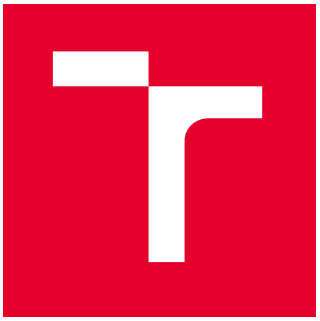


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

AUTOMOBILOVÉ SBĚRNICE V PROSTŘEDÍ MATLAB SIMULINK

CAR INDUSTRY COMMUNICATION PROTOCOLS IN MATLAB SIMULINK

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Ondřej Sikora**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Lukáš Pohl, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Ondřej Sikora ***ID:*** 209467

Ročník: 2 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Automobilové sběrnice v prostředí MATLAB Simulink

POKYNY PRO VYPRACOVÁNÍ:

1. Proveďte rešerši sběrnic aktuálně používaných v automobilovém průmyslu
2. Vytvořte MATLAB Simulink model komunikující po vybrané sběrnici prostřednictvím podporovaného komunikačního zařízení
3. Vytvořte databázový soubor obsahující alespoň jedno řídicí a jedno řízené zařízení
4. Ověřte funkčnost vytvořeného databázového souboru na reálné komunikaci

DOPORUČENÁ LITERATURA:

VOSS, Wilfried. A comprehensible guide to controller area network. Copperhill Media, 2008.

PARET, Dominique. Multiplexed networks for embedded systems: CAN, LIN, flexray, safe-by-wire.. John Wiley & Sons, 2007.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 17.5.2023

Vedoucí práce: Ing. Lukáš Pohl, Ph.D.

doc. Ing. Petr Fiedler, Ph.D.
předseda rady studijního programu

Abstrakt

Tato diplomová práce se zabývá nejrozšířenějšími automobilovými komunikačními standardy. Je zde zpracována rozsáhlá teoretická část, na kterou navazuje praktická část práce. Praktická část je řešena za použití vývojového kitu Nucleo F446RE.

Klíčová slova

LIN sběrnice, CAN sběrnice, Vehicle Network Toolbox, Simulink, Vector

Abstract

This thesis deals with the most widely used automotive communication standards. There is an extensive theoretical part, which is followed by a practical part of the thesis. The practical part is solved using the Nucleo F446RE development kit.

Keywords

LIN bus, CAN bus, Vehicle Network toolbox, Simulink, Vector

Bibliografická citace

SIKORA, Ondřej. *Automobilové sběrnice v prostředí MATLAB Simulink* [online]. Brno, 2023 [cit. 2023-04-27]. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/151852. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Lukáš Pohl.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	**Ondřej Sikora**
VUT ID studenta:	**209467**
Typ práce:	***Diplomová práce***
Akademický rok:	***2022/23***
Téma závěrečné práce:	**Automobilové sběrnice v prostředí MATLAB Simulink**

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucího závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

.

V Brně dne 16. května 2023 ----------

podpis autora

Poděkování

Mé poděkování patří vedoucímu diplomové práce, panu Ing. Lukášovi Pohlovi, Ph.D. Děkuji za cenné rady, připomínky a postřehy.

V Brně dne 16. května 2023 ----------

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM ZKRATEK

ABS	Antilock Brake System
ACK	Acknowledge
ACW	Anti Clockwise rotation
ADAS	Advanced Driver Assistance System
CAN	Contorller Area Network
CANdb	CAN database
CAN-FD	Contorller Area Network – Flexible Datarate
CRC	Cyclic Redundancy Check
CSMA/CR	Carrier Sense Multiple Access with Collision Detection
CW	Clockwise rotation
D2D	Domestic Digital Bus
DCS	Dynamic Control Segment
DTC	Diagnostic Trouble Code
ECU	Electronic Control Unit (mikrokontrolér)
EOF	End of Frame
ESP	Electronic Stability Program
EV	Electric Vehicle
FIFO	First In, First Out
HAL	Hardware Abstraction Layer
HEV	Hybrid Electric Vehicle
IFS	Interframe Space
IoT	Internet of Things
ISO	International Organization for Standardization
LIN	Local Interconnection Network
LKA	Lane Keeping Assist
LVDS	Low Voltage Differential Signaling
MIL	Malfunction Indicator Light
MOST	Media Oriented System Transport
OBD	On-Board Diagnostics
OSI	Open Systém InterConnect
PID	Protected Identifier
PWM	Pulse-Width Modulation
RTH	Receiver Threshold
RTL	Receiver Threshold Level
RTOS	Real-Time Operating System
RTR	Remote transmission Request
SAE	Society of Automotive Engineers
SAS	Steering Wheel Angle Sensor
SCI	Serial Communication Interface X Static Control Segment
SOF	Start of Frame
TDMA	Time Division Multiple Access
UART	Universal Asynchronous Receiver-Transmitter

Úvod

Automobilový průmysl se neustále vyvíjí a rychlý pokrok v komunikačních systémech vozidel hraje klíčovou roli při zvyšování bezpečnosti, efektivity a celkového zážitku z jízdy. Automobilové sběrnice slouží jako nepostradatelná komunikační páteř, která umožňuje výměnu dat mezi komponenty ve vozidle. Pochopení různých typů automobilových sběrnic a jejich praktických aplikací je nezbytné pro vývoj inovativních automobilových systémů.

Tato práce je motivována snahou o hlubší pochopení oblasti automobilové komunikace a klade si za cíl provést průzkum používaných automobilových sběrnic. Primárně se zaměří na obsáhlou rešerši sběrnic LIN a CAN, přičemž se bude zabývat i dalšími významnými protokoly, jako jsou MOST, Ethernet a FlexRay.

V návaznosti na teoretický průzkum zde bude uvedena také praktická část, která rovněž přispěje k rozšíření znalostí a lepšímu pochopení problematiky sběrnic.

Dalším cílem této práce je demonstrovat funkčnost vybrané sběrnice vytvořením souboru obsahujícího jedno řízené a jedno řídicí zařízení. Zařízení bude využívat mikrokontrolér Nucleo F446RE. A bude komunikovat prostřednictvím sběrnice CAN, čímž se prokáže praktická použitelnost zvolené sběrnice pro komunikaci v reálném čase v automobilových systémech.

1. AUTOMOBILOVÉ SBĚRNICE

Automobilová elektronika je stále a rychle se rozvíjejícím odvětvím. A to s rostoucím počtem bezpečnostních, asistenčních a informačních zařízení, které se stávají standardem v nových vozidlech. Vnitřní komunikační sítě jsou nezbytné pro zabezpečení požadovaných funkcí. Současná vozidla obecně používají řadu různých síťových protokolů k integraci nových funkcí do systému automobilu. Úkolem interní sítě je primárně co nejefektivněji zprostředkovat vzájemný přenos informací mezi dílčími jednotkami vozu, za současného dodržení časových požadavků. Proto je správný výběr sběrnice podstatnou záležitostí již při návrhu vozidel.

V moderních vnitrovozových komunikačních systémech existuje několik sítí pro datovou komunikaci. Primárně se pracuje se šesti datovými protokoly:

LIN
CAN
SAE J1939
Ethernet
MOST
FlexRay

Uvedené protokoly jsou vzájemně diferencovány svou konstrukcí, technologií výroby a rozdílným způsobem datového přenosu. Každý z nic má své výhody i nedostatky, a tak nelze říct, který z nich by byl výrazně lepším oproti ostatním. Jednotlivé vrstvy jsou reprezentovány částí fyzického propojení obsahující sběrnicové rozhraní, vodiče a jejich zakončení. Dále přidruženou vrstvou datového spoje a programovatelnou vrstvou mikrokontroléru.

- Protokol **LIN** se používá pro nízkorychlostní aplikace, které obvykle nevyžadují přísné časování.
- **CAN** je v současnosti tím nejrozšířenějším protokolem. Propojuje kritické funkce automobilu, příkladem může být hnací ústrojí vozu, ovládání motoru a automatické převodovky. Dále bezpečnostní systémy vozu jako je ABS a ESP. Rovněž je standardním rozhraním pro získávání dat prostřednictvím OBD konektoru z vozidel.
- **Ethernet** je v sériových automobilech relativně nově používaný. Má však velký potenciál pro růst kvůli vysokorychlostnímu přenosu dat.
- Síť **MOST** je oproti ostatním primárně určena jako nosič dat infotainmentu a jinde ve voze se nepoužívá.
- **FlexRay** má vysokou přenosovou rychlost a odolnost vůči chybám.

S postupnou integraci dat z velkého počtu moderních senzorů pro poskytování asistenčních aplikací pro řidiče nutně souvisí i požadavek na rychlejší, kvalitnější a spolehlivější síťové komunikační technologie.

1.1 Automobilové domény

Automobilový systém je rozdělen do čtyř hlavních domén z hlediska funkce, kterou vykonává: [1]

1. **Hnací ústrojí** – Zahrnuje funkce motoru a převodovky. Tato doména je kritická a vyžaduje odezvu od systému v reálném čase.

2. **Podvozek** – Doména zahrnuje brzdový systém, odpružení a řízení, jenž rovněž zajišťují funkce kritické z hlediska bezpečnosti vozidla.

3. **Karoserie** – Propojuje uživatelské funkce vozu jako např. signály z přístrojové desky, stěračů, světel a ovládání oken. Oproti předešlým, pro tyto funkce není obecně vyžadováno, aby byla jejich odezva plněna v reálném čase a nejsou kritické z hlediska bezpečnosti.

4. **Infotainment** – Funkce spravující různé komunikační, informační, zábavní služby a moderní asistenty. Příkladem je navigace, přehrávač hudby a další bezdrátová rozhraní.

Zmíněné domény se odlišují z hlediska funkcí, které poskytují, dále kvality a výkonu sítě, kterou vyžadují. Na základě těchto rozdílů je každou doménou používán jiný, nejvhodnější komunikační protokol. Mezi typické motivace pro zavedení komunikace patří:

1. **Snížení nákladů na kabely** – Systém kabelových svazků je třetím nejdražším a nejtěžším systémem hned po motoru a podvozku [2]. Za účelem snížení výrobních nákladů a budoucí spotřeby paliva je vhodné používat kódované komunikační linky.

2. **Úspora místa** – Vzhledem k tomu, že moderní vozidla jsou stále více obohacena o nové elektronické a elektrotechnické zařízení, vedení svazků je stále náročnější. Dále jsou kabely náchylné na elektromagnetické rušení a jsou jedním z nechtěných zdrojů tepla.

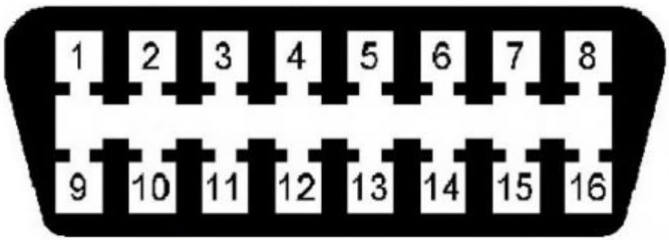
3. **Požadavky na šířku pásma** – V současné době mají některá vozidla až 70 ECU, s přibližně 2500 signály [3] pro interní přenos. Kromě toho neustále rostou požadavky na šířku pásma ze strany ADAS systémů a dalších nových aplikací. Takto rozsáhlou výměnu dat je možné usnadnit pouze pomocí kódované komunikace.

4. **Spolehlivost** – K zvýšení spolehlivosti komunikace bezpochyby přispívá digitální přenos signálů. Výhody má v robustnosti signálu oproti tradičním analogovým vodičům. Kromě toho digitální signály také umožňují šifrování pro další zvýšení zabezpečení datového toku.

1.2 Standard OBD

Standard OBD je soubor specifikací pro monitorování a přenášení interních zpráv skrze konektor automobilu do připojeného zařízení. Jedná se o vestavěný autodiagnostický systém vozidla.

Existují dva základní standardy pro OBD. První je označován jako OBD1 – ten byl v minulosti využíván výrobci vozů bez mezinárodních standardů. V těchto dobách výrobci vozů také nestandardizovali DTC. Tento diagnostický kód je používán k identifikaci poruch v různých systémech automobilu. Pokud palubní diagnostický systém vozidla detekuje poruchu, do počítače vozidla (do paměti DTC) je uložen kód dané poruchy. Při vzniku závažné poruchy se tato informace dostane také k řidiči prostřednictvím viditelné indikace MIL.



Obrázek 1.1: Konektor OBD2

Postupem času byla vytvořena norma OBD2, jejím cílem bylo diagnostikovat vozidla všech výrobců stejným konektorem a standardem. Specifikace OBD2 poskytuje standardizované hardwarové rozhraní – 16pinový konektor viz *obrázek 1.1: Konektor OBD2*. Tento konektor je dále rozdělen na typy A a B. Vizuálně se rozlišují pouze tvarem podélné středové části oddělující dvě horizontální řady pinů. Typ A se obvykle vyskytuje v automobilech, zatímco typ B je běžný u středně těžkých a těžkých nákladních vozidel. Podstatný rozdíl je na 16. pinu, kde OBD2 – typ A má +12 V, zatímco typ B má +24 V.

V Evropě jsou automobilky zavázány OBD2 normu dodržovat. Konkrétně od roku 2000 pro benzínové motory a od roku 2003 pro dieselové motory. Musí se jí řídit a do svých vozů povinně implementovat OBD2 konektory. Které zároveň musí být umístěny v zóně u řidiče poblíž středové konzole [1].

Norma OBD2 má tři fyzicky odlišné komunikační rozhraní.

ISO 9141	Evropská a asijská norma
ISO 15765	CAN BUS
SAE J1850 PWM	Pulse Wide Modulation

Význam pinů pro konektor OBD2, odpovídající normě SAE J1962 [4], je znázorněn v následující tabulce Význam jednotlivých pinů OBD2.

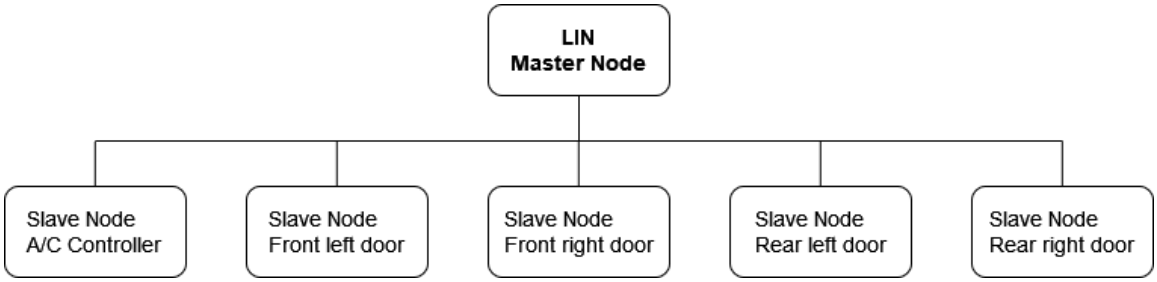
Tabulka 1.: Význam pinů OBD2

PIN	***FUNKCE PINU***
1	*nespecifikováno*
2	*SAE J1850 PWM Bus (+); nebo VPW Bus*
3	*nespecifikováno*
4	*kostra vozidla*
5	*komunikační kostra*
6	*CAN-Bus High (J-2284)*
7	*komunikační linka K-line*
8	*nespecifikováno*
9	*nespecifikováno*
10	*J1850 PWM Bus (-)*
11	*nespecifikováno*
12	*nespecifikováno*
13	*nespecifikováno*
14	*CAN-Bus Low (J-2284)*
15	*inicializační linka L-line; nebo 2. K-line (ISO 9141-2)*
16	*palubní napětí (+12 V nebo +24 V)*

2. LIN

LIN je levná, jednolinková, nízkorychlostní a snadno implementovatelná komunikační sběrnice typu master – slave. Vyvinutá byla na konci devadesátých let konsorciem společností pod společným názvem LIN. Vznikla z důvodů potřeby implementovat do vozů sběrnici, která je levnější alternativou k CAN sběrnicím. A to pro méně důležité prvky komunikační sítě ve vozidle, které jsou časově méně kritické. Používaná je pro řízení mechatronických zařízení a prvků komfortní výbavy, kde zároveň není potřeba pracovat s velkým objemem dat. Takový přenos pak probíhá rychlostí do 20kbit/s [5].

Každé zařízení, které je připojené do komunikace je nazýváno uzlem sběrnice. Síť LIN sestává vždy z jednoho hlavního zařízení iniciujícího komunikaci. Má pouze jeden hlavní (master) uzel a až 16 jemu podřízených (slave) uzlů. Jedno zařízení vykonává funkci kontroly sběrnice a ostatních zařízení. Různé uzly získávají přístup k síti na základě rozvrhové tabulky, která je uložena v hlavním uzlu. Slave uzly se synchronizují s masterem při každém přenosu a hlavičky zprávy zůstávají synchronizovány až po zbytek datového rámce.



Obrázek 2.1: Diagram použití LIN

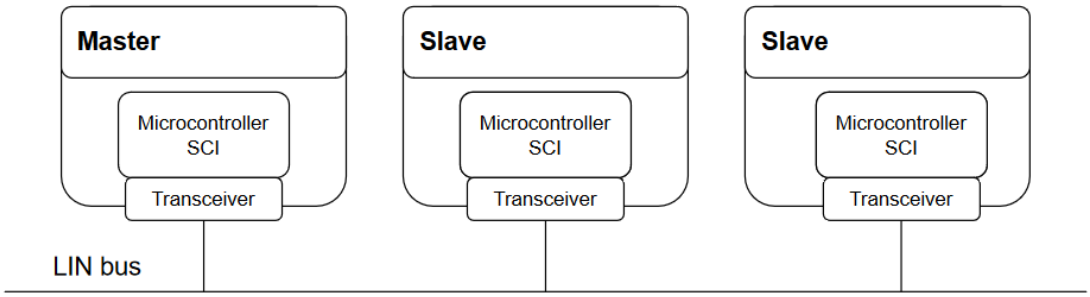
Veškerá komunikace na sběrnici je řízena master uzlem (viz *obr. 2.1: Diagram použití LIN)*. Zároveň je prováděna kontrola správnosti doručení zprávy adresátovi. Master odešle požadavek určenému uzlu jako záhlaví (začátek rámce) a respondér odpoví jako rámec odpovědí. Existuje také případ, kdy master odešle respondentovi záhlaví a rámec odezvy a ten pouze naslouchá bez odpovědi.

Tato předvídatelná povaha umožňuje inteligentní plánování zpráv [6]. A situace zaručují předvídatelný a definovaný provoz sběrnice, což z větší části neumožňuje kolize, protože komunikaci vždy zahajuje tentýž uzel. Nicméně tato vlastnost je i velkou nevýhodou master – slave sběrnice. Neboť v případě, že selže master uzel, tak s ním selže i celá komunikace.

Díky řídicímu uzlu není potřeba zahrnout do vyhodnocování komunikace arbitraci, tak jako tomu je u CAN sítě. Délka sběrnice LIN by nikdy neměla přesáhnout 40 metrů.

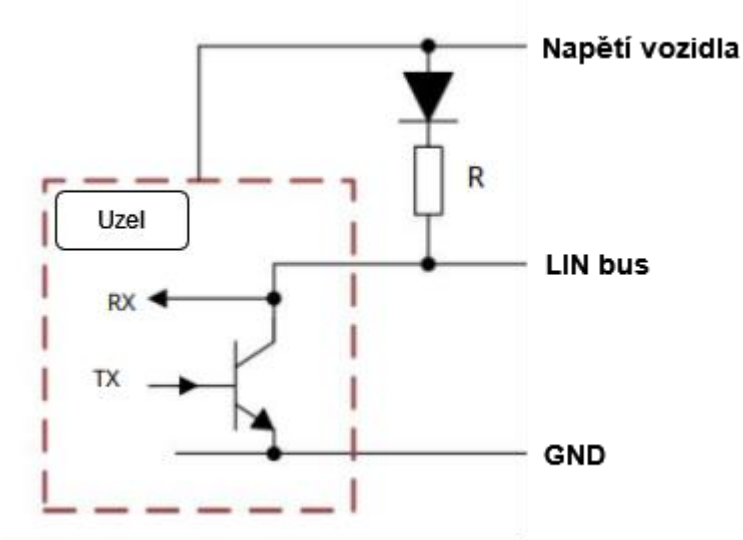
2.1 Architektura sběrnice LIN

Master je umístěn na jednom konci sběrnice a slouží k řízení celé komunikace. Slave jsou umístěny na druhém konci sběrnice a komunikují s masterem na základě jeho požadavků.



Obrázek 2.2.: Fyzická struktura LIN

Mikrokontrolér se používá k řízení komunikace s transceiverem přes rozhraní SCI. Toto rozhraní nahradilo UART ve většině aplikací LIN. Viz *Obrázek 2.2.: Fyzická struktura LIN.*



Obrázek 2.3: Schéma Pull – up rezistoru

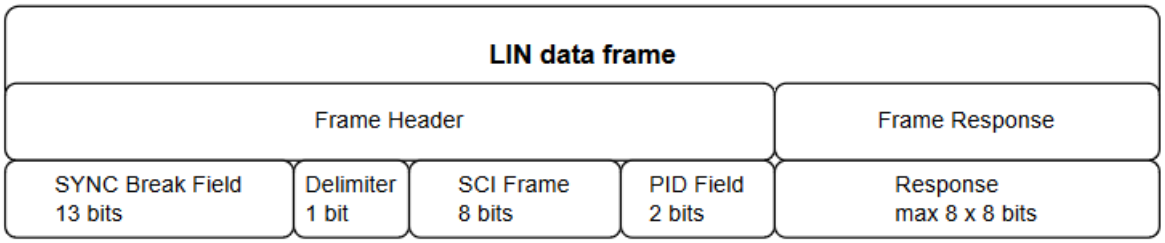
Přenos sběrnice LIN vyžaduje pouze jeden vodič a používá se pomalejší komunikační rychlost, aby se předešlo problémům s vyzařovanými emisemi. Všechny uzly jsou pasivně připojeny ke sběrnici. Pull-up rezistor (viz *obrázek 2.3: Schéma Pull – up rezistoru*) se používá k zajištění toho, aby sběrnice byla na úrovni napájecího napětí, když jsou uzly ve vypnutém stavu.

2.2 Datový rámec LIN

Každá zpráva LIN je definována specifickou strukturou: první částí je úvod zprávy a druhá část jsou již samotná data. Úvodní token je vždy přenášen a rozdělen na:

- přerušení synchronizace
- pole synchronizace
- chráněný identifikátor (PID)

Kde PID je definující část. Hlavička se skládá celkem z alespoň 13 bitů pro přerušení SYNC, 1 oddělovacího bitu, 10 bitů pole SYNC (1 startovací bit, 8 bitů pro synchronizaci a 1 stop bit) a 10 bitů identifikátoru (1 startovací bit, 6 bitů pro identifikátor, 2 bity pro paritu a 1 stop bit).



Obrázek 2.4: Rámec LIN zprávy

Datová část odpovědi je odeslána úlohou respondéru. Odpověď je rozdělena na datové bajty, doprovázená kontrolním součtem. Kontrolní součet ověřuje, zde během přenosu nedošlo k žádným chybám. Každý uzel může přijmout odezvu rámce, ale který uzel ji skutečně potřebuje, závisí na LIN Description File (LDF). Odezva je složena z celkem 10 bitů, pro každý bajt dat (1 start bit, 8 bitů pro data, 1 stop bit), až 8 datových bajtů a 10 bitů pro kontrolní součet (1 start bit, 8 bitů pro řešení kontrolního součtu, 1 stop bit). Viz *obrázek 2.4: Rámec LIN zprávy*.

2.3 Fyzická vrstva LIN

Fyzická vrstva LIN je založena na normě ISO 9141. Je přizpůsobena napětí baterie vozidla, v zapojení přes odpor a diodu (pouze master uzel) a je připojena k transceiveru každého jednoho uzlu.

Transceiver LIN převádí bitovou logiku z mikrokontroléru na vyšší napěťové úrovně jako přenos po sběrnici a naopak. TXD (vysílání) a RXD (příjem) transceiveru LIN umožňují komunikaci sběrnice prostřednictvím přenosu napětí [6].

Typické úrovně napětí pro TXD a RXD jsou pro většinu úrovní mikrokontrolerů rovny 3,3 V, nebo 5 V. Sběrnice a transceivery LIN obvykle pracují při napětí v rozsahu od 9 V do 18 V, ale některé mohou dosáhnout až 30 V (v závislosti na účelu aplikace).

2.4 Speciální režimy transceiveru

U většiny moderních transceiverů LIN existují režimy, které pomáhají se specifickými potřebami aplikací. Většina uvedených funkcí je určena pro systémy, které se zaměřují na nízkou spotřebu energie.

1. **Režim spánku** – Tento klidový režim se používá pro úsporu energie, když není LIN transceiver potřeba v žádné části systému. Zařízení je v méně funkčním stavu, ale je stále schopno monitorovat sběrnici LIN pro případné signály probuzení.

2. **Pohotovostní režim** – je také režim s nízkou spotřebou. Hlavní rozdíl mezi pohotovostním a klidovým režimem je, že se zařízení po požadavku na probuzení do pohotovostního režimu dostane rychleji a je dříve připraveno vysílat potřebná data.

3. CAN

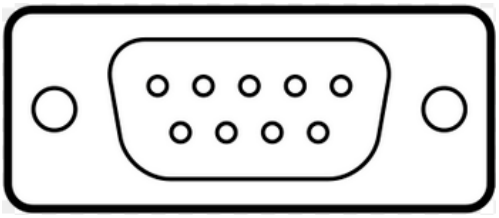
Sběrnice CAN je komunikačním standardem vyvinutým společností Robert Bosch, který byl uveden na trh v roce 1986. Definuje funkcionalitu první a druhé vrstvy síťového modelu OSI.

3.1 Charakteristika CAN

Protokol CAN se nepoužívá pouze pro automobilové aplikace, nýbrž je zastoupen napříč různými odvětvími průmyslu. Je definován normou ISO 11898, která popisuje fyzickou a linkovou vrstvu protokolu. Pod fyzickou vrstvu spadají elektrické, řídící a mechanické prvky. Zatímco linