsem aplikaci *Torque* a marně jsem čekal na spojení. Zkontroloval jsem vizuálně adaptér a všechny kontrolky na něm byly zhasnuté. Stiskl jsem tedy vypínač a očekával jsem reakci. Nic se nestalo. Mačkal jsem tedy vypínač tak dlouho, než kontrolky rozsvítily. Pak se aplikace přeci jen s adaptérem propojila a přenos dat fungoval. Nebyl to ale přesvědčivý start.

Koncem listopadu chytá mého otce zpravidla amok a jakmile se objeví první prodejci vánočních stromků, on neváhá a startuje auto, aby se stal jejich prvním zákazníkem. Sice si doma vyměňujeme udivené pohledy, proč řeší v listopadu Vánoce, ale zase máme v obýváku vždycky krásný strom. Něco na tom asi bude. Zatímco táta se sestrou tancovali okolo stromečku, já jsem ležel na podlaze jeho nového Superbu a nechápal, proč se ten hloupý[31] adaptér nepřipojí a nepřipojí. Kontrolky nesvítily a spojení s aplikací jsem během 30 minut nezažehl ani jednou. Že by moderní vozy nespolupracovaly? Pro jistotu jsem tedy vytáhl Markův půjčený adaptér a zkusil spojení. Naběhlo okamžitě.

Zdálo se, že se adaptér po jednom jediném úspěšném pokusu odebral do věčných lovišť, což potvrdily i večerní testy v kamarádově VW Golfu, kde můj adaptér nefungoval, zatímco Markův modrý šlapal jako hodinky. Byl jsem otrávený a odhodlaný vadný kus reklamovat. Před tím jsem ale chtěl mít jistotu, a tak jsem vzal adaptér do hardwarové laboratoře Fakulty informačních technologií ČVUT.

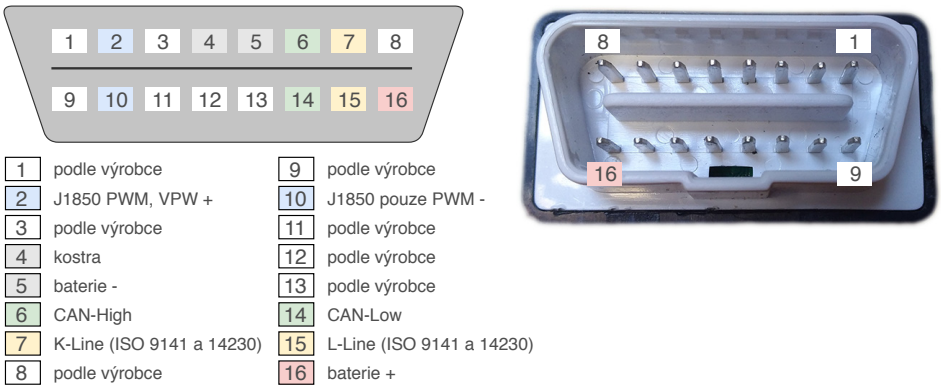
Den 39.

Objevil jsem, že k testování adaptéru není třeba reálné vozidlo, ale bohatě postačí zdroj napětí, který je schopen generovat 12 V [42]. Pokud připojím adaptér k napájení a konektory ke sběrnicím nechám nezapojené, adaptér je schopen se přes bluetooth standardně připojit k telefonu a reagovat na AT příkazy (popsáno v sekci 3.2.1). To bohatě stačí k ověření funkčnosti. Doma bohužel zdroj napětí nemám, ale v hardwarové laboratoři jich je k dispozici mnoho.

Podle obrázku 1.5 jsem na pin 16 zapojil zdroj napětí. Piny 4 a 5 jsem spojil se zemí, na zdroji napětí jsem nastavil 12 V a stiskl jsem vypínač na adaptéru. Nic se nestalo, to jsem očekával. Pro jistotu jsem stejné zapojení provedl i s modrým adaptérem od Marka. A opět nic! Rána pod pás! Začal jsem zběsile kontrolovat zapojení a na internetu dohledávat obrázky propojení. Pak se mi v hlavě rozsvítila žárovka. Všechny dostupné obrázky OBD-II portu zobrazují konektor typu samice. ELM327 má ale kolíky, je tedy nutné zapojovat všechno zrcadlově obráceně podle obrázku 4.3.

Jakmile jsem zapojil konektory zrcadlově, modrý adaptér se rozblikal. Pomocí aplikace Serial Bluetooth Terminal jsem mohl odesílat libovolné AT příkazy a přijímal jsem odpovědi. Funkční zapojení adaptéru na laboratorní zdroj napětí je na obrázku 4.4 vlevo. Vpravo je pak totéž zapojení vadného adaptéru

[31] V tu chvíli jsem měl na jazyku mnohem jadrnější výrazy.



Obrázek 4.3: OBD-II port samice vlevo a vpravo význam pinů na adaptéru ELM327

Viecar. Jak je na obrázku 4.4 vidět, kontrolky zůstaly i po stisku zapínacího tlačítka zhasnuté. Ještě ten den jsem odeslal čínskému obchodu *shopping-newfrog* reklamaci i s fotodokumentací z laboratoře.

A když už jsem na eBay byl, rovnou jsem koupil OBD-II prodlužovací kabel. Bude přeci jen příjemnější sledovat adaptér při komunikaci vsedě (jestli svítí správné kontrolky), než přitom lézt po špinavé zemi v autě. Po nedobrých zkušenostech s adaptérem za 95 Kč jsem se tentokrát rozhodl do své diplomové práce investovat třícifernou sumu a za kabel jsem dal 145 Kč[32]. To se mi vyplatilo, protože v následujících měsících jsem v souvislosti s kabelem žádné potíže nezaznamenal.

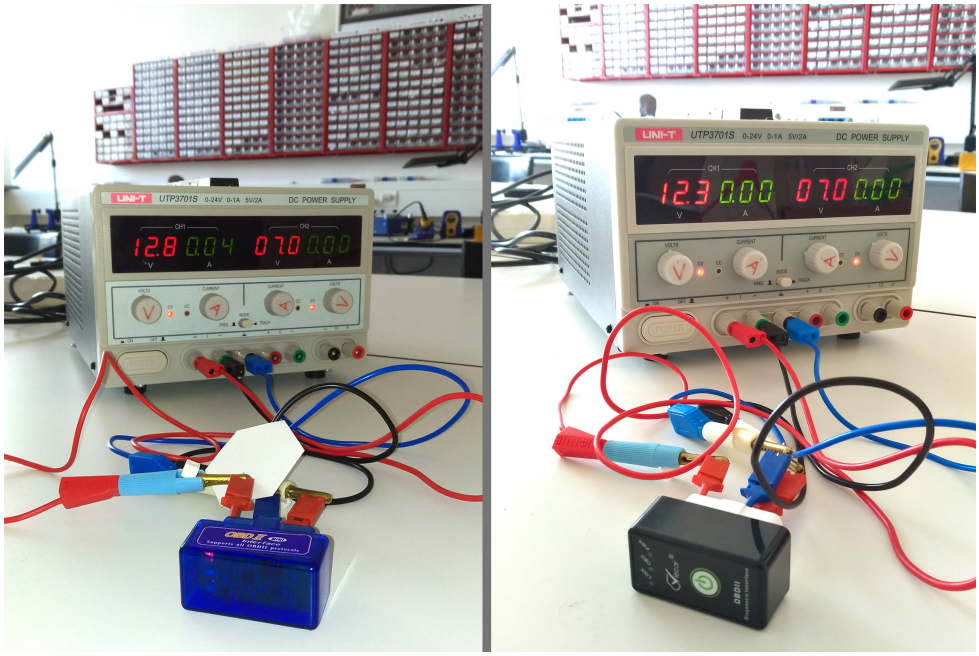
Den 45.

Vybaven modrým adaptérem ELM327 a aplikací Serial Bluetooth Terminal jsem se vydal do své Octavie na první testy. Zkoušel jsem odesílat AT příkazy, abych otestoval jestli komunikace vůbec funguje:

```
já odesílám      | AT RV     (požadavek na hodnotu napětí)
adaptér odpovídá | 12.1V>    (správná odpověď)
já odesílám      | AT 0105   (nesmyslný požadavek)
adaptér odpovídá | ?>        (správná odpověď)
```

Adaptér reagoval na příkazy přesně podle [34], a tak jsem se osmělil a zkusil komunikaci přímo s autem. Vycházel jsem z normy ISO 15031-5 [26] popsané v kapitole 3.4. Její část jsem si vytiskl asi na 10 stran A4 a v autě jsem z ní poté přepisoval příkazy do terminálu.

[32]Ceny v korunách závisí na aktuálním kurzu amerického dolaru. Většina čínských obchodníků nabízí své zboží za dolary.



Obrázek 4.4: Testování funkčnosti adaptérů ELM327 v hardwarové laboratoři

```
já odesílám   | 01 05        (teplota chladící kapaliny)
auto odpovídá | 410539>      (správná odpověď)
já odesílám   | 01 0C        (otáčky motoru)
auto odpovídá | 410C0E22>    (správná odpověď)
```

V prvním případě jsem se ptal na teplotu chladící kapaliny. Na tento požadavek jsem obdržel jsem 1 datový byte odpovědi: $A = 39_{hex} = 57_{dec}$. Vzorec v normě [26] udává, že teplotu chladící kapaliny spočítám následovně:

$$teplota = A - 40 = 57 - 40 = 17$$

Zjistil jsem, že teplota chladící kapaliny je 17 °C. To je možné, protože měření probíhalo v prosinci, venkovní teplota byla okolo 0 °C a já měření prováděl na parkovišti před domem. Motor byl přitom zcela studený. Měřil jsem jsem i další parametry, část měření je zachycena na snímku obrazovky (obrázek 4.5), který jsem pořídil při měření. Kompletní záznam z měření je v příloze této práce v sekci C.1 na straně 120.

Abych ten úspěšný den pěkně oslavil, vydal jsem se večer na online nákup na eBay. Vzhledem k tomu, že jediný funkční adaptér je půjčený, rozhodl jsem se pořídit další. Černobílý Viecar je v reklamaci a i kdybych dostal peníze zpět, nikdo mi ho asi nezprovozní. Rozhodl jsem se najít adaptér, který je nejpodobnější funkčnímu adaptéru od Marka. A pro změnu jsem se rozhodl být úsporný, vzal jsem tedy modrý *ELM327 V2.1 OBD2 CAN-BUS OBDII Bluetooth Car Auto Diagnostic Interface Scanner* za 69 Kč. Navrch jsem přihodil

```
10:17:26.127 Connecting to OBDII ...
10:17:26.561 Connected
10:17:40.727 AT @1
10:17:40.743 OBDII to RS232 Interpreter>10:17:52.690 AT RV
10:17:52.713 12.1V>10:17:59.045 ATRV
10:17:59.066 12.1V>10:18:26.978 AT@2
10:18:26.990 ?>10:18:34.463 AT @2
10:18:34.482 ?>10:18:57.722 AT0105
10:18:57.734 ?>10:18:59.485 AT0105
10:18:59.501 ?>10:19:03.678 AT 0105
10:19:03.686 ?>10:19:08.391 AT 01 05
10:19:08.408 ?>10:19:13.442 01 05
10:19:13.509 410539>10:19:31.901 0105
10:19:31.970 410539>10:20:44.821 0110
10:20:44.890 41100056>10:21:17.378 0133
10:21:17.450 413363>10:21:55.521 010C
10:21:55.588 410C0000>10:25:10.166 010C
10:25:10.227 410C0E22>10:25:31.360 010C
10:25:31.428 410C1356>10:26:09.854 0111
10:26:09.917 4111D8>10:26:55.633 0111
10:26:55.697 4111D8>10:27:42.367 012F
10:27:42.577 NO DATA>10:28:06.049 012F
10:28:06.261 NO DATA>10:28:38.826 011C
10:28:38.893 411C06>10:29:18.702 010C
10:29:18.778 410C0000>10:31:07.313 0100
```

Obrázek 4.5: Zjišťování dat z vozidla – měření prováděno 3. 12. 2017

sadu deseti propojovacích krokosvorek za 40 Kč. Budou se hodit při zapojování adaptéru do laboratorního zdroje napětí.

Den 66.

K pohodové atmosféře Vánočních svátků přispělo kladné vyřízení reklamace vadného adaptéru značky Viecar. Dostal jsem zpět všech 95 Kč, které jsem už před časem investoval do dalšího adaptéru. (Kam také jinam, že?) Pod stromečkem jsem našel malou elektrikářskou sadu. Obsahuje měřící šňůry zakončené závity a na ně je možné přišroubovat mnoho zakončení – jehlu, banánek, očko a hlavně krokosvorky (obrázek 4.6). Jeden z dárků, které mi udělaly opravdu velkou radost!

Den 80.

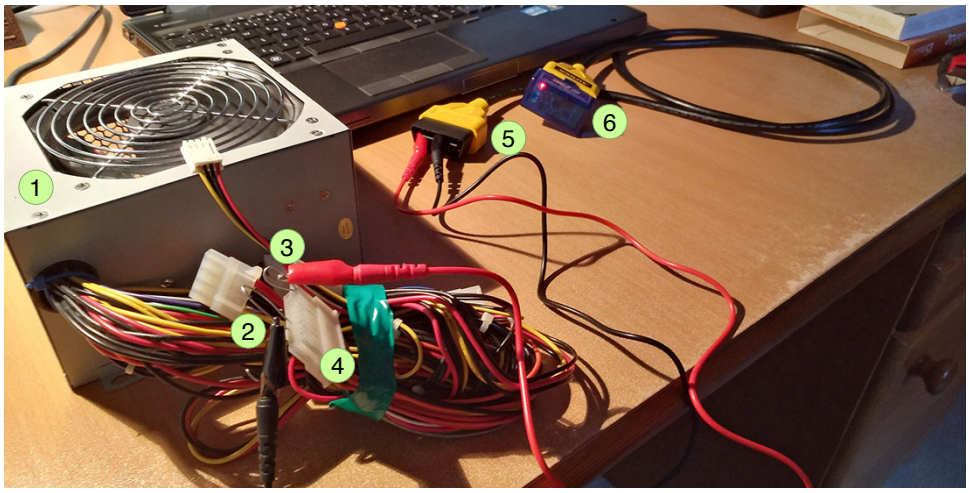
Byl to velký den. Podařilo se mi vyřešit dlouhodobý problém týkající se testování mojí mobilní aplikace. Když jsem doposud potřeboval otestovat spojení mezi adaptérem a mobilní aplikací, měl jsem dvě možnosti. Buď jít do hardwarové laboratoře a tam zapojit adaptér do laboratorního zdroje nebo jít do auta a zapojit adaptér do OBD-II portu. Jenže laboratoř je v Praze, kde nemám k dispozici auto na testování. Doma v Mladé Boleslavi jsem měl k dispozici auto, ale k němu na dvůr musím přes tři poschodí a patery dveře. Především jsem ale na dvoře neměl žádné zázemí. Není tam elektřina ani stůl, což je při vývoji čehokoliv velmi nepraktické. A ta zima. Sníh se objevil možná na jeden týden, zato mrazy byly poctivé až do března. Po 20 minutách testování jsem obvykle necítil prsty a musel jsem testování ukončit.

Obrázek 4.6: Měřící šňůry (vpravo) a k nim koncovky – zprava jehly, oko, banánek a krokosvorky

Kupovat laboratorní zdroj se mi nechtělo. Je to velké zařízení, které nemám kam uskladnit a pak bych pro něj neměl využití. Hledal jsem způsob, jak vytvořit zdroj 12 V napětí, až jsem si uvědomil, že mám hluboko v zásuvce počítačový zdroj z kdysi dávno rozebraného počítače po dědovi. Ten počítač byl tehdy dobrý už jen na hraní. Při práci se zasekával, výkon byl ve srovnání s dnešními telefony desetinový. Když jsem ho ale rušil, přišlo mi líto zbavovat se všeho, a tak jsem si z něj mnoho dílů nechal, že snad budou jednou k něčemu dobré. Ta chvíle právě nastala!

Každý počítačový zdroj (i ty 15 let staré), generuje stejnosměrné napětí 3,3 V, 5 V a 12 V. Problém je v tom, že dokud ho nepřipojíte k základové desce, tak se nerozjede. Je v něm pojistka. Dá se ale elegantním trikem ošálit. Zdroj obsahuje mnoho konektorů a mezi nimi i velký 20pinový konektor. Je-li v tomto konektoru propojen pin 14 se zemí, zdroj se spustí. Poté je na pinu 10 stejnosměrné napětí 12 V. Z jiných pinů 20pinového konektoru je možné čerpat i 3,3 V nebo 5 V, detaily zde již neuvádím. Zájemci si mohou detailní popis propojení i jiná využití počítačového zdroje dohledat ve velmi podrobném návodu v [43], odkud jsem čerpal.

Pomocí kabelů s krokosvorkami a OBD prodlužovacího kabelu jsem propojil adaptér ELM327 se zdrojem napětí. Dosáhl jsem tak stejného efektu, jako bych použil laboratorní zdroj napětí (například na obrázku 4.4). Celá sestava je zachycena na obrázku 4.7 včetně popisu důležitých částí. Od této chvíle jsem mohl spustit adaptér doma ve svém pokoji a mohl jsem testovat svoji Android aplikaci okamžitě bez nutnosti chodit do auta. Adaptér připojený do zdroje napětí je schopen komunikovat přes bluetooth a reagovat na AT příkazy (sekce 3.2.1).



1 napájecí zdroj do počítače
2 propojené piny 14 a 15 (0 V)
3 pin 10: 12 V
4 20pinový konektor
5 prodlužovací OBD kabel
6 adaptér ELM327

Obrázek 4.7: Zapojení adaptéru ELM327 do napájecího zdroje z počítače

Den 92.

Z Číny dorazil třetí adaptér ELM327. Je velmi podobný Markovu adaptéru. Rozměry má stejné, kryt je také modrý, ale neprůhledný. Otestoval jsem ho jak na domácím zdroji, tak v autě s pomocí aplikace Serial Bluetooth Terminal a funguje správně. Přitom jsem si vzpomněl na rozbitý adaptér Viecar. Byl nefunkční a po úspěšné reklamaci už jsem nemohl nic ztratit. Rozhodl jsem se tedy adaptér rozebrat a pokusit se ho opravit.

Sundal jsem plastový kryt. Adaptér se uvnitř skládá ze dvou desek, jež jsou osazeny mnoha elektronickými součástkami. Na horní desce je vypínací tlačítko. Připojil jsem adaptér na zdroj a stiskl jsem tlačítko. LED diody se rozblikaly! Po chvíli se mi podařilo spojit se s mobilní aplikací a komunikovat

Obrázek 4.8: Adaptér Viecar – vlevo originální, uprostřed bez obalu, vpravo s vyvrtanou dírou na tlačítko

pomocí AT příkazů. Závada byla pouze v tom, že kryt byl příliš tuhý a vypínací tlačítko přes něj nešlo stisknout. Vzal jsem vrtačku a do plastového krytu jsem vyvrtal díru. Vrtal jsem na nízké otáčky, takže při tom kryt dokonce ani nepraskl. Okraje otvoru jsem začistil hrubým smirkovým papírem. Za půl hodiny jsem byl hotový.

Dále jsem objevil, že horní deska je uchycena z jedné strany pouze na jediném pinu, který navíc není zcela pevný. S horní deskou bylo možné pohybovat nahoru a dolů v rozmezí asi 2 mm. Při neopatrné manipulaci by se snadno mohly přerušit další podpůrné piny, a proto jsem se rozhodl horní desku podepřít sirkou. Na práci s tavnou pistolí ale nejsem odborník, navíc mám neohrabané prsty, a tak jsem požádal svoji sestru. Ta úspěšně absolvovala kurz *Dovedné ruce* — lepšího odborníka jsem si nemohl přát. S pomocí pinzety sirku z obou stran přilepila k deskám tavným lepidlem. Výsledek našeho kutilského odpoledne je na obrázku 4.8. Vlevo je původní výrobek, uprostřed adaptér bez obalu již vyztužen sirkou a vpravo adaptér s upraveným obalem.

