TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Komunikační síť experimentálního elektrického automobilu

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika
Studijní obor: 3902T005 – Automatické řízení a inženýrská informatika

Autor práce: **Bc. Zoltán Dolenský**
Vedoucí práce: Ing. Pavel Jandura, Ph.D.
Konzultant: Ing. Lukáš Krčmář

Liberec 2022

Prohlášení

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

Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.

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

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

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

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

16. 5. 2022 Bc. Zoltán Dolenský

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání diplomové práce

Komunikační síť experimentálního elektrického vozidla

Jméno a příjmení: **Bc. Zoltán Dolenský**
Osobní číslo: M19000162
Studijní program: N2612 Elektrotechnika a informatika
Studijní obor: Automatické řízení a inženýrská informatika
Zadávající katedra: Ústav mechatroniky a technické informatiky
Akademický rok: **2021/2022**

Zásady pro vypracování:

1. Proveďte rešerši aktuálních prostředků pro komunikační systémy elektrických automobilů.
2. Seznamte se s aktuální realizací elektroniky experimentálního elektromobilu eŠus.
3. Pro experimentální vozidlo navrhněte inovované řešení komunikační sítě řídicích jednotek. Cílem je dosažení přehledné architektury, která bude tvořit testovací platformu pro vývoj softwarových technologií automobilů.
4. Realizujte software centrální řídicí jednotky umožňující řízení pohonů vozidla pomocí komunikační sběrnice.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	40–50 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	Čeština

Seznam odborné literatury:

[1] F. Nouvel, W. Gouret, P. Mazério, and G. Zein, „Automotive Network Architecture for ECUs Communications," 2009, pp. 69–90. doi: 10.4018/978-1-60566-338-8.ch004.

[2] K. Ismail, A. Muharam, and M. Pratama, „Design of CAN Bus for Research Applications Purpose Hybrid Electric Vehicle Using ARM Microcontroller," Energy Procedia, vol. 68, Apr. 2015, doi: 10.1016/j.egypro.2015.03.258.

[3] P. Hank, S. Müller, O. Vermesan, and J. Van Den Keybus, „Automotive Ethernet: In-vehicle networking and smart mobility," in 2013 Design, Automation Test in Europe Conference Exhibition (DATE), Mar. 2013, pp. 1735–1739. doi: 10.7873/DATE.2013.349.

[4] C.-W. Hung, Y.-M. Guan, S.-T. Yu, and W.-H. Hu, „X–Y Platform Synchronous Control with CANopen," J. Robot. Netw. Artif. Life, vol. 6, no. 4, pp. 254–257, Feb. 2020, doi: 10.2991/jrnal.k.200221.005.

Vedoucí práce:	Ing. Pavel Jandura, Ph.D. Ústav mechatroniky a technické informatiky
Konzultant práce:	Ing. Lukáš Krčmář Ústav mechatroniky a technické informatiky
Datum zadání práce:	12. října 2021
Předpokládaný termín odevzdání:	16. května 2022

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. Ing. Josef Černohorský, Ph.D.
vedoucí ústavu

V Liberci dne 12. října 2021

Abstrakt

Tato diplomová práce se zabývá aktuálními trendy v oblasti komunikačních technologií automobilových systémů. V rešerši jsou představeny nejčastěji zastoupené protokoly a různá řešení architektury komunikačních sítí vozidel. Následně je provedena analýza komunikační sítě experimentálního elektrického automobilu Laboratoře elektromobility, zvaného eŠus. Na základě této analýzy je pro toto vozidlo navržena nová architektura komunikační sítě. Motivací pro vznik práce je snaha z eŠusu vytvořit experimentální platformu pro testování technologií, které jsou předmětem stávajícího a budoucího výzkumu laboratoře elektromobility.

Klíčová slova: automobilová platforma, komunikační síť, řídicí jednotka, CAN, protokol, sběrnice

Abstract

This master thesis provides insight into current trends in the field of communication technologies of automotive systems. The research part introduces the most frequently represented protocols and various solutions of vehicle communication network architecture. Subsequently, an analysis of the communication network of the experimental electric vehicle of the Electromobility laboratory, called eŠus, is carried out. Based on this analysis, a new communication network architecture is designed for the vehicle. The motivation for the creation of this thesis is the effort to make the eŠus an experimental platform for testing technologies that are the subject of current and future research of the electromobility laboratory.

Keywords: vehicle platform, communication network, control unit, CAN, protocol, bus

Poděkování

Na úvod chci poděkovat panu Ing. Pavlu Jandurovi, Ph.D., za svěřenou důvěru a umožnění spolupráce na projektu automobilu eŠus, představujícím poměrně komplexní problematiku. Dále děkuji konzultantovi práce Ing. Lukášovi Krčmářovi za pravidelné diskuze vedoucí ke konstruktivnímu řešení dílčích problémů a Ing. Ondřejovi Machovi za vývoj a údržbu hardware využitého pro realizaci praktické části práce. Velké díky patří i mé rodině a blízkým za podporu během celého studia.

Acknowledgment

Many thanks for my dear colleagues from both Czech and German MicroNova, who shared a lot of knowledge regarding automotive solutions with me through regular technical discussions. Especially thanks to Dipl.-Ing. Franz Dengler for the interview explaining many uncertainties related to automotive networking and testing.

Last but not least, thanks to Dipl.-Ing. Michael Hillman from company Embedded Office for enrichment of open source world with state of the art implementation of CANopen protocol stack library and for allowing me to contribute on such project.

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

ISO	International Organization for Standardization
CAN	Controller Area Network
CIA	CAN In Automation
CAN-FD	Controller Area Network with Flexible Data Rate
FDF	Flexible Data Format
AUTOSAR	Automotive Open System Architecture
LIN	Local Interconnect Network
WIFI	Wireless Fidelity
LTE	Long Term Evolution
PID	Protected Identifier
UART	Universal Asynchronous Receiver-Transmitter
SOA	Service Oriented Architecture
MMDS	Multi Motor Drive System
DPFP	Double Precision Floating Point
BMS	Battery Management System
TDMA	Time Division Multiple Access
TSN	Time Sensitive Networking
API	Application Programming Interface
OBD	On Board Diagnostics
SDO	Service Data Object
PDO	Process Data Object
LSS	Layer Settings Services
DMA	Direct Memory Access
GPIO	General Purpose Input/Output
ROM	Read Only Memory

Slovník použitých pojmů

protokol Konvence nebo standard, podle kterého probíhá přenos informace mezi dvěma nebo více koncovými body.

sběrnice Metalické nebo optické médium zprostředkující spojení mezi dvěma nebo více koncovými body za účelem přenosu informace. Formát informace přenášené po sběrnici je specifikován protokolem.

rozhraní Prostředek vzájemné interakce softwarových nebo hardwarových komponent. Typicky se jedná o popis struktury registrů hardwarové periferie nebo soubor datových typů a funkcí softwarové knihovny. V případě software bývá rozhraní označováno také jako API.

ISO-OSI model Referenční příklad řešení komunikace v počítačových a telekomunikačních sítích. Rozděluje komunikaci do několika nezávislých vrstev za účelem dosažení jejich nahraditelnosti. Využívá se pro návrh a standardizaci komunikačních protokolů.

master-slave Model síťové komunikace, kde jeden master uzel řídí provoz slave uzlů a určuje podmínky, za jakých smí přistupovat k přenosovém médiu za účelem vysílání.

publish-subscribe V překladu vydávat-odebírat. Vydavatel při vysílání zprávy přímo nespecifikuje jejího příjemce, ale charakterizuje ji jako určité téma. Podobným způsobem odběratel vyjadřuje zájem o jedno nebo více témat bez vědomí, od kterého uzlu zprávu obdrží.

unit test Způsob testování software, kterým jsou testovány jednotlivé jednotky zdrojového kódu, sady jednoho nebo více počítačových programů spolu s přidruženými kontrolními údaji, postupy a provozní postupy. Cílem je zjistit, zda je testovaný objekt vhodný pro použití.

Seznam použitých jednotek

B	1 byte
m	1 metr
kbps	1 kilobit za sekundu
Mbps	1 megabit za sekundu
Gbps	1 gigabit za sekundu

1 Úvod

Elektronické systémy moderních automobilů zahrnují značné množství prvků zajišťujících různorodou funkcionalitu, od jednoduchých snímačů a akčních členů až po výkonné výpočetní jednotky. Propojení jednotlivých komponent vozidla tak, aby bylo z hlediska latence a bezpečnosti dosaženo optimálních podmínek pro sdílení informací různé priority, představuje náročný úkol. Proto lze v posledních letech ve veřejně dostupných zdrojích zaznamenat snahu výrobců automobilů o adaptaci nových technologií vedoucích ke zjednodušení architektury komunikačních sítí vozidel při zajištění vysoké konektivity. Vzhledem k pestrému charakteru informací sdílených mezi jednotlivými uzly palubní sítě lze v každém moderním automobilu nalézt různé typy komunikačních protokolů, z nichž každý je vhodný pro určité aplikace. Aktuální architektura automobilových platforem například umožňuje nasazení protokolu Ethernet společně s konvenčními síťovými technologiemi, například Wifi či LTE. Díky stále větším nárokům na software, poskytující zákazníkovi rozličné služby, je tedy na výzkum, standardizaci a aplikaci nových komunikačních technologií v posledních letech kladen vysoký důraz. V rešerši této práce jsou představeny protokoly nejčastěji zastoupeny v moderních vozidlech.

Mezi aktuálně nejvíce diskutované trendy v automobilovém průmyslu patří také elektromobilita. Ačkoliv trhu stále dominují spalovací motory, automobily s elektrickým pohonem se každým rokem těší vyšší popularitě. Na mnoha univerzitách proto vznikla snaha o realizaci nejrůznějších experimentálních elektrických vozidel za účelem výzkumu technologií spojených právě s tímto odvětvím automobilového průmyslu. V rámci budování Laboratoře elektromobility vznikl i na Technické univerzitě v Liberci experimentální elektromobil, zvaný eŠus. Toto vozidlo prošlo během své existence mnoha úpravami a sloužilo také k reprezentaci univerzity na různých společenských událostech. Primárním účelem eŠusu je však testování algoritmů regulace elektrických pohonů nebo procesu nabíjení trakční baterie. Vzhledem ke stavu elektronických systémů a konfiguraci hnacího ústrojí vozidla však již některá měření není možné provádět. Proto vznikl požadavek na realizaci nové verze elektromobilu, zaměřující se primárně na elektroniku a její propojení. Praktická část této práce se zabývá právě návrhem nové komunikační sítě a implementací její dílčí části. Cílem je z elektromobilu eŠus vytvořit flexibilní a rozšiřitelnou platformu, umožňující právě testování rozličných progresivních technologií.

2 Teoretická část

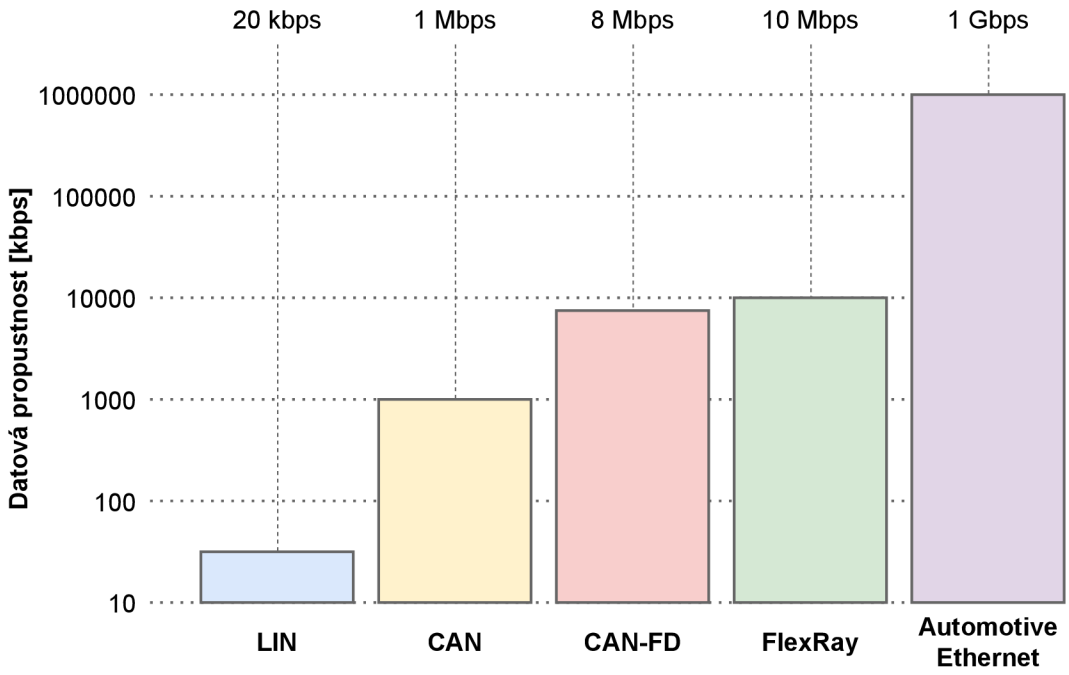
Teoretická část práce obsahuje rešerši protokolů a koncepcí komunikační architekury nejčastěji zastoupených v infrastruktuře moderních automobilových platforem. Cílem není detailně popsat jednotlivé technologie a jejich parametry, nýbrž informovat o tom, jaká je motivace jejich vzniku a v jakých aplikacích jsou typicky nasazeny. Bohužel kvůli utajení nelze srovnat konkrétní řešení komunikačních sítí jednotlivých modelových řad automobilů dostupných na trhu. Vychází se tedy z informací z veřejných zdrojů, dostupných převažně díky společnostem zabývajících se vývojem elektronických komponent pro automobilový průmysl. Rešerse také popisuje protokol CANopen, jehož implementace tohoto byla také použita pro přenos informací v nové architektuře komunikační sítě mezi řídicími jednotkami realizovanými v rámci experimentální části této práce.

Následně je představen experimentální elektromobil eŠus vyvinutý na Technické univerzitě v Liberci. Kapitola věnovaná jeho problematice zahrnuje historii vozidla a stručný popis klíčových komponent. Jako základ pro experimentální část práce jsou analyzovány nedostatky jeho aktuální architektury komunikační sítě a stanoveny požadavky na realizaci nové.

Vzhledem k obtížné dostupnosti informací ohledně reálné situace v automobilovém průmyslu byl za účelem objasnění témat týkajících se právě komunikačních technologií automobilu osloven specialista ze společnosti MicroNova AG. Ta se zabývá testováním elektronických systémů budoucích automobilů. Oslovený odborník žádost přijal a přepis poskytnutého rozhovoru lze nalézt v příloze A.1.

2.1 Komunikační technologie automobilů

Elektronika automobilu v dnešní době představuje komplexní systém zahrnující mnoho dílčích funkcí, od řízení osvětlení až po pokročilé algoritmy autonomní jízdy. V posledních letech také narůstá zájem o připojení automobilů k internetu za účelem jejich interakce mezi sebou či s okolní infrastrukturou. Proto lze v elektronických systémech automobilových platforem nalézt rozličné druhy komunikačních protokolů sloužících ke sdílení informací mezi jednotlivými řídicími jednotkami. Tyto protokoly se liší nejen datovou propustností (viz obr. 2.1) nebo cenou nasazení, ale například i způsobem řízení přístupu k přenosovému médiu [1]. Následující část práce se zabývá aktuálně v automobilovém průmyslu nejčastěji diskutovanými protokoly CAN-FD, LIN, FlexRay a Automotive Ethernet.



Obrázek 2.1: Srovnání komunikačních protokolů vozidel dle datové propustnosti

2.1.1 CAN-FD

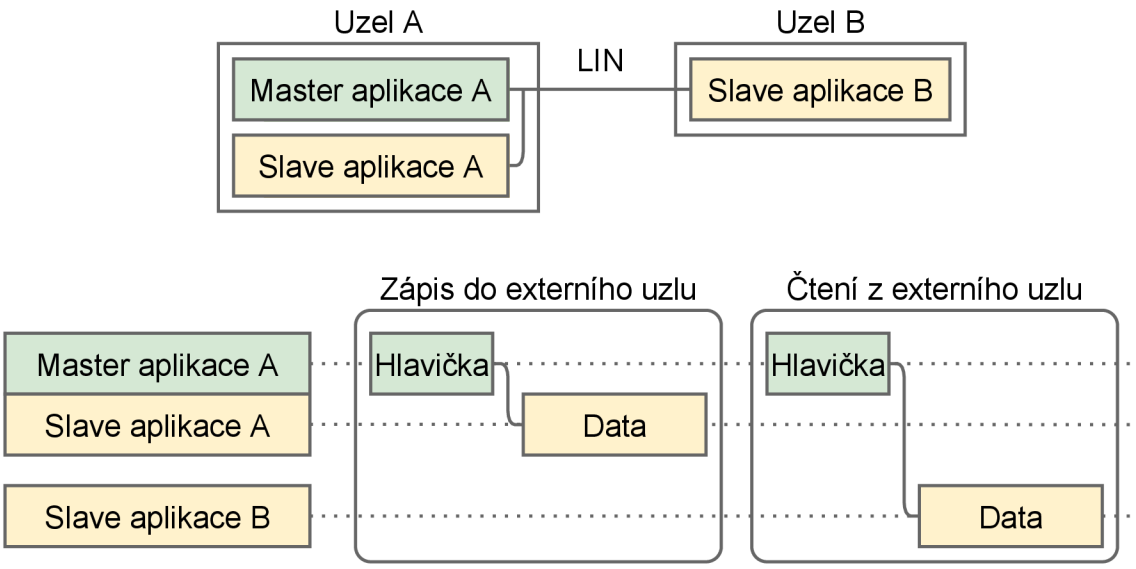
Protokol ISO 11898-1 neboli CAN-FD vznikl v roce 2012 jako rozšíření stávajícího protokolu ISO 11898 (CAN). Motivací k tomuto rozšíření byl narůstající objem informací sdílených mezi řídicími jednotkami vozidel, obzvláště v kontextu aktualizace jejich firmware. Vzhledem k velikosti binárních obrazů firmware jednotek se při použití klasického CAN, omezeného velikostí jednoho datového rámce na 8 B, prodlužoval proces aktualizace jednotky i na několik minut. Hlavním důvodem není nízká přenosová rychlost, ale segmentace přenášených dat do mnoha rámců. Každý rámec navíc musí obsahovat informace týkající se režie přenosu, lišící se dle použitého transportního protokolu. CAN-FD umožňuje vysílání rámce o velikosti až

64 B, tedy osmkrát více než v případě klasického CAN. Tímto se výrazně snižuje režie potrebná k segmentovanému přenosu informací a samotný proces aktualizace firmware probíhá rychleji [25]. Protokol je kompatibilní, nikoliv však zaměnitelný se starším standardem. Arbitráž na sběrnici probíhá stejnou rychlostí jako v případě CAN (maximálně 1 Mbit/s u CAN 2.0B), při vysílání datové části rámce lze aktivovat zrychlení přenosu až na 8 Mbit/s. K rozlišení mezi klasickým a FD rámcem slouží bit FDF v jeho hlavičce. Ten standard CAN označuje jako rezervu pro budoucí rozšíření, tudíž jej nijak nevyužívá a na rámce s jeho hodnotou rovno 1 nereaguje. Díky tomu lze CAN i CAN-FD zařízení připojit na stejnou sběrnici aniž by došlo ke kolizím či narušení stability komunikace [2]. Protokol CAN-FD tvoří základ komunikační sítě automobilu a typicky je nasazován v systémech reálného času, které zodpovídají například za úlohy spojené s převodovkou či přístrojovým panelem vozidla.

2.1.2 LIN

Standard ISO 17987-Local Interconnect Network (LIN) popisuje jednoduchý a cenově dostupný protokol založený na komunikačním modelu master-slave. Zařízení se ke sběrnici připojují jediným vodičem, komunikace je tedy asynchronní a poloduplexní. K jednomu kanálu sběrnice může být připojeno zároveň 16 zařízení. Aktuální specifikace podporuje přenosové rychlosti až do 20 kbit/s na vzdálenost 40 m. Přenos komunikačních rámců zahajuje vždy master zařízení vysláním hlavičky a identifikátoru dotazu (PID), v reakci na který oslovené slave zařízení musí v určeném čase odpovědět odvysíláním příslušného datového obsahu. V případě zápisu master vysílá celý rámec včetně dat a slave zařízení na dotaz neodpovídá. Přenos informací je tedy řízen časově zavedením fixní lhůty odpovědi. Pokud slave po dobu stanovenou specifikací neodpoví na dotaz, přenos je vyhodnocen jako neúspěšný. Zvláštním případem přenosu je takzvaný event-triggered frame, tedy aperiodický přenos vyvolaný nějakou událostí v jednom z uzlů. V takové situaci může odpovědet na dotaz více slave jednotek, vlivem čehož dojde ke kolizi na sběrnici. Master musí být schopen tuto kolizi detekovat a následně postupně oslovit všechny slave, které mohly kolizi způsobit. Tato sekvence je pevně stanovena konfigurací master jednotky [3].

Protokol LIN je nasazován v aplikacích jednoduchých elektronických prvků vozidla, nejčastěji snímačů a akčních členů, které nevyžadují vysokou datovou propustnost. Doplňuje tak protokol CAN jako cenově dostupná a méně komplexní alternativa. K implementaci LIN zařízení totiž stačí mikrokontrolér s periferií UART doplněn o budič sběrnice. Takový hardware je cenově dostupnější, než pokročilé mikrokontroléry s podporou CAN komunikace [4].



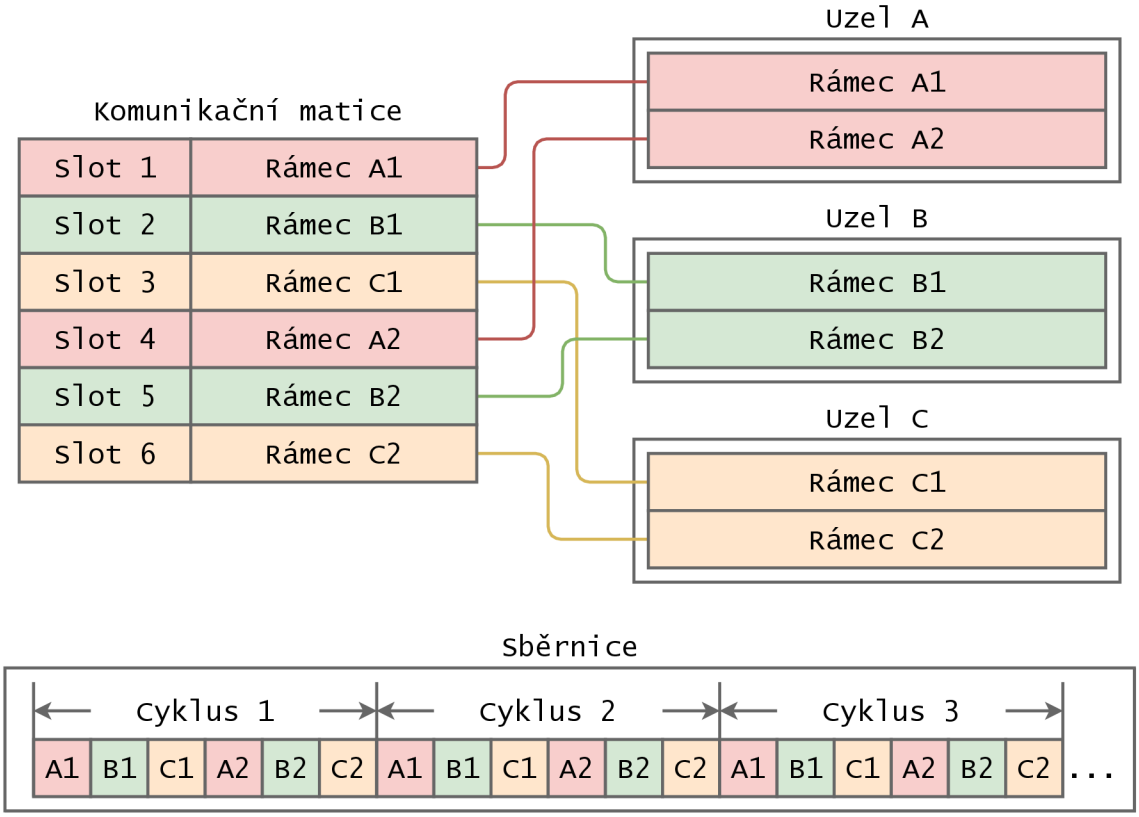
Obrázek 2.2: Aplikační model komunikace pomocí protokolu LIN. Oddělením master a slave funkcionality je dosaženo jejich vzájemné nezávislosti, což usnadňuje implementaci. V případě master uzlu mají obě aplikace přístup k přenosovému médiu, komunikaci mezi nimi lze tedy zaznamenat i na sběrnici. Toho je využíváno pro zápis dat do některého z externích slave uzlů.

2.1.3 FlexRay

Vývoj protokolu FlexRay je spojen s přísnými požadavky na stabilitu a bezpečnost komunikace v kritických systémech vozidla, mezi které patří například hnací ústrojí nebo elektronické brzdy. Výrazným rozdílem proti protokolu CAN je použití metody TDMA, tedy časově řízeného přístupu k přenosovému médiu (viz obr. 2.3) tvořenému krouceným párem metalických vodičů nebo optickou linkou. Komunikace probíhá periodicky a přenos dat je rozdělen na 64 cyklů. Každý komunikační cyklus se skládá ze statického a dynamického segmentu. Ve statickém segmentu má každý uzel na sběrnici přidělena časová okna zvaná sloty, kdy je očekáváno vysílání příslušného rámce. V případě, že příslušný uzel nemá v daném slotu připravena data, vysílá takzvaný null frame, tedy prázdný rámec. Díky tomuto mechanismu lze snadno detekovat výpadek jednoho z uzlů. Maximální velikost jednoho komunikačního rámce protokolu FlexRay může dosahovat hodnoty 254 B. Ve statickém segmentu mají všechnny rámce stejnou velikost pro zajištění konstatní doby trvání jednotlivých slotů. V případě dynamického segmentu lze již velikost vysílaných rámců měnit. Omezení dynamického segmentu však spočívá v maximální době jeho trvání, stanovené konfigurací. Statický segment tedy slouží primárně k vysílání procesních dat, zatímco dynamický umožňuje diagnostiku či aktualizaci software, aniž by došlo k zahlcení sběrnice [5].

Díky časově řízenému přístupu na sběrnici je FlexRay deterministickým protokolem v dnešní době nasazovaným převažně v systémech trakce či řízení. Ačkoliv s integrací FlexRay do automobilových platforem narůstají výrobní náklady, značně se také zvyšuje spolehlivost a bezpečnost vozidel tímto protokolem vybavených [6].

Proti CAN výrazně vyšší přenosová rychlost FlexRay společně s nárustem výpočetního výkonu mikrokontrolérů také umožňuje sloučení více palubních systémů do jedné řídicí jednotky, což vede k výraznému zjednodušení komunikační infrastruktury vozidla.

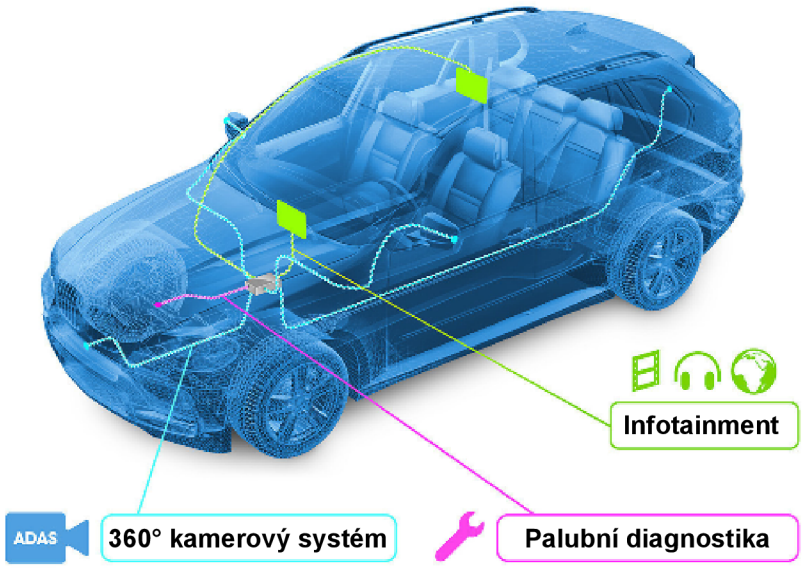


Obrázek 2.3: Princip TDMA v síti FlexRay. Základem je komunikační matice obsahující informace o všech vysílaných rámcích, jejich obsahu a jednotkách, kterým je přidělena úloha vysílače či příjimače.

2.1.4 Automotive Ethernet

Velkou nevýhodou doposud zmíněných komunikačních protokolů je nedostatečná kompatibilita s konvenční síťovou infrastrukturou. Jelikož koncepty budoucích modelových řad automobilů počítají s vysokou konektivitou, vznikla snaha o standardizaci Ethernetu pro použití právě v automobilovém průmyslu. Ethernet byl poprvé nasazen ve vozidlech BMW v roce 2013 (viz obr. 2.4). První verze standardu 10BASE-T1 vznikla také v roce 2013 a popisuje fyzickou vrstvu OSI modelu schopnou stabilního provozu ve vozidlech při zajištění datové propustnosti 10 Mbit/s. Později byly standardizovány další rozšíření protokolu, aktuální 1000BASE-T1 (2016) nabízí přenosovou rychlost až 1 Gbit/s na vzdálenost 15 m. Přenosovým médiem je opět kroucený pár metalických vodičů [7]. Ethernetu nadřazené protokoly nepodstoupily žádnou změnu a zůstavají tedy kompatibilní s konvenční síťovou infrastrukturou. Na straně výrobců automobilů došlo následně k začlenění etherne-

tové komunikace do standardu AUTOSAR, kde překlad komunikace mezi řídicími jednotkami, založené na datových signálech (viz kapitola 3.2.2), zajišťuje modul SocketAdaptor (SoAd) [8]. Ethernet v moderním automobilu nachází uplatnění primárně v páteřních sítích s vysokou datovou propustností, kamerových a lidarových systémech či v infotainmentu. Díky jeho kompatibilitě s protokoly jako je Wifi nebo LTE lze zajistit připojení vozidla k internetu, což otevírá nové možnosti vývoje software pro elektroniku moderních vozů. Příkladem mohou být vzdálená aktualizace řídících jednotek či technologie spojené s autonomním řízením. V poslední době se také diskutuje o možnosti nahradit Ethernetem veškeré klasické protokoly přítomné v automobilu. Vzhledem k charakteristice ethernetové komunikace však nelze aktuálně zajistit latence potřebné pro provoz některých palubních systémů. Problematické jsou také možnosti deterministického řízení přístupu ke sběrnici a synchronizace času mezi jednotlivými uzly. Tato problematika udávala směr výzkumu automobilových technologií v posledních letech a zasloužila se o adaptaci nových technologií jako například Time Sensitive Networking (TSN), umožňující prioritizaci přenášených paketů. V síti podporující TSN lze přerušit probíhající vysílání v případě požadavku na přenos paketu s vyšší prioritou. Tuto funkcionalitu zajišťuje síťový přepínač. Od TSN se očekává snížení latence přenášených dat na hodnoty v řádu jednotek mikrosekund [9].



Obrázek 2.4: BMW x5 2013, představující historicky první veřejně známé nasazení Ethernetu v palubní síti vozidla. Použitá technologie zvaná OPEN (One Pair Ethernet) tehdy zajišťovala primárně komunikaci s kamerovými systémy automobilu v reálném čase [10].

2.2 Síťová infrastruktura vozidel

Jak již bylo zmíněno, vlivem stále vyšších požadavků na komfort či bezpečnost automobilů lze v posledních letech zaznamenat značný nárůst koplexity jejich elektronických systémů. Obzvlášť u prémiových modelů může počet řídicích jednotek dosahovat i vyšších desítek kusů. Princip těchto zařízení je založen na sdílení datových signálů, ať už interně v rámci jednoho modulu, tak s ostatními příslušníky sítě prostřednictvím komunikačních sběrnic. Signály reprezentují vstupní a výstupní hodnoty různých datových typů potřebné pro software běžící v jednotkách [11]. Následující kapitola popisuje přístupy k návrhu komunikačních sítí vozidel a jejich v důsledku narustajících požadavků na výbavu moderního automobilu. Pro pochopení textu a přiložených obrázků je nejprve nutné v kontextu této kapitoly definovat pojmy:

I/O Z anglického input/output (vstup/výstup), úroveň software, která zajišťuje prostřednictvím hardware cílového zařízení měření vstupních veličin a ovladání výstupů. Takové zařízení tedy zastává funkci snímače či akčního členu.

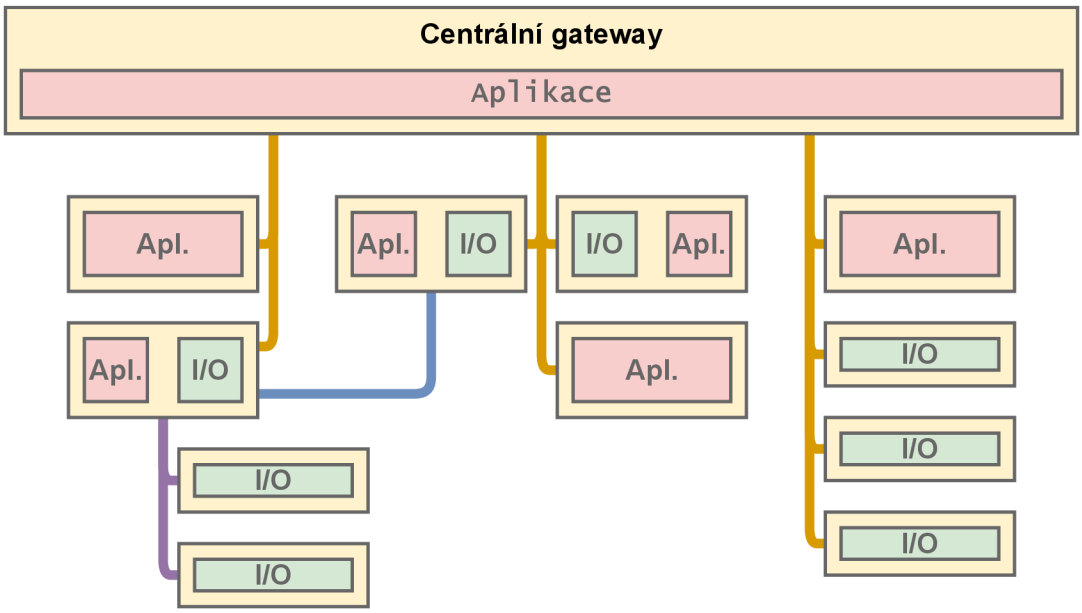
Aplikace Logická úroveň software zodpovědná za sdílení informací či zpracování vstupních a výpočet výstupních hodnot. Je přenositelná a v ideálním případě nezávislá na hardware či operačním systému. Se svým okolím interaguje prostřednictvím proměnných uložených typicky v operační paměti zařízení.

Poznámka: Jako I/O nejsou na obrázcích označeny ovladače komunikačních rozhraní, jejichž implementace je samozřejmě vyžadována u všech zařízení včetně výhradně aplikačních jednotek.

2.2.1 Distribuovaná architektura

Nejznámějším a historicky nejčastěji zastoupeným konceptem síťové infrastruktury vozidla je distribuovaná architektura. Z pohledu software jsou I/O i aplikace distribuovany do mnoha vzájemně komunikujících modulů (viz obr. 2.5). Každá z těchto jednotek představuje implementaci jedné dílčí funkce automobilu, příkladem mohou být tempomat či posilovač řízení. Z důvodu rozdílných požadavků na odezvu či bezpečnost jsou jednotky rozděleny do několika menších podsítí, starajících se o dílčí část funkcionality vozidla [12]. Typickým příkladem jsou například podsítě pohonu, podvozku, komfortu a infotainmentu. Jejich propojení je zajištěno pomocí takzvané gateway jednotky. Ta slouží k přeposílání důležitých zpráv mezi uzly v různých podsítích. Gateway jednotka zároveň umožňuje díky přístupu do celé sítě automobilu servisní zásahy a diagnostiku. Aplikace gateway typicky slouží pouze k přeposílání požadovaných zpráv mezi jednotlivými podsítěmi, neobsahuje žádnou službu rozhodovací úrovně software [13]. Neustálé rozšiřování funkcionality moderních automobilů by při zachování aktuálních konvencí vedlo ke značnému nárustu komplexity přítomné elektroniky a množství kabeláže potřebné k jejímu propojení. Klasická distribuovaná architektura zde již naráží na hranice svých možností

a pro další vývoj bylo tedy nutné nalézt alternativu, umožňující zjednodušení celkové síťové infrastruktury vozidla. Koncept distribuovaného software ssebou také nese bezpečnostní rizika, jelikož s nárustem množství řídicích jednotek roste pravděpodobnost selhání systému jako celku.



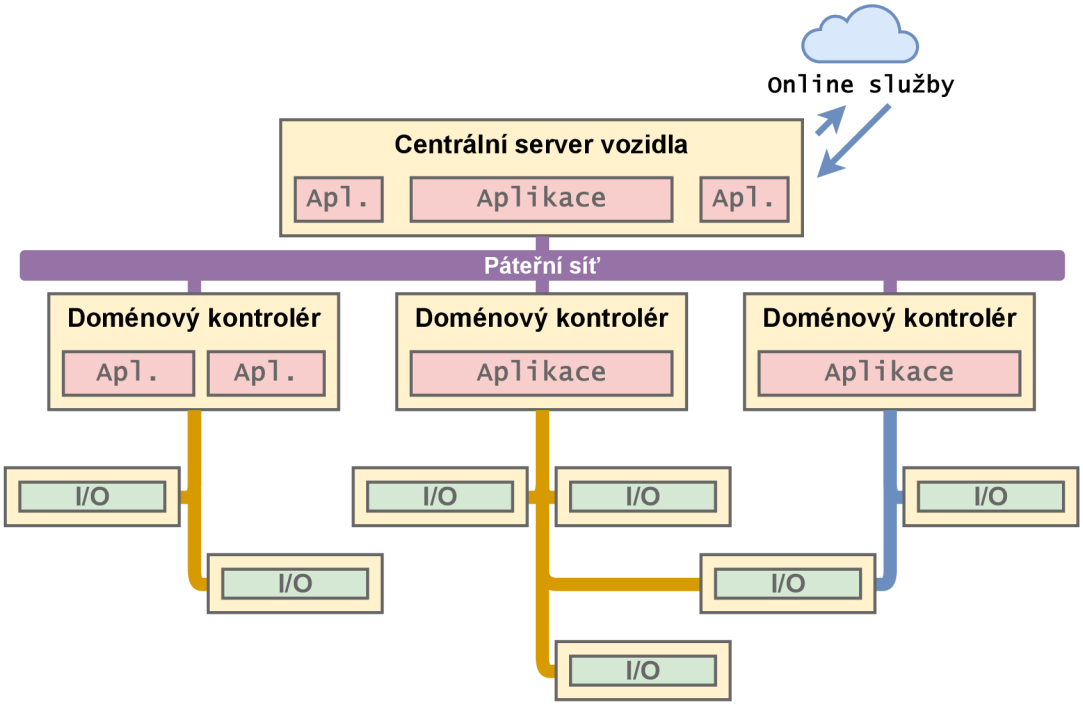
Obrázek 2.5: Příklad struktury distribuované architektury. Některé z uzlů mohou vyžadovat vzájemné propojení i mimo gateway jednotku z důvodu snížení latence přenosu určitých procesních informací.

2.2.2 Doménová architetura

Začlenění Ethernetu a jeho nadřazených protokolů do portfolia komunikačních technologií vozidel otevřelo nové možnosti vývoje software pro palubní elektroniku. Výrazným milníkem byla například adaptace softwarové architektury orientované na služby (SOA) ve vysokoúrovňových systémech automobilů. Jedná se o technologii dobře známou díky rozmachu webových aplikací, využívající komunikační model klient-server [14]. Služby jsou definovány jako programy spuštěné na cílovém zařízení buď přímo v hostitelském operačním systému nebo v případě použití stále více populárních technologií kontejnerizace v rámci izolovaného prostředí kontejneru [15]. Každá ze služeb reprezentuje dílčí funkcionalitu cílového systému a interaguje se svým okolím prostřednictvím virtuálního či fyzického síťového rozhraní. Vyhodou takového řešení je jeho vysoká flexibilita, jelikož lze dynamicky určovat, které služby jsou v daném okamžiku aktivní. V případě aktualizace jednotlivých služeb dochází k přepisu pouze dílčí části software, což výrazně redukuje čas i režii potřebnou k přenosu dat. Připojení automobilů k internetu umožňuje zajistit některé ze služeb prostřednictvím cloudových řešení [14].

Nasazení SOA a slučování dílčích částí distribuovaného firmware, motivovaného snahou o snížení počtu řídicích jednotek, napomohlo ke vzniku doménové architektury. Její pricip spočívá, podobně jako v případě 2.2.1, v rozdělení komunikační síte

na menší částečně autonomní části zvané domény. Ty zajišťují určitou funkcionalitu automobilu, například regulaci pohonů či služby infotainmentu. Výrazným rozdílem však je přesun většiny aplikací příslušné domény do výkonných kontrolérů vzájemně propojených páteřní sítí, využívající protokolů s vysokou datovou propustností (FlexRay, Ethernet). Ostatní uzly doménových podsítí pak zastávají pouze funkci I/O, tedy snímače nebo akčního členu. Doménový kontrolér, vybaven zpravidla vícejádrovým mikrokontrolérem s operačním systémem reálného času, zajišťuje veškeré aplikační úlohy pro příslušnou doménu. Aplikace vyšší úrovně pak zprostředkovává, právě díky adaptaci SOA, centrální server vozidla formou služeb. Jeho úkolem je také zajištění konektivity za účelem přístupu ke službám mimo vozidlo.



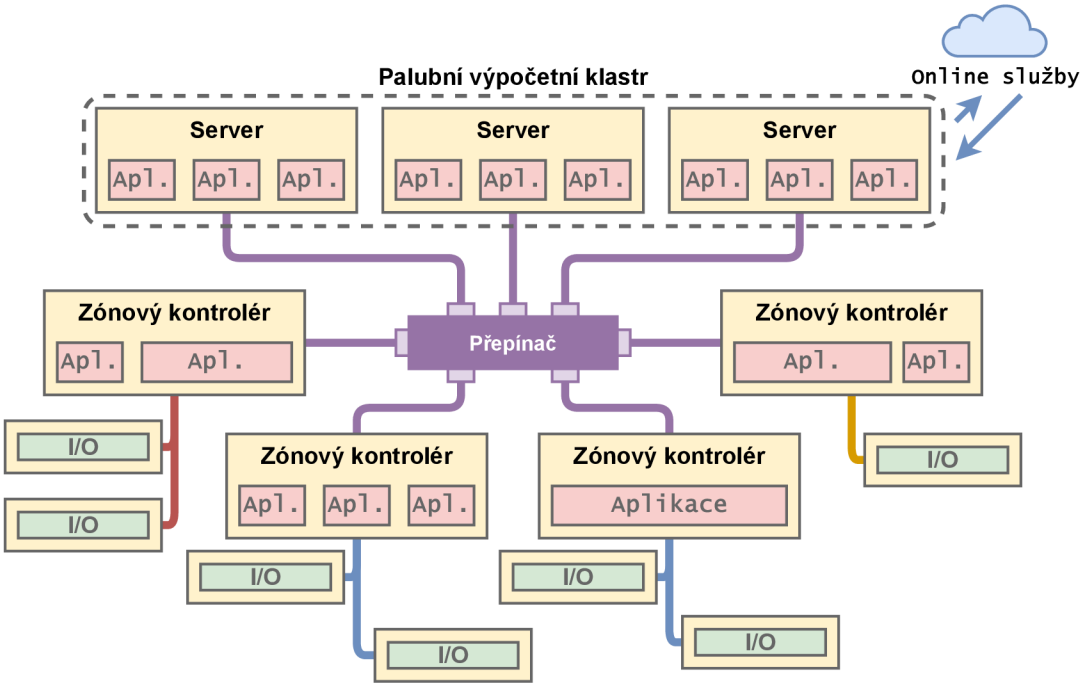
Obrázek 2.6: Příklad doménové architektury. Aplikace doménových kontrolérů zajišťuje primárně úlohy reálného času a zpracování dat z doménové sítě a jejich normalizaci pomocí konfiguračních či kalibračních parametrů. Centrální server pak realizuje přístup k online službám či rozličné funkce komfortu a infotainmentu.

2.2.3 Zónová architektura

Ačkoliv jsou Ethernet, SOA či online služby přítomny již ve vozidlech s doménovou či dokonce i distribuovanou architekturou (formou dodatečného rozšíření výbavy), s plným využitím jejich potenciálu je počítáno až v případě architektury zónové (viz obr. 2.7). První veřejně známou implementaci této architektury představila společnost Tesla se svým elektromobilem Model Y. Díky nasazení zónové architektury došlo k výrazné redukci kabeláže vozidla na hodnoty v řádu stovek metrů [26]. Právě problematika kabeláže, související s cenovou dostupností návrhu, vede společnosti k diskuzi o zónové architektuře jako o možném nástupci doménové architek-

tury v budoucích platformách. Například společnost Cariad, spadající do koncernu Volkswagen, představila v roce 2021 nadcházející platformu Trinity, využívajících celé spektrum moderních síťových technologií k dosažení vysoké konektivity a schopnosti plně autonomní jízdy, dokonce v kontextu komerčního vuyžití (představen byl koncept taxi služby bez obsluhy) [27]. Tesla dnes již nabízí ve svých automobilech zpoplatněné rozšíření funkcionality skrze balíčky aktualizací software, distribuované vzdáleně přes internet [28]. Přítomnost konvenčních síťových technologií, jako jsou právě Ethernet, WiFi a LTE, umoňuje také interakci s okolními automobily či infrastrukturou (semafory, nabíjecí stanice).

Na straně vozidla se o poskytování většiny služeb stará výpočetní klastr skládající se z několika vysoce výkonných jednotek. Prostor automobilu je rozdělen na několik zón spravovaných zónovým kontrolérem (logickou úvahou se nabízí rozdělení na čtyři kvadranty). K němu jsou v rámci jedné nebo více podsítí připojeny všechny snímače a akční členy potřebné pro daný segment vozidla. Vzájemné propojení zónových kontrolérů pak zajišťuje opět páteřní vysokorychlostní ethernetová síť, realizovaná pomocí centrálního přepínače [29]. Většina aplikací je realizována formou služeb v centrálním klastru, zónový kontrolér zajišťuje lokálně funkcionalitu specifickou pro daný segment a vyžadovanou v reálném čase. Uživateli jsou zpřístupněny značné možnosti personalizace komfortu či zábavy a prostředníctvím online služeb lze nabídnout různá rozšíření funkcionality prostředníctím "app store", tedy internetových obchodů, podobně jako v případě chytrých telefonů. Tato skutečnost transformuje automobil z pouhého dopravního prostředku na softwarově orientovaný síťový produkt, podobně jak je dnes běžné na trhu spotřební elektroniky [30].



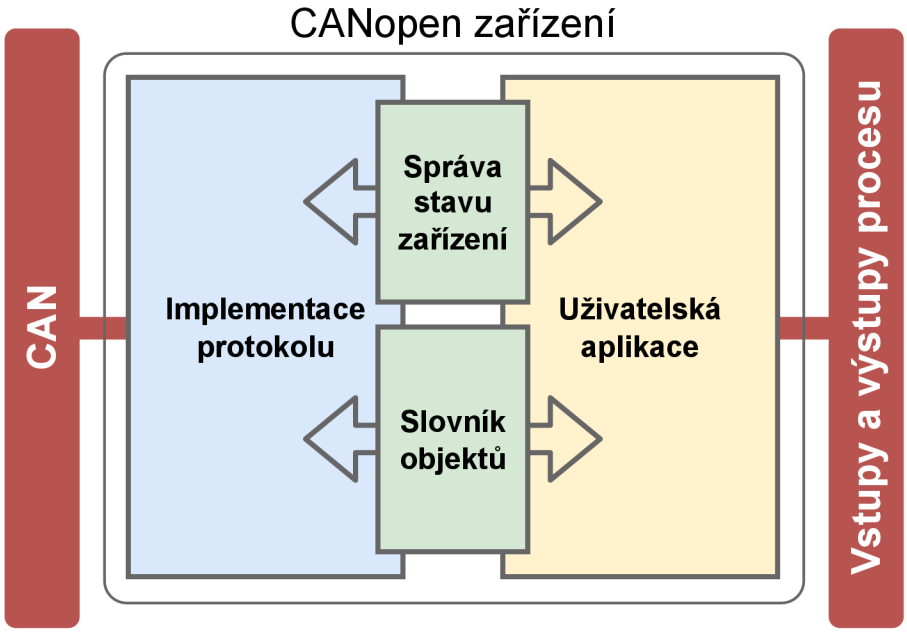
Obrázek 2.7: Příklad zónové architektury. Kritickým bodem je v tomto případě centrální síťový přepínač, zajišťující spojení uzlů páteřní sítě.

2.3 Komunikační protokol CANopen

Vlivem nárůstu popularity protokolu CAN a jeho postupné integrace i v aplikacích mimo automobilový průmysl vznikla v minulosti potřeba standardizované aplikační vrstvy (dle referenčního ISO-OSI modelu) pro zařízení komunikující právě prostřednictvím CAN. Cílem je usnadnění vývoje, konfigurace a diagnostiky těchto zařízení. Standard protokolu CAN totiž popisuje pouze fyzickou a linkovou vrstvu, čímž poskytuje značnou flexibilitu při návrhu komunikační sítě.

V případě vývoje komplexních systémů, jako jsou právě automobily nebo například výrobní linky, se však může bez stanovených pravidel postupným přidáváním komunikačních rámců výrazně zkomplikovat výsledné řešení. Komunikační rámec CAN je navíc omezen velikostí 8 B a v případě nutnosti výsílání větších paketů typických pro diagnostiku nebo aktualizace je již nutná přítomnost transportní vrstvy. Neposledním problémem je chybějící network management funkcionalita, tedy vzdálená správa stavu jednotlivých uzlů sítě.

CANopen spadá do rodiny takzvaných fieldbus protokolů používaných primárně v průmyslových aplikacích. Jedná se o protokol založený na takzvaných objektech organizovaných pomocí slovníku objektů, sloužícího zároveň jako aplikační rozhraní. Každý z objektů je identifikován pomocí jedinečného indexu o velikosti 16 bit a subindexu s velikostí 8 bit používaného v případě polí a struktur. Slovník objektů obsahuje veškeré informace týkající se daného zařízení včetně stávající konfigurace nebo identifikace jeho výrobce. Přitomny jsou také informace specifické pro uživatelskou aplikaci.



Obrázek 2.8: Aplikační model implementace CANopen zařízení. Uživatelská aplikace interaguje s implementací protokolu pouze skrze softwarové rozhraní a nemusí se tedy zabývat aktuálním provozem na přenosovém médiu.

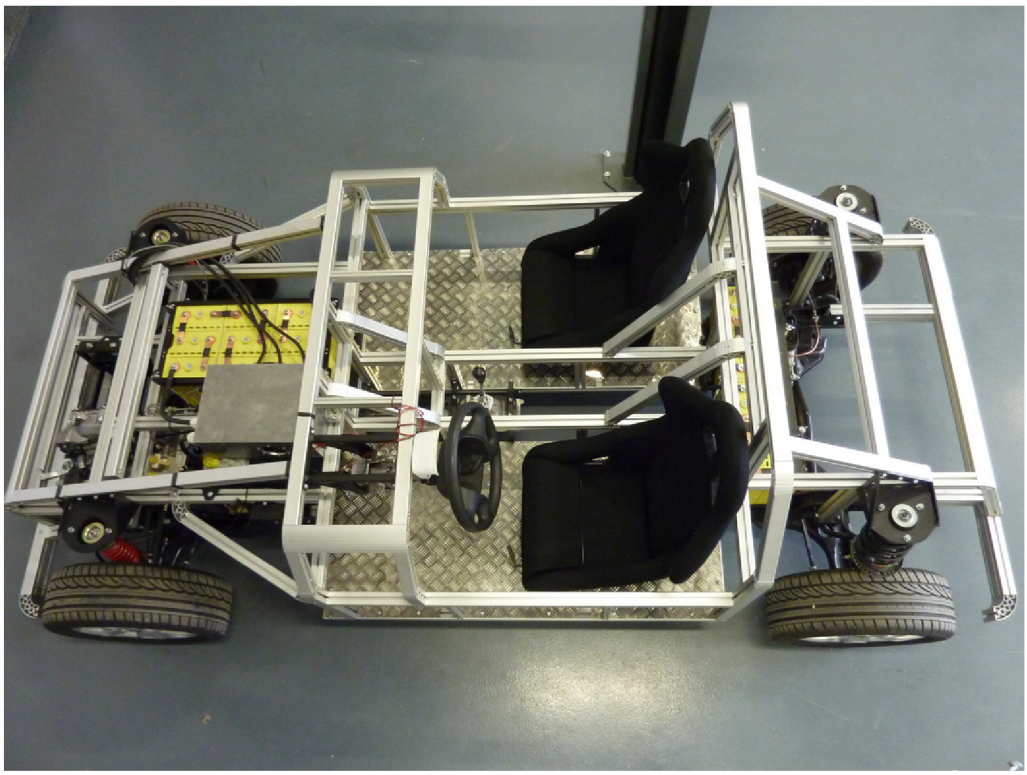
Protokol využívá komunikační model master-slave, na sběrnici se tedy typicky vyskytuje jeden master, další zařízení fungují jako slave. Úkolem master uzlu je řídit síťový provoz, detekovat závady a případně zprostředkovat uživateli diagnostický přístup do sítě. Existuje i varianta CANopen umožňující přítomnost několika master uzlů, které si mezi sebou předávají dle potřeby token udělující držiteli právo řídit síťový provoz. Tohoto je využíváno přimárně v redundatních systémech, na které jsou kladeny přísné bezpečnostní požadavky [16].

Ačkoliv je CANopen označován za aplikační vrstvu ISO-OSI modelu, jsou v něm zahrnuty dílčí protokoly zajišťující funkcionalitu nižších vrstev. Například protokol SDO, který je používán ke vzdálenému přístupu do slovníku jednotlivých uzlů, umožňuje segmentovaný přenos dat o velikosti vyšší než dovoluje komunikační rámec protokolu CAN. Lze jej tedy označit za transportní vrstvu. SDO využívá komunikační model klient-server a každý uzel dle standardu musí implementovat serverovou část, která vysílá data pouze pokud je explicitně požádán klientem. Funkcionalitu síťové vrstvy, zajišťující automatickou identifikaci a adresaci uzlů na sběrnici, zprostředkovává protokol LSS. Pomocí něj lze zároveň měnit přenosovou rychlost komunikace. Pro sdílení dat v rámci procesů běžících v reálném čase slouží protokol PDO, který proti SDO omezuje maximální velikost vysílaných dat na jeden komunikační rámec. Dosaženo je tím snížení latence a režie potřebné v případě použití transportního protokolu. Způsob vysílání PDO rámců a jejich obsah lze nastavit zápisem do příslušných indexů slovníku objektů. Přenos lze zahajovat synchronně i asynchronně vůči ostatním uzlům sítě a vysílající strana v případě PDO neočekává potvrzení vysílaných dat, čemuž odpovídá komunikační model publish-subscribe [17]. Cílem CANopen je izolovat uživatelskou aplikaci od hardwarové vrstvy koncového zařízení a standardizovat některá přenášená data. Interakce protokolu a aplikace probíhá výhradně skrze slovník objektů a rozhraní pro správu stavu zařízení.

Přestože je CANopen flexibilním komunikačním protokolem, není příliš využíván v případě automobilového průmyslu. Důvodem může být způsob, jakým standard nakládá se zařízeními určitého typu. Jak již bylo zmíněno, slovník objektů poskytuje obecné rozhraní mezi aplikací a hardware, nepopisuje však význam uložených procesních dat. K tomu slouží takzvané profily zařízení, upřesňující datové struktury pro zařízení se specifickou funkcí. Aktuálně existuje několik těchto rozšíření standardu, mezi kterými lze zmínit například CiA 402, profil pro frekvenční měniče a serva, nebo CiA 417 pro výtahové systémy [18]. Jelikož jsou v případě automobilů v různých modelech od různých výrobců použity odlišné technologie a jejich konfigurace, není možná standardizace univerzálního popisu vozidla, ačkoliv lze najít mnoho společných prvků. Navíc s vývojem standardu AUTOSAR, který definuje aplikační rozhraní s důrazem na automobily, není o CANopen v automobilovém průmyslu takový zájem.

2.4 Experimentální vozidlo eŠus

Vozidlo eŠus vzniklo v rámci budování Laboratoře elektromobility na Technické univerzitě v Liberci. Je tedy výsledkem spolupráce mezi Ústavem mechatroniky a technické informatiky (MTI) a Katedrou vozidel a motorů (KVM). Vývoj začal již v roce 2011 s cílem vytvořit elektrické vozidlo sloužící jako platforma pro testování a demonstraci technologií, kterými se pracovníci zmíněných ústavů a studenti univerzity zabývají. Koncepce vozidla počítala se dvěma důležitými body. Prvním bylo omezení maximálního napětí baterie na 60 V z důvodu bezpečnosti, jelikož je s elektronikou vozidla často manipulováno. Navazujícím druhým bodem je pak snadná přístupnost ke všem klíčovým částem elektroniky právě kvůli časté manipulaci. Díky splnění těchto bodů je vozidlo snadno servisovatelné a rozšiřitelné o další potenciální výbavu. První vyrobený funkční vzorek vznikl v roce 2013, kdy zároveň začal i vývoj druhé generace, která by již koncepčně více odpovídala homologovaným sériovým vozidlům, ačkoliv homologace eŠusu původně nebyla zamýšlena [19].



Obrázek 2.9: První generace experimentálního vozidla eŠus [31]

Do dnešní doby prošlo vozidlo již několika vylepšeními a úpravami. Druhá generace vozidla, představená v roce 2016, přinesla značné rozdíly primárně v konstrukci trakční baterie, která byla vybavena novými články a umístěna do podvozku konstrukce. Díky zajištění nových bezpečnostních prvků mohlo být napětí baterie navýšeno na 100 V. Úspěšně se podařilo zprovoznit systém řízení natočení zadních kol. Významným problémem však zůstává řešení komunikační sítě řídicích jednotek.

Jelikož v posledních letech došlo u eŠusu k postupnému rozšíření jeho funkcionality, zvýšilo se také množství elektroniky v něm přítomné. Základ vozidla tvoří komponenty ze sériového automobilu Škoda, některé systémy byly zakoupeny jako komerční produkt a některé vznikly v rámci studentských prací. Při implementaci systému natáčení zadní nápravy [20] byl použit stejný snímač úhlu natočení jako v případě volantu. To způsobilo problém na sběrnici CAN, kde oba moduly vysílají rámce se stejným identifikátorem. Při provozu na jednom kanálu sběrnice tedy docházelo pravidelně ke kolizím a nestabilitě systému. Jelikož se jedná o jednotky ze sériového automobilu, nelze tento problém vyřešit úpravou kódu. Nakonec byla síť rozdělena na dvě části oddělěné gateway jednotkou v podobě vývojového kitu Arduino, zajišťující přeposílání zpráv z jednoho fyzického kanálu na druhý. Problematická je také aktuální realizace elektroniky hnacího ústrojí vozidla. Regulaci pohonu přední nápravy zajišťuje trakční měnič Gen4 od společnosti Sevcon. Ten pracuje v režimu master. Trakční jednotka tedy představuje samostatný uzavřený systém. Z hlediska koncepce vozidla je tento režim provozu nežádoucí, jelikož na něm není možné možná úprava firmare pro potřeby testování různých algoritmů regulace pohonů. O správu trakční baterie vozidla se stará zakoupená jednotka BMS od společnosti Ewert Energy Systems. Ta také neumožňuje úpravu firmware a navíc není nijak propojena s ostatními komponentami vozidla [21].

Experimentální část práce se zabývá řešením několika na vozidle přítomných problémů. Prvním je skutečnost, že každá elektronická komponenta vykonává určitou specifickou funkci bez napojení na zbytek systému. Z hlediska koncepce vozidla je sice zajištěna snadná fyzická manipulace s elektronikou, ze strany firmware je však přístup k datům dílčích systémů vozu v neuspokojivém stavu. Každé rozšíření o novou funkcionalitu mnohdy spočívá nejen ve změně firmware zodpovědné jednotky, mnohdy musí dojít i aktualizaci jednotek ostatních, aby měl nový algoritmus přístup k potřebným datům. S algoritmizací souvisí také druhý problém, kterým je aktuální konfigurace trakčního subsystému. V souvislosti s výzkumem aktuálně probíhajícím na MTI vznikl zájem o realizaci testovací platformy typu MMDS. Pohon vozidla budou tedy zajišťovat tři elektromotory, jeden na přední nápravě a dva vzadu. Každý z pohonů bude následně řízen samostatným trakčním měničem v režimu slave. Vyžadována je tedy nová řídicí jednotka trakce, zajišťující jejich obsluhu. Důvodem k tomuto kroku je zároveň také požadavek na vznik řídicího hardware v režii TUL, který umožní neomezené úpravy firmware. Dosažením tohoto cíle lze na cílovém hardware například spouštět simulační modely z prostředí Matlab Simulink, exportované do programovacího jazyka C. Modely mohou obsahovat implementaci libovolných algoritmů pro řízení elektrických pohonů. Nezávislý pohon levého a pravého kola zadní nápravy umožní také testování technologie torque vectoring, spočívající v řízení krouticího momentu jednotlivých kol vozidla. Nasazením takové technologie v trakčních systémech lze dosáhnout lepší manévrovatelnosti a bezpečnosti automobilů [22]. Problematika těchto algoritmů je mimo rozsah této práce, řešena je pouze realizace dílčí části komunikační a řídicí platformy, která umožní jejich testování. Cílem je rozšířit elektroniku vozidla o všechny požadované systémy a následně navrhnout jejich propojení tak, aby vznikla přehledná a snadno rozšiřitelná komunikační síť, demonstrující aktuální trendy v automobilovém průmyslu. Pro

rané testovaní nové sítě je nutné implementovat novou řídicí jednotku trakce a nainstalovat ji do vozidla. Zaroveň by měl vzniknout prostor pro budoucí studentské práce vzbuzující zájem o řídicí elektroniku a komunikační infrastrukturu vozidel.

2.4.1 Klíčové komponenty vozidla

V této kapitole jsou popsány klíčové komponenty obsažené v automobilu. Některé z nich již bylu v minulosti popsány v rámci publikací a závěrečných prací [23] [20] a proto jsou uvedeny pouze jejich nejdůležitější parametry s referencí na podrobnější informace. Aktuálně přítomnou elektroniku lze rozdělit do čtyř skupin dle funcionality:

- energetické zdroje a jejich správa
- obsluha pohonů
- ovládání vozidla
- vizualizace jízdních informací

Systém je napájen trakční baterií umístněnou v podvozku vozidla. Ta se skládá z 10 modulů v konfiguraci 3s4p, použitými články jsou Saft VL41M s nominálním napětím 3,6 V a energií 144 Wh. Klíčové parametry baterie jsou uvedeny v tabulce 2.1. Nabíjení baterie aktuálně zajišťuje palubní nabíječka s maximálním nabíjecím výkonem 2,2 kW. Budoucí verze vozidla počítá s novou baterií postavenou na cylindrických článcích formátu 18650 nebo 21700, s 400V nebo 800V technologií a integrací rychlonabíjecího standardu CCS Typ-2 umožňující nabíjení výkonem až 100 kW. Vozidlo je také vybaveno autobaterií postavenou na článcích technologie LFP sloužící pro napájení elektroniky palubním napětím o maximální velikosti 14,6 V [21].

Tabulka 2.1: Přehled parametrů trakční baterie elektromobilu eŠus

Parametry trakční baterie			
Nom. napětí [V]	Nom. energie [kWh]	Nom. proud [A]	Krátkodobý proud [A]
108	17,2	600	960

Pro potřeby monitorování baterie a balancování jednotlivých článků je přítomen systém správy baterie (z anglického battery management system neboli BMS) od společnosti Ewert Energy Systems. Tento modul umožňuje správu baterií až do konfigurace 36s a balancování článků proudem 100 mA. BMS je vybavena dvěma komunikačními kanály sběrnice CAN, možností plné konfigurace vysilaných dat a podporou protokolů J1939 a OBD2. Jeden z kanálů je aktuálně využíván pro komunikaci s palubní AC nabíječkou a druhý k diagnostickému a konfiguračnímu přístupu pomocí softwarového nástroje poskytovaného výrobcem zařízení.

Pro úlohy spojené s trakcí je ve vozidle instalován trakční měnič Gen4 Size6, jehož výrobcem je společnost Sevcon. Výrobek patří do rodiny kontrolérů pro řízení čerpadel nebo pohonů v zařízeních napájených baterií. Podporované jsou synchronní (PMAC) i asynchronní motory. Měniče jsou vyráběny v několika variantách odlišujících se primárně nominálním provozním napětím a proudem. Rozdíly lze také najít v provedení pouzdra, existují modely chlazené vzduchem i vodou. Měnič může být provozován v režimu master i slave. V případě režimu master zastává zařízení funkci centrální řídicí jednotky vozidla. Aktuálně přítomný model Gen4 Size6 (viz tabulka 2.2) je vzduchem chlazený a společně s diferenciálem zajišťují pohon přední nápravy. Jedním z požadavků na novou implementaci architektury vozidla je rozšíření této konfigurace o řízení zadní nápravy, každého kola samostatně. Pro tento účel byly zvoleny dva trakční měniče od stejného výrobce s názvem Gen4 Size4. Od předního měniče se liší pouze rozměry a nižšími hodnotami nominálního a krátkodobého proudu. Jejich konfigurace a způsob ovládání zůstavají stejné jako v případě Size6.

Tabulka 2.2: Přehled parametrů trakčních měničů Sevcon

Model	Parametry		
	Provozní napětí [V]	Nominální proud [A]	Krátkodobý proud [A]
Gen4 Size4	72–120	140	420
Gen4 Size6		220	660

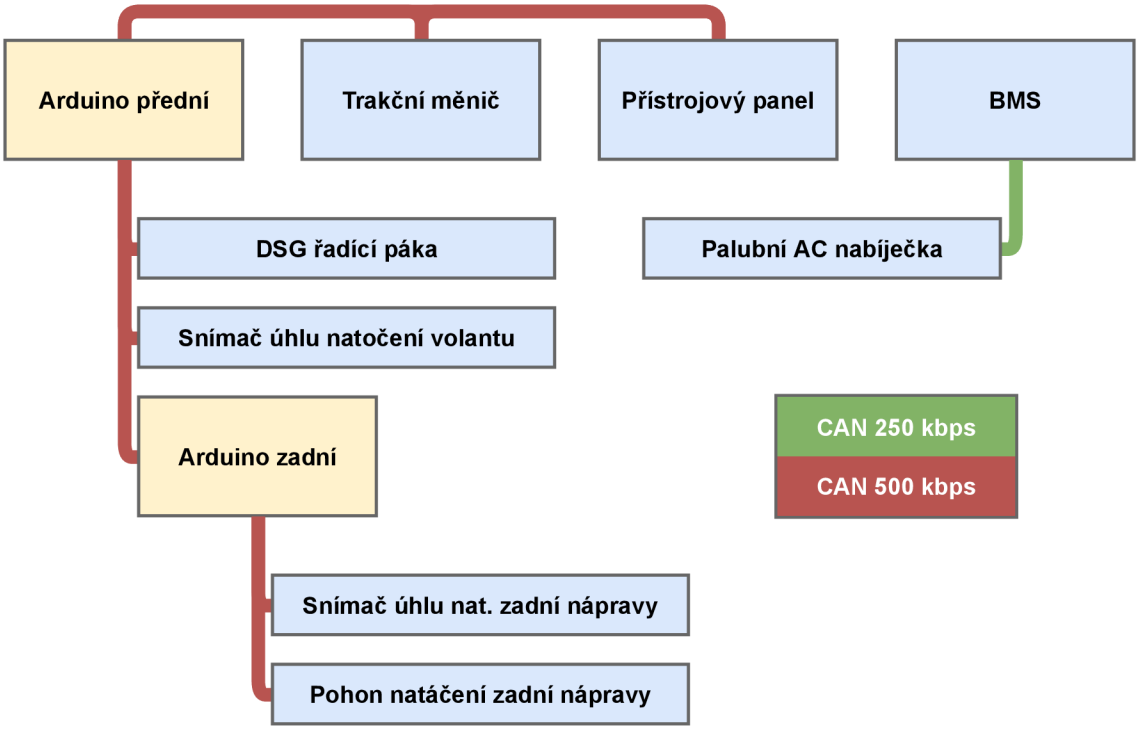
2.4.2 Architektura komunikační sítě

Původní komunikační síť automobilu eŠus byla založena na trakčním měniči Sevcon v režimu master. Ten tvoří centrální řídicí jednotku vozidla. Do měniče je zavedena většina vstupů a výstupů, potřebných k řízení vozidla, formou diskrétních či analogových signálů. O zpracování dat vysílaných komponentami sériového automobilu Škoda se starají dvě řídicí jednotky v podobě vývojového kitu Arduino. Tyto jednotky převádí formát dat vysílaných snímači právě na diskrétní a analogové signály, zavedené do trakčního měniče. Zároveň je díky jejich použití rozdělena sběrnice na více kanálů, čímž se předchází kolizím identifikátoů CAN, vzniklým z důvodů použití identických snímačů úhlu natočení volantu pro přední i zadní nápravu v rámci implementace funkce přiřizování zadní nápravy. Problém a jeho řešení jsou podrobněji popsány v kapitole 3.2.3. Jednotka správy baterie (BMS) tvoří společně s palubní AC nabíječkou samostanou síť oddělenou od zbytku elektroniky.

Tento způsob realizace sítě představuje pro vozidlo potenciální riziko, jelikož není nijak zajištěno sdílení informací o baterii s trakčním měničem. Nejčastěji diskutovanou situací v souvislosti s elektromobilem eŠus je případ, kdy dojde k přehřátí baterie a BMS ve vysílaných datech sníží maximální možnou hodnotu proudu odebíraného z baterie. Informace se však nedostane do aplikace v trakčním měniči a ta neomezí maximální výstupní výkon. Vlivem této chyby návrhu může za určitých podmínek dojít až k nevratnému poškození baterie.

Dalším problémem původního návrhu je nedostatečná flexibilita nastavení

a úpravy firmware v jednotkách. Jediné komponenty s možnosti úpravy aplikace jsou totiž právě jednotky Arduino, které však neobsahují žádnou rozhodovací logiku. Automobil by měl tvořit experimentální platformu pro testování různých alogritmů pro správu baterie či řízení trakce, což původní realizace elektroniky neumožňovala.



Obrázek 2.10: Původní komunikační síť vozidla eŠus

2.4.3 Požadované změny

Na návrh nové komunikační sítě vozidla byly společně s vedoucím a konzultantem práce stanoveny tyto požadavky:

- propojení všech komponent vozidla vybavených komunikační sběrnicí
- přidání dvou trakčních měničů umožňujících individuální řízení zadních kol
- přenesení rozhodovací úrovně aplikace do komponent s možností úpravy firmware
- snadná konfigurace a diagnostika parametrů jednotek pomocí standardizovaného protokolu
- realizace jednoho z konceptů síťové architektury moderních vozidel představených v kapitole 2.2

3 Experimentální část

Cílem experimentální části byl dle zadání návrh nové síťové architektury elektromobilu eŠus a realizace řídicí jednotky trakce vozidla. Kapitola navíc popisuje řídicí jednotku podvozku, realizovanou mimo rozsah zadání práce. Pro potřeby komunikace mezi jednotkami pomocí protokolu CANopen byla použita veřejně dostupná knihovna *canopen-stack* od společnosti Embedded Office. Přestože existuje několik zdarma dostupných implementací protokolu, volba pravě této této knihovny je podložena následujícími body:

- přehledná struktura zdrojového kódu a podrobná dokumentace
- striktně dodrženo použití statické alokace paměti (problematika dynamické alokace u vestavěných systémů viz [32])
- přítomnost hardwarové abstrakce, knihovna je plně nezávislá na architektuře či modelové řadě procesoru
- přítomnost unit testů (viz slovník pojmů) pro každou komponentu knihovny
- knihovna je nasazena v reálných průmyslových řešeních společnosti Embedded Office a tedy pravidelně testována z důvodu zajištění stability a bezpečnosti

Použitá knihovna je určena pro implementaci zařízení typu CANopen slave, původně tedy neobsahovala funcionalitu, kterou vyžaduje master. Jelikož v nové architektuře vozdidla eŠus vystupuje ve své doméně řídicí jednotka trakce právě jako master, bylo pro potřeby komunikace s trakčními měniči nutné knihovnu rozšířit o následující funkce:

SYNC producer Periodické vysílání rámců zahajujících synchronní přenos procesních dat pomocí protokolu PDO

SDO client Klientská část SDO protokolu umožňující vzdálený přístup do slovníku uzlů přítomných na sběrnici

Implementované funkce byly schváleny a přídány do hlavního repozitáře knihovny. Podmínkou pro schválení tohoto rozšíření bylo splnění testů přidaných komponent, o jejichž specifikaci dle standardu CiA 301 se postaral Dipl. Ing. Michael Hillman, zodpovědný za údržbu knihovny. Všechny požadované testy proběhly bezchybně a příspěvek se tedy stal plnohodnotnou funkcionalitou nové verze vydání *canopen-stack 4.2.0* a byl v rámci licence Apache 2.0 zpřístupněn pro veřejnost [33].

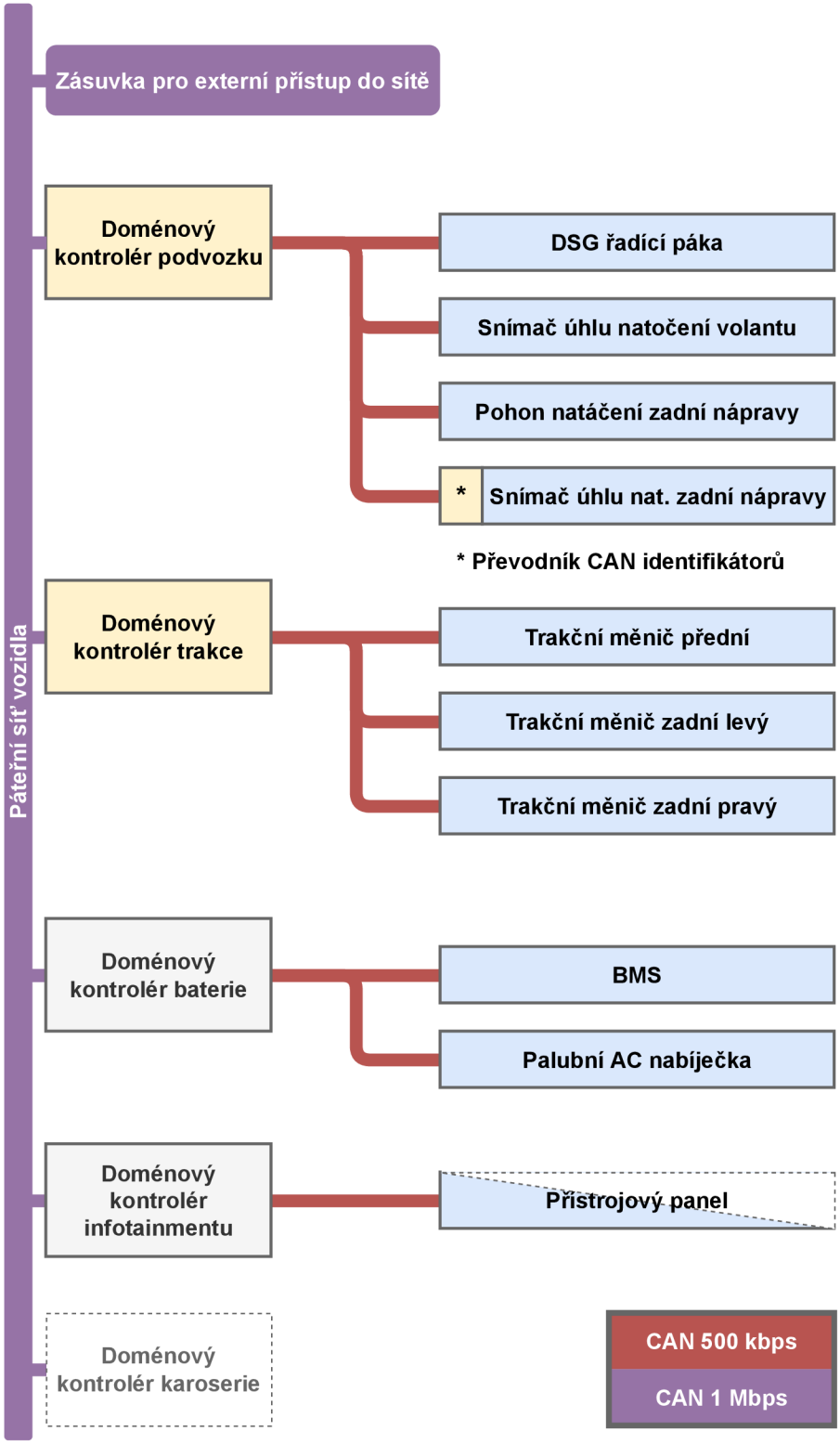
3.1 Návrh nové síťové architektury vozidla

V kapitole 2.4.1 byly komponenty vozidla eŠus rozděleny do několika skupin dle funcionality. Z tohoto rozdělení vychází také návrh nové architektury komunikační sítě. Ačkoliv se jako zajímavější a přínosnější jeví snaha o realizaci a ověření vlastností koncepce zónového řízení, není v souvislosti s dostupnými prostředky takový postup možný. Mnoho komponent přítomných v automobilu má uzavřený firmware bez možnosti jeho úpravy, jelikož se jedná o zakoupené komerční produkty nebo moduly získané ze sériových vozidel. Díky studenstkým pracím, které vznikly v souvislosti s vozidlem eŠus, je alespoň přibližně znám způsob komunikace s těmito jednotkami. Tímto je zajištěna veškerá funcionalita potřebná k provozu vozidla. Kompletní popis protokolu je duševním vlastnictvím výrobce těchto jednotek a zpravidla bývá utajen. Jako časově a finančně dostupnějším řešení se tedy ukázala realizace doménové architektury. Po prvotní analýze ve spolupráci s konzultantem práce bylo rozhodnuto, že jejím nasazením lze při správném postupu dosáhnout všech požadavků stanovených v 2.4.3.

Postup při realizaci nové architektury vozidla tedy nejprve vyžadoval rozdělení použitých komponent do doménových sítí řízených doménovým kontrolérem. Doménové kontroléry byly následně propojeny pátěřní sítí vozidla. Aktuální objem sdílených informací nevyžaduje nasazení protokolů jako FlexRay či Ethernet, využit byl tedy CAN s přenosovou rychlostí 1 Mbit/s. Návrh nové komunikační sítě lze vidět na obrázku 3.1.

Pro minimální implementaci, představující pojízdné vozidlo, počítá prvotní architektura s doménovými kontroléry podvozku, trakce, baterie a infotainmentu. Implementace trakční a podvozkové domény byla dokončena v rámci této práce, domény baterie a infotainmentu by měly vzniknout v rámci budoucích závěrečných prací ve spolupráci s jinými studenty. Pro komunikaci v páteřní síti byl zvolen protokol CANopen, a to z následujících důvodů:

- jedná o standardizovaný a dobře zdokumentovaný protokol
- je na něm založena komunikace s trakčními měniči použitými ve vozidle
- poskytuje značnou flexibilitu v kontextu nasazení a konfigurace
- existuje několik zdarma dostupných veřejných implementací



Obrázek 3.1: Nová síťová architektura elektromobilu eŠus. Bloky označené modrou barvou představují zakoupené komponenty. Béžová barva reprezentuje prvky řešené v rámci této práce a šedá možnosti budoucího vývoje.

3.2 Řídicí jednotka podvozku

Prvním krokem v realizaci nové sítě vozidla byla implementace řídicí jednotky podvozku. Jednotka slouží jako doménový kontrolér pro úlohy spojené s řízením vozidla. Obecné požadavky na řídicí jednotku v souvislosti s vozidlem eŠus byly po diskuzi s konzultantem práce stanoveny následovně:

- komunikace s komponentami ze sériového vozidla Škoda Auto
- snímání polohy plynového pedálu
- snímání stavu koncového spínače brzdového pedálu
- řízení natočení zadní nápravy vozidla
- sdílení procesních dat s ostatními doménovými kontroléry

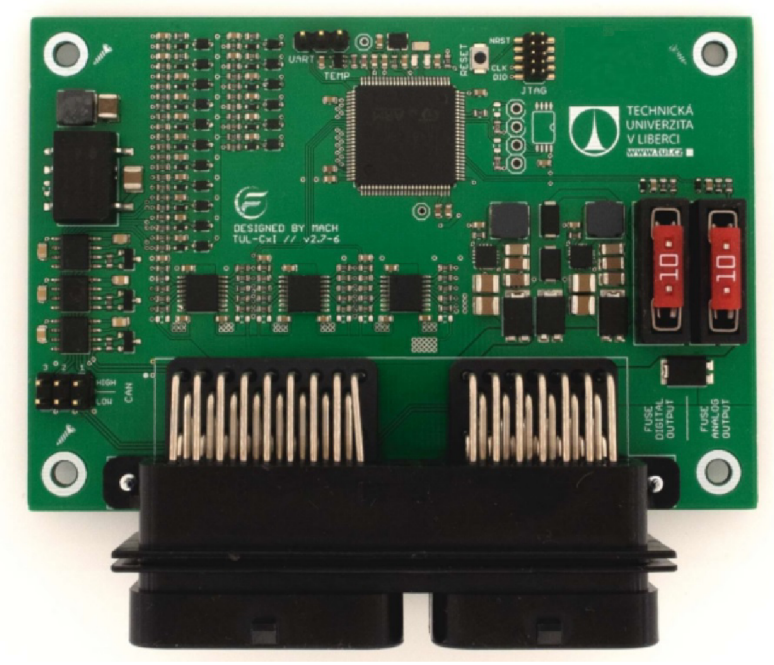
Na prvním kanálu sběrnice CAN jednotka komunikuje s I/O periferiemi jako jsou například řadící páka, volant či plynový pedál. Zpracovává tedy všechny události vyvolané řidičem a vyhodnocuje, zdali aktuální stav podvozku umožňuje bezpečnou jízdu. Druhý kanál již slouží ke spojení s ostatními doménovými kontroléry pomocí protokolu CANopen prostřednictvím páteřní sítě. Zde jsou vysílány PDO související se stavem podvozku. Na tuto jednotku byla také přesunuta služba řízení natáčení zadní nápravy, kterou dříve zajišťovala jedna z přítomných vývojových desek Arduino.

3.2.1 Hardware

Pro implementaci řídicí jednotky podvozku byl použit hardware založen na mikrokontroléru s jádrem ARM Cortex-M4. Jedná o desku vyvinutou přímo na Technické univerzitě v Liberci, v rámci realizace projektu ve spolupráci s průmyslem. Hardware je navržen tak, aby jej bylo možné vyrábět v malých sériích a nasazovat na dalších projektech vyžadujících funkce, kterými disponuje. Komunikaci zajišťují tři kanály CAN2.0B s maximální přenosovou rychlostí 1 Mbit/s. Zařízení lze napájet napětím až do velikosti 16 V. Vstupy a výstupy jednotky jsou s tímto napětím také kompatibilní. Proudovou ochranu zajišťují automobilové pojistky. Digitální výstupy jsou vybaveny inteligentními spínacími prvky Texas Instruments TPS2HB16F se zavedenou zpětnou vazbou, umoňující snímání aktuálně odebíraného proudu. Tato funkce značně usnadňuje diagnostiku výstupů. Připojení jednotky ke zbytku systému umožňuje konektor certifikovaný pro použití v automobilových aplikacích.

3.2.2 Firmware

Aplikace implementovaná v jednotce (viz příloha A.2) využívá prostředí operačního systému reálného času FreeRTOS. Úlohy spravované operačním systémem jsou rozděleny, podobně jako služby aplikace, na systémové a uživatelské. Jedna úloha (také označována jako vlákno) může zajišťovat běh jedné nebo více dílčích funkcí



Obrázek 3.2: Hardware řídicí jednotky použité jako doménový kontrolér podvozku

aplikace. Systémové úlohy zprostředkovávají primárně interakci s hardware jednotky a předzpracování dat. Patří sem obsluha komunikačních rozhraní a přerušení, vzorkování vstupních veličin a řízení přístupu k aplikačním datům. Uživatelská aplikace tedy nepřistupuje explicitně k hardware, se svým okolím interaguje pomocí implementace aplikačního rozhraní a proměnných různých datových typů alokovaných v operační paměti mikrokontroléru. Tím je dosaženo přenositelnosti zdrojového kódu na libovolnou architekturu mikroprocesoru, což poskytuje flexibilitu ve výběru hardware nebo možnost automatizovaného testování aplikace bez nutnosti nasazení na cílovém zařízení.

Jak již bylo zmíněno, jedna ze systémových funkcí aplikace jednotky podvozku zajišťuje komunikaci s komponentami ze sériového vozidla Škoda Auto a zpracovává jimi vysílané signály. Podrobný popis komunikace s těmito jednotkami není z důvodu utajení znám, typicky však zařízení nasazena v automobilovém průmyslu obsahují firmware implementovaný v souladu se standardem AUTOSAR a struktura komunikace je tvořena proprietárně pro každou automobilovou platformu [24]. Standard AUTOSAR definuje v souvislosti s komunikačními službami pojmy:

PDU Reprezentace dat obsažených v komunikačních rámcích. Rámec může obsahovat jeden nebo, v případě agregace, více PDU [11]. Cílem PDU je abstrakce přenášených dat díky čemuž odpadá závislost na použitém protokolu. Příkladem může být situace, kdy gateway jednotka přijme PDU na kanálu FlexRay a vzápětí jej vysílá na CAN.

Signál Aplikační datový objekt nesoucí informaci předávanou mezi jednotlivými účastníky komunikace. Nemá fixní datový typ, definován je šířkou v bitech. Signály jsou mapovány do jednoho či více PDU na specifickou pozici [11].

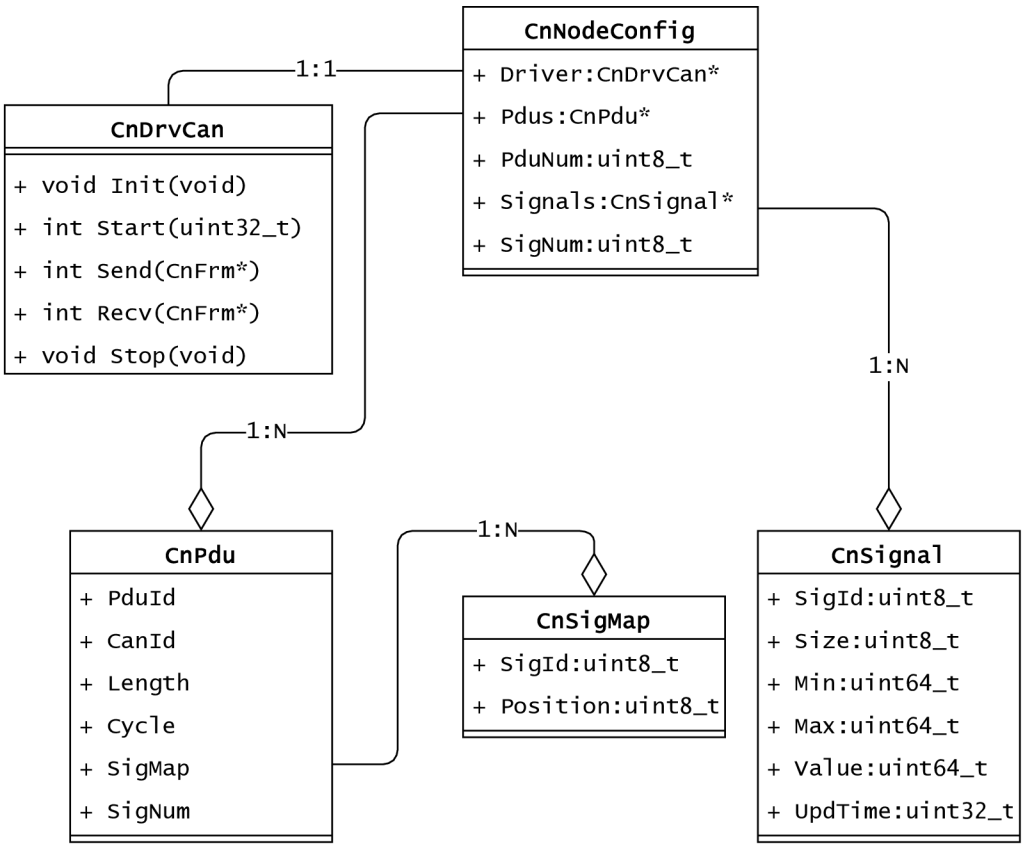
V případe protokolu CAN volbu identifikátorů komunikačních rámců standard AUTOSAR nespecifikuje. Identifikátory jsou přiřazeny při vývoji komunikační sítě dané automobilové platformy. Proti tomu standard CiA 301 [17] specifikuje rozsahy identifikátorů pro jednotlivé služby protokolu CANopen. Z toho vyplývá výrazná odlišnost mezi zmiňovanými standardy, díky které lze vyloučit možnost použití knihovny *canopen-stack* pro komunikaci s jednotkami Škoda Auto. Pro potřeby zpracování rámců vysílaných těmito jednotkami byla proto vyvinuta knihovna *can-node*. Tato knihovna sice není plnohodnotnou implementací standardu AUTOSAR, využívá však v souvislosti s protokolem CAN stejného přístupu k datům. Konfigurace této knihovny definovaná datovým typem *CnNodeConfig* (viz obr. 3.3) se skládá z následujících částí:

CnDrvCan Ukazatel na uživatelskou implementaci rozhraní sloužící k vysílání a přijmu komunikačních rámců CAN. Zavedením tohoto rozhraní je zajištěno hardwarové nezávislosti knihovny. Pro potřeby testovánní lze tuto implementaci nahradit simulovaným prostředím, například virtuální sběrnicí.

CnPdu Ukazatel na pole PDU zpracovaných službou. PDU je reprezentováno vlastním identifikátorem, CAN identifikátorem, délkou a cyklem. Mapování více PDU do jednoho komunikačního rámce není podporováno. Hodnota cyklu slouží k automatickému vysílání či kontroly periodického příjmu.

CnSignal Ukazatel na pole signálů. Podobně jako u PDU, každému ze signálů je přiřazen jedinečný identifikátor, pomocí kterého k němu přistupuje aplikace. U signálů je použita také časová značka poslední aktualizace a povolený rozsah jejich hodnoty.

CnSigMap Ukazatel na pole mapování signálů. Každý záznam propojuje pomocí identifikátorů signálu a PDU. Také je definovaná pozice signálu, která se může pro různá PDU lišit.



Obrázek 3.3: Vývojový diagram konfigurace knihovny *can-node*

Aplikační rozhraní knihovny umožňuje uživateli čtení nebo zápis dat do signálů či vypínání a zapínaní periodického vysílání PDU. Další funkce rozhraní slouží k inicializaci a spouštění knihovny. Uživatelem vytvořená konfigurace je předána knihovně pomocí funkce *CnNodeInit()*. Knihovna je spuštěna voláním *CnNodeStart()*. Následně je ve vlákně operačního systému v periodě 1 ms volána funkce *CnNodeProcess()*, zajišťující interní operace knihovny. Ačkoliv není knihovna závislá na použití operačního systému, je nutné zajistit její správné časování. Budoucí vývoj počítá se zavedením nového rozhraní pro obsluhu časovače, díky kterému si již knihovna bude řešit časování autonomně bez nutnosti zásahu uživatele. Z popisu konfigurace a rozhraní knihovny lze vyvodit, že není určena pouze pro potřeby této práce a lze ji nasadit v libovolné aplikaci pro služby spojené s CAN komunikací založené na signálech.

Mimo obsluhy komunikace obsahuje systémová část aplikace služby pro periodické snímání polohy plynového pedálu, zprostředkované dvěma analogovými vstupy. Pomocí digitálního vstupu je také snímán stav koncového snímače brzdového pedálu. Analogově digitální převodník mikrokontroléru STM32F4 je vybaven funkcí sekvenčního čtení několika kanálů a zápisu změřených hodnot do paměti pomocí periferie DMA. Zároveň lze spouštět tyto měřící sekvence hardwarovým časovačem. Spojením těchto funkcí lze dosáhnout plně autonomního periodického snímání až 16 kanálů, bez nutnosti zásahu procesoru. Ověřená konfigurace stabilně vzorkuje tři

analogové kanály s rozsahem 16 bit v periodě 10 ms. Jelikož toto nastavení vyhovuje stanoveným požadavkům (poloha plynového pedálu je počítána dokonce v periodě 20 ms), nebyly dále testovány možnosti konfigurace periferie. Výstupy vzorkování jsou uloženy v operační paměti a jejich převod na fyzickou hodnotu napětí provádí pouze v případě potřeby uživatelská služba.

3.2.3 Kolize identifikátorů protokolu CAN

V teoretické části bylo zmíněno použití snímače natočení úhlu volantu ze sériového vozidla Škoda Auto pro potřeby implementace služby přiřizování zadní nápravy. Jelikož je stejná komponenta použita i pro snímání úhlu natočení volantu, docházelo by v doménové síti podvozku k pravidelným kolizím. Obě jednotky totiž požívají stejné identifikátory protokolu CAN pro přenos procesních dat. Jelikož je úprava jejich firmware vyloučena, musel být tento problém vyřešen jiným způsobem.

První navrhované řešení počítá se zakoupením nového snímače, jenž umožní alespoň částečnou konfiguraci komunikačních parametrů. Tato varianta by však vyžadovala úpravu podvozku vozidla, konkrétně ukotvení tohoto snímače na hřídel serva natáčejícího zadní nápravu.

Jako cenově i časově dostupnější řešení se ukázalo použití převodníku identifikátorů v podobě miniaturního zařízení (viz příloha A.8), vyvinutého laboratoří pro potřeby jiných projektů. Tento hardware, pracovně nazývaný "Zlodějka", je vybaven mikrokontrolérem ARM a dvěma kanály protokolu CAN. Lze jej napájet stejným způsobem jako ostatní jednotky, tedy palubním napětím 12 V. Firmware tohoto zařízení obsahuje jednoduchou logiku, která inkrementuje identifikátor rámců CAN příchozích ze snímače natočení volantu Škoda Auto a přeposílá je na druhý kanál, připojený k doménové síti podvozku. Toto řešení sice zanáší do komunikace zpoždění v řádu jednotek µs, procesní data ze snímače jsou však vysílána v periodě 20 ms. Zpoždění se tedy znatelně neprojevuje na kvalitě komunikace. V laboratoři bylo k dispozici mnoho již osazených desek tohoto přípravku a proto nevznikly jeho nasazením ve vozidle žádné další finanční náklady. Zařízení je možné vzhledem k jeho rozměrům přichytit přímo na kryt cílové komponenty.

3.3 Řídicí jednotka trakční domény

Společně se správou energetických zdrojů a podvozku tvoří řízení trakce základ funkcionality elektromobilu. Přítomnost trakční domény v komunikační síti automobilu eŠus nabízí možnost implementace a testování algoritmů spojených právě s řízením pohonů vozidla. Pro tyto potřeby bylo nutné pro automobil eŠus vyvinout trakční doménový kontrolér splňující následující požadavky:

- obsluha trakčních měničů Sevcon Gen4 prostředníctvím protokolu CANopen
- diagnostický přístup k jednotce, možnost monitorování aktuálního stavu
- integrace a spuštění Simulink modelů vyexportovaných do jazyka C
- běh všech přítomných služeb v reálném čase

3.3.1 Hardware

Vzhledem k požadavkům na nový trakční kontrolér bylo nutné zvolit hardware nabízející dostatečný výkon pro běh aplikace v reálném čase nehledě na spotřebu. Použití stejného hardware jako v případě řídicí jednotky podvozku by v budoucnu mohlo představovat problémy s výpočtem jednotlivých kroků nasazeného Simulink modelu vedoucí k porušení podmínky reálného času, v případě trakční domény definovaného periodou 1 ms. Proto bylo nutné zvolit modelové řady mikrokontrolérů vybavené jedním nebo více jádry s výkonem vyšším než v případě jednotky podvozku. Do výběru byly nakonec zařazeny tři čipy splňující stanovené požadavky.

STM32H747

Zástupce nejvýkonnější řady nabízené firmou STMicroelectronics. Kombinuje jádra Cortex-M7 na frekvenci 480 MHz a Cortex-M4 s maximální frekvencí 240 MHz. Pro potřeby řízení přístupu jader k prostředkům čipu je implementován hardwarový semafor s podporou přerušení. Velkou výhodou je v případě prvního jádra hardwarová akcelerace DPFP, urychlující aritmetiku s plovoucí čárkou s dvojnásobným rozlišením (double). Tento datový typ v základu využívá k provádení výpočtů právě Simulink a nebylo by nutné optimalizovat z něj exportovaný zdrojový kód. Velikost operační paměti dosahuje 1 MB a pro uložení programu jsou k dispozici 2 MB paměti FLASH. Pro potřeby komunikace je čip vybaven třemi kanály CAN-FD, obsluhované prostředníctvím periferie FDCAN.

STM32H723

Jednojádrový mikrokontrolér patřící do stejné řady a vybaven stejnou strukturou periferií jako STM32H747. Rozdílem je přítomnost pouze jednoho jádra Cortex-M7, tentokrát však s maximální frekvencí 550 MHz. Přítomna je také akcelerace DPFP.

RT1170

Aktuálně nejnovější dvoujádrový zástupce řady mikrokontrolérů NXP. Stejně jako STM32H747, kombinuje jádra Cortex-M7 a Cortex-M4. Rozdílem jsou však maximální frekvence 1 GHz a 400 MHz. Opět přítomna podpora DPFP a tří kanálů sběrnice CAN-FD. Rozdílem je tzv. cross-over architektura čipu, spočívající v uložení programu v externí paměti. K dispozici je 256 kB interní ROM pro zavaděč. Součástí výbavy jsou tři Ethernet rozhraní, z nichž jedno podporuje zmiňovanou technologii Time Sensitive Networking (viz 2.1.4).

Díky struktuře trakční doménové sítě není vyžadováno, aby doménový kontrolér obsahoval I/O funkcionalitu (viz 2.2), veškerá jeho interakce s okolím probíhá prostřednictvím datových sběrnic. Proto u zmiňovaných zástupců byly srovnávany pouze výkon procesorových jader a možnosti komunikačních rozhraní, periferie sloužící k obsluze vstupů a výstupů nejsou pro potřeby realizace jednotky důležité. Z hlediska výkonu se jako nejvhodnější jeví NXP RT1170, jeho nasazení však přináší několik problémů spojených s implementací zavaděče a obsluhou externích pamětí. Dalším problémem je v případě vícejádrových mikrokontrolérů nutnost řízení přístupu jader ke sdíleným prostředkům, tedy paměti a periferiím. Aplikace vyžaduje implementaci zamykacích mechanismů a algoritmů pro identifikaci datových struktur na specifických adresách v paměti. Program je totiž kompilován samostatně pro každé jádro a adresy globálních proměnných se tedy mohou lišit.

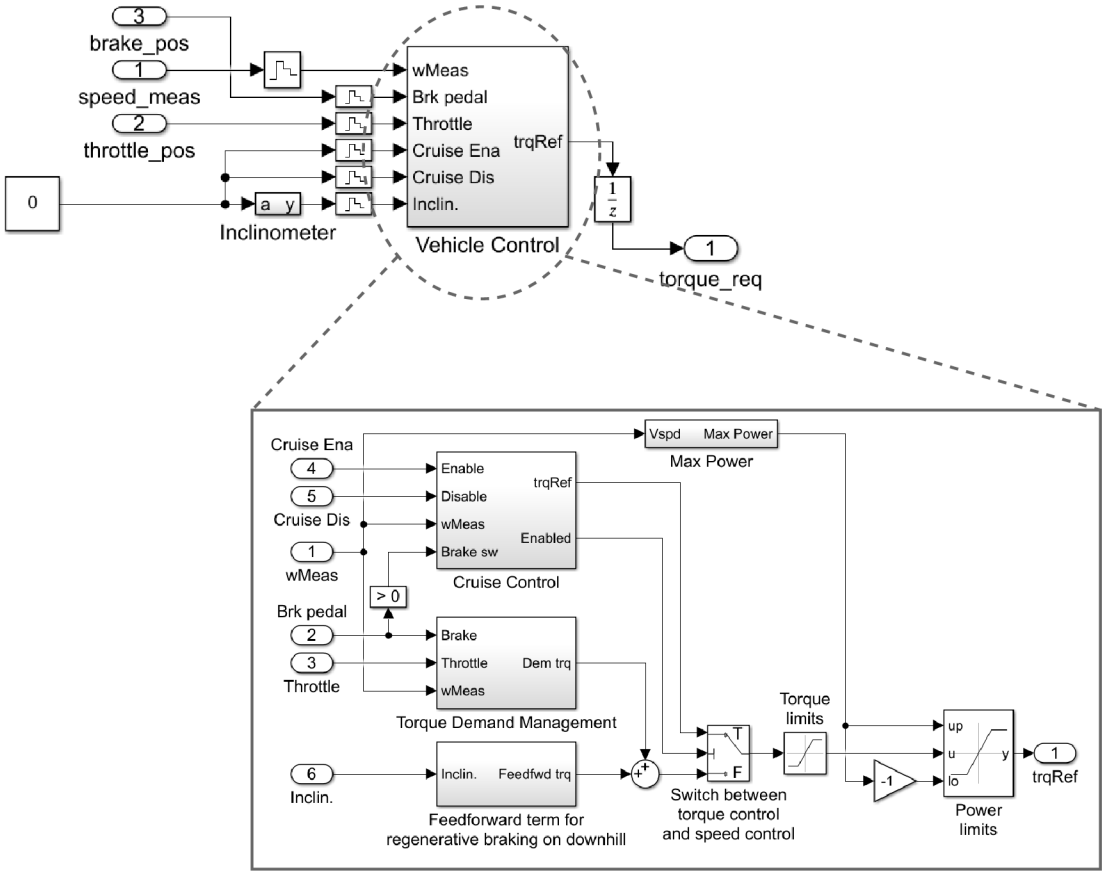
Vzhledem k rozsahu této práce byl nakonec z časových důvodů zvolen za účelem ověření návrhu architektury sítě jednojádrový model STM32H723. Tento mikrokontrolér byl použit k realizaci funkčního vzorku jednotky postaveném na vývojové desce Nucleo, nabízené výrobcem. Deska byla rozšířena o tři budiče fyzické vrstvy protokolu CAN a možnost napájení palubním napětím 12 V formou tzv. shield modulu (viz příloha A.9). Budoucí prototyp řídicí jednotky trakce ve formě proprietarního hardware již počítá s použitím právě NXP RT1170. Vzhledem ke struktuře firmware nepředstavuje budoucí migrace na jiný mikrokontrolér problém, pro nasazení bude stačit implementace ovladače periferie CAN a zamykacích mechanismů umožňujících sdílení prostředků mezi více jádry.

3.3.2 Firmware

Aplikace řídicí jednotky je založena na protokolu CANopen a představuje implementaci zažízení typu master. Základem je tedy knihovna *canopen-stack* [33], která zajišťuje veškeré služby spojené s komunikací. Na jednom kanálu sběrnice probíhá komunikace s doménovou sítí pomocí protokolu CAN, ve které jsou přítomny trakční měniče v režimu slave, obsluhující pohony vozidla. Druhý kanál slouží ke spojení s páteřní sítí vozidla. Jednotka v doménové síti vystupuje jako CANopen master a je zodpovědná za vysílání synchronizačních rámců zahajujících přenos příslušných procesních dat. Zároveň monitoruje aktuální stav trakčních měničů a případné chyby nahlašuje v páteřní síti. Výpočet procesních parametrů pro měniče zajišťuje právě vyexportovaný Simulink model, zkompilovaný společně se zdrojovým kódem ostatních dílčích komponent aplikace. Pro správu jednotlivých úloh aplikace byl opět,

stejně jako v případě doménového kontroléru podvozku, použit operační systém reálného času FreeRTOS.

Možnosti exportu a nasazení modelu na cílovém zařízení byly ověřeny na dílčí části simulace elektrického automobilu, určené pro testování pomocí metody hardware-in-loop [34]. Algoritmus v simulaci zajišťuje výpočet požadovaného momentu jednoho pohonu na základě aktuální hodnoty rychlosti a poloh plynového a brzdového pedálu. Přitomny jsou také funkce tempomatu či rekuperace na základě sklonu vozidla při brždění. Ty byly ale vypnuty, jelikož s aktuální výbavou elektromobilu eŠus není možné poskytnout vyžadované vstupní informace. Použitý model nebyl nijak optimalizován s cílem zjistit, jaká je v případě použitého mikrokontroléru náročnost aritmetických operací s datovým typem *double*.



Obrázek 3.4: Model regulace momentu hnacího ústrojí, použitý pro ověření možností exportu a nasazení na cílovém hardware. Pro potřeby testování byly vypnuty funkce tempomatu a sklonoměru. Ty vyžadují vstupy, jimiž vozidlo eŠus není vybaveno.

K exportu modelů slouží rozšíření software Matlab, zvané Embedded Coder. Tento plugin umožňuje nejen export do zdrojového kódu v jazyce C a C++, ale i možnost značné optimalizace výsledného kódu. V případě některých proměnných modelu lze nahradit v základu použitý datový typ double jinými, čímž lze dosáhnout značného zrychlení aritmetických operací na cílovém zařízení. Podporovány jsou

i optimalizace specifické pro cílovou architekturu procesoru. Model vyexportovaný z prostředí Matlab Simulink má následující strukturu:

Vstupy a výstupy Globální proměnné různých datových typů představující vstupy a výstupy modelu. Pomocí těchto proměnných lze zajistit interakci modelu se zbytkem aplikace.

Initialize() Funkce sloužící k inicializaci modelu a jeho nastavení do výchozího stavu. Volání proběhne při startu aplikace jednotky. V případě spuštění modelu na počítači jsou voláním této funkce alokovány prostředky potřebné pro výpočet.

Step() Klíčová funkce představující výpočet jednoho kroku modelu. Pro zajištění funcionality implementované modelem je nutné zajistit její periodické volání.

Terminate() Tato funkce slouží k ukončení výpočtu modelu a jeho uvedení do výchozího stavu. V případě běhu modelu na počítači dochází při volání také k uvolnění prostředků alokovaných pro výpočty. Na cílovém zařízení však její použití aktuálně není nutné, jelikož k restartu modelu v aplikaci jednotky nedochází. Uvedení proměnných modelu do výchozího stavu zajistí po resetu zařízení právě volání *Initialize()*.

V případě řídicí jednotky trakce zajišťuje běh modelu v periodě 1 ms vlákno operačního systému reálného času FreeRTOS. Před voláním funkce *Step()* jsou všechny vstupní proměnné modelu aktualizovány na hodnotu v příslušném čase přítomnou ve slovníku objektů CANopen. Po provedení výpočtu jsou opět do slovníku zapsány aktualní hodnoty výstupních proměnných.

Pro zjištění náročnosti modelu na hardwarové prostředky bylo provedeno měření pomocí software SEGGER SystemView, který slouží ke kolekci a vizualizaci diagnostických dat aplikace běžící na vestavěném systému. Pomocí tohoto nástroje lze vizualizovat běh celé aplikace, včetně přerušení a vláken operačního systému, na časové ose. Nasbíraná data slouží také k výpočtu různých statistik spojených s diagnostikovanou aplikací. Použitím SystemView bylo zjištěno, že v případě nasazeného modelu trvá výpočet jednoho kroku průměrně 2,7 µs a příslušné vlákno operačního systému využívá 0,03 % procesorového času. Vizualizaci měření lze nalézt v příloze A.10. Model přímo určený pro vozidlo eŠus bohužel nebyl vedoucím práce dodán, tudíž jej nebylo možné vyzkoušet na řídicí jednotce a změřit jeho náročnost na hardwarové prostředky cílového zařízení. Z naměřených hodnot však lze soudit, že hardware řídicí jednotky poskytuje dostatečnou výkonovou rezervu pro implementaci aplikace splňující podmínku reálného času, stanovenou na 1 ms.

4 Možnosti budoucího vývoje

Díky návrhu nové komunikační sítě a implementaci řídicích jednotek bylo dosaženo základní funcionality zajišťující pojízdný stav automobilu eŠus. Do budoucna se však otevírají nové možnosti budoucího vývoje jeho elektroniky.

Regulace pohonů

Regulace pohonů vozidla vyžaduje implementaci a nasazení algoritmů zajišťujících výpočet parametrů pro přítomné trakční měniče. Zde se nabízí výzkum progresivních technologií jako například *torque vectoring* s cílem zajistit optimální distribuci momentu mezi jednotlivá kola vozidla na základě jejich aktuálních otáček či úhlu natočení volantu.

Trakční doména

Realizovaná trakční jednotka vyžaduje náhradu aktuálního řešení v podobě vývojové desky za proprietární hardware splňující bezpečnostní požadavky pro použití v automobilu. V případě použití vícejádrových mikrokontrolérů je nutná implementace algoritmů sloužících k zajištění sdílení prostředků mezi jednotlivými jádry.

Správa baterie

Doménová síť baterie nebyla řešena v rámci této práce a proto je do budoucna nutná implementace jejího doménového kontroléru, který zajistí komunikaci se systémem správy baterie a komponentami určenými pro nabíjení vozidla. Zde se opět otevírají možnosti implementace a testování různých algoritmů, například pro řízení nabíjecího procesu nebo monitorování izolačniho stavu výkonových prvků elektrické sítě.

Infotainment

U elektromobilu eŠus je aktuální výbava infotainmentu omezena pouze na digitální přístrojový panel, dodávaný k trakčním měničům Sevcon. Ten zobrazuje pouze základní jízdní informace. Jelikož se jedná a experimentální vozidlo, není vyžadována přítomnost funkcí komfortu, jako například rádio či navigace. Pro potřeby vizualizace či nastavení některých parametrů v rámci testování vozidla se nabízí realizace nejen nového přístrojového panelu, ale i centrálního dotykového panelu, podobně jako je dnes například u automobilů značky Tesla.

Osvětlení

Aby bylo možné s elektromobilem provádět legálně testovací jízdy na veřejných komunikacích, je v budoucnu plánováno požádat o jeho registraci a přidělení poznávacích značek určených pro prototypy. Schválení takového požadavku je podmíněno splněním základních bezpečnostních předpisů. Jedním z nich je právě přítomnost osvětlení. Pro zajištění této funkcionality by bylo nutné realizovat doménu karoserie se všemi potřebnými prvky.

5 Závěr

Tato práce objasňuje aktuální trendy komunikačních technologií využívaných v automobilovém průmyslu. Z rešerše vyplývá výrazný vliv protokolu Ethernet na vývoj budoucích automobilových platforem. Přiložený rozhovor se specialistou z oboru testování elektronických systémů vozidel tento trend potvrzuje. Překvapivým zjištěním byla redukce délky kabeláže potřebné k propojení palubních systémů z jednotek kilometrů na stovky metrů díky nasazení Ethernetu v kombinací se zónovou architekturou komunikační sítě. Potvrzen je také neustále narůstající význam software v moderních automobilech, díky čemuž lze nejen poskytovat rozličné služby navyšující komfort uživatele, ale i zajistit vzdálenou diagnostiku či aktualizaci.

Praktická část práce se zabývá návrhem nové komunikační architektury experimentálního elektromobilu eŠus, vyvíjeného na Technické univerzitě v Liberci. Zároveň byla implementována dílčí část řídicích jednotek této nové architektury. Nová komunikační síť společně s rozšířením hnacího ústrojí vozidla o dva měniče umožňující samostatné řízení pohonů zadních kol tedy nyní umožňuje testování technologií, které jsou předmětem výzkumu Laboratoře elektromobility. Vozidlo nyní tvoří novou experimentální platformu pro budoucí vývoj a poskytuje prostor k rozšíření stávající funkcionality například formou dalších závěrečných prací ve spolupráci se studenty fakulty. Platforma tedy může v budoucnu přispět i ke zvýšení zájmu o komunikační technologie a elektronické systémy vozidel.

Přínosy autora v rámci této práce tedy jsou:

1. návrh nové architektury komunikační sítě elektromobilu eŠus a realizace řídicích jednotek podvozkové a trakční domény
2. příprava trakční domény vozidla pro řízení dvojice pohonů na zadní nápravě
3. implementace knihovny *can-node*, sloužící ke komunikaci založené na signálech dle standardu AUTOSAR, a rozšíření knihovny *canopen-stack* o funkcionalitu vyžadovanou v nové komunikační síti vozidla
4. řešení kolizí identifikátorů CAN vlivem přítomnosti dvou identických komponent vozidla na jedné sběrnici pomocí specializovaného hardware
5. analýza možností a návrh koncepce dalšího vývoje v souvislosti s elektromobilem eŠus
6. podíl na návrhu hardware řídicích jednotek, využívaného Laboratoří elektromobility pro realizaci projektů ve spolupráci s průmyslem

Použitá literatura

[1] VDOVIĆ, Hrvoje; BABIC, Jurica; PODOBNIK, Vedran. Automotive Software in Connected and Autonomous Electric Vehicles: A Review. *IEEE Access*. 2019, roč. 7, s. 166365–166379. ISSN 2169-3536. Dostupné z DOI: 10.1109/ACCESS.2019.2953568.

[2] HARTWICH, F.; BOSCH, R. CAN with Flexible Data-Rate. In: *iCC 2012: 13th international CAN Conference*. Hambach Castle, DE: CAN in Automation, 2012, s. 9.

[3] FIJALKOWSKI, B.T. Local Interconnect Networking. In: FIJALKOWSKI, B. T. (ed.). *Automotive Mechatronics: Operational and Practical Issues: Volume I*. Dordrecht: Springer Netherlands, 2011, s. 57–59. Intelligent Systems, Control and Automation: Science and Engineering. ISBN 978-94-007-0409-1. Dostupné z DOI: 10.1007/978-94-007-0409-1_5.

[4] RUFF, M. Evolution of local interconnect network (LIN) solutions. In: *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No.03CH37484)*. Orlando, FL, USA: IEEE, 2003, sv. 5, 3382–3389 Vol.5. ISBN 0-7803-7954-3. Dostupné z DOI: 10.1109/VETECF.2003.1286317. ISSN: 1090--3038.

[5] HUSE, Markus Iversen. *FlexRay Analysis, Configuration Parameter Estimation, and Adversaries*. Trondheim, NO, 2017. Diplomová práce. Norwegian University of Science a Technology. Vedoucí práce Sverre HENDSETH.

[6] XU, Yi-Nan et al. Implementation of FlexRay protocol with an automotive application. In: *2008 International SoC Design Conference*. 2008, sv. 02, s. II.25–II.28. Dostupné z DOI: 10.1109/SOCDC.2008.4815675.

[7] HANK, Peter et al. Automotive Ethernet: In-vehicle networking and smart mobility. In: *2013 Design, Automation Test in Europe Conference Exhibition*. 2013, s. 1735–1739. Dostupné z DOI: 10.7873/DATE.2013.349. ISSN: 1530--1591.

[8] *AUTOSAR: Specification of Socket Adaptor*. Munich, Germany, 2021. Standard, AUTOSAR 416. Automotive Open System Architecture. Dostupné také z: https://www.autosar.org.

[9] IXIA. *Automotive Ethernet: An Overview*. 2017. 915-3510-01 Rev. A. Ixia Technologies. Dostupné také z: https://www.ixiacom.com.

[10] BOATRIGHT, Robert B et al. *Automotive Ethernet - The Definitive Guide*. USA: Intrepid Control Systems, 2014. ISBN 978-0-9905388-0-6. Dostupné z DOI: 10.4271/099053880X.

[11] *AUTOSAR: Specification of Communication*. Munich, Germany, 2017. Standard, AUTOSAR 015. Automotive Open System Architecture. Dostupné také z: https://www.autosar.org.

[12] RACU, Razvan et al. Automotive Software Integration. In: *2007 44th ACM/IEEE Design Automation Conference*. San Diego, CA, USA: IEEE, 2007, s. 545–550. Dostupné z DOI: 10.1145/1278480.1278619. ISSN: 0738--100X.

[13] LIEDER, Roland. Gateway processor evolution in automotive networks. In: *iCC 2017: 16th international CAN Conference*. Nuremberg, DE: CAN in Automation, 2017, s. 6.

[14] GOPU, G. L.; KAVITHA, K. V.; JOY, James. Service Oriented Architecture based connectivity of automotive ECUs. In: *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*. 2016, s. 1–4. Dostupné z DOI: 10.1109/ICCPCT.2016.7530358.

[15] KUGELE, Stefan; HETTLER, David; PETER, Jan. Data-Centric Communication and Containerization for Future Automotive Software Architectures. In: *2018 IEEE International Conference on Software Architecture (ICSA)*. 2018, s. 65–6509. Dostupné z DOI: 10.1109/ICSA.2018.00016.

[16] FARSI, M.; RATCLIFF, K. An introduction to CANopen and CANopen communication issues. In: *IEE Colloquium on CANopen Implementation (Digest No. 1997/384)*. London, UK: IET, 1997, s. 2/1–2/6. Dostupné z DOI: 10.1049/ic:19971322.

[17] *CiA 301: Application layer and communication profile*. Nuremberg, DE: CAN in Automation, 2002-02-13. Standard, CiA 301. Dostupné také z: https://www.can-cia.org/.

[18] BARBOSA, M.B.M.; SILVA CARVALHO, A. da; FARSI, M. A CANopen I/O module: simple and efficient system integration. In: *IECON '98. Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (Cat. No.98CH36200)*. Aachen, Germany: IEEE, 1998, sv. 1, 155–159 vol.1. ISBN 0-7803-4503-7. Dostupné z DOI: 10.1109/IECON.1998.723962.

[19] JANDURA, Pavel; BUKVIC, Martin. Lightweight Battery Electric Vehicle for Educational Purposes. *Applied Mechanics and Materials*. 2013, roč. 390, s. 281–285. Dostupné z DOI: 10.4028/www.scientific.net/AMM.390.281.

[20] HYŠKA, Tomáš. *Návrh systému řízení zadních kol pro experimentální elektrické vozidlo*. Liberec, CZ, 2020. Bakalářská práce. Technická univerzita v Liberci. Vedoucí práce Pavel JANDURA.

[21] BŘOUŠEK, Josef; BUKVIC, Martin; JANDURA, Pavel. Experimental Electric Vehicle eŠus Gen2. *Journal of Middle European Construction and Design of Cars*. 2016, roč. 14. Dostupné z DOI: 10.1515/mecdc-2016-0007.

[22] GHOSH, Jyotishman; TONOLI, Andrea; AMATI, Nicola. A Torque Vectoring Strategy for Improving the Performance of a Rear Wheel Drive Electric Vehicle. In: *2015 IEEE Vehicle Power and Propulsion Conference (VPPC)*. 2015, s. 1–6. Dostupné z DOI: 10.1109/VPPC.2015.7352887.

[23] ROMPOTL, Tomáš. *Návrh systému přiřizování zadní nápravy experimentálního elektromobilu*. Liberec, CZ, 2018. Bakalářská práce. Technická univerzita v Liberci. Vedoucí práce Pavel JANDURA.

[24] NOUVEL, Fabienne et al. Automotive Network Architecture for ECUs Communications. In: 2009, s. 69–90. ISBN 978-1-60566-338-8. Dostupné z DOI: 10.4018/978-1-60566-338-8.ch004.

Online reference

[25] *Classical CAN v/s CAN FD: Decoding Their Data Transfer Capabilities and Compatibility with the Bootloader Software* [Embitel Technologies Pvt. Ltd.] [online]. 2018-07-20 [cit. 2022-04-16]. Dostupné z: https://www.embitel.com.

[26] LAMBERT, Fred. *Tesla reveals revolutionary new wiring architecture to help robots build upcoming cars like Model Y* [Electrek - EV and Tesla news] [online]. 2019-07-22 [cit. 2022-03-23]. Dostupné z: https://electrek.co.

[27] DIESS, Herbert. *NEW AUTO Strategy Presentation* [online]. Wolfsburg, DE, 2021 [cit. 2021-07-15]. Dostupné z: https://www.volkswagenag.com. NEW AUTO Strategy Presentation.

[28] *Tesla over-the-air upgrades* [Tesla] [online]. 2019-12-18 [cit. 2022-04-18]. Dostupné z: https://www.tesla.com.

[29] NADAV, Idan. *The Evolutionary Path to Zonal E/E Architecture* [GuardKnox Cyber Technologies] [online]. 2021-02-16 [cit. 2022-04-11]. Dostupné z: https://blog.guardknox.com.

[30] KURUMBUDEL, Prashanth Ram. *SOA, Network Protocols and Connected Car Framework for Future Vehicles* [TelematicsWire] [online]. 2020-03-18 [cit. 2021-10-27]. Dostupné z: https://www.telematicswire.net.

[31] KOČÁRKOVÁ, Jaroslava. *Máme vlastní školní elektromobil, říká se mu eŠus.* [T-uni] [online]. 2013-05-24 [cit. 2021-10-23]. Dostupné z: https://tuni.tul.czech.

[32] WHITE, Frames Catherine. *How bad is it to use dynamic datastuctures on an embedded system?* [StackOverflow] [online]. 2017-08-17 [cit. 2021-12-15]. Dostupné z: https://stackoverflow.com.

[33] HILLMANN, MICHAEL. *canopen-stack: Free CANopen Stack* [git]. Wangen im Allgäu, DE, 2021. Ver. 4.2.0. Dostupné také z: https://github.com.

[34] *Electric Vehicle Configured for HIL* [MathWorks] [online]. 2022 [cit. 2022-04-25]. Dostupné z: https://www.mathworks.com.

A Přílohy

A.1 Rozhovor: Dipl.-Ing. Franz Dengler

Name: Dipl.-Ing. Franz Dengler

Profession: Senior Expert Systems Engineering, MicroNova Gmbh, responsible for product management, new technologies and innovation.

Education: Electrical Engineering, University of Applied Sciences Munich
Computer Science, Fernuniversität in Hagen

In your opinion, what is currently the biggest issue related to automotive networking?

There are a lot of issues and it is difficult to name THE biggest issue. Descending order below.

- Increasing complexity at a reasonable price
- Security issues. Car is a mobile computer with remote access
- Safety, reliability, redundancy. Autonomous driving requires high levels of security.

What is the role of conventional networking technologies like Ethernet, Wifi or LTE in modern vehicles?

Convential Ethernet is too expensive vor vehicles. Automotive Ethernet has been standardized. It uses two wire connection for bidirectional communication on a simple twisted pair cable. Meanwhile up to 10GB/s! Convential Ethernet technology is currently not suited for all applications, especially safety related functions. However technologies like TSN (time sensitive networking) coming from industrial automation will fill this gap. I expect that when that technology is available Ethernet will become the dominant network in a vehicle.

WiFi I see mainly for consumer applications like mobile devices in cars. Initially it was discussed to have WiFi technology as a base for CAR2x applications. But it looks like if technology moves more into direction 5G. LTE and 5G will play a major role in Car to X applications (C2X).

What are the advantages and disadvantages of adopting such technologies in context of vehicle systems?

Advantages are about cost. It is a mass technology that needs not to be developed specially for cars. Disadvantaes are that all of the problems present in current applications of technologies are now mirrored into automotive. Especially all the security issues.

Might they fully replace "classic" bus systems like CAN, LIN or FlexRay in future?

In case of CAN and LIN, this depends on the price. As long as they are a few cents cheaper they will stay. Border is 10 Mbit/s where Bosch introduced CAN-XL that should be able to transmit whole IP-Frames. Here there is a good chance that CAN-XL will not be used in vehicles. FlexRay will be in the long term replaced by Automotive Ethernet. As FlexRay is used in areas like gearbox, ESP, motor there are high safety requirements. It took a long time to get this for FlexRay running. And so I expect a slow adaptation because the current areas are established and running. And innovation is not so high there.

In this thesis, two concepts of automotive networks are presented besides currently most common gateway architecture, domain and zonal. Which one is the most aspiring successor of gateway arch. for near future vehicles?

In the near future the zonal architecture is probably the most aspiring sucessor. Reason is again the cost.

Is it possible to define modern and future automobiles as software-oriented consumer product instead of as a machine?

I think there will be a layered approach. Technical areas where development is quite far and new developments are not expected in a big amount (e.g. vehicle dynamics) and there are safety related things will be developed in the old way (Classic Autosar) there will be used the old approach. Here there is only evolution. For new things, architecture will become much more software centered. This can be found in Autosar Adaptive which is based on a service oriented architecture. An examples is SOME/IP where services communicate. Service discovery is made at runtime. New vehicle architectures are currently already defined in an service oriented manner.

Does application centralization in vehicle networks present possible safety issue?

All the problem from the classic internet technologies are transfered to the car. There are already a lot of famous hacks.

How does your company fit into automotive development?

The software and system company MicroNova offers innovative products, solutions and services for various industries. In Vierkirchen, north of Munich and at eight other locations in Germany and the Czech Republic, our Testing experts are working on innovative solutions for testing electronic control and regulation components.

According to rising volume of shared data in vehicle network, how is it possible to evaluate and ensure safety of design?

I think we clearly must differentiate between safety and security. Safety must be designed into the product (see ISO 26262). How to ensure safety in areas like automated driving is a matter of current research. There are no established methods currently.The problem is less the amount of data in the network. It is more how to ensure that reaction on that data is correct. Especially as in applications like AD there are no equal situations. Security is a different topic. Here all mechanims done in other areas like access control or digital signatures must be applied.

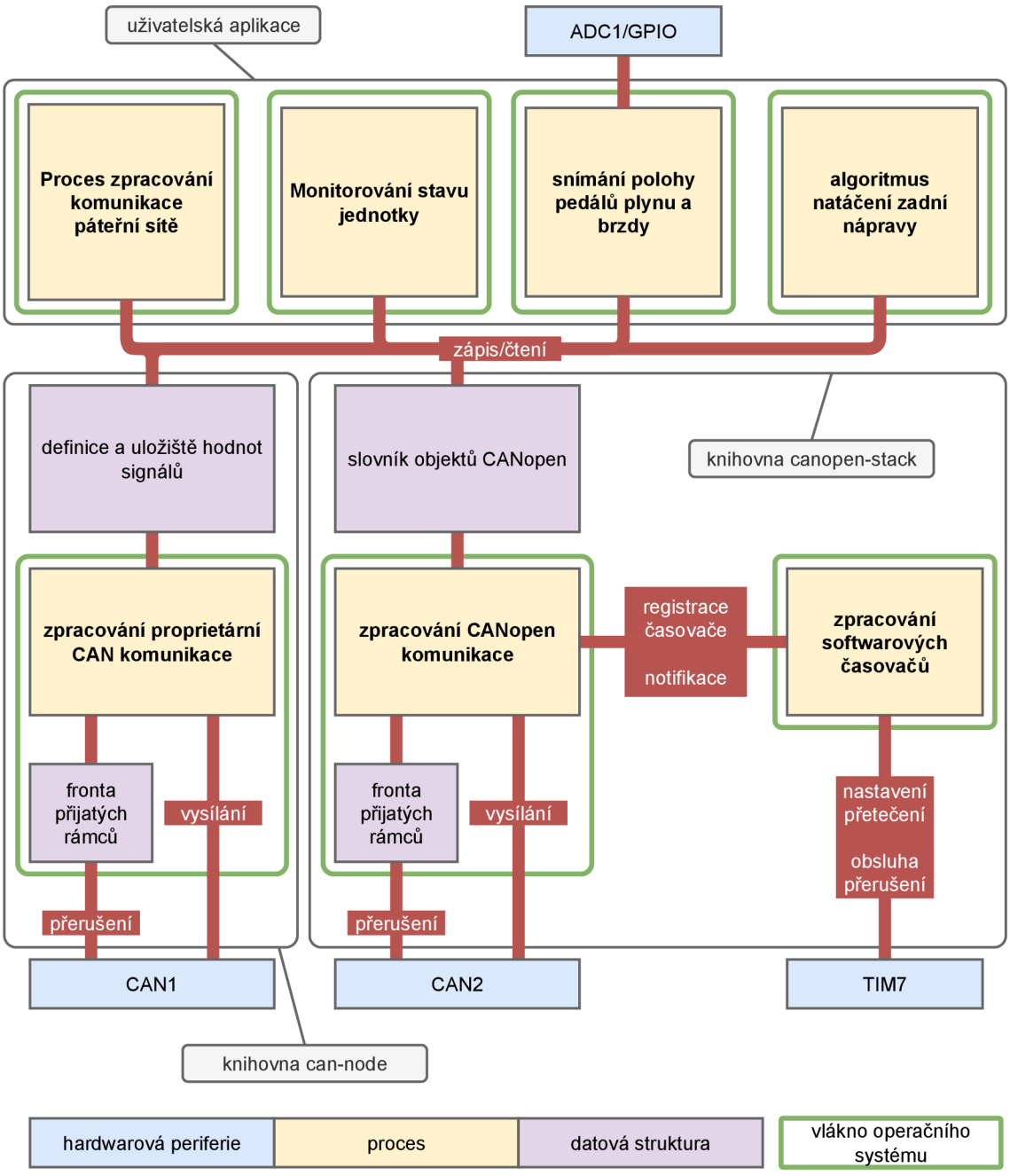
Is networking tested on application level only or does testing even include protocol level evaluation (bit manipulation etc.)?

Bit manipulation is interesting only in context to ensure that low level protocols work, resulting typically into discarded messages. Testing on protocol level will be much more important.

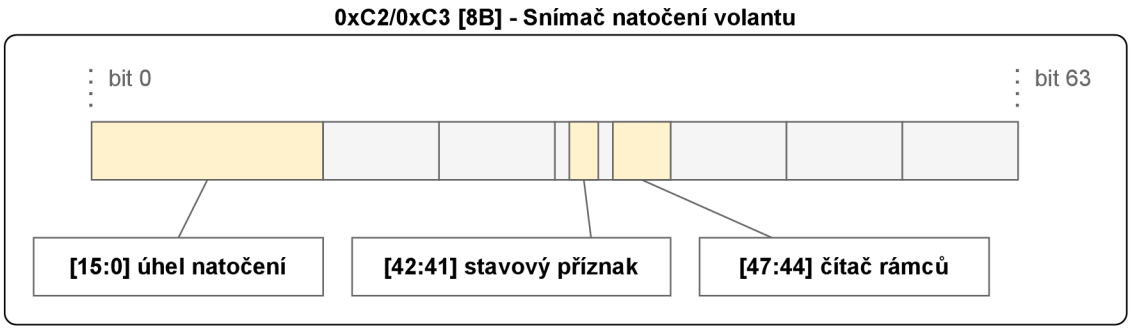
Is it possible to evaluate network design before even having physical prototypes of control units?

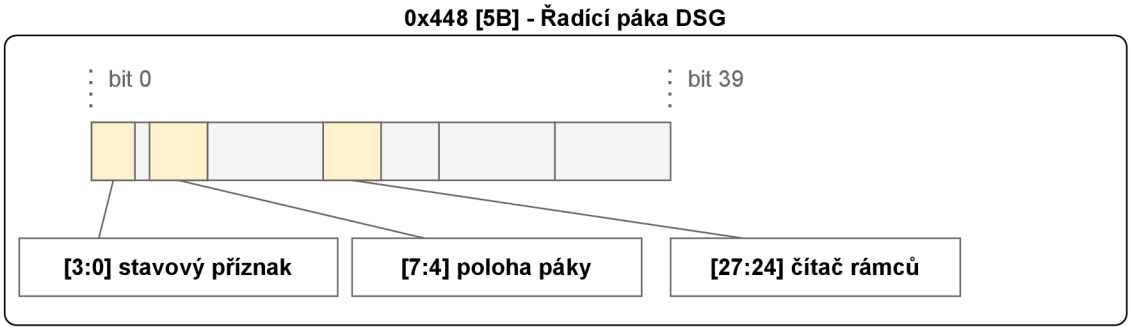
I have a telecommunications background. And simulation of network scenarios was state of the art there. A typical product used was OPNET. As far as I can remember the focus was more on load scenarios (maximum delays, usage of connection lines and so on). However I found also some applications e.g. for CAN protocol. I cannot estimate if it is worth the effort to do modelling on that level.

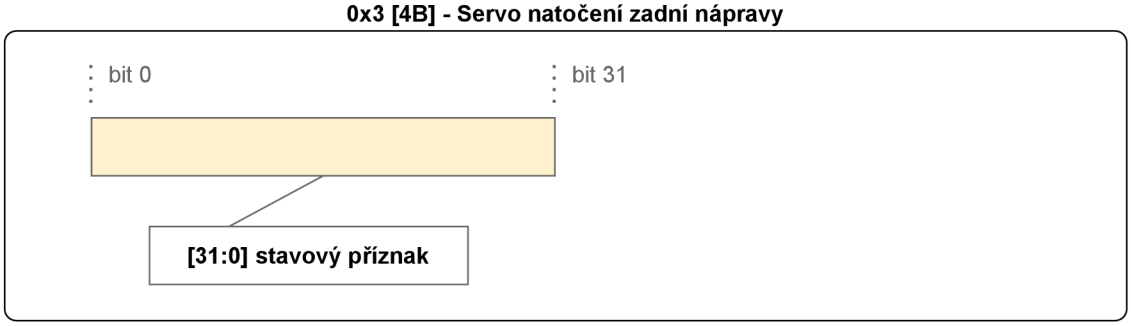
A.2 Schéma firmware řídicí jednotky podvozku

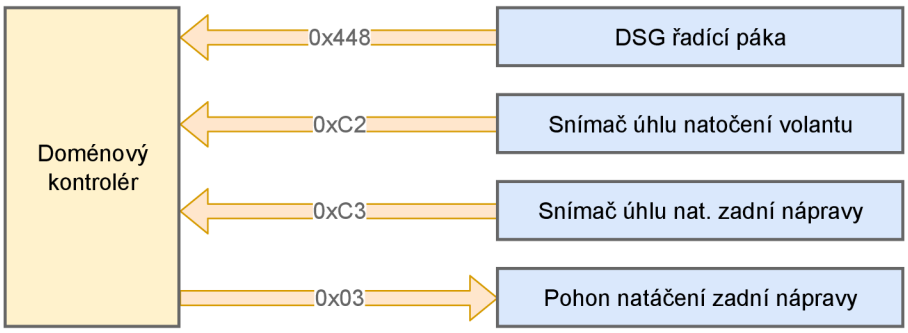


A.3 Popis komunikačních rámců doménové sítě podvozku

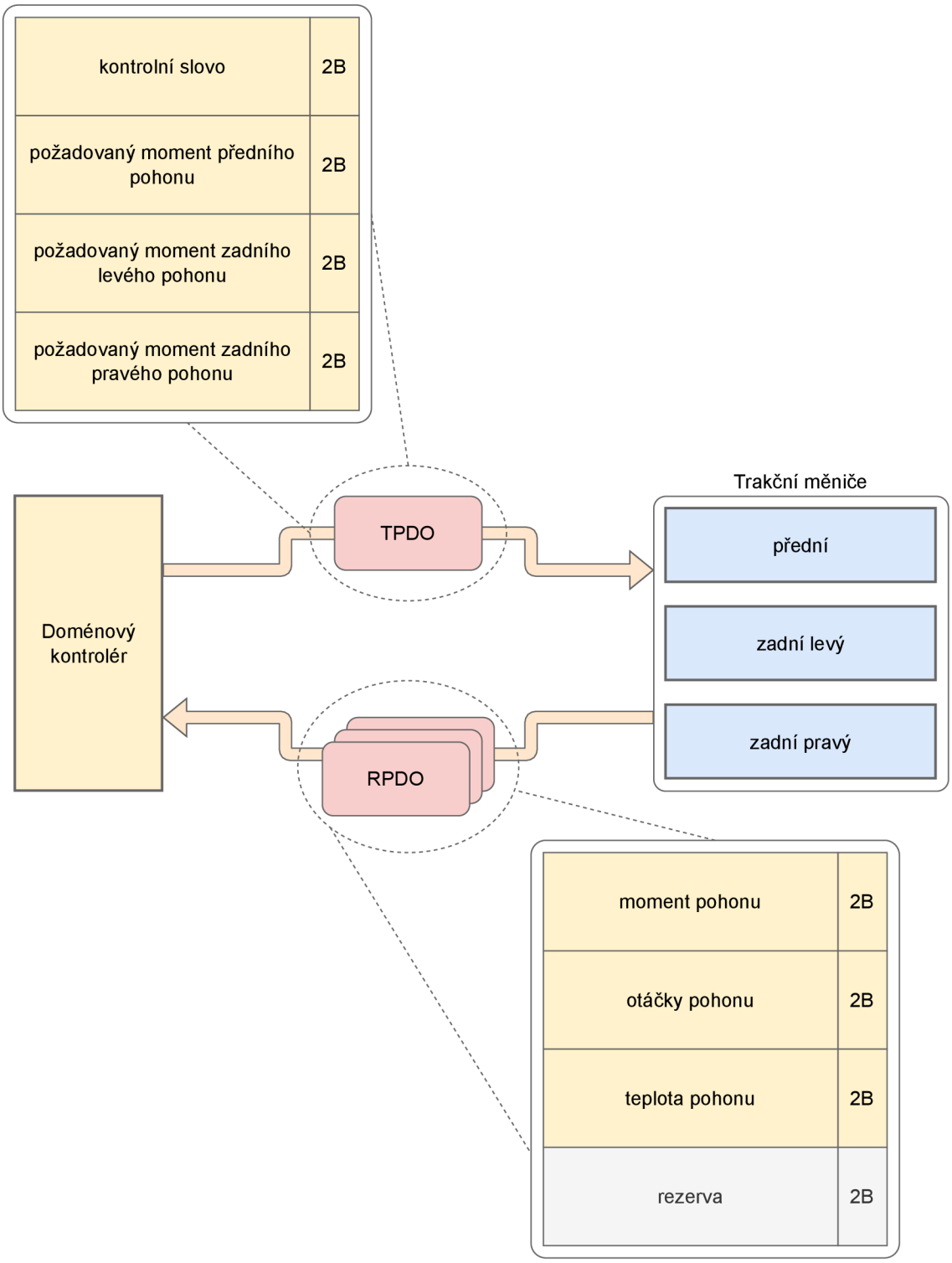




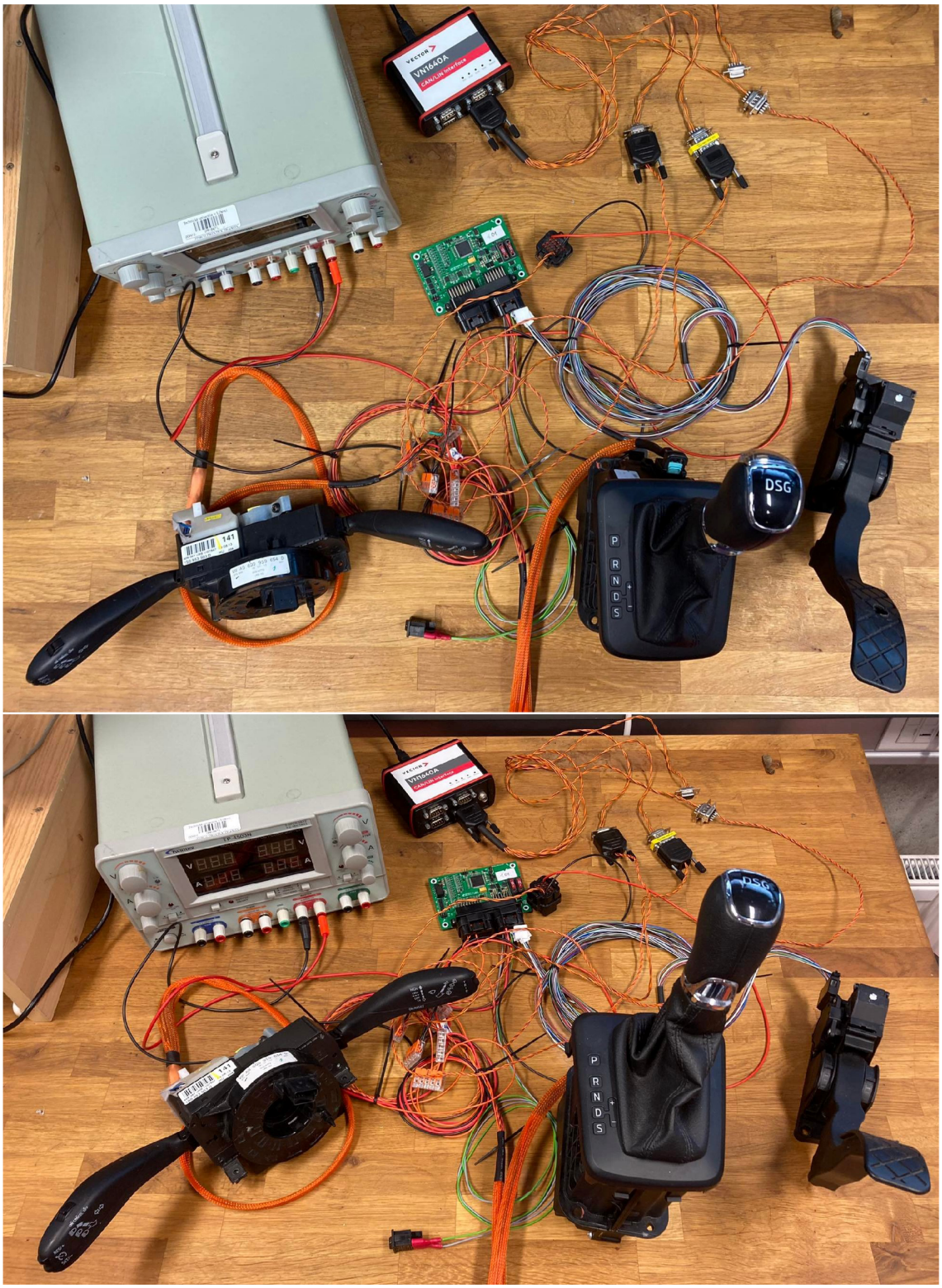




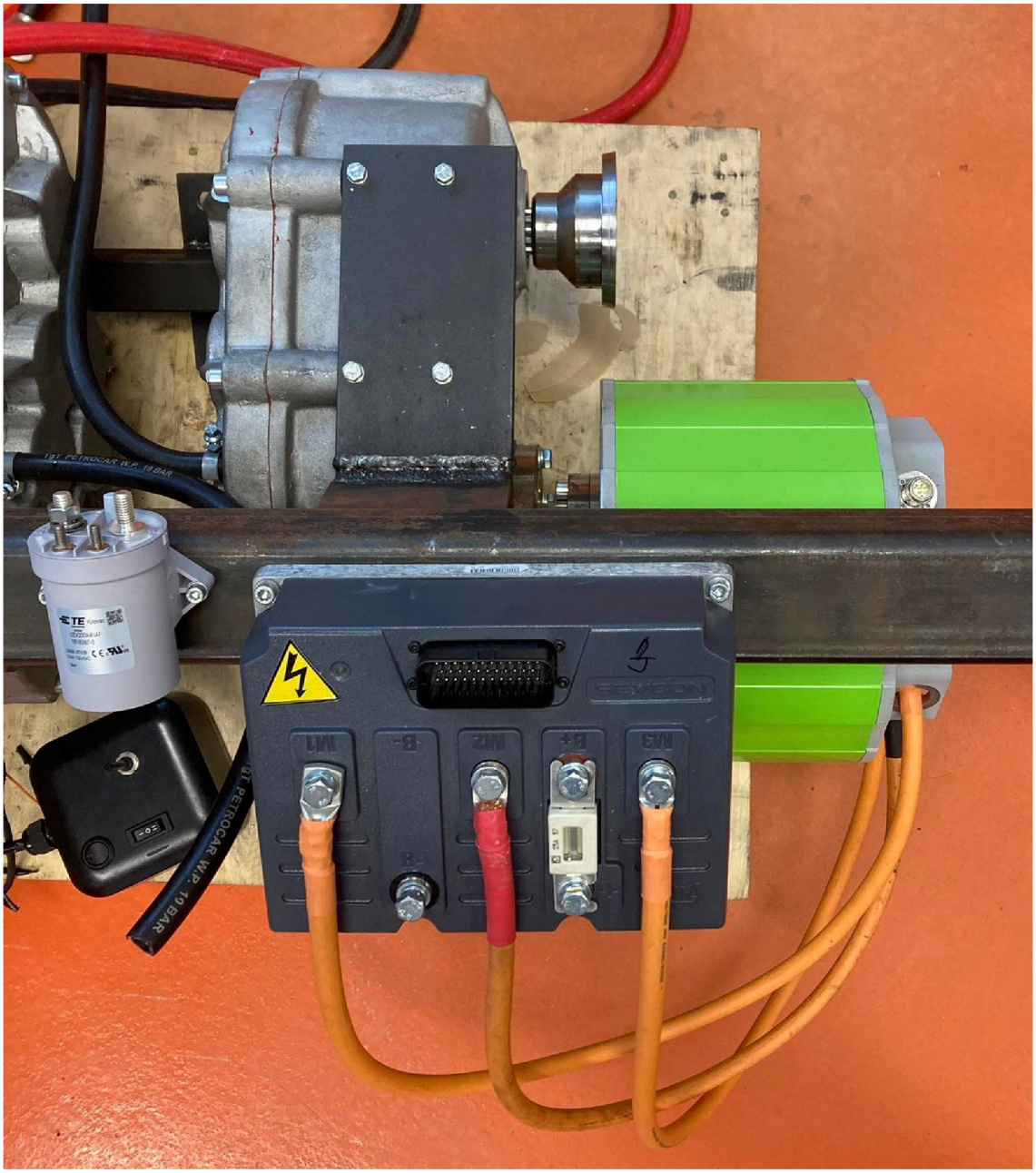
A.4 Struktura CANopen komunikace doménové sítě trakce



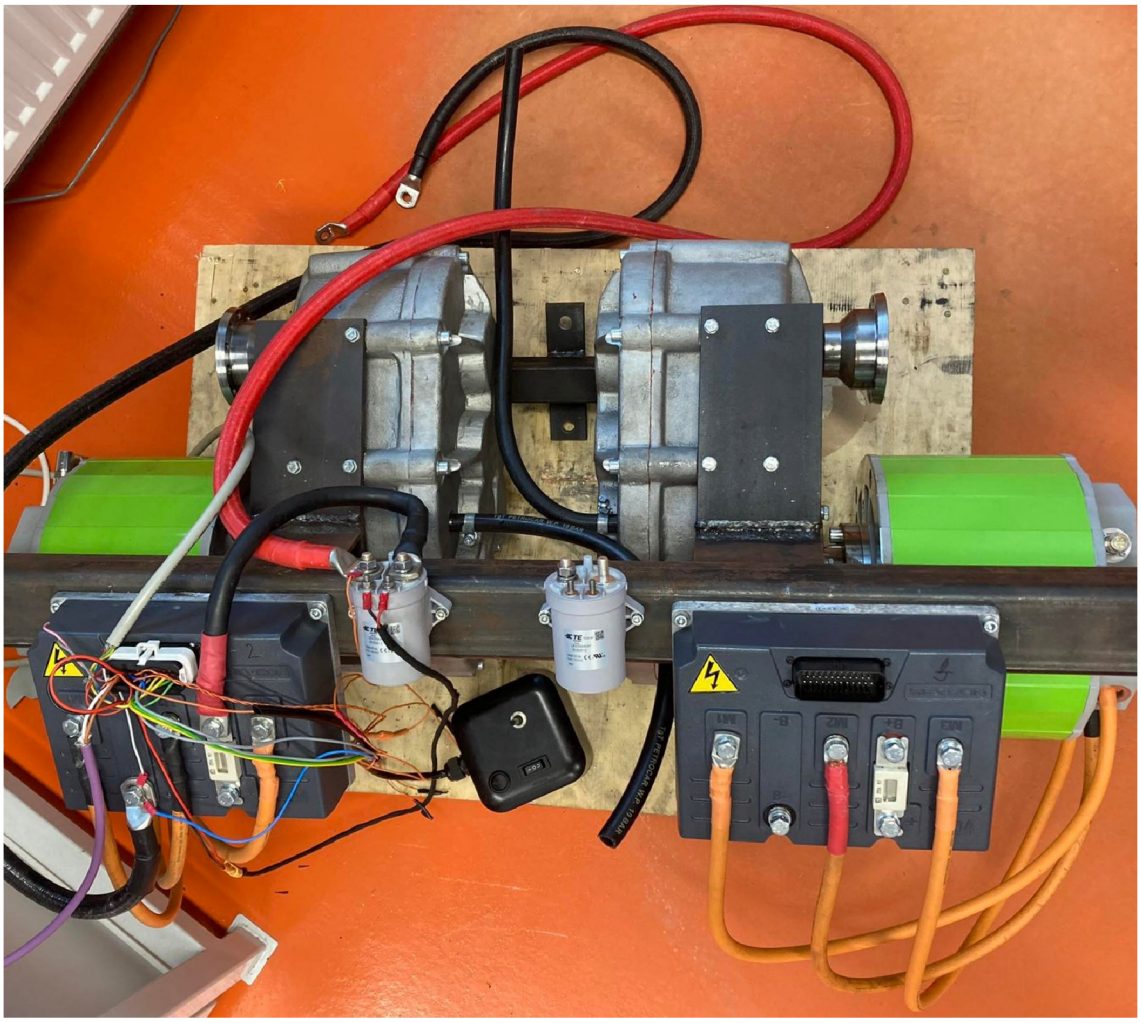
A.5 Snímek laboratorního zapojení doménové sítě podvozku



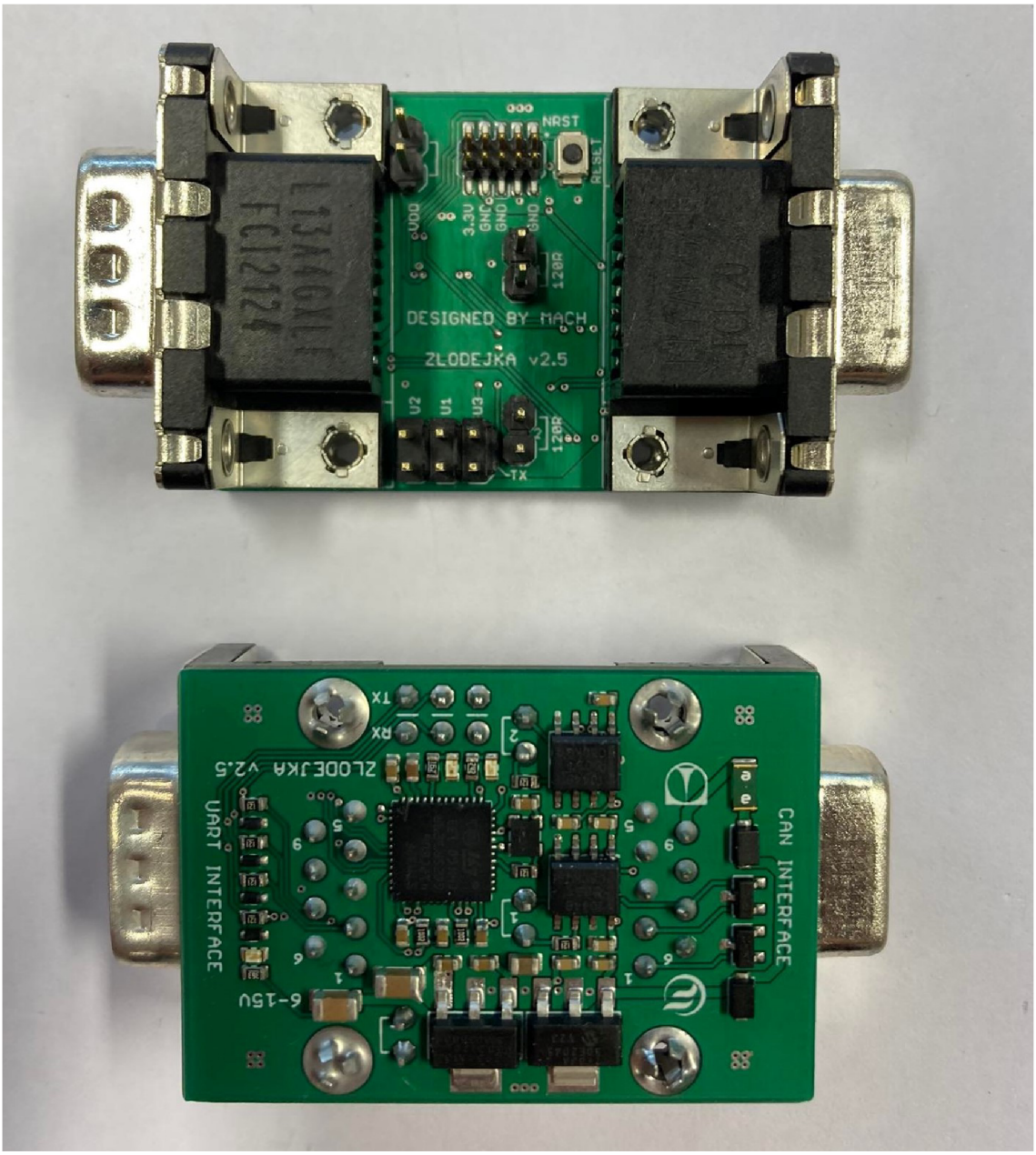
A.6 Snímek hnacího ústrojí zadní nápravy 1



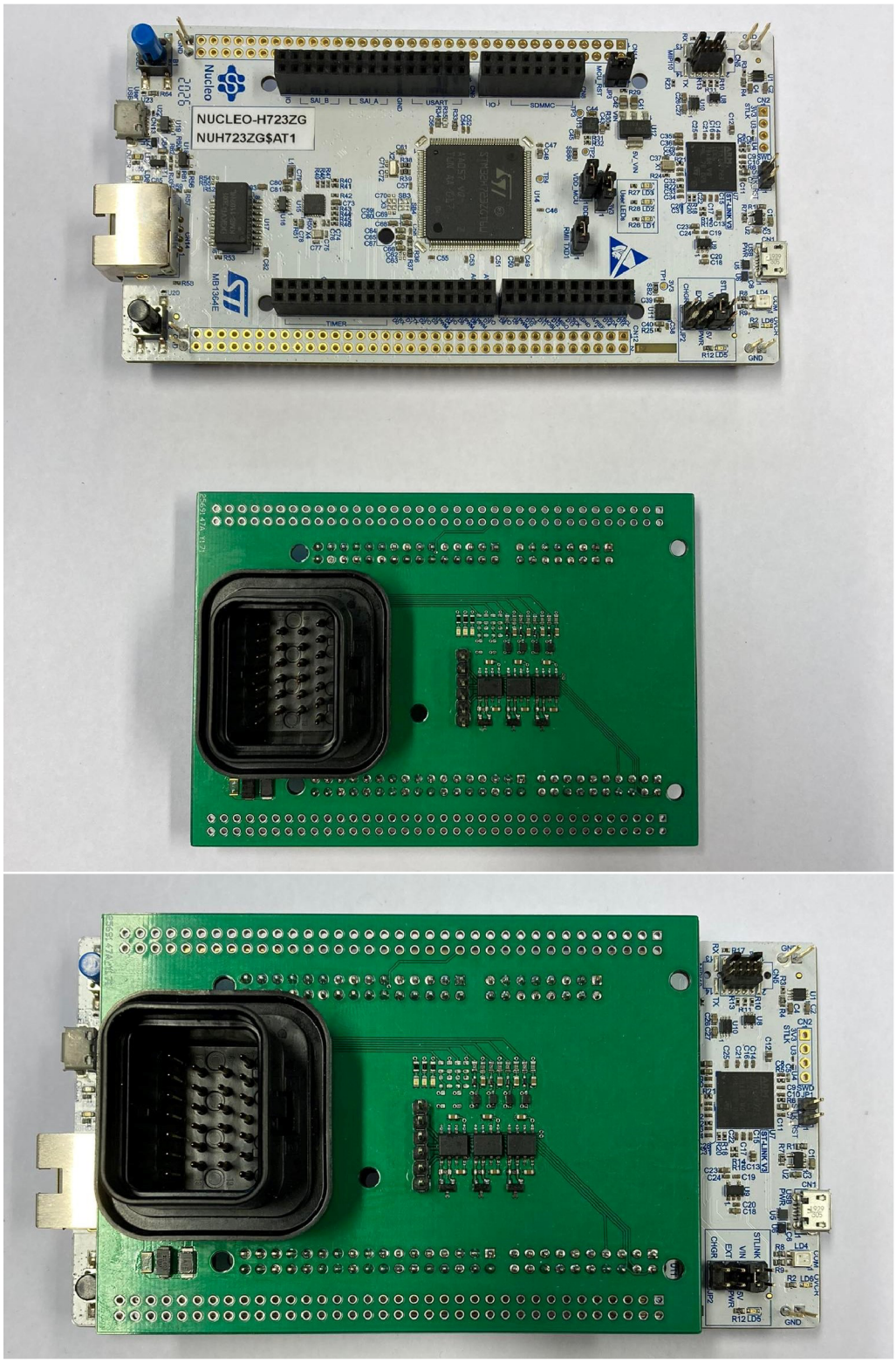
A.7 Snímek hnacího ústrojí zadní nápravy 2



A.8 Snímek hardware převodníku identifikátorů CAN



A.9 Snímek funkčního vzorku řídicí jednotky trakce



A.10 Snímek metriky testovaného modelu