Od té chvíle jsem měl celkem 3 funkční adaptéry ELM327, jejichž parametry jsou shrnuty v tabulce 4.1 a všechny tři jsou jsou zobrazeny na obrázku 4.9. Vlevo je adaptér od Marka, uprostřed opravený adaptér Viecar a vpravo adaptér Tom.

Tabulka 4.1: Přehled adaptérů ELM327 použitých při realizaci

pojmenování	*kryt*	*datum pořízení*
Marek	modrý průhledný	20. 10. 2017
Viecar	černo-bílý neprůhledný	24. 11. 2017
Tom	modrý neprůhledný	19. 1. 2018

Obrázek 4.9: Adaptéry ELM327 použité při realizaci – zleva Marek, Viecar a Tom

Shrnutí

Sehnal jsem celkem 3 adaptéry pro komunikaci s vozidlem. Všechny 3 jsou schopné komunikovat s aplikací *Serial Bluetooth Terminal* pomocí AT příkazů a všechny jsou schopné číst údaje z vozidla podle normy ISO 15765-4. To jsem testoval ve voze Škoda Octavia II. generace a Škoda Superb III. generace. Navíc se mi oproti původním předpokladům podařilo vytvořit vhodné prostředí pro testování mobilní aplikace v domácích podmínkách.

4.2 Tvorba aplikace OBD Robot

Cílem tohoto tématického celku je vytvoření aplikace pro mobilní operační systém Android, která je schopná komunikovat s adaptérem ELM327. Ve spolupráci s tímto adaptérem je aplikace nejen schopná číst data z vozidla, ale také odesílat příkazy, kterými je schopná vozidlo částečně ovládat.

K vývoji aplikace **OBD Robot** jsem používal program Android Studio [44]. Při tvorbě aplikace jsem zpočátku postupoval podle návodu na webových stránkách *Lemberg Solutions Blog* [39]. Autor návodu Ruslan Yanchysin využívá při tvorbě aplikace softwarovou knihovnu *OBD-Java-API Library* [45], která je k dispozici zdarma pod licencí Apache. To znamená, že zdrojový kód je možné volně kopírovat s uvedením autora. Knihovna usnadňuje odesílání a zpracování příkazů především důkladným ošetřením chyb a nezbytnými konverzemi, které mi značně usnadnily práci s příkazy. Knihovna například při čtení příkazu čeká na povinný ukončovací znak `>`, který adaptér ELM327 musí poslat za každou odpovědí a naopak automaticky přidá byte s hodnotou $\texttt{OD}_{hex}$, který dává adaptéru informaci, že zpráva od uživatele končí.

Při vývoji jsem se kromě výše zmíněných zdrojů inspiroval i aplikací *Android OBD-II Reader* od Paula Pirese, jejíž zdrojové kódy jsou volně k dispozici [46]. Aplikace také využívá knihovnu *OBD-Java-API Library* [45] a na jediné obrazovce zobrazuje souhrnně přibližně 20 – 30 údajů z vozidla, které získává přes bluetooth pomocí OBD-II adaptéru.

Aplikaci jsem vytvářel podle obrázku 3.5 na straně 51 v sekci 3.3. Její realizaci jsem rozdělil na několik základních bloků:

- vytvoření spojení přes bluetooth,
- vytvoření terminálové obrazovky, která umí přijímat a odesílat příkazy,
- rozdělení aplikace na terminál (libovolná komunikace) a aktuální data (jen čtení dat),
- vytvoření obrazovky s aktuálními daty o vozidle.

Den 40.

Po instalaci Android Studia jsem vytvořil prázdný projekt a začal jsem postupně vytvářet aplikaci krok po kroku podle návodu v [39]. Vytvořil jsem 2 základní aktivity[33]. První z nich kontroluje zapnutý bluetooth a zobrazí seznam párovaných zařízení s mobilním telefonem. Ve druhé aktivitě jsem programoval terminál pro komunikaci s adaptérem ELM327.

```
btnPaired.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        pairedDevicesList();
    }
});
```

Výše uvedený kód zajišťuje obsluho tlačítka *Ukaž párovaná zařízení*. Po kliku na něj volám metodu `pairedDevicesList()`, která zjistí všechna párovaná zařízení s telefonem, vytvoří z nich seznam a ten vypíše na obrazovku telefonu. Každá z položek obsahuje jméno párovaného zařízení a jeho MAC adresu[34]. Na obrázku 4.10 je vidět, jak vypadá rozbalený seznam párovaných zařízení.

Po kliku na položku v seznamu se zavolá metoda `myListClickListener()`, která zahájí bluetooth komunikaci s vybraným zařízením. Pokud zařízení nepodporuje bluetooth profil SPP[35], propojení se nezdaří. V opačném případě se otevře komunikační kanál bez další kontroly, jestli se jedná o OBD-II adaptér.

Tato část práce šla poměrně rychle, bez větších odchylek jsem se držel návodu. Problém nastal ve chvíli, kdy jsem chtěl importovat knihovnu *OBD-Java-API Library*. Pouhé přidání do projektu nestačilo. Když jsem chtěl použít třídy z knihovny, překladač hlásil chybu, že použité třídy nezná. Zjistil jsem, že je třeba z knihovny vytvořit `JAR` archiv a ten do projektu přidat. (Projekt $\longrightarrow$ Externí knihovny $\longrightarrow$ vložím knihovnu. Poté kliknu na knihovnu pravým tlačítkem myši a zvolím *Přidat jako knihovnu*.)

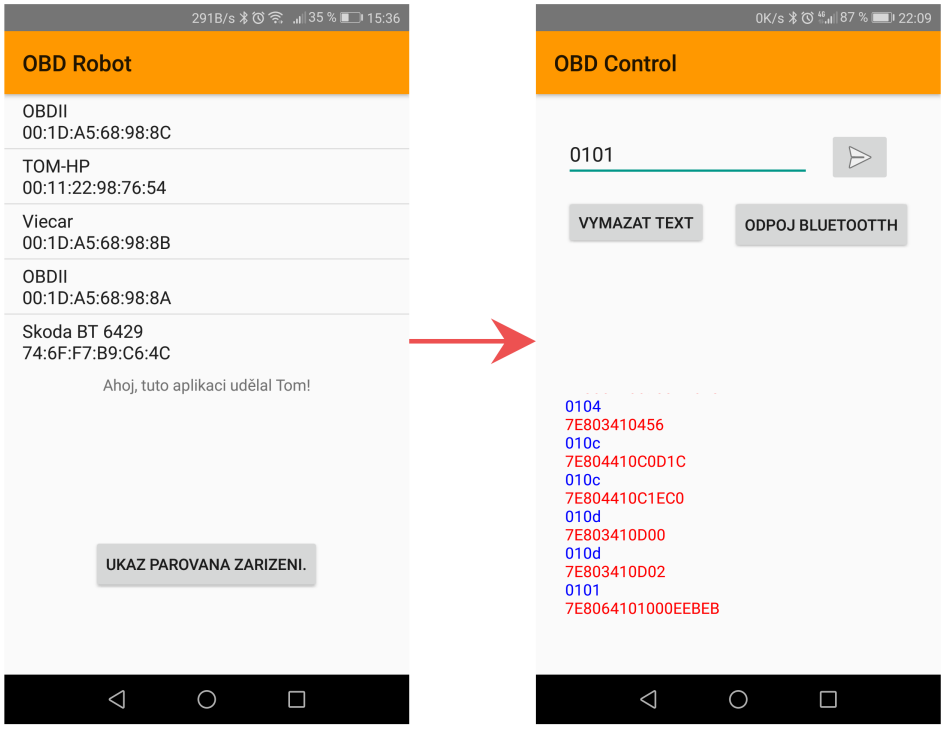
Den 80.

Vývoj mi šel ztuha, kvůli testování i těch nejmenších detailů jsem musel vždy do auta, což mě odrazovalo od častých testů. V důsledku toho jsem ale zase déle hledal chyby, protože jsem je musel hledat ve větším objemu odvedené práce. Naštěstí se mi ten den podařilo vytvořit testovací prostředí, díky němuž jsem mohl i doma v pokoji testovat funkčnost aplikace. Detailně to popisuji v sekci 4.1 na straně 68. Pak už jsem až do konce práce používal při vývoji stále stejný scénář:

[33] Aktivitou se v prostředí Androidu myslí jedna obrazovka.

[34] MAC adresa je jednoznačný identifikátor libovolného zařízení, které komunikuje v síti (tedy i přes bluetooth) [14].

[35] O bluetooth profilech pojednává sekce 3.3 na straně 49.

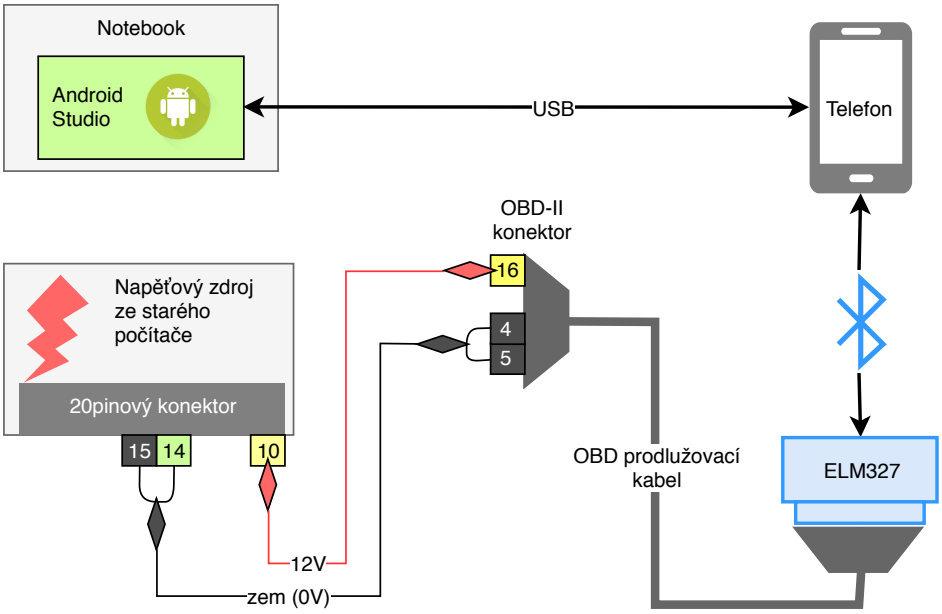


Obrázek 4.10: OBD Robot – raná vývojová verze – vlevo seznam párovaných zařízení a vpravo obrazovka terminál

1. Implementoval jsem nějakou funkcionalitu.
2. Připojil jsem mobilní telefon k notebooku přes USB kabel a zapnul bluetooth.
3. Připojil jsem adaptér ELM327 k napěťovému zdroji.
4. Exportoval jsem novou verzi aplikace přímo do mobilního telefonu.
5. Ihned jsem aplikaci v telefonu spustil a testoval jsem, jak funguje.

Na obrázku 4.11 jsou znázorněna všechna propojení (jak drátová tak bezdrátová), nezbytná k tomu, aby výše uvedený scénář vývoje korektně fungoval. Potřeboval jsem notebook, USB kabel, telefon, adaptér ELM327, OBD prodlužovací kabel, propojovací vodiče s krokosvorkami a napěťový zdroj. Aby se zdroj spustil, musí být pin 14 ve dvacetibitovém konektoru spojen se zemí (tu představuje pin 15).

Při testování jsem odhalil, že příkazy, na které má přijít chybová odpověď `NO DATA` tuto odpověď negenerují. Místo toho se aplikace zasekávala. Zjistil jsem, že problém je v použité knihovně pro zpracování OBD příkazů ve třídě



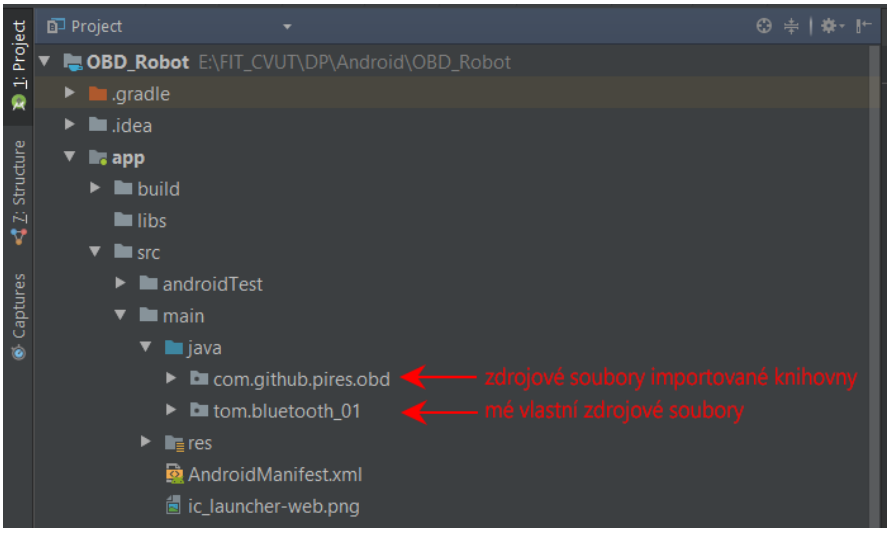
Obrázek 4.11: Propojení všech komponent při vývoji aplikace OBD Robot

`ObdCommand` a v její metodě `readResult(InputStream in)`. Zde se volala funkce `checkForErrors()`, která v případě zjištění chyby vyhodila výjimku. Tu bylo třeba ošetřit, což jsem já nedělal. U mě ale výskyt chyby není problém, prostě jsem chtěl, aby se chybová hláška vypsala do terminálu stejně jako kladná odpověď. Vyřešil jsem to jednoduše – zcela jsem odstranil volání funkce `checkForErrors()` z metody `readResult(...)`.

To se však ukázalo značně problematické. Knihovna importovaná do projektu jako `JAR` archiv není určená k editaci. Musel jsem tedy přistoupit k jinému způsobu vložení knihovny do projektu. Vytvořil jsem ve svém projektu novou složku s názvem `com.github.pires.obd`. Stáhl jsem knihovnu z [45] ve formátu `ZIP` a všechny obsažené zdrojové soubory jsem vložil do nově vytvořené složky (viz obrázek 4.12). Později jsem musel při použití libovolné funkce z knihovny importovat ručně třídu, ve které se nachází, ale po přeložení program fungoval správně.

Dále jsem v obrazovce terminál nastavil pro lepší orientaci barvu odesílaných dat na modrou a barvu přijímaných dat na červenou. Vylepšil jsem grafiku aplikace, především rozmístění elementů na obrazovce, nastavil jsem oranžové barevné schéma a přidal jsem ikonu aplikace.

ikona aplikace *OBD Robot*



Obrázek 4.12: Způsob vložení knihovny *OBD-Java-API Library* do projektu

Na závěr jsem si nechal to nejdůležitější – přejmenování aplikace. Z provizorního názvu *Bluetooth01* se právě zrodil *OBD Robot*. Název sice není příliš originální, ale je stručný a je z něj jasné, k čemu aplikace slouží.

Den 84.

Nové malé SUV[36] Škoda Karoq se začalo prodávat v říjnu roku 2017. Necelé tři měsíce poté jsem měl to štěstí nejen se na něj podívat, ale provést na něm i první zátěžové testy mojí aplikace. Propojení vozidla s aplikací se zdařilo. Pomocí terminálu jsem odesílal požadavky na čtení aktuálních údajů z jízdy. Podařilo se číst například rychlost, otáčky motoru i okolní teplotu a jejich hodnoty porovnat s hodnotami zobrazenými na ukazatelích v autě. Vizuálně jsem ověřil správnost. Zprávy z vozidla jsem četl v syrovém stavu a hodnoty parametrů jsem musel zatím dopočítávat ručně podle normy [4]. Vzhledem k tomu, že aplikace neměla v té době implementovaný export zaznamenaných dat do souboru, přikládám jako doklad o měření snímky obrazovky. Téměř kompletní záznam z měření je v příloze této práce v sekci C.2 na straně 122.

V tu chvíli jsem si říkal, že by se export dat do textového souboru velice hodil, a tak jsem se rozhodl, že ho do aplikace přidám. To jsem ještě netušil, že mi přidání této funkce zabere téměř celý den.

Den 123.

O Vánocích a zkouškovém období jsem diplomovou práci poslal na vedlejší kolej a vrátil se k ní až na začátku letního semestru. Přivítání s Android

[36] SUV – sportovně užitkové vozy

Studiem nebylo zcela vřelé, vývojové prostředí je podle mého názoru docela nepřehledné a první desítky minut jsem opět hledal kam kliknout, abych našel zdrojové soubory, abych si zobrazil náhledy obrazovek nebo abych si našel definici nějaké funkce.

Zabýval jsem se ukládáním zpráv do textových souborů. Vytvořil jsem si pro tento účel funkci `saveLogToFile()`. Dlouhou dobu mi trvalo pochopit, že aplikace v Androidu může ukládat data do svého privátního prostoru a cokoliv jiného je pro ni externí úložiště. Do privátního prostoru aplikace jsem data ukládat nemohl, protože tam uložená data jsou čitelná jen pro aplikaci, která je vytvořila. Já ale potřebuji uložený soubor nalézt pomocí správce souborů a být schopen ho zkopírovat do počítače. Je matoucí, že pro tento účel musím ukládat do externího úložiště, přestože se fyzicky jedná o interní zabudovanou flash paměť mobilního telefonu a nikoliv o vyjímatelnou SD kartu. Řekne-li se externí úložiště, běžný uživatel Androidu si představí SD kartu nebo *cloudovou* službu[37], což se stalo i mně.

Po mnoha hodinách jsem nakonec s pomocí různých zdrojů a především online příručky *Android Developers* [47] problém vyřešil. Zdrojový kód funkce `saveLogToFile()` je uveden níže. Není kompletní a z důvodu lepší čitelnosti je zkrácen. Úplné zdrojové kódy jsou na přiloženém CD. Komentáře ke kódu následují hned pod ním.

```
01: String state = Environment.getExternalStorageState();
02:    if (Environment.MEDIA_MOUNTED.equals(state)) {
03:        File root = Environment.getExternalStorageDirectory();
04:        String PATH_NAME = root.getAbsolutePath() + "/OBD_Robot";
...
05:        if (!dir.exists()) {
...
06:        }
07:
08:        File file = new File(dir, LOG_FILE_NAME);
09:        FileOutputStream fos = new FileOutputStream(file);
10:
11:            int listSize = clog.size();
12:            for (int i = 0; i < listSize; i++) {
13:                fos.write(clog.get(i).toString().getBytes());
14:            }
...
15:    } else {
16:        msg("Error: Externi uloziste je nedostupne.");
17:    }
```

- Řádek 1 – Zjistím stav externího úložiště.
- Řádky 2, 15, 16, 17 – Zkontroluji dostupnost externího úložiště. Pokud není dostupné, vypíšu chybovou hlášku.

[37] Velké internetové firmy jako Google nebo Microsoft poskytují online úložiště, kam je možné pomocí aplikací nahrávat obsah přímo z mobilního telefonu. Takovým úložištím se lidově říká *cloud*.

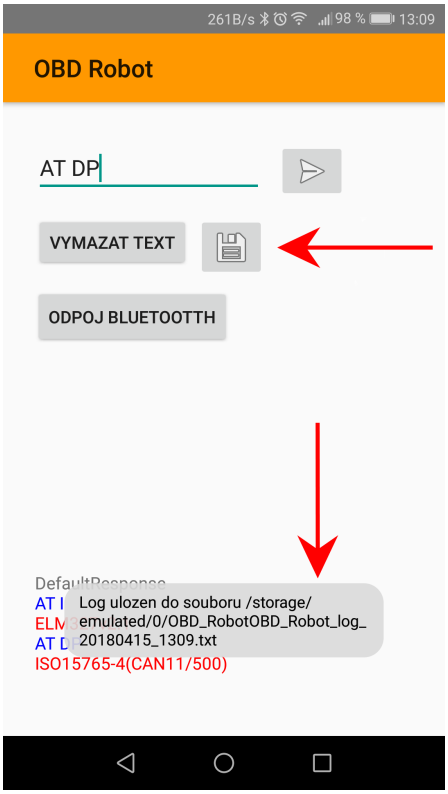
- Řádky 3, 4 – Zjistím adresu kořenového adresáře a vytvořím řetězec reprezentující vlastní adresář s názvem `OBD_Robot`.

- Řádky 5, 6 – Kontroluji existenci vlastního adresáře. Pokud adresář neexistuje, pokusím se ho vytvořit. Když operace selže, ukončím činnost.

- Řádky 8, 9 – Vytvořím zápisový *stream* `fos`.

- Řádky 11, 12, 13, 14 – Po jednom zapíšu do souboru prostřednictvím *streamu* `fos` řádky příchozích i odchozí zpráv, které se zobrazují v terminálu.

S výše uvedeným kódem aplikace funguje, ale až poté, co jsem jí explicitně ve správci aplikací systému Android povolil ukládání souborů do úložiště telefonu. Na obrázku 4.13 je červenou šipkou zvýrazněna ikona, pomocí níž lze záznamy uložit. Dole je šipkou vyznačen potvrzovací dialog, který signalizuje úspěšné uložení.



Obrázek 4.13: OBD Robot – Ukládání záznamů do souboru

Den 161.

Měl jsem už poměrně dobře zvládnutou obrazovku s terminálem a chtěl jsem přidat obrazovku, kde se budou automaticky zobrazovat aktuální data o jízdě. Vytvořil jsem tedy v Android Studiu novou aktivitu a do ní jsem vložil element `ListView`. Funkce `updateParametersList3`[38] do tohoto elementu přiřazovala položky. Cyklicky odesílala žádosti o data a zpracovávala odpovědi. Vytvořil jsem jednoduché menu, v němž byly pouze 2 položky – Terminál a Aktuální data. Pomocí menu jsem se přepínal mezi dvěma obrazovkami.

Při přepínání z jedné obrazovky do druhé jsem ale pozoroval mírné zasekávání aplikace a zjistil jsem, že aplikace vždy ukončí a znovu obnoví bluetooth spojení s adaptérem. Kvůli tomu mizela v terminálu historie odeslaných příkazů a odpovědí. Aplikace sice fungovala, ale z uživatelského hlediska byla velice nepřívětivá. Sjednal jsem si proto schůzku s odborníkem na tvorbu Android aplikací Tomášem Krabačem [48].

Vysvětlil mi, že původní aktivita se s odchodem na novou aktivitu automaticky ukončí, přestože je součástí jedné aplikace. Ke změně obsahu na obrazovce není nutné používat novou aktivitu, ale je vhodné použít takzvaný *frame*[39] (rámec). Jedna aktivita v sobě může sdružovat více rámců, které podle přání uživatele zobrazuje nebo skrývá.

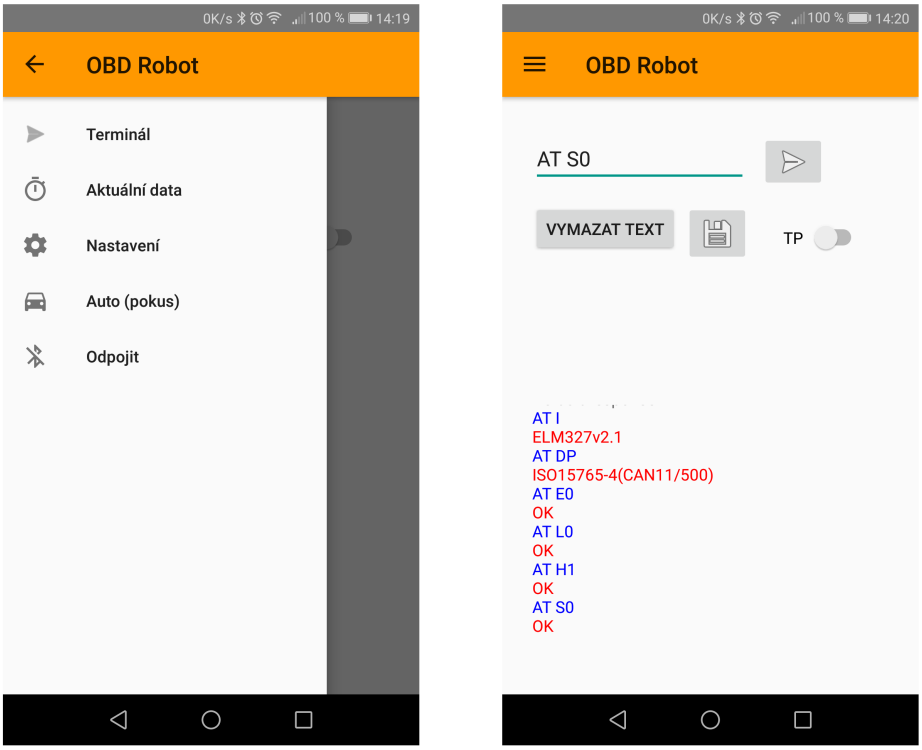
Přepsali jsme společně aplikaci a od té doby obsahuje 2 aktivity:

- seznam párovaných zařízení – zobrazí se při spuštění aplikace
- hlavní aktivita – je spuštěná, když je aplikace připojená k OBD-II adaptéru

Do hlavní aktivity se vkládají rámce Terminál a Aktuální data, která se zobrazují podle toho, co uživatel stiskne v menu. Už se při přepnutí neukončí spojení a uživatel se může libovolně přepínat mezi těmito obrazovkami, aniž by se mu z jedné z nich ztratila data. Tlačítko, ukončující bluetooth spojení jsme společně přesunuli z rámce Terminál do menu. Tam ho uživatel má přístupnější. Každý uživatel může upřednostňovat práci s jinou obrazovkou a v případě potřeby ukončit spojení je to z obou do menu na jeden dotyk. Na obrázku 4.14 vlevo je vidět rozbalené menu aplikace. Jedná se o ranou verzi softwaru, a tak jsou v menu i položky, které se nakonec do finální verze aplikace nedostaly. Vpravo na obrázku 4.14 je téměř finální verze obrazovky Terminál (nyní již implementované jako frame).

[38] Nyní tento úkol zastává funkce `onMessageEvent(CommandsEvent event)`. Detaily se nachází pod nadpisem „Den 183.“ v sekci 4.2 na straně 81.

[39] Pozor, *frame* (rámec) v Androidu je zcela něco jiného než *frame* (rámec) CAN zprávy používaný v sekci 1.4.3.



Obrázek 4.14: OBD Robot – Zobrazení rozbaleného menu vlevo a obrazovka terminál vpravo

Den 170.

Dokončoval jsem práce na obrazovce s aktuálními daty. Čtení dat ve vozidle probíhalo vcelku dobře, problém nastal při testování s adaptérem ELM327 na zdroji napětí. Bez připojeného vozidla se nedostaví na žádný dotaz odpověď, a tak by aplikace měla zobrazovat u všech hodnot buď nuly nebo informaci o tom, že data jsou nedostupná. Místo toho se však aplikace zasekávala. Po krátkém bádání jsem zjistil, že mnou používaná softwarová knihovna [45] využívá při zpracování odpovědi funkci `performCalculations()`, která odstraňuje z odpovědi všechny následující textové řetězce:

```
"\\s"                  (mezery, tabulátory)
"BUS INIT", "BUSINIT"  (inicializace sběrnice)
"\\."                  (nový řádek)
"SEARCHING"            (vyhledávání komunikačního protokolu)

"UNABLETOCONNECT"      (není možné spojení s~autem)
```

Takové řetězce může posílat adaptér v různých situacích a tyto řetězce se mohou přimíchat do správné odpovědi od vozidla. Je nutné je odstranit, aby knihovna mohla provést správný přepočet přijatých bytů na požadovanou hodnotu některého parametru. Poslední řetězec `UNABLETOCONNECT` v původní verzi nebyl. Posílá ho adaptér ve chvíli, kdy nemá spojení s autem, což byl právě případ při testování se zdrojem napětí. Ve chvíli, kdy se řetězec `UNABLETOCONNECT` měl interpretovat jako hodnota nějakého parametru, spadl program do ošetřené výjimky, protože řetězec obsahoval jiné než hexadecimální znaky (A – F, 0 – 9). Poté, co jsem přidal řetězec `UNABLETOCONNECT` mezi slova určená k odstranění, program se přestal zasekávat.

Zamířil jsem tedy hrdě do auta a aplikaci jsem ihned vyzkoušel. Zjistil jsem, že hodnota motor se točí rychlostí asi 4000 otáček za minutu, což byl nesmysl. Auto stálo v klidu a motor běžel na volnoběh. I ostatní údaje byly zjevně nesmyslné. Naštěstí jsem rychle věděl, kde může být problém. Po každém vysunutí z OBD-II konektoru se adaptér ELM327 resetuje do výchozího nastavení. To při měření způsobilo, že adaptér ELM327 zprávy od auta formátoval jinak, než očekávala knihovna *OBD-Java-API Library*. Postupným experimentováním jsem zjistil, že aby knihovna dobře počítala hodnoty parametrů, je nutné mít v adaptéru vypnuté zobrazování CAN identifikátorů, vypnuté vkládání mezer a vypnuté echo (tedy opakování zadaného požadavku). Abych toho docílil, přidal jsem před periodické zjišťování hodnot jednorázovou funkci, která vše potřebné pomocí AT příkazů nastaví:

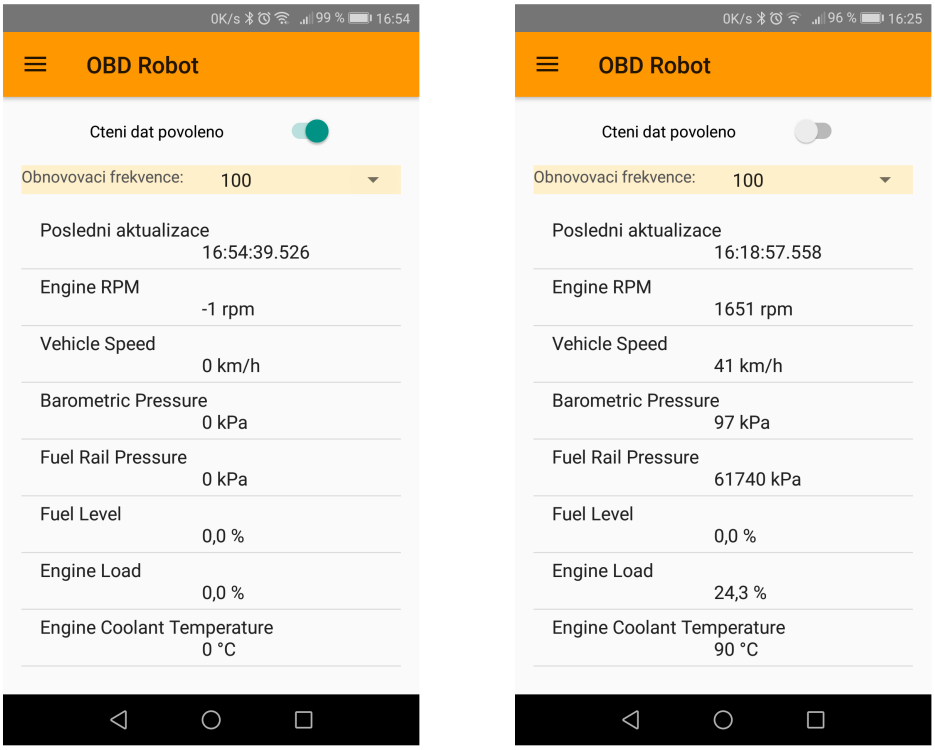
```
private void initAdapter(){
    sendOBDCommand("AT H0");    // CAN hlavičky vypnout
    sendOBDCommand("AT E0");    // echo vypnout
    sendOBDCommand("AT S0");    // mezery vypnout
}
```

Při dalším testu jsem již v autě viděl správné hodnoty. Na obrázku 4.15 je vlevo vidět, co se na obrazovce aktuální data zobrazí, když je adaptér ELM327 připojen ke zdroji napětí. Tedy nemá s autem fyzické spojení. Na obrázku 4.15 vpravo jsou aktuální data a jejich hodnoty, jak jsem je získal při správném zapojení ve vozidle (zapnutý motor). Ještě se nejedná o finální verzi aplikace, proto jsou na obrázku 4.15 názvy parametrů v anglickém jazyce převzaté přímo z použité knihovny.

Den 183.

Poté, co jsem na obrazovku s aktuálními daty přidal sledování více než 5 parametrů současně, začal jsem pozorovat zasekávání celé aplikace. Při 10 parametrech už byla aplikace citelně zpomalena. Požádal jsem proto o pomoc Tomáše Krabače, aby mi pomohl celou situaci objasnit [49].

Programy a aplikace mohou využívat pro svůj běh více vláken. Má-li počítač více procesorových jader, mohou být vlákna vykonávána pomocí různých



Obrázek 4.15: OBD Robot – Zobrazení obrazovky s aktuálními daty – vlevo bez připojení k autu a vpravo s připojením k autu

procesorových jader současně. Aplikace *OBD Robot* běžela do té doby pouze na jediném vlákně. Proto se se zvyšujícím počtem parametrů snižovala odezva na příkazy od uživatele.

S využitím knihovny *EventBus* [50] a pomocí Tomáše Krabače jsem přepsal aplikaci tak, že hlavní vlákno zajišťuje pouze výpis a obnovení dat. Aktuální hodnoty parametrů z vozidla zjišťuje funkce `onMessageEvent(CommandsEvent event)` běžící ve vedlejším vlákně. Když funkce zajistí aktuální hodnoty, pošle je jako seznam řetězců do hlavního vlákna a vytvoří speciální událost. Na událost reaguje funkce `void onMessageEvent(UpdateData event)`, která aktualizuje data na obrazovce. Pokud je spuštěné získávání parametrů, opět se aktivuje zjišťování aktuálních hodnot ve vedlejším vlákně.

Shrnutí

Podle zadání práce jsem měl vytvořit aplikaci pro Android, která dokáže komunikovat s vozidlem, číst data z vozidla a také příkazy do vozidla odesílat. To jsem splnil. Aplikace OBD Robot zobrazí uživateli po připojení k OBD-II adaptéru dvě možné obrazovky – Terminál a Aktuální data. V terminálu

je možné libovolná komunikace s vozidlem, uživatel ale musí znát správné příkazy. O tom, jaké příkazy posílat, pojednává sekce 4.3. Na obrazovce Aktuální data může sledovat uživatel hodnoty mnoha vybraných parametrů, aniž by musel příkazy znát.

Nad rámec zadání se mi podařilo vyrobit aplikaci s velmi čistým, srozumitelným a přehledným uživatelským rozhraním. Umí navíc exportovat záznam komunikace do textového souboru, s čímž jsem v návrhu původně nepočítal. Z analýzy dostupných aplikací v kapitole 2 vyplynulo, že v oficiálním obchodě Google Play není ani jedna aplikace, která by kombinovala libovolné odesílání příkazů pomocí terminálu a zobrazovaní aktuálních dat z vozidla. Díky této kombinaci je aplikace OBD Robot unikátní.

Návod k instalaci a obsluze aplikace OBD Robot je v příloze D této práce na straně 127. Zde je také krátká sekce s řešením nejběžnějších potíží.

4.3 Odhalování interní komunikace ve vozidle

Cílem tohoto tématického celku je nalezení příkazů pro ovládání vozidla pomocí adaptéru ELM327 a aplikace v mobilním telefonu. Součástí tématického celku je i nalezení správného postupu, který musí být dodržen, aby vozidlo na příkazy správně reagovalo.

Protože jsem měl pro testování k dispozici pouze vozidla koncernu Volkswagen, odvíjel se postup práce i výsledky od možností, které tyto vozy nabízí. Postup i metody odhalování komunikace jsem zákonitě přizpůsoboval tak, abych dosáhl výsledků na automobilech, které jsem měl k dispozici. A je proto zcela logické, že někdo jiný, kdo by měl k dispozici jiné automobily, by mohl dosáhnout s použitím mého postupu výsledků zcela jiných.

Den 75.

Podle návrhu v sekci 3.5 jsem se rozhodl nejprve provést monitoring sběrnice – takzvaný *sniffing*, abych získal nějaká počáteční data. Následně jsem měl v plánu data analyzovat a extrahovat z nich jednotlivé zprávy. Proto jsem důkladně pročítal manuál k mikrokontroléru ELM327 [34] a hledal jsem v něm stopy postupu, jak takový monitoring s pomocí adaptéru ELM327 provést.

V manuálu [34] jsem na straně 21 nalezl příkaz `AT MA`, který je popsán doslova takto: „Tento příkaz nastaví ELM327 do režimu monitoringu sběrnice, ve kterém kontinuálně monitoruje a zobrazuje všechny zprávy na OBD sběrnici. Pro zastavení monitoringu jednoduše odešlete do ELM327 jakýkoliv znak a poté čekejte na odpověď v podobě znaku `>`.“

Byl jsem nadšený, že jsem našel, co jsem hledal. Ihned jsem zašel do auta, vložil jsem adaptér do OBD portu. Spustil jsem aplikaci *Serial Bluetooth Terminal* a napsal jsem do příkazového řádku `AT MA`. Nestalo se nic. Čekal jsem

asi minutu, ale nezobrazilo se ani písmenko. Přitom motor byl spuštěný, takže nějaká komunikace na sběrnici jistě probíhala. Příčiny mohly být následující:

1. špatně nastavený adaptér
2. nefunkční adaptér
3. problém v autě (dělám něco, co auto nepodporuje)

Vrátil jsem se tedy zklamaně zpět ke studiu manuálu k ELM327 a hledal jsem další nastavení, která mohou ovlivňovat monitoring sběrnice. Objevil jsem příkazy, které mohou monitoring ovlivnit. Jsou shrnuté v tabulce 4.2. Mohlo se stát, že jsem měl špatně nastavenou masku a ELM327 mi jen zprávy nezobrazovalo.

Tabulka 4.2: AT příkazy, které mohou ovlivnit monitoring sběrnice

příkaz	*vysvětlení*
`AT MA`	spustí monitoring sběrnice
`AT AL`	povolí příjem dlouhých zpráv (delších než 8 bytů)
`AT CF XXX`	nastaví filtr; XXX jsou 3 hexadecimální cifry; když přijde zpráva, ELM327 ji zpracuje jen tehdy, pokud CAN ID přijaté zprávy odpovídá filtru u bitů, kde je maska nastavená na `'1'`
`AT CM XXX`	nastaví masku; XXX jsou 3 hexadecimální cifry; tam, kde jsou bity masky nastaveny na `'1'`, se porovnává CAN ID s nastaveným filtrem

Před dalším pokusem jsem příkazem `AT CM 000` nastavil masku na samé nuly, aby se nic s filtrem neporovnávalo. Povolil jsem příkazem `AT AL` příjem dlouhých zpráv a spustil jsem monitoring. Bohužel se stále nic nedělo. Na displeji mobilního telefonu se nevypsala ani jedna zachycená zpráva.

Mohl jsem mít tedy vadný adaptér, ale pro jiné příkazy fungoval dobře. Rozhodl jsem se tedy, že si sjednám schůzku s odborníkem na CAN komunikaci z firmy Digiteq Automotive Slavomírem Mikuleckým. Firma Digiteq Automotive je dodavatelem elektronických komponent do automobilů značky Škoda Auto a Volkswagen. V plánu bylo probrat můj přístup k monitoringu sběrnice a zjistit nějaké informace o tom, jak probíhá komunikace na CAN sběrnicích ve vozidlech Škoda.

Den 97.

Firma Digiteq Automotive má 3 pobočky – v Praze, Mladé Boleslavi a Plzni. V době, kdy jsem psal diplomovou práci, jsem byl v této firmě zaměstnán na částečný úvazek a objevil jsem zde plno lidí, kteří mi předali cenné informace

o tom, jak probíhá komunikace na CAN sběrnici ve vozech koncernu Volkswagen. Se Slavomírem Mikuleckým jsme se sešli v pražské pobočce v místě jeho pracoviště [3].

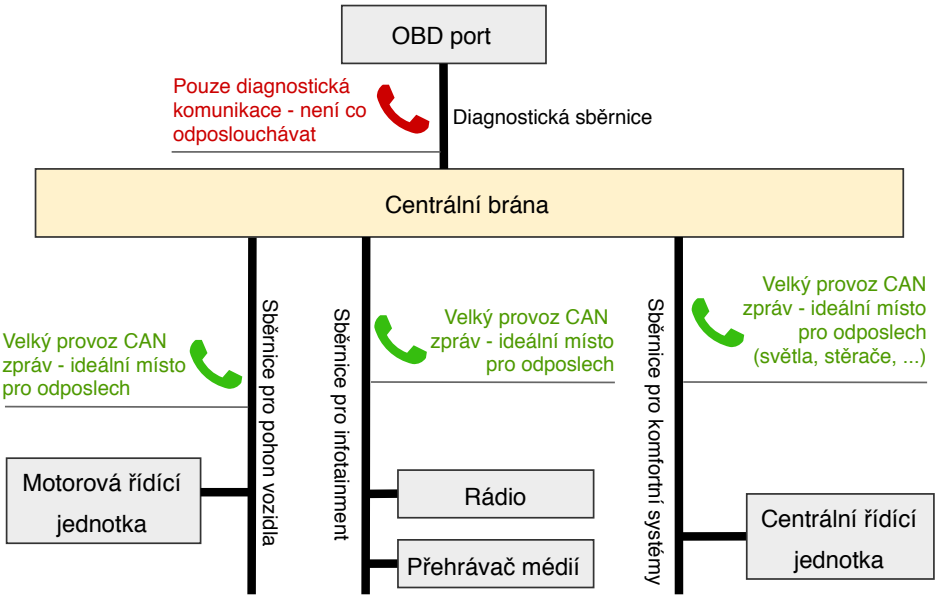
Slavomír mi ukázal mapu sběrnic a vysvětlil mi, že CAN sběrnice je ve vozech skupiny Volkswagen rozdělena na více podsběrnic (viz ilustrační obrázek 1.2 na straně 5) a že diagnostická podsběrnice je vyčleněna mimo ostatní podsběrnice. Centrálním bodem mezi všemi sběrnicemi je brána *(gateway)*, která přeposílá mezi podsběrnicemi jen ty zprávy, u kterých je to nutné. Je zodpovědná za to, že do diagnostické podsběrnice (kde je OBD-II port) přeposílá pouze diagnostickou komunikaci a nic jiného. Proto jsem při spuštěném monitoringu neviděl žádné zprávy. Obrázek 4.16 ilustruje, které podsběrnice má smysl monitorovat.

Abych mohl číst a analyzovat zprávy, které ovlivňují například rozsvícení světel, spuštění rádia nebo stěračů, musel bych se dostat přímo na CAN pro komfortní systémy, kde tato komunikace probíhá. Způsoby, jak toho docílit jsou 2:

1. Změnit software v centrální bráně, aby posílala komunikaci i na diagnostický CAN.

2. Místo do OBD-II portu se připojit až za bránu přímo na komfortní CAN.

První způsob je pro mě vyloučen. I kdybych to technicky dokázal provést, nikdo mi nepůjčí auto, abych to reálně otestoval. Riziko nežádoucích účinků je



Obrázek 4.16: Smysluplnost odposlechu podsběrnic CAN

příliš velké. Druhý způsob je rovněž problematický. K tomu, abych se dostal k vodičům, musel bych odstranit plastové kryty v kabině. Tyto kryty jsou uchyceny pomocí plastových háčků a jsou navrženy tak, aby se snadno při výrobě montovaly. Se sundaváním se nepočítá a kdo není zkušený pracovník servisu a neví, kde přesně použít sílu, snadno plastové háčky nevratně zlomí. Kryt už pak na svém místě nebude držet a při otřesech může vypadávat. Dalším problémem je připojení k vodičům sběrnice. Musel bych odstranit gumové těsnění, to má ale svůj význam. Automobily se používají za každého počasí a vodiče s poškozeným těsněním mohou začít korodovat. Dostávám se tak opět do stavu, že nutná úprava by mohla způsobit poškození vozu. S tím vědomím mi automobil na testy nikdo nepůjčí.

Na závěr schůzky jsem se dozvěděl, že jednotky komunikují na aplikační úrovni pomocí UDS protokolu. Začel jsem si tedy zjišťovat, jak tento protokol funguje a k čemu je dobrý. Výsledky průzkumu jsou v kapitole 1 v sekci 1.6.1.

Den 130.

Byl jsem trochu zaseknutý na mrtvém bodě. Smířil jsem se s tím, že monitoring sběrnice je pro mě zapovězený, a tak jsem veškeré úsilí zaměřil na studium UDS protokolu. Doufal jsem, že snad bude cesta, jak pomocí něj zjistit alespoň nějaké informace. Objevil jsem, že existuje UDS služba `22`, pomocí které je možné číst data podle identifikátoru. Neměl jsem ale žádnou představu o tom, jak takovou službu použít.

Šel jsem dělat pokusy do auta (Škoda Octavia II. generace). Použil jsem aplikaci *Serial Bluetooth Terminal*, protože moje vlastní aplikace *OBD Robot* v té době ještě nebyla dokončena natolik, aby byla schopná spolehlivě odesílat to, co jsem potřeboval. Odesílal jsem zprávy `22XX`, kde jsem za `XX` postupně dosazoval všechny kombinace. Odpovědí bylo vždy jen `NODATA`. Když jsem odeslal jen `22`, odeslala se znovu poslední platná zpráva a na ni přišla odpověď. Otestoval jsem také sekvenci `AT CF 000`, `AT CM 000`, `AT ST FF` a `AT MA`. Tedy monitoring sběrnice s vypnutým filtrem a maximální dobou čekání na odpověď. Reakce z ELM327 byla `OK`, ale při monitoringu se neukázalo. Čekal jsem přes 60 s. Zkoušel jsem zařadit, popojet, stáhnout okno, pustit stěrače, světla, nic se neukázalo. Detaily k měření jsou v sekci C.3 na straně 122.

Den 140.

Moje předchozí pokusy s UDS službou `22` nepřinesly žádné výsledky. Požádal jsem proto opět o schůzku Slavomíra Mikuleckého, aby mi názorně předvedl, jakým způsobem data z vozu získat [51]. Sešli jsme se opět v sídle firmy Digiteq Automotive v Praze a tentokrát jsme si vzali na pomoc automobil Volkswagen Caddy III. generace (rok výroby 2014) a notebook se softwarem CANoe [52]. Profesionální nástroj CANoe[40] dokáže kromě jiného monitorovat sběr-

[40] Časově neomezená licence pro podnikové využití na jeden počítač stojí 8000 € [53].

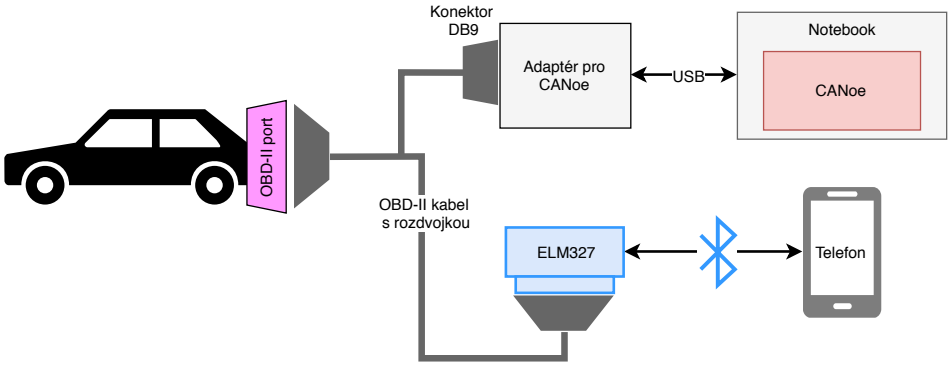
nici a přehledně filtrovat zprávy na ní probíhající. Použili jsme ho z několika následujících důvodů.

1. Abychom ověřili, že na diagnostické CAN sběrnici skutečně neprobíhá jiná komunikace než diagnostická.

2. Abychom vyloučili možnost, že příkaz `AT MA` funguje špatně nebo adaptér ELM327 je porouchaný.

3. Abychom zjistili, jaké CAN ID používá u odeslaných zpráv adaptér ELM327.

Připojili jsme CANoe, adaptér ELM327, notebook a mobilní telefon podle schématu na obrázku 4.17. Zkoušeli jsme posílat dotazy, na které by měla přijít odpověď, ale vždy bez úspěchu. Nezjistili jsme, v čem je problém, přesto bylo měření užitečné, protože odpovědělo na výše uvedené otázky.



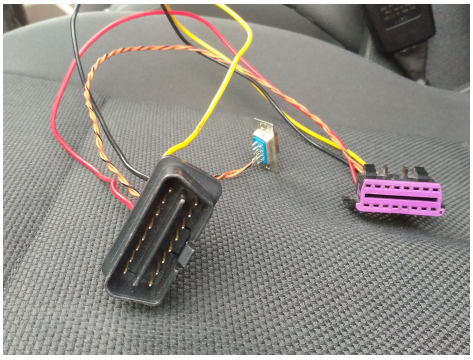
Obrázek 4.17: Schéma zapojení při měření na diagnostické CAN sběrnici 8. 3. 2018

1. Sledování pomocí CANoe potvrdilo, že na diagnostické sběrnici, žádná komunikace kromě diagnostické přítomna není.

2. Příkaz `AT MA` vykonaný adaptérem ELM327 nejspíše funguje správně. Neukazuje nic, protože nemá co ukázat.

3. Adaptér ELM327 používá při odesílání zpráv CAN identifikátor s hodnotou $7DF_{hex}$.

Během měření jsem nastavil adaptéru ELM327 CAN identifikátor na hodnotu, kterou používá při diagnostice profesionální software, ale ani na takové zprávy nepřišla od vozidla žádná odpověď. Záznam z měření je v sekci C.4 na straně 124. Na sdruženém obrázku 4.18 je obrazová dokumentace z měření.



(a) Detail OBD-II rozdvojky

(b) Volkswagen Caddy

(c) Celkové propojení

Obrázek 4.18: Získávání dat z vozidla prostřednictvím UDS

Den 145.

Měření na Volkswagenu Caddy příliš světla do problému nevneslo, hledal jsem proto další odborníky, kteří by mě dokázali nasměrovat správným směrem. Dostal jsem doporučení na Miroslava Čermáka, který v Digitequ programuje jednotky na CAN sběrnici. Sešli jsme se v pražské kanceláři [54]. Mirek programuje jednotky, které se umisťují přímo na nějakou podsběrnici a problém s obcházením brány tedy při své práci řešit nemusí.

K diagnostickým úkonům se v Digitequ využívá diagnostický software ODIS[41] [55]. Navrhl, že bych mohl vyvolat například stažení okénka nebo rozsvícení světel nějakým diagnostickým testem z ODISu a přitom bych od-

[41] ODIS je profesionální diagnostický nástroj pro automobilové vývojáře a servis značek koncernu Volkswagen. Vývojáři používají verzi *Engineering*, jejíž roční licence stojí 22 000 Kč.

poslouchával komunikaci svým zařízením ELM327. Následně bych se pokusil komunikaci ze své mobilní aplikace zopakovat. Tuto myšlenku jsem se rozhodl dále rozvíjet a začal jsem hledat někoho, kdo se dobře vyzná v diagnostice.

Den 148.

Dostal jsem odpovědi na své zvídavé maily od Martina Dočekala a Radka Šťastného. Mají v Digitequ na starosti diagnostickou komunikaci. Z naší mailové korespondence ze dne 12. 3. 2018 cituji ty nejzajímavější pasáže:

1. „Sledování komunikace po diagnostickém CANu je možné pouze ve chvíli, kdy probíhá komunikace mezi testerem a jednotkou, tester (OBD) odesílá žádosti a *gateway* na ně odpovídá. Před vyčtením dat je potřeba zaslat jednotce nějaké informace, abys vůbec mohl vyčítat.“

2. „Pokud jsi posílal zprávu s ID, které žádná jednotka nepřijímá, pak jsi ani odpověď přijmout nemohl.“

3. „Stačí měnit jeden Bit v daných originálních zprávách a nebude ti vycházet *StuffBit*[42] a nebo CRC[42]. Následně můžeš generovat své zprávy. Ale bude to složité na implementaci, protože si takto můžeš zarušit i své zprávy. Potřebuješ nějaký algoritmus na rozeznání.“

4. „V příloze zasílám popis k diagnostickému požadavku pro rádio. Samozřejmě, že službu $\texttt{2F}_{hex}$ (*OI-Control* – test akčních členů) můžeš použít i na jiné jednotky, ale je třeba znát parametry a identifikátory pro akční členy.“

Bod číslo 4 byl pro následný průběh realizace zlomový. Pokud si člověk provádějící diagnostiku vygeneruje z příslušné databáze dokumentaci k vybrané platformě[43], může poté pomocí UDS služby `2F` provádět testování akčních členů. Akční člen je ve vozidle prakticky cokoliv – ručička tachometru, větrák, světlo, stěrač, ... Samotný test akčního členu je připraven v řídící jednotce, která akční člen obsluhuje. UDS zpráva iniciuje spuštění testu s parametry, které ovlivňují například dobu běhu testu. Stačí přitom být připojený do diagnostického portu OBD-II, není třeba ani změna softwaru v bráně *(gateway)* ani přímý přístup k podsběrnicím.

Domluvil jsem se s Radkem Šťastným na osobním setkání přímo v diagnostické laboratoři, kde si testy akčních členů budu moci vyzkoušet na vlastní kůži. Cítil jsem, že se blíží velké finále mého výzkumu.

[42] Detaily k pojmům *StuffBit* a CRC přináší sekce 1.4.3 na straně 15.
[43] Platforma sdružuje několik modelů aut vyráběných v určitém období.

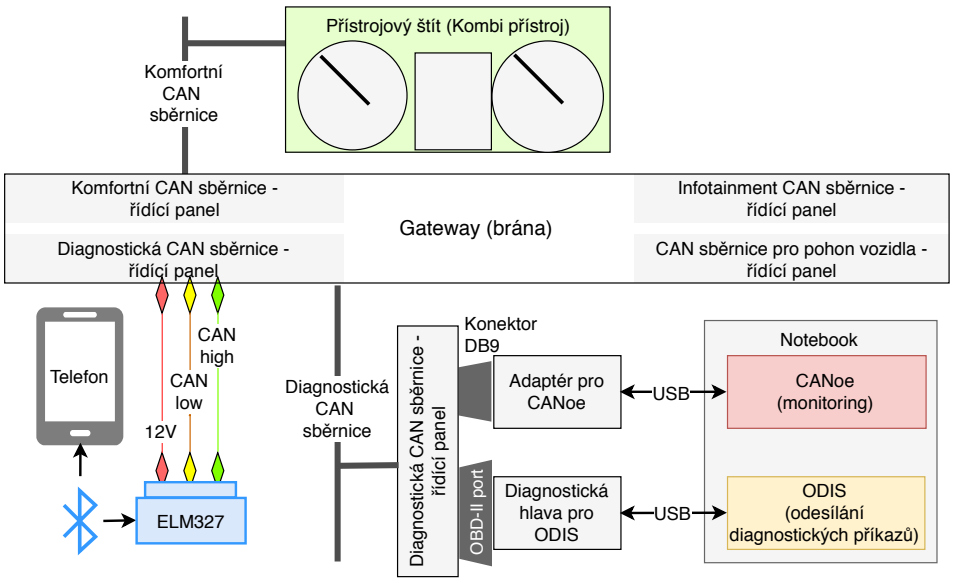
Den 159.

Breadboard je základová deska pro konstrukci elektronických prototypů [56]. Původní význam anglického slova je prkénko pro krájení chleba a není divu, že první breadboardy dostaly toto pojmenování. Jednalo se o dřevěné desky na nichž byly uspořádány součástky do nějakého obvodu. Vzhled se do dnešní doby značně změnil a desky už dřevěná prkénka nepřipomínají. Jsou to zpravidla hustě perforované bílé desky, do kterých se snadno zasazují součástky. V českém jazyce se pro termín breadboard používá pojem *nepájivé pole*. Jedná se však o malé desky o velikosti maximálně jednotek decimetrů čtverečních určené pro spojování malých elektronických součástek. V automobilovém průmyslu se pojem breadboard používá pro obrovské stěny osazené veškerou elektronikou, která se v autě nachází [57]. Stěny mohou zaujímat plochu až 20 m^2 a je na nich k nalezení vše od volantu, přes jednotlivé řídící jednotky až po větráky klimatizace a zadní světla. Jde vlastně o zcela rozložené auto bez hnacího ústrojí a karoserie. A zde už jistě každý cítí, že používat pojem *nepájivé pole*, by bylo zcela zcestné. Proto se budu držet dále v textu anglického originálu.

Digiteq Automotive má breadboardů několik, každý z nich je sestaven pro nějakou platformu. Já jsem společně s Radkem Šťastným obsadil breadboard pro model Škoda Yeti a další příbuzné modely. Z důvodu ochrany intelektuálního vlastnictví uvádím na obrázku 4.19 pouze ilustrační fotografii breadboardu, kterou jsem převzal z [58].

Obrázek 4.19: Breadboard – ilustrační snímek

Kromě samotného breadboardu jsme pro testy měli k dispozici programy ODIS a CANoe, adaptér ELM327 Viecar a mobilní aplikaci *Serial Bluetooth Terminal*. Vlastní aplikace *OBD Robot* ještě nebyla v té době hotová. Testovali jsme prostřednictvím UDS služby `2F` akční členy. Přitom jsem měl zapojený do diagnostické CAN sběrnice adaptér ELM327, se kterým jsem prováděl monitoring sběrnice. Komponenty jsme měli zapojené podle schématu na obrázku 4.20. Po odposlechnutí komunikace jsem se ji snažil zopakovat skrze své nástroje.



Obrázek 4.20: Schéma propojení komponent při testování akčních členů na breadboardech 27. 3. 2018

Nejprve jsem roztáčel ventilátor klimatizace. Postupoval jsem následovně:

1. Radek Šťastný vygeneroval z databáze diagnostické dokumentace [59] a [60] ve formátu PDF k platformě, na které jsme měřili.

2. Propojili jsme všechna zařízení podle schématu na obrázku 4.20.

3. Vyhledal jsem v diagnostické dokumentaci [59] v sekci `2F` příkaz k provedení testu větráku. Jedná se o 8 bytů na aplikační úrovni, s byty transportní vrstvy je třeba odeslat celkem 11 bytů a to budou 2 fyzické CAN zprávy. (Shrnutí síťových vrstev je v sekci 1.7.)

4. Spustil jsem příkazem `AT MA` monitoring sběrnice na adaptéru ELM327.

5. Odeslal jsem příkaz pro aktivaci ventilátoru. Dal se ihned do pohybu, což bylo znát na první poslech podle hlasitého hučení.

6. Zjistil jsem, že:
   - ODIS je přepnutý v diagnostické relaci a periodicky vysílá příkaz *Tester present*. (Přehled UDS relací je v tabulce 1.6 na straně 26 a v přidružené sekci.)
   - Před samotným spuštěním testu, odesílá ODIS inicializační příkaz, který předá řízení větráku z rukou řídící jednotky klimatizace do rukou diagnostického nástroje.
   - Zpráva se skutečně odeslala prostřednictvím 2 CAN rámců *(framů)*.
7. Vyhledal jsem v programu ODIS správné CAN ID. Zprávy pro každý akční člen musí mít přesně daný CAN identifikátor. Pokud bych odeslal zprávu se správným obsahem, ale špatným identifikátorem, brána ji zahodí jako neplatnou. Diagnostické zprávy pro klimatizaci mají jiný identifikátor než zprávy pro přístrojový štít a ty zase mají jiný identifikátor než zprávy pro řídící jednotku airbagů, atp.
8. Pokusil jsem se sekvenci příkazů zopakovat z aplikace *Serial Bluetooth Terminal*, ale adaptér ELM327 příkazy neodeslal. Zůstal ve stavu, kdy čeká na další vstup od uživatele.

Pátral jsem po tom, proč adaptér neodeslal zadané příkazy, a z dokumentace [34] jsem zjistil, že mikrokontroléry ELM327 umí odeslat zprávu dlouhou maximálně 7 bytů. Experimentováním jsem ale došel k faktu, že můj adaptér Viecar dokáže odeslat zprávu o maximální délce 4 byty. Adaptér neumí využívat transportní protokol k tomu, aby odesílal zprávy přesahující rozsah jednoho CAN rámce, což je velký problém. Drtivá většina diagnostických příkazů v UDS službě `2F` se skládá minimálně z 8 bytů.

Zopakoval jsem postup odposlechu také při testu přístrojového štítu (kombi přístroj). Po odeslání příkazu z ODISu zhasly všechny kontrolky uvnitř ciferníků a otáčkoměr se nastavil na hodnotu 3000 (viz obrázek 4.21). Po 20 sekundách se uvedl přístrojový štít do původního stavu. V [60] jsem nalezl příkaz, který vrací řízení přístroje zpět do rukou řídící jednotky a má délku přesně 4 byty. Odeslal jsem příkaz z mobilního telefonu a ciferníky se uvedly do původního stavu. Byl jsem tedy schopen test z mobilu ukončit, ale ne ho spustit.

Dále se mi podařilo příkazem `11 03` resetovat všechny řídící jednotky. Takový příkaz musí být odeslán s CAN identifikátorem $\texttt{700}_{hex}$, což značí, že je určen všem jednotkám. Rovněž se mi podařilo přepnout se z mobilního telefonu do diagnostické relace příkazem `10 03`.

V laboratoři jsem strávil téměř 6 hodin. Při analýze monitorovaných zpráv jsem pochopil fungování protokolu ISO-TP a jeho roli na CAN sběrnici. Zpočátku měření jsem nerozuměl tomu, proč odeslané zprávy začínají odlišně než je uvedeno v dokumentaci. Až po důkladném seznámení s transportním protokolem jsem pochopil, že „to divné“ na začátku zpráv jsou PCI byty protokolu

ISO-TP. Více detailů z měření na breadboardech a ukázky dat naměřených při monitoringu sběrnice přináší kapitola 5 v sekci 5.3.

Obrázek 4.21: Přístrojový štít na breadboardu po spuštění testu

Shrnutí

Ukázalo se, že monitoring sběrnice v současně vyráběných vozech koncernu Volkswagen není z diagnostického portu OBD-II možný. Port je připojen k podsběrnici, která je bránou *(gateway)* odstíněna od veškerého provozu na ostatních podsběrnicích. Pro úspěšný monitoring by bylo nutné odstranit kryty, odmontovat části vozu a připojit se pomocí krokosvorek přímo k vodičům jedné z podsběrnic. Druhou možností by byla změna softwaru v bráně tak, aby odesílala komunikaci z jiných podsběrnic i na diagnostickou podsběrnici. Obojí jsem neučinil, protože jsem neměl k dispozici žádný automobil, na kterém by to bylo možné provádět a otestovat.

Změnil jsem proto původní plán a uchýlil jsem se k odposlechu diagnostických zpráv během diagnostické relace. Ukázalo se, že zprávy jsou příliš dlouhé na to, aby je mohl můj adaptér ELM327 reprodukovat. V testovací laboratoři se podařilo i s takovým omezením vyvolat zhasnutí přístrojového štítu nebo resetovat všechny řídící jednotky. Obojí bylo iniciováno z mobilní aplikace a odesláno pomocí adaptéru ELM327. To ale není možné reprodukovat v sériově vyráběném automobilu.

Z důvodu velké časové náročnosti výzkumu jsem byl nucen ukončit projekt i za cenu toho, že jsem v reálném vozidle nebyl schopen pomocí adaptéru vyvolat žádnou akci. Je ale vypracován dostatečný základ, z něhož je možné pokračovat v rámci jiné práce ať už s použitím jiných automobilů nebo jiných adaptérů. Odhaduji, že výběr, nákup, seznámení a výzkum s pomocí jiného

kvalitnějšího adaptéru by zabraly minimálně další dva měsíce. Tolik času už jsem v době zjištění výsledků neměl.

Přestože pokusy o reálné ovládání skončily neúspěšně, považuji výsledek svého průzkumu v této oblasti za úspěch a důkaz, že i s takto levným a jednoduchým vybavením lze dosáhnout určitých výsledků. Používal jsem ten nejlevnější adaptér ELM327 v ceně 95 Kč a neschopnost odeslat dostatečně dlouhou zprávu pro mě vlastně není překvapením. Naopak, je příjemným zjištěním, že kromě tohoto zásadního nedostatku neexistují další technická omezení na straně adaptéru, která by bránila monitoringu sběrnice a následné analýze dat.

4.4 Souhrnné výsledky realizace

Realizaci práce jsem rozdělil do tří tématických celků, které na sobě byly téměř nezávislé. V prvním tématickém celku (sekce 4.1, strana 62) jsem sehnal 3 adaptéry pro komunikaci s vozidlem. Všechny 3 komunikují s aplikací *Serial Bluetooth Terminal* i s mojí vlastní aplikací *OBD Robot* a jsou schopné číst údaje z vozidla.

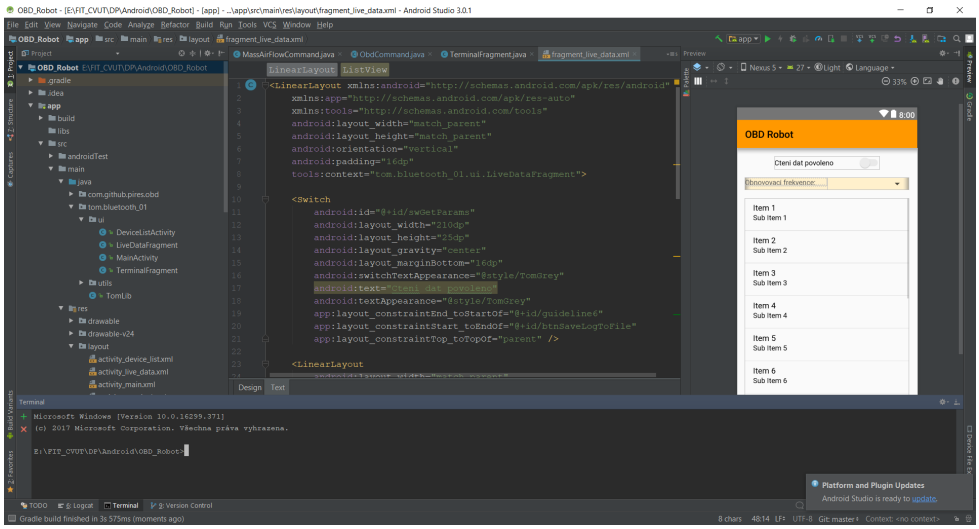
Ve druhém tématickém celku (sekce 4.2, strana 72) jsem vytvářel vlastní aplikaci *OBD Robot*, která je schopná číst aktuální data z vozidla. Dokáže také pomocí terminálu odesílat do vozidla libovolný příkaz a zobrazit „syrovou“ odpověď na něj přesně tak, jak vypadá na CAN sběrnici. Nad rámec zadání jsem vytvořil aplikaci s velmi čistým, srozumitelným a přehledným uživatelským rozhraním, která dokáže exportovat záznam komunikace do textového souboru. Aplikace kombinuje libovolné odesílání příkazů pomocí terminálu a zobrazovaní aktuálních dat z vozidla. Díky tomu je *OBD Robot* mezi analyzovanými aplikacemi v kapitole 2 unikátní.

Ve třetím tématickém celku (sekce 4.3, strana 83) jsem se snažil odhalit příkazy pro ovládání vozidla, které se vyskytují na CAN sběrnici. K tomu jsem chtěl použít monitoring sběrnice, ale ukázalo se, že není možné ho na vozidlech, která jsem měl k dispozici, použít. Brána *(gateway)* z bezpečnostních důvodů blokuje monitoring sběrnice z diagnostického portu. Proto jsem změnil původní plán a rozhodl jsem se k odposlechu diagnostických zpráv během diagnostické relace. Zprávy se podařilo odposlechnout, ale ukázalo se, že mnou použitý adaptér ELM327 nedokáže posílat dostatečně dlouhé zprávy. Pokud bych ale měl vhodný adaptér, zprávy pro ovládání vozidla bych odesílat mohl. Výsledek přesto považuji za úspěch, protože jsem dokázal, že i s použitím nejlevnější dostupné techniky je při dostatečné znalosti možné odesílat vozidlu příkazy, na které zareaguje.

4.5 Technické prostředky použité při práci

4.5.1 Software

K realizaci této práce jsem v počítači použil vývojové prostředí *Android Studio* [44] (viz obrázek 4.22), softwarovou knihovnu *OBD-Java-API Library* [45], program pro diagnostickou komunikaci *ODIS* [55], program pro sledování komunikace na CAN sběrnici *CANoe* [52] a program pro sledování sériové komunikace *Advanced Serial Port Terminal* [36].

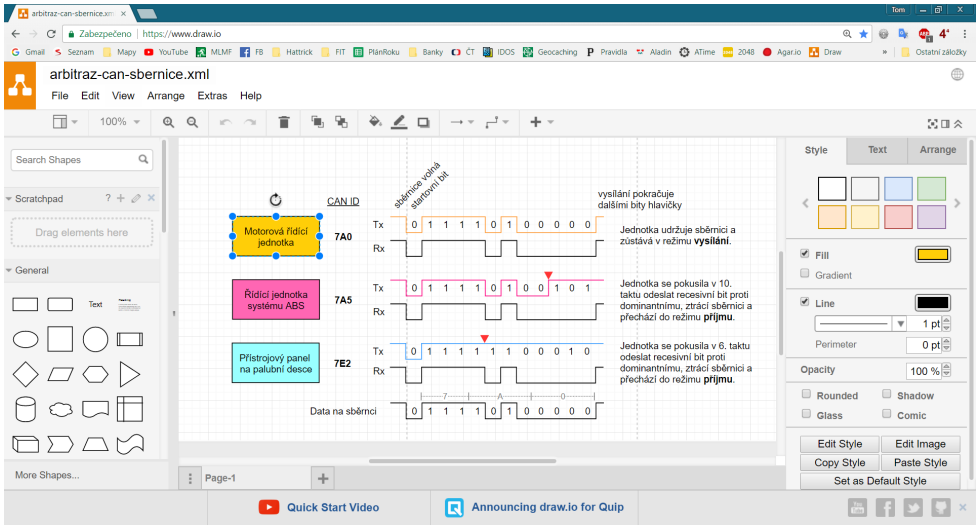


Obrázek 4.22: Android Studio při vývoji aplikace OBD Robot

V mobilním telefonu jsem používal aplikaci pro komunikaci po sériové lince *Serial Bluetooth Terminal* [37].

K vytvoření textové části diplomové práce ve formátu `PDF` jsem použil sázecí systém LaTeX [61] a textový editor *Sublime Text* [62]. Vektorové obrázky schémat a diagramů jsem tvořil především v online nástroji *draw.io* [63] (viz obrázek 4.23) a některé v programu *Inkscape* [64].

Není-li uvedeno jinak, jsou veškeré obrázky, fotografie a schémata moje dílo vytvořené přímo pro účely této práce.



Obrázek 4.23: Draw.io – nástroj pro kreslení vektorových diagramů

4.5.2 Hardware

Pro zprostředkování komunikace mezi vozidlem a mobilním telefonem jsem používal neznačkový adaptér založený na mikrokontroléru ELM327 [34]. Detaily k použitým adaptérům poskytuje podkapitola 4.1 a přehled existujících adaptérů poskytuje sekce 3.1.

Při testování jsem použil automobily Škoda Octavia II. generace (rok výroby 2011), Škoda Superb III. generace (rok výroby 2017), Škoda Karoq (rok výroby 2017) a Volkswagen Caddy III. generace (rok výroby 2014). Všechna vozidla pocházela ze sériové výroby a nebyla pro účely testování nijak upravována.

Kapitola 5

Testování

Cílem této kapitoly je prokázat, že výsledky a dosažená řešení v kapitole Realizace jsou platná a funkční. Proto jsem absolvoval mnoho testovacích schůzek, jízd a měření, z nichž existuje jak textová, tak obrazová dokumentace. Než se pustím do testovacích detailů, je nutné pro udržení kontextu připomenout strukturu kapitoly Realizace. Stejnou strukturu udržuje i kapitola Testování. Realizace se skládá z následujících tří tématických celků:

1. pořízení adaptéru ELM327, jeho nastavení, práce s ním a čtení údajů o vozidle – sekce 4.1 od strany 62
2. vytvoření aplikace OBD Robot – sekce 4.2 od strany 72
3. odhalování interní komunikace ve vozidle – sekce 4.3 od strany 83

Ať už je řeč o libovolném tématickém celku, mnoho testovacích schůzek a měření bylo nedílnou součástí realizace. Vyplynuly z nich závěry, bez nichž by následný vývoj realizace nedával smysl. Proto jsem mnohé testy a jejich průběhy zmiňoval už v rámci kapitoly 4 a zde je uvedu už jen výčtově ve formě odkazů na příslušné stránky.

5.1 Testování funkčnosti adaptérů ELM327

K dispozici jsem měl 3 adaptéry ELM327. Jejich přehled je v tabulce 4.1 a na obrázku 4.9 v sekci 4.1 na straně 71.

ELM327 Marek

ELM327 Viecar

ELM327 Tom

Abych vyloučil chybu na straně vlastní aplikace, testoval jsem adaptéry pomocí mobilní aplikace *Serial Bluetooth Terminal*. Testovacím prostředím pro mě byl vždy jeden z následujících automobilů:

- Škoda Octavia II. generace (rok výroby 2011)
- Škoda Superb III. generace (rok výroby 2017)
- Škoda Karoq (rok výroby 2017)
- Volkswagen Caddy III. generace (rok výroby 2014)

Mohl bych použít i libovolný jiný osobní vůz vybavený OBD-II portem. V automobilu jsem postupoval podle následujícího testovacího scénáře:

1. Nastartoval jsem automobil.
2. Zasunul jsem adaptér do OBD-II portu.
3. Vizuálně jsem zkontroloval, že adaptér je zapnutý. K tomu slouží červená indikační LED dioda na vrchní straně adaptéru.
4. Spojil jsem adaptér s mobilní aplikací *Serial Bluetooth Terminal*.
5. Odesílal jsem základní AT příkazy, například:

```
AT DP     (zobrazí použitý protokol)
AT E0     (vypne opakování zadaného příkazu)
AT H1     (zobrazuje hlavičky zpráv, především CAN ID)
AT S1     (nastaví zobrazování mezer)
AT ST FF  (nastaví dobu čekání na odezvu auta na max.)
```

6. Kontroloval jsem správnost odezvy na příkazy podle [34].
7. Odesílal jsem příkazy pro čtení dat podle normy ISO 15765-4 [4], například:

```
01 0C     (žádost o~aktuální otáčky motoru)
01 0D     (žádost o~aktuální rychlost vozidla)
09 02     (žádost o~VIN kód vozidla)
```

8. Kontroloval jsem správnost odezvy porovnáním s ukazateli v automobilu (například otáčkoměr, teploměr, tachometr).

Funkčnost adaptéru Marek potvrdilo testování, jehož detaily jsou uvedeny pod nadpisem „Den 45.“ v sekci 4.1 na straně 66. Funkčnost adaptérů Tom a Viecar potvrdilo testování, jehož detaily jsou uvedeny pod nadpisem „Den 92.“ v sekci 4.1 na straně 70. Kompletní záznamy z testů jsou v příloze této práce v sekci C na straně 119.

Ve vzácných případech („Den 39.“, sekce 4.1, strana 65) jsem testoval funkčnost adaptéru připojením na zdroj napětí. Připojil jsem adaptér podle schématu 4.11 na straně 75. Přitom není třeba spojení s autem a CAN sběrnicí, aby bylo možné otestovat bluetooth spojení a tudíž i reakce na AT příkazy. Tímto způsobem jsem postupoval v případě adaptéru Viecar, který se jevil být nefunkční. Později se ale ukázalo, že závada byla mechanická a šla snadno odstranit (sekce 4.1, strana 70). Pro testování následných tématických celků jsem tak měl k dispozici celkem 3 plně funkční adaptéry.

5.2 Testování aplikace OBD Robot

Testování aplikace v autě bylo nedílnou součástí realizace po každé větší změně kódu. Finální verze aplikace prošla dvěma důkladnými testy (viz tabulku 5.1). Při obou testech jsem použil adaptéry ELM327 Marek a ELM327 Viecar. Výsledky byly s použitím obou adaptérů shodné, nebudu tedy dále v textu zdůrazňovat, který adaptér byl použit.

Tabulka 5.1: Testování aplikace OBD Robot

Název	*datum*	*testovací vozidlo*
Test 1	7. 4. 2018	Škoda Karoq (rok výroby 2017)
Test 2	20. 4. 2018	Škoda Octavia (rok výroby 2011)

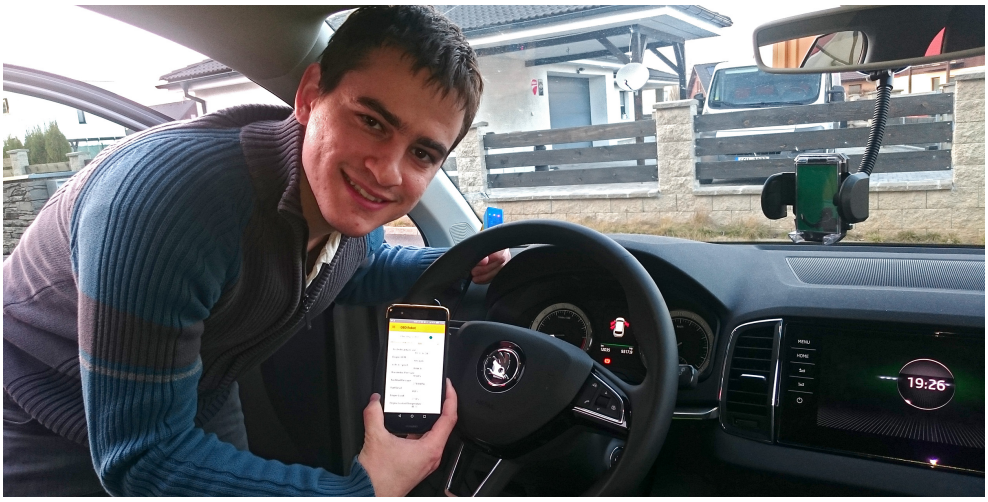
5.2.1 Test 1

Testoval jsem ve vozidle Škoda Karoq (rok výroby 2017, obrázek 5.1a). Obrazovku *Terminál* jsem testoval podle testovacího scénáře na straně 98 naprosto stejným způsobem, jako jsem v předchozí sekci testoval adaptéry.

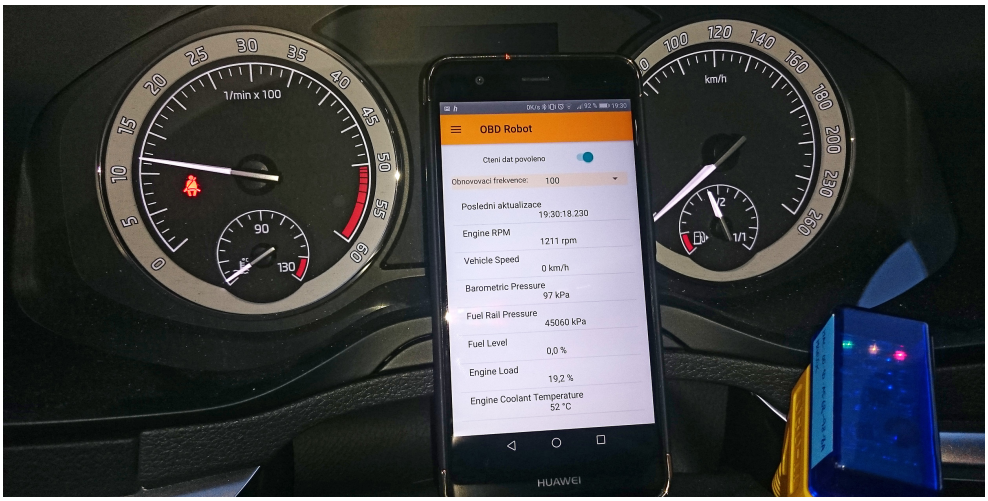
Obrazovku *Aktuální data* jsem testoval během stání a během jízdy. Během stání jsem se zapnutým motorem testoval například teplotu chladící kapaliny nebo stav paliva v nádrži. Během jízdy jsem měl mobilní telefon připnutý v držáku na předním skle a hodnotu rychlosti a otáček motoru jsem porovnával s hodnotami na originálních přístrojích na palubní desce. Obrazová dokumentace z testování je na sdruženém obrázku 5.1. Především obrázek 5.1c jasně dokládá soulad mezi hodnotami v mobilní aplikaci a na přístrojovém štítě automobilu.

(a) Škoda Karoq

(b) Pohled zvenku



(c) Interiér vozu Škoda Karoq



(d) Detail aplikace, přístrojového štítu a adaptéru

Obrázek 5.1: Testování aplikace ve voze Škoda Karoq

5.2.2 Test 2

K testu jsem použil automobil Škoda Octavia II. generace (rok výroby 2011). Testoval jsem společně s Tomášem Krabačem, který mi asistoval a zároveň pomáhal řešit odhalené problémy přímo na místě. Během testování jsme se přesouvali mezi hardwarovou laboratoří a parkovištěm. Na základě výsledků testu jsme se rozhodli odstranit panel s obnovovací frekvencí a obnovovat data maximální možnou rychlostí. Dále jsem nahradil dočasný řetězec „Parametr 2“ za „Napětí“ (dole na obrázku 5.2b). Funkčnost a rychlost obnovování dat jsme testovali na laboratorním zdroji napětí (obrázek 5.2c) a po závěrečném odladění jsme se přesunuli opět na parkoviště a aplikaci jsme znovu otestovali. Snímky obrazovky z testu jsou na sdruženém obrázku 5.2.

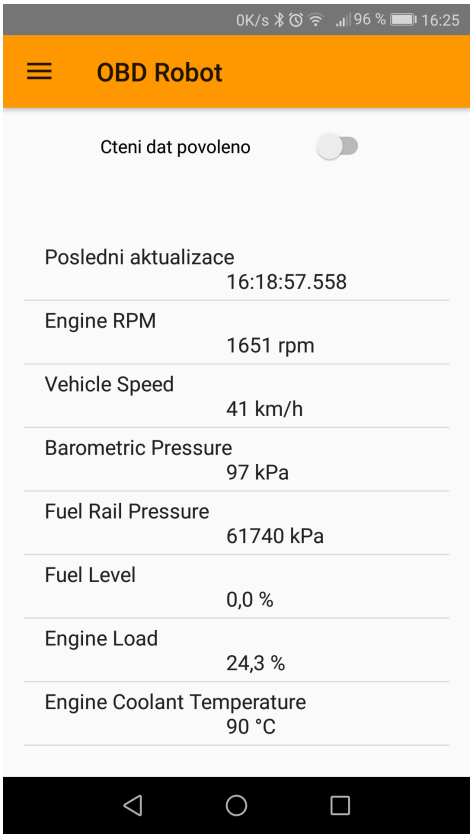
Oba testy potvrdily, že aplikace funguje správně a s výjimkou stavu paliva zobrazuje všechny údaje. Čtení údaje „stav paliva v nádrži“ není vozidly značky Škoda podle normy ISO 15765-4 [4] podporováno. Problém tedy není na straně aplikace, ale u použitých automobilů.

5.3 Odesílání příkazů do vozidla

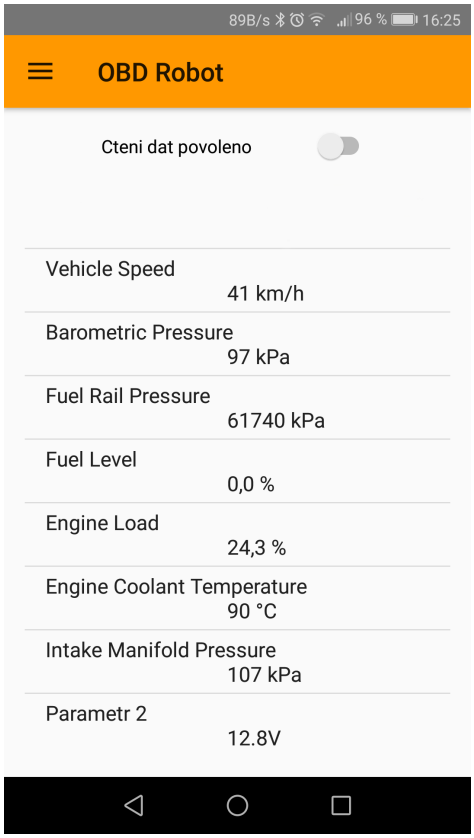
Stejně jako v předchozích tématických celcích je i zde nemyslitelné, aby realizace probíhala bez průběžného testování. O něm se v celé sekci 4.3 kapitoly Realizace zmiňuji. Průzkum navíc ukázal, že z důvodu dobrého zabezpečení není možné odposlouchávat komunikaci ve vozech značky Škoda. Nemělo smysl vytvářet algoritmus nebo pracovní postup k odhalování interní komunikace, který bych zde mohl testovat. Následující odstavce budou proto pojaty jako detailní protokol ze závěrečného měření na breadboardech, kde došlo k největšímu průlomu mého výzkumu (nadpis „Den 159.“ v sekci 4.3 na straně 90).

Při prvním měření jsem použil Adaptér Viecar. Zapojil jsem všechny součásti podle schématu na obrázku 4.20 na straně 91. Nastavil jsem adaptér následujícími AT příkazy:

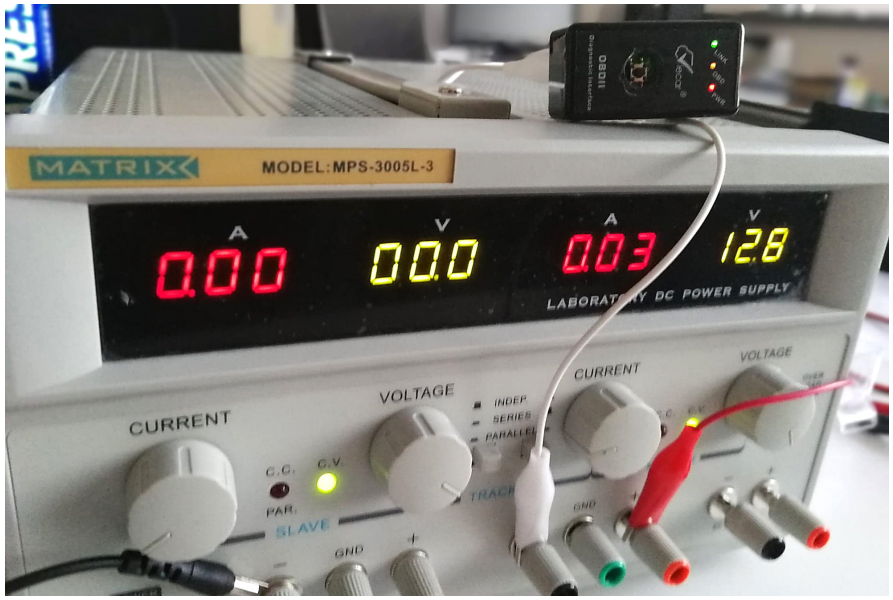
```
AT L1      (za každou odpověď přidá nový řádek)
AT H1      (zobrazuje CAN ID u~každé zprávy)
AT S1      (zobrazuje za každým bytem mezery)
AT AL      (povolí čtení zpráv delších než 1 frame)
AT E0      (vypne opakování zadaného příkazu)
AT SP6     (nastaví protokol na CAN 11/500)
AT CF 000  (nastaví filtr)
AT CM 000  (nastaví masku)
```



(a) Obrazovka Aktuální data



(b) Obrazovka Aktuální data



(c) Testování aplikace na zdroji napětí v laboratoři

Obrázek 5.2: Testování aplikace ve voze Škoda Octavia a v laboratoři

K diagnostické sběrnici byly připojeny programy ODIS (pro odesílání diagnostických testů) a CANoe (pro monitorování). Na svém adaptéru ELM327 jsem spustil monitoring sběrnice příkazem `AT MA`. Pozoroval jsem pouze následující komunikaci:

```
08:28:36.777 700 02 3E 80
08:28:38.764 700 02 3E 80
08:28:40.779 700 02 3E 80
... (zkráceno)
```

Jiné zprávy než tyto na diagnostické sběrnici neprobíhaly. Zprávu `3E 80` generoval program ODIS a jedná se o příkaz *Tester present*, který musí diagnostické zařízení periodicky vysílat, pokud je přepnuté v jiné než výchozí diagnostické relaci (viz sekci 1.6.1 od strany 23). Z komunikace je dále zřejmé, že zprávy odeslané programem ODIS mají CAN ID rovno 700_{hex}.

Odeslal jsem z programu ODIS příkaz ke spuštění testu přístrojového štítu a pozoroval jsem v mobilní aplikaci prostřednictvím příkazu `AT MA`, co se na sběrnici děje.

```
11:17:47.178 77E 04 62 01 00 00
11:17:47.292 714 04 2F 01 40 02          (příprava)
11:17:47.350 77E 03 7F 2F 78
11:17:47.352 77E 04 6F 01 40 02
11:17:47.457 714 10 08 2F 01 40 03 14 FF (spuštění testu)
11:17:47.457 77E 30 0F 05
11:17:47.469 714 21 FF FF 55 55 55 55 55 (spuštění testu)
11:17:47.469 77E 10 08 6F 01 40 03 14 FF (kladná odpověď)
11:17:47.469 714 30 00 00
11:17:47.469 77E 21 FF FF AA AA AA AA AA (kladná odpověď)
... (zkráceno)
```

ODIS odesílá zprávy s CAN ID 714_{hex}, zatímco řídící jednotka přístrojového štítu odpovídá zprávami s identifikátorem $77E_{hex}$. CAN zpráva na fyzické úrovni nemůže být delší než 8 B, proto je příkaz se spuštěním testu rozdělen na 2 zprávy. Po očištění od metadat transportního protokolu ISO-TP a sloučení dostávám:

```
2F 01 40 02             (příprava)
2F 01 40 03 14 FF FF FF (spuštění testu)
```

Nastavil jsem adaptéru ELM327 identifikátor příkazem `AT SH 714` a poté jsem odeslal první příkaz. Vizuálně se nic nestalo, ale v programu CANoe bylo vidět, že od řídící jednotky přišla kladná odpověď. Pokračoval jsem odesláním druhého příkazu, ale adaptér ELM327 příkazy neodeslal. Zůstal ve stavu, kdy čeká na další vstup od uživatele. Experimentálně jsem zjistil, že adaptér nedokáže odeslat zprávu delší než 4 B. Otestoval jsem tedy, jestli můj adaptér

Viecar dokáže odeslat 4 byty dlouhou zprávu „předej řízení zpět do rukou řídící jednotky – `2F014000`“. Spustil jsem test z programu ODIS a pak jsem ho odesláním tohoto příkazu ze svého adaptéru zase zrušil. Ručička na ciferníku se vrátila na nulu a na displeji přístrojového štítu se rozsvítily kontrolky, které svítily před testem. Následně jsem otestoval i resetování všech jednotek, na což podle výpisu z programu CANoe zareagovaly řídící jednotky také pozitivně.

Výše uvedený test prokázal, že jsem schopen odeslat příkaz do vozidla a vozidlo na něj zareaguje. Jsem ale limitován délkou příkazu 4 byty.

Celé testování jsem se pokusil zopakovat se stejným nastavením i s pomocí adaptéru ELM327 Marek. Zjistil jsem, že monitoring pomocí příkazu `AT MA` zobrazuje velmi odlišné hodnoty než adaptér Viecar, přestože nastavení obou adaptérů bylo shodné.

```
13:22:58.161 at ma
13:22:58.199 OK
13:22:58.206 > 00 00 00 00 02 3E 80 00 00 00 00 00
13:23:00.580   00 00 00 00 02 3E 80 00 00 00 00 00
13:23:02.588   00 00 00 00 02 3E 80 00 00 00 00 00
```

Hlavičky CAN zpráv se nezobrazují, místo nich se zobrazuje před i za zprávou mnoho nul, které ve skutečnosti na sběrnici nejsou (srovnáno s výpisem komunikace z programu CANoe). Po spuštění monitoringu odpovídá adaptér ELM327 Marek `OK` a `>`, což není chování, které by mělo podle dokumentace [34] být. Zkoušel jsem také odeslat krátkou čtyřbytovou zprávu, ale nic se nestalo. Testoval jsem mnohé kombinace nastavení adaptéru, ale při žádném se nepodařilo zprávu odeslat. Ukázalo se tedy, že adaptér ELM327 Marek je pro tuto činnost nepoužitelný.

Úplné záznamy komunikace z testování jsou dohromady dlouhé přes 1300 řádků, proto nejsou obsaženy v příloze C, ale pouze na přiloženém CD.

5.4 Shrnutí testů

Testy prvního tématického celku prokázaly, že zakoupené adaptéry ELM327 jsou funkční a mohou sloužit k následnému vývoji aplikace a odesílání příkazů do vozidla. V sekci 5.2 jsem prokázal, že aplikace *OBD Robot* splňuje zadání a je schopná data z vozidla jak číst, tak je do něho prostřednictvím obrazovky *Terminál* i odesílat. Z testů vyplynulo, že vozidla značky Škoda nepodporují zjišťování stavu paliva podle podle normy ISO 15765-4.

V sekci 5.3 jsem zjistil, že prostřednictvím adaptéru ELM327 Viecar lze do vozidla odesílat pouze příkazy, jejichž délka nepřesáhne 4 byty. Monitoring sběrnice je s pomocí tohoto adaptéru možný. Adaptér ELM327 Marek není schopen odeslat do vozidla jiné příkazy než určuje diagnostická norma ISO 15765-4. Adaptér sice dokáže sběrnici monitorovat, ale nezobrazuje zprávy přehledně a mnoho důležitých informací (například identifikátory zpráv) nezobrazuje vůbec.

Závěr

V diplomové práci jsem se zabýval návrhem a implementací mobilní aplikace *OBD Robot* pro systém Android. Cílem bylo analyzovat existující aplikace a podle jejich výhod či nevýhod navrhnout aplikaci novou. Dalším úkolem bylo seznámit se s fungováním CAN sběrnice, systémem OBD-II a s použitím zjištěných informací vytvořit aplikaci schopnou periodického čtení aktuálních informací o stavu vozidla. Výsledkem práce je aplikace, která dokáže číst nejen aktuální data z vozidla, ale je také schopna odesílat do vozidla příkazy, pomocí nichž je možné ovládat součásti vozu. Aplikace prošla mnoha testy a ty potvrdily funkčnost podle návrhu.

Po seznámení s CAN sběrnicí a prostudování nutné teorie jsem vypracoval analýzu existujících aplikací, které umí číst aktuální data prostřednictvím CAN sběrnice a jsou volně dostupné. Analýza ukázala, že pro monitorování údajů vozidla v reálném čase existuje poměrně široká nabídka aplikací. Ty se však vzájemně velmi liší jak množstvím poskytovaných údajů, tak svým grafickým zpracováním a ovladatelností. Ani jedna ze zkoumaných aplikací nemá rozhraní pro odesílání libovolných příkazů do vozidla a prostor k vylepšení skýtá u mnoha z nich také uživatelské rozhraní. Aby se nová aplikace v tomto segmentu uchytila, musí podporovat zobrazení více údajů na jedné obrazovce, přečíst z vozidla velké množství údajů a přehledně je zobrazit. Zároveň musí být co nejjednodušší, aby ji nový uživatel mohl začít používat intuitivně bez nutnosti číst návod. Z těchto požadavků byla formulována specifikace nové aplikace *OBD Robot*.

Samotná aplikace však ke sledování dat nestačí. Komunikaci mezi automobilem a mobilním telefonem zajišťuje adaptér, který se umístí do OBD-II portu vozidla. Z dostupných možností jsem vybral jednu z nejdostupnějších variant na trhu – adaptér ELM327, založený na stejnojmenném mikrokontroléru od firmy ELM Electronics. Adaptér ELM327 je malé zařízení ve tvaru kvádru, jehož nejdelší strana měří 5 cm a s mobilní aplikací komunikuje prostřednictvím bezdrátového standardu bluetooth. V zahraničních e-shopech je

ho možné pořídit v přepočtu už za 100 Kč. Pořídil jsem jich raději více, abych vyloučil možná selhání.

Aplikace *OBD Robot* funguje ve dvou základních režimech – *Terminál* a *Aktuální data*. Terminál umožňuje libovolnou komunikaci s vozidlem. Obrazovka *Aktuální data* zobrazuje hodnoty mnoha vybraných parametrů, které se stále aktualizují. Aplikaci jsem podrobil dvěma důkladným testům na vozech značky Škoda s použitím dvou různých adaptérů ELM327. Oba testy potvrdily, že aplikace dokáže v reálném čase data číst a zobrazovat uživateli a tato data jsou ve shodě s údaji zobrazovanými na vestavěných ukazatelích dat uvnitř vozidla. Jediná hodnota, která se nezobrazovala správně, byl stav paliva v nádrži. Zjistil jsem ale, že čtení tohoto údaje není vozidly značky Škoda podle normy ISO 15765-4 podporováno.

Nad rámec zadání se mi podařilo vytvořit aplikaci se srozumitelným a přehledným uživatelským rozhraním. Umí navíc exportovat záznam komunikace do textového souboru. Z analýzy dostupných aplikací vyplynulo, že v oficiálním obchodě Google Play není ani jedna aplikace, která by kombinovala libovolné odesílání příkazů pomocí terminálu a zobrazovaní aktuálních dat z vozidla. Díky této kombinaci je aplikace *OBD Robot* unikátní.

Aplikace měla původně umět ovládat nekritické součásti vozu (rozsvícení světel, stahování okének, atp.), nakonec ale slouží pouze jako rozhraní mezi odborníkem a automobilem. Skrze obrazovku *Terminál* je sice možné poslat příkaz k nějaké akci, ale jen pokud je známo, jak tento příkaz vypadá. Výzkum ukázal, že monitoring sběrnice v současně vyráběných vozech koncernu Volkswagen není z diagnostického portu možný, což je zásadní překážkou při odhalování skryté komunikace. Z bezpečnostních důvodů blokuje monitoring sběrnice z diagnostického portu takzvaná brána *(gateway)*. Pro úspěšný monitoring by bylo nutné odstranit kryty, odmontovat části vozu a připojit se přímo k vodičům jedné z podsběrnic. Druhou možností by byla změna softwaru v bráně. Ani jedno z možných řešení jsem nemohl aplikovat, protože jsem neměl k dispozici vhodný testovací automobil.

Proto jsem částečně změnil původní plán a rozhodl se k odposlechu diagnostických zpráv během diagnostické relace. Zprávy se podařilo odposlechnout, ale ukázalo se, že použitý adaptér ELM327 nedokáže posílat dostatečně dlouhé zprávy. V testovací laboratoři se podařilo i s takovým omezením vyvolat zhasnutí přístrojového štítu a resetovat všechny řídící jednotky. Obojí bylo iniciováno z mobilní aplikace a odesláno pomocí adaptéru ELM327. Výsledek považuji za úspěch, protože bylo možné ukázat, že i s použitím nejlevnější dostupné techniky je při dostatečné znalosti možné odesílat vozidlu příkazy, na které zareaguje.

Aplikace je nyní k dispozici komukoliv se zájmem zkoumat ovládání svého automobilu z mobilního telefonu. S použitím vhodného adaptéru je možné odeslat jakýkoliv příkaz. Bude-li aplikace součástí úspěšného výzkumu, může sloužit například k vypínání světel, motoru nebo stahování okének.

Literatura

[1] Volkswagen Deutschland – Assistenzsysteme: So funktioniert Trailer Assist. [online], 2018, [cit. 2018-02-28]. Dostupné z: https://www.volkswagen.de/de/technologie/assistenzsysteme/einfacher-einparken.html#item=1&powerLayer=assistenzsysteme/trailer-assist.display

[2] Roadshow: Audi's new A8 offers proper hands-off autonomy in traffic. [online], 2017, [cit. 2018-04-24]. Dostupné z: https://www.cnet.com/roadshow/news/audis-new-a8-is-designed-to-let-you-play-candy-crush-in-rush-hour-traffic-safely/

[3] Mikulecký, S.: Komunikace mezi jednotkami ve vozidlech Škoda Auto. [osobní konzultace], Digiteq Automotive s.r.o., Novodvorská 994/138, Praha, [2018-01-24].

[4] ISO 15765-4:2005(E): Road vehicles – Diagnostic communication over Controller Area Network (DoCAN) – Part 4: Requirements for emissions-related systems. Standard, International Organization for Standardization, Geneva, CH, 2005.

[5] ISO 14230-4:2000(E): Road vehicles – Diagnostic systems – Keyword Protocol 2000 – Part 4: Requirements for emission-related systems. Standard, International Organization for Standardization, Geneva, CH, 2000.

[6] Valasek, C.; Charlie, M.: Remote Exploitation of an Unaltered Passenger Vehicle. [online], 2015, [cit. 2018-02-25]. Dostupné z: http://illmatics.com/Remote%20Car%20Hacking.pdf

[7] EUR-Lex: Přístup k právu Evropské unie: Directive 98/69/EC of the European parliament and of the Council. [online], prosinec 1998, [cit. 2018-02-25]. Dostupné z: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31998L0069:EN:HTML

[8] Smith, C.: *The Car Hacker's Handbook*. San Francisco: no starch press, 2016, ISBN 1-59327-703-2.

[9] Polák, K.: Sběrnice CAN. [online], červen 2003, [cit. 2018-03-04]. Dostupné z: http://www.elektrorevue.cz/clanky/03021/index.html

[10] National Instruments: Controller Area Network (CAN) Overview. [online], srpen 2014, [cit. 2018-03-04]. Dostupné z: http://www.ni.com/white-paper/2732/en/

[11] ISO 11898-1:2003(E): Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signalling. Standard, International Organization for Standardization, Geneva, CH, 2003.

[12] ISO 11898-2:2003(E): Road vehicles – Controller area network (CAN) – Part 2: High-speed medium access unit. Standard, International Organization for Standardization, Geneva, CH, 2003.

[13] ISO 11898-3:2006(E): Road vehicles – Controller area network (CAN) – Part 3: Low-speed, fault-tolerant, medium-dependent interface. Standard, International Organization for Standardization, Geneva, CH, 2006.

[14] Smotlacha, V.: Úvod. Modely sítí. [přednáška]. Technická zpráva, Fakulta informačních technologií, ČVUT, Praha, [2018-02-19].

[15] National Instruments: CAN Physical Layer and Termination Guide. [online], srpen 2016, [cit. 2018-03-06]. Dostupné z: http://www.ni.com/white-paper/9759/en/

[16] Wiesinger, J.: Der CAN-Bus – Grundlagen von Automobil Bussystemen. [online], únor 2018, [cit. 2018-03-06]. Dostupné z: https://www.kfztech.de/kfztechnik/elo/can/can_grundlagen_1.htm

[17] Bosch: *CAN Specification Version 2.0*. 1991, [online], [cit. 2018-03-11]. Dostupné z: http://esd.cs.ucr.edu/webres/can20.pdf

[18] Matematicko-Fyzikální fakulta, Univerzita Karlova v Praze, Katedra didaktiky matematiky: Konjunkce. [online], 2010, [cit. 2018-04-02]. Dostupné z: http://www.karlin.mff.cuni.cz/~portal/logika/?page=konjunkce

[19] Kolektiv autorů encyklopedického institutu ČSAV: *Malá československá encyklopedie*. Praha: ACADEMIA, nakladatelství Československé akademie věd, 1984.

[20] Novák, J.: Diagnostická komunikace [přednáška]. Technická zpráva, Fakulta elektrotechnická, ČVUT, Praha, [2017-10-15].

[21] ISO 15765-2:2004(E): Road vehicles – Diagnostic communication over Controller Area Network (DoCAN) – Part 2: Transport protocol and network layer services. Standard, International Organization for Standardization, Geneva, CH, 2004.

[22] ISO 15765-3:2004(E): Road vehicles – Diagnostics on Controller Area Networks (CAN) – Part 3: Implementation of unified diagnostic services (UDS on CAN). Standard, International Organization for Standardization, Geneva, CH, 2004.

[23] ISO 14229-1:2006(E): Road vehicles – Unified diagnostic services (UDS) – Part 1: Specification and requirements. Standard, International Organization for Standardization, Geneva, CH, 2006.

[24] ISO 14229-2:2013(E): Road vehicles – Unified diagnostic services (UDS) – Part 2: Session layer services. Standard, International Organization for Standardization, Geneva, CH, 2013.

[25] ISO 14229-3:2012(E): Road vehicles – Unified diagnostic services (UDS) – Part 3: Unified diagnostic services on CAN implementation (UDSonCAN). Standard, International Organization for Standardization, Geneva, CH, 2012.

[26] ISO 15031-5:2006(E): Road vehicles – Communication between vehicle and external equipment for emissions-related diagnostics – Part 5: Emissions-related diagnostic services. Standard, International Organization for Standardization, Geneva, CH, 2006.

[27] downloadAPK.net: Remote EX for Nissan. [online], červenec 2014, [cit. 2018-03-24]. Dostupné z: `http://downloadapk.net/Remote-EX-for-NISSAN.html`

[28] Bayer, S.; Hirata, K.; Oka, D. K.: Towards a Systematic Pentesting Framework for In-Vehicular CAN Networks. *escar Europe, Munich*, 2016.

[29] Kuchera, K.: How to hack a car – a quick crash-course. [online], červen 2017, [cit. 2018-03-24]. Dostupné z: `https://medium.freecodecamp.org/hacking-cars-a-guide-tutorial-on-how-to-hack-a-car-5eafcfbbb7ec`

[30] Smith, K.: a complete guide to hacking your vehicle bus on the cheap & easy. [online], březen 2013, [cit. 2018-03-24]. Dostupné z: `https://theksmith.com/software/hack-vehicle-bus-cheap-easy-part-1/`

[31] Kubátová, H.: Systémy na čipu - Úvod [přednáška]. Technická zpráva, Fakulta informačních technologií, ČVUT, Praha, [2017-10-03].

[32] ELM327 miniguide - Read Before Buying. [online], prosinec 2008, [cit. 2018-03-24]. Dostupné z: http://www.ebay.co.uk/gds/ELM327-mini-guide-Read-Before-Buying-/10000000009829308/g.html

[33] 8devices: USB2CAN. [online], 2018, [cit. 2018-04-24]. Dostupné z: https://www.8devices.com/products/usb2can

[34] ELM327 OBD to RS232 Interpreter. [online], 2018, [cit. 2018-03-24]. Dostupné z: https://www.elmelectronics.com/wp-content/uploads/2016/07/ELM327DS.pdf

[35] Simon Tatham: *PuTTY*. [software]. [cit. 2018-04-09]. Dostupné z: https://www.putty.org/

[36] Eltima Software: *Advanced Serial Port Terminal*. [software]. [cit. 2018-04-09]. Dostupné z: https://www.eltima.com/products/serial-port-terminal/

[37] Kai Morich: *Serial Bluetooth Terminal*. [software]. [cit. 2018-04-09]. Dostupné z: https://play.google.com/store/apps/details?id=de.kai_morich.serial_bluetooth_terminal

[38] SparkFun Electronics: Bluetooth Basics. [online], 2016, [cit. 2018-04-02]. Dostupné z: https://learn.sparkfun.com/tutorials/bluetooth-basics

[39] Yanchyshyn, R.: How-to Guide for OBDII Reader App Development. [online], květen 2014, [cit. 2018-03-24]. Dostupné z: https://blog.lemberg.co.uk/how-guide-obdii-reader-app-development

[40] Wikipedia contributors: OBD-II PIDs — Wikipedia, The Free Encyclopedia. [online], 2018, [cit. 2018-04-02]. Dostupné z: https://en.wikipedia.org/w/index.php?title=OBD-II_PIDs&oldid=826257496

[41] Ústav pro jazyk český Akademie věd ČR: Čmuchal – Internetová jazyková příručka. [online], 2004, [cit. 2018-04-12]. Dostupné z: http://prirucka.ujc.cas.cz/?slovo=cmuchal

[42] Vetinari Developement: Testing ELM327 without the car. [online], 2017, [cit. 2017-11-27]. Dostupné z: http://vetinari.pl/2017/03/07/testing-elm327-without-the-car/

[43] Smith, J. P.: Turn a Computer Power Supply into Bench Power. [online], duben 2014, [cit. 2018-04-12]. Dostupné z: https://makezine.com/projects/computer-power-supply-to-bench-power-supply-adapter/

[44] Google Inc: *Android Studio*. [software]. [cit. 2018-04-09]. Dostupné z: `https://developer.android.com/studio/index.html`

[45] Paulo Pires: *OBD-II Java API*. [softwarová knihovna]. [cit. 2018-04-14]. Dostupné z: `https://github.com/pires/obd-java-api`

[46] Paulo Pires: *Android OBD-II Reader*. [software]. [cit. 2018-04-14]. Dostupné z: `https://github.com/pires/android-obd-reader`

[47] Android Developers: Save Files on Device Storage. [online], 2018, [cit. 2018-04-15]. Dostupné z: `https://developer.android.com/training/data-storage/files.html#WriteExternalStorage`

[48] Krabač, T.: Vývoj android aplikace – implementace menu. [osobní konzultace], Technická 2710/6, Praha, [2018-03-29].

[49] Krabač, T.: Vývoj android aplikace – odesílání zpráv pomocí knihovny EventBus. [osobní konzultace], Thákurova 9, Praha, [2018-04-20].

[50] Markus Junginger: *Eventbus*. [softwarová knihovna]. [cit. 2018-04-28]. Dostupné z: `https://github.com/greenrobot/EventBus`

[51] Mikulecký, S.: Získávání dat z vozidla prostřednictvím UDS. [osobní konzultace], Digiteq Automotive s.r.o., Novodvorská 994/138, Praha, [2018-03-08].

[52] Vector Informatik: *CANoe*. [software]. [cit. 2018-04-16]. Dostupné z: `https://vector.com/vi_canoe_en.html`

[53] Pintr, P.: Ceny licencí profesionálních diagnostických nástrojů používaných v automobilovém průmyslu. [osobní konzultace], Digiteq Automotive s.r.o., Ptácká 74, Mladá Boleslav, [2018-04-10].

[54] Čermák, M.: Komunikace jednotek na CAN sběrnici. [osobní konzultace], Digiteq Automotive s.r.o., Novodvorská 994/138, Praha, [2018-03-13].

[55] DNE Elektronik-Systeme GmbH: *Offboard Diagnostics Information System*. [software]. [cit. 2018-04-16]. Dostupné z: `http://www.dne-elektronik.de/en/index.html`

[56] Warren Young: What is a „Breadboard“? [online], 2015, [cit. 2018-04-18]. Dostupné z: `http://tangentsoft.net/elec/breadboard.html`

[57] Šťastný, R.: Testování akčních členů na breadboardech. [osobní konzultace], Digiteq Automotive s.r.o., Ptácká 74, Mladá Boleslav, [2018-03-27].

[58] Gaycarboys.com: Ford's Global Development System and SUV Expertise Underpin Development of All-New Global SUV. [online], 2014, [cit. 2018-04-18]. Dostupné z: https://gaycarboys.com/2014/08/24/fords-global-development-system-and-suv-expertise-underpin-development-of-all-new-global-suv/

[59] EV_ClimaAutoBasis_A01723.odx: Diagnosis documentation – Climatronic (auto) Basis (User mode). Technická zpráva, Volkswagen AG, Wolfsburg, 2012.

[60] EV_KombiUDSVDDRM09_A05733.odx: Diagnosis documentation Kombi_UDS_VDD_RM09 (User mode). Technická zpráva, Volkswagen AG, Wolfsburg, 2015.

[61] TeX Users Group: *LaTeX*. [software]. [cit. 2018-04-15]. Dostupné z: http://www.latex-project.org/ftp.html

[62] Sublime HQ Pty Ltd: *Sublime Text 3.0*. [software]. [cit. 2018-04-14]. Dostupné z: http://www.sublimetext.com/3

[63] JGraph Ltd.: *draw.io*. [software]. [cit. 2018-04-16]. Dostupné z: https://www.draw.io/

[64] Free Software Foundation, Inc.: *Inkscape 0.92*. [software]. [cit. 2018-04-14]. Dostupné z: https://inkscape.org/en/download/

Příloha **A**

Seznam pojmů a zkratek

Algoritmus
: přesný postup řešení nějakého typu problému

Aplikační vrstva sítě
: obsahuje „jádro“ aplikací, které má smysl standardizovat (například HTTP)

AT příkazy
: sada krátkých textových řetězců začínajících znaky „AT“
slouží pro nastavení adaptéru ELM327

Breadboard základová deska pro konstrukci elektronických prototypů
: v automobilovém průmyslu se jedná o všechny elektronické systémy vozu, které jsou rozložené v laboratoři, propojeny a připraveny k testům

CAN Controller Area Network
: komunikační sběrnice požívaná v motorových vozidlech určená ke vzájemné komunikaci řídících a jiných jednotek a také komunikační protokol, který se na této sběrnici používá

CANoe
: profesionální software, který kromě jiného dokáže monitorovat sběrnici a přehledně filtrovat zprávy na sběrnici

CRC Cyclic redundancy check
: krátký kód vypočítaný z dat, která zabezpečuje
před přenosem se vypočítá a přidá na konec přenášených dat, po přenosu se vypočítá znovu a porovná s obdrženým CRC, jsou-li oba shodné, víme, že data nebyla přenosem nijak změněna

DTC Diagnostic trouble code
diagnostický chybový kód
slouží ke specifikování závady ve vozidle, skládá se z písmene a číslic

ECU Electronic control unit
elektronická řídící jednotka
v této práce je zkratka ECU použita zcela výjimečně, ale v anglické literatuře se využívá velmi hojně pro jakoukoliv řídící jednotku ve vozidle

ELM327
mikrokontrolér od firmy ELM Electronics, který slouží transformaci komunikace z počítače do automobilu a obráceně
adaptér obsahující mikrokontrolér ELM327, který navíc přidává například bezdrátovou komunikaci a jiná vylepšení

Fuzzing
metoda, jejímž cílem je nalézt zranitelnosti systému
opakovaně se posílají poškozená nebo záměrně neplatná data do systému a monitoruje se odezva

Fyzická vrstva sítě
fyzická vrstva definuje elektrické vlastnosti rozhraní a určuje, jaké napětí bude považováno za logickou jedničku a jaké za logickou nulu
umožňuje přenos bitů kanálem

Hackování
proces, při kterém se počítačový expert snaží dostat do systému, ke kterému nemá přístup
systém je pro něj „černá bedna“, o které nic neví
cílem je vytěžit ze systému maximum informací

Heuristika
řešení na základě kvalifikovaného odhadu
používá se u problémů, kde není znám přesný algoritmus

HUD Head-up display
průhledový displej, který promítá údaje na čelní sklo vozidla do zorného pole řidiče

Infotainment systém Informační a multimediální systém
systém umístěný v přední centrální části automobilu mezi sedadly řidiče a spolujezdce, který se skládá z velkého displeje a slotů pro externí zařízení
účelem je umožnit posádce vozu ovládat z jednoho místa většinu komfortních funkcí vozu a zároveň dostávat informace o aktuální jízdě, stavu a nastavení vozidla

ISO International Organization for Standardization
mezinárodní organizace pro standardizaci

ISO-TP ISO Transport Protocol
protokol, jehož úkolem je zajistit služby transportní vrstvy sítě
definován standardem ISO 15765-2

ISO/OSI model Open System Interconnection
určuje, jak dekomponovat komunikaci na libovolné síti do hierarchicky uspořádaných vrstev
každá vrstva řeší nějaký problém, přitom využívá služeb vrstvy nižší a poskytuje služby vrstvě vyšší

Linková vrstva sítě
linková vrstva zajišťuje detekci a případnou korekci chyb, formátuje data do rámců a zajišťuje přístup k lince

MAC adresa Media Access Control
jednoznačný identifikátor libovolného zařízení, které komunikuje v síti
není možné ji změnit, je přiřazena již z výroby a skládá se z 6 bytů

MCU mikrokontrolér
malý počítač na jediném integrovaném obvodě, obvykle speciálně navržen pro jeden účel

OBD On-Board Diagnostic
standardizovaný diagnostický systém
umožňuje získávat ze všech vozidel, které se dnes vyrábí, informace o vozidle, aktuální data a informace o chybách prostřednictvím jednotného protokolu a portu

ODIS Offboard Diagnostics Information System
profesionální diagnostický nástroj pro automobilové vývojáře a servisy značek koncernu Volkswagen

PID Parameter identifier
identifikátor parametru
kód, který diagnostické zařízení využívá k získání hodnoty zvoleného parametru z vozidla

Sniffing
metoda sběru dat z vozidla, která spočívá v bezzásahovém monitorování CAN sběrnice

SUV Sportovně užitkové vozy
kategorie osobních vozů

Transportní vrstva sítě
přizpůsobuje data tak, aby je bylo možné poslat prostřednictvím nižších vrstev
protokoly této vrstvy jsou implementovány pouze v koncových účastnících

UDS Unified Diagnostic Services
unifikované diagnostické služby
služby usnadňující diagnostiku vozidla

VIN Vehicle identification number
identifikační číslo vozidla
kód složený ze 17 znaků (čísla, písmena), který jednoznačně identifikuje každé motorové vozidlo

Volkswagen Group koncern Volkswagen
podle ročních prodejů vozů se jedná o jednoho z největších výrobců automobilů na světě
skupina se skládá z celkem 12 značek – Audi, Bentley, Bugatti, Ducati, Lamborghini, MAN, Porsche, Scania, Seat, Škoda, Volkswagen a Volkswagen užitkové vozy

Příloha **B**

Obsah přiloženého CD

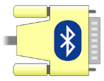
- `readme.txt` ..... stručný popis obsahu CD
- `OBD Robot` ..... adresář s instalací aplikace OBD Robot a návodem
  - `OBD_Robot_v2.0.apk` ..... instalační balíček aplikace OBD Robot pro systém Android
  - `Manuál k aplikaci OBD Robot.pdf` .... návod k instalaci a používání aplikace ve formátu PDF
- `src` ..... zdrojové kódy Android aplikace OBD Robot
  - `OBD_Robot.zip` ..... projekt pro Android Studio se zdrojovými kódy k aplikaci OBD Robot
- `measure` ..... záznamy z měření a testování v automobilech
  - `2017-11-27_laboratoř_Praha`
  - `2017-12-03_SerialBluetoothTerminal`
  - `2017-12-09_Piston`
  - `2018-01-07_domácí_podmínky`
  - `2018-01-11_OBD_Robot`
  - `2018-01-22_adaptér_Viecar`
  - `2018-02-26_Octavia`
  - `2018-03-08_e4t_Caddy`
  - `2018-04-08_OBD_Robot_Karoq`
  - `2018-04-20_laboratoř_Praha`

`text` ..... adresář s elektronickou verzí diplomové práce
- `src` ..... zdrojové soubory textové části práce
  - `img` ..... všechny obrázky použité v diplomové práci
- `Sledování_stavu_vozidla_Tomáš_Zimmerhakl.pdf` .... text práce ve formátu PDF
- `Zadání_diplomové_práce_Tomáš_Zimmerhakl.pdf` ... zadání práce ve formátu PDF

Příloha C

Záznamy z měření

Tato kapitola obsahuje kompletní záznamy z provedených testů a měření ve vozidlech. Přehled je v souhrnné tabulce C.1. Záznamy jsou v textové podobě, nebyly nijak upravovány a zde jsou prezentovány přesně tak, jak byly měřícími nástroji uloženy. Pro měření byly v mobilním telefonu použity tyto aplikace:

 Serial Bluetooth Terminal

 OBD Robot

Tabulka C.1: Přehled záznamů z měření v kapitole C

datum pořízení záznamu	*testované vozidlo*	*použitý adaptér*	*aplikace v mobilním v telefonu*
3. 12. 2017	Škoda Octavia II. generace (rok výroby 2011)	ELM327 Marek	
11. 1. 2018	Škoda Karoq (rok výroby 2017)	ELM327 Marek	
26. 2. 2018	Škoda Octavia II. generace (rok výroby 2011)	ELM327 Marek	
8. 3. 2018	Volkswagen Caddy III. generace (rok výroby 2014)	ELM327 Marek	

C.1 3. 12. 2017 Škoda Octavia

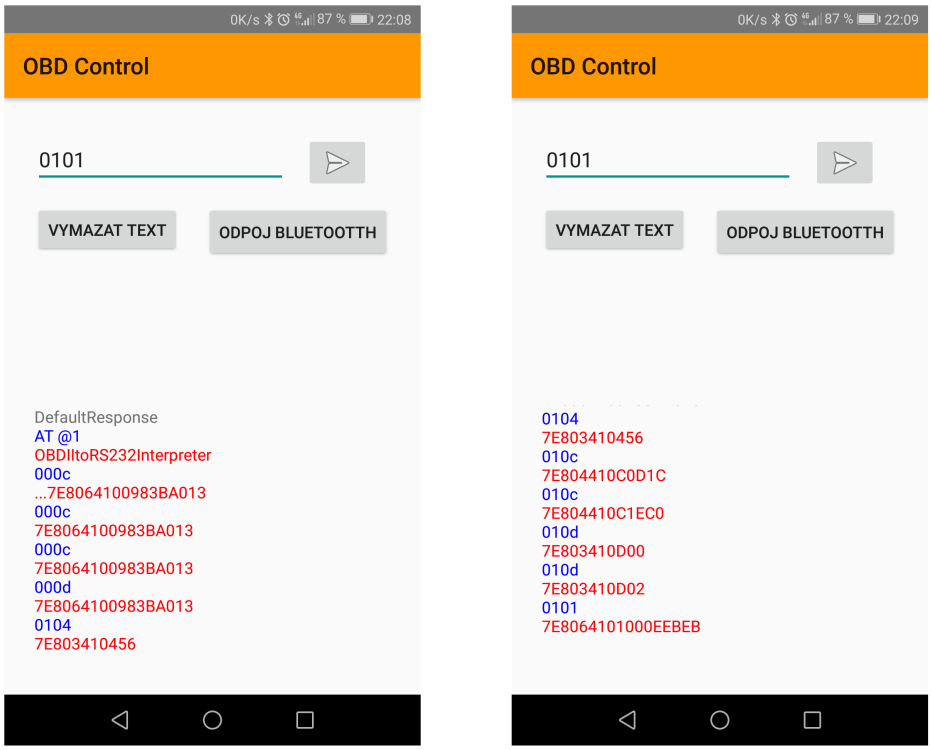
Testoval jsem ve voze Škoda Octavia II. generace (rok výroby 2011). Použil jsem modrý adaptér ELM327 s průhledným krytem od Marka s MAC adresou `00:1D:A5:68:98:8A`. V mobilním telefonu jsem použil aplikaci *Serial Bluetooth Terminal*. Detaily k měření jsou v sekci 4.1 na straně 66.

```
10:17:26.127 Connecting to OBDII ...
10:17:26.561 Connected
10:17:40.727 AT @1
10:17:40.743 OBDII to RS232 Interpreter
>10:17:52.690 AT RV
10:17:52.713 12.1V
>10:17:52.690 AT I
10:17:52.713 ELM327 v2.1
>10:17:52.690 AT DP
10:17:52.713 ISO 15765-4 (CAN 11/500)
>10:17:59.045 ATRV
10:17:59.066 12.1V
>10:18:26.978 AT@2
10:18:26.990 ?
>10:18:34.463 AT @2
10:18:34.482 ?
>10:18:57.722 AT0105
10:18:57.734 ?
>
10:18:59.485 AT0105
10:18:59.501 ?
>
10:19:03.678 AT 0105
10:19:03.686 ?
>
10:19:08.391 AT 01 05
10:19:08.408 ?
>
10:19:13.442 01 05
10:19:13.509 410539
>10:19:31.901 0105
10:19:31.970 410539
>10:20:44.821 0110
10:20:44.890 41100056
>10:21:17.378 0133
10:21:17.450 413363
>10:21:55.521 010C
10:21:55.588 410C0000
>10:25:10.166 010C
10:25:10.227 410C0E22
>10:25:31.360 010C
10:25:31.428 410C1356
>10:26:09.854 0111
10:26:09.917 4111D8
>10:26:55.633 0111
10:26:55.697 4111D8
```

```
>10:27:42.367 012F
10:27:42.577 NO DATA
>10:28:06.049 012F
10:28:06.261 NO DATA
>10:28:38.826 011C
10:28:38.893 411C06
>10:29:18.702 010C
10:29:18.778 410C0000
>10:31:07.313 0100
10:31:07.382 4100983BA013
>10:33:11.410 03
10:33:11.480 4300
>10:33:54.206 03
10:33:54.282 4300
10:33:54.287
10:33:54.287 >10:33:57.218 03
10:33:57.280 4300
10:33:57.303
10:33:57.308 >10:34:06.078 010C
10:34:06.140 410C0000
10:34:06.168
10:34:06.168 >10:34:22.586 03
10:34:22.651 4300
>10:34:23.967 03
10:34:24.040 4300
>10:34:27.098 03
10:34:27.162 4300
>10:34:54.857 03
10:34:54.930 4300
>10:34:55.844 03
10:34:55.929 4300
>10:35:09.330 03
10:35:09.390 4300
>10:35:10.262 03
10:35:10.320 4300
>10:35:26.024 03
10:35:26.090 4300
>10:35:27.049 03
10:35:27.110 4300
>4300
>4300
>10:37:39.943 at rv
10:37:39.963 12.2V
>10:39:25.916 0142
10:39:25.980 41422F08
>10:41:13.318 010D
10:41:13.389 410D00
>10:41:41.560 010D
10:41:41.629 410D09
>10:41:44.278 010D
10:41:44.339 410D08
>
```

C.2 11. 1. 2018 Škoda Karoq

Testoval jsem ve voze Škoda Karoq (rok výroby 2017). Použil jsem modrý adaptér ELM327 s průhledným krytem od Marka. MAC adresa adaptéru je `00:1D:A5:68:98:8A`. V mobilním telefonu jsem použil aplikaci *OBD Robot*. Detaily k měření jsou v sekci 4.2 na straně 76.



Obrázek C.1: Záznam z testování aplikace OBD Robot 11. 1. 2018

C.3 26. 2. 2018 Škoda Octavia

Testoval jsem ve voze Škoda Octavia II. generace (rok výroby 2011). Použil jsem modrý adaptér ELM327 s průhledným krytem od Marka s MAC adresou `00:1D:A5:68:98:8A`. V mobilním telefonu jsem použil aplikaci *Serial Bluetooth Terminal*. Detaily k měření jsou v sekci 4.3 na straně 86.

```
16:07:10.982 Connecting to OBDII ...
16:07:11.657 Connected
>16:08:06.461 010C
16:08:06.552 7E8 04 41 0C 00 00
>16:09:34.299 at cf 000
16:09:34.313 OK
```

```
>16:09:45.537 at cm 000
16:09:45.584 OK
>16:09:51.740 at ma
16:09:51.803 OK
>16:10:15.332 at h1
16:10:15.367 OK
>16:10:37.222 010c
16:10:37.320 7E8 04 41 0C 00 00
>16:10:46.593 at ma
16:10:46.604 OK
>16:12:20.697 010c
16:12:20.910 NO DATA
>16:15:03.076 010c
16:15:03.146 7E8 04 41 0C 0F 4E
>16:15:34.928 at st ff
16:15:34.939 OK
>16:15:43.539 22 00
16:15:43.781 NO DATA
>16:15:50.289 22 01
16:15:50.504 NO DATA
>16:15:52.782 22 02
16:15:53.048 NO DATA
>16:15:56.051 22 03
16:15:56.285 NO DATA
>16:15:59.290 22 04
16:15:59.513 NO DATA
>16:16:02.029 22 05
16:16:02.293 NO DATA
>16:16:04.787 22 06
16:16:05.042 NO DATA
>16:16:11.484 22 07
16:16:11.810 NO DATA
>16:16:24.204 2205
16:16:24.439 NO DATA
>16:16:27.297 220550
16:16:27.548 NO DATA
>16:16:36.001 2208
16:16:36.265 NO DATA
>16:16:40.308 2209
16:16:40.543 NO DATA
>16:16:42.713 220a
16:16:42.956 NO DATA
>16:16:45.438 220b
16:16:45.655 NO DATA
>16:16:47.400 220v
16:16:47.461 ?
>16:16:51.672 220b
16:16:51.894 NO DATA
>16:16:54.001 220c
16:16:54.219 NO DATA
>16:16:55.786 220d
16:16:56.051 NO DATA
>16:16:57.920 220e
16:16:58.188 NO DATA
```

```
>16:16:59.434 220f
16:16:59.661 NO DATA
>16:17:01.923 220g
16:17:01.958 ?
>16:17:07.040 2210
16:17:07.337 NO DATA
>16:17:09.350 221q
16:17:09.356 ?
>16:17:11.072 2211
16:17:11.377 NO DATA
>16:17:14.948 2212
16:17:15.171 NO DATA

... (kráceno)

>16:28:59.614 0901
16:28:59.846 NO DATA
>16:29:11.585 0902
16:29:11.697 7E8 10 14 49 02 01 54 4D 42
7E8 21 48 45 36 31 5A 36 43
7E8 22 32 30 32 32 38 32 35
>16:29:16.988 Disconnected from device
```

C.4 8. 3. 2018 Volkswagen Caddy

Testoval jsem ve voze Volkswagen Caddy III. generace (rok výroby 2014). Použil jsem modrý adaptér ELM327 s průhledným krytem od Marka s MAC adresou `00:1D:A5:68:98:8A`. V mobilním telefonu jsem použil aplikaci *Serial Bluetooth Terminal*. Detaily k měření jsou v sekci 4.3 na straně 86.

```
16:14:08.491 atdp
ISO 15765-4 (CAN 11/500)
>16:17:49.038 ath1
OK
>16:19:42.320 010c
7E8 04 41 0C 00 00
>16:20:54.617 010c
7E8 04 41 0C 00 00
>16:25:06.363 at sh 700
OK
>16:25:16.837 0101
7E8 06 41 01 00 0E E8 00
>16:25:59.528 3e80
NO DATA
>16:26:49.059 at e0
OK
>16:27:02.738 3e80
16:27:02.972 NO DATA
>16:30:51.104 at st ff
16:30:51.124 OK
>16:33:45.911 at sh 773
16:33:45.928 OK
```

```
>16:34:05.949 22f187
16:34:06.991 NO DATA
>16:35:28.270 0101
16:35:29.306 NO DATA
>16:35:55.212 at sh 7df
16:35:55.226 OK
>16:36:01.778 0101
16:36:01.819 7E8 06 41 01 00 0E E8 00
>16:36:16.939 at sh 773
16:36:16.954 OK
>16:36:21.689 0101
16:36:22.725 NO DATA
>16:42:49.082 at ma
16:47:38.350 at sh 773
16:47:38.368 STOPPED
>?
>16:47:53.346 at sh 773
16:47:53.359 OK
>16:48:33.265 220b20
16:48:34.303 NO DATA
>
```

Příloha D

Manuál k aplikaci OBD Robot

Tato příloha usnadňuje uživatelům seznámení s aplikací OBD Robot. Aplikace je dostupná ve formátu `APK` na CD, které je součástí této diplomové práce. Pro použití aplikace je nutné mít OBD-II adaptér, který zprostředkovává komunikaci mezi vozidlem a mobilním telefonem.

D.1 Nutné kroky před použitím aplikace

Pokud jste se rozhodli použít aplikaci OBD Robot, předpokládá se, že máte k dispozici automobil, z něhož budete chtít číst data, nebo který budete chtít ovládat nebo monitorovat. Potom je nutné, abyste měli k dispozici:

- automobil, který má OBD-II port a podporuje OBD-II diagnostiku (všechny v EU vyrobené po roce 2003)
- mobilní telefon s operačním systémem Android 4.3 nebo novější
- adaptér, který lze připojit k vozidlu prostřednictvím OBD-II portu a který komunikuje s mobilním telefonem přes bluetooth

D.2 Instalace aplikace

1. Na CD přiloženém k této práci nalezněte soubor „OBD_Robot.apk“. Umístění souborů na CD je zobrazeno v příloze B.
2. Soubor „OBD_Robot.apk“ zkopírujte do mobilního telefonu.
3. Zkontrolujte v nastavení systému, že máte oprávnění instalovat aplikace z neznámých zdrojů.
4. Najděte pomocí správce souborů aplikaci v mobilním telefonu a poklepáním na ni ji nainstalujte.

5. Potvrďte případný dotaz stiskem tlačítka „OK“.

6. Ujistěte se ve správci aplikací, že aplikace OBD Robot má oprávnění používat bluetooth a zapisovat do interního úložiště telefonu.

D.3 Základní použití

Přesvědčte se, že OBD-II adaptér je dobře zasunutý v OBD-II portu vozidla. Obvykle je to indikováno svícením červené nebo zelené LED diody. Pokud připojujete aplikaci k adaptéru poprvé, je nejprve nutné párovat telefon a adaptér. To se dá udělat v nastavení telefonu v sekci Bluetooth. Heslo k párování je obvykle „1111“ nebo „1234“.

Spusťte aplikaci OBD Robot. Na úvodní obrazovce stiskněte tlačítko *Ukaž párovaná zařízení*. Zobrazí se seznam, vyberte z něj položku, která odpovídá OBD adaptéru, tedy například „OBDII“ nebo „OBD“. Po výběru zařízení se aplikace pokusí připojit k adaptéru. Pokud se to podaří, objeví se obrazovka Terminál. Do řádku nahoře můžete psát libovolný příkaz pro vozidlo nebo pro adaptér. Na obrazovce se dále nachází tyto ikony:


odešle příkaz napsaný v řádku do adaptéru


uloží komunikaci do textového souboru

spustí periodické odesílání příkazu `TP` (každé 3 s)


zobrazí menu aplikace

Při prvním spuštění je nejprve nutné zvolit správný komunikační protokol, který používá vaše vozidlo. Pokud víte, který to je, můžete napsat do terminálu `AT SP X`, kde `X` nahradíte číslem protokolu. Pokud nevíte, odešlete `AT SP 0` a adaptér si protokol najde sám. Ve vozech značky Škoda se používá protokol „ISO 15764-4 (CAN 11/500)“ a má číslo 6 (je nutné odeslat `AT SP 6`).

Po odeslání příkazu se odeslaný příkaz objeví v dolní části obrazovky modrou barvou a odpověď na něj červenou barvou. Nejnovější zpráva je zobrazena vždy nejníže. Pokud se zprávy na obrazovku nevejdou, starší se skryjí na úkor novějších. Starší zprávy je ale možné zobrazit posunem prstu v oblasti zpráv.

Z menu je možné se dostat do obrazovky Aktuální data, nebo se odpojit od adaptéru. Na obrazovce Aktuální data je nutné povolit přepínačem periodické

čtení dat. Po zapnutí se objeví mnoho parametrů a jejich hodnoty, které se budou aktualizovat maximální možnou rychlostí. Obnovovací frekvence dat závisí na automobilu a použitém adaptéru a může se pro různá vozidla velmi lišit. Před odchodem z obrazovky Aktuální data doporučuji přepínačem vypnout čtení dat, šetříte tím výpočetní prostředky telefonu a snižujete pravděpodobnost zahlcení CAN sběrnice v automobilu.

D.4 Řešení běžných problémů

1. Aplikace se nedokáže spojit s adaptérem v autě
   - Zkuste adaptér vysunout z OBD-II portu a zasunout ho zpět.
   - Pokud to nepomůže, zkuste aplikaci OBD Robot vypnout a zapnout.
2. Omylem jsem vymazal celý záznam komunikace v obrazovce Terminál.
   - Vymazaný text nelze obnovit.
3. Proč aplikace neukazuje na konci odpovědí od adaptéru povinný znak „>“?
   - Aplikace tento znak pro přehlednost úmyslně skrývá.
4. Musím přidávat za každou odeslanou zprávu povinný byte „0D“ a jak to udělám?
   - Ne, aplikace povinný byte „0D“ odesílá automaticky.
5. Jaké je výchozí nastavení adaptéru? Ukazuje to aplikace?
   - Výchozí nastavení každého adaptéru je jiné, aplikace to neukazuje. Můžete to zjistit ručně odesíláním AT příkazů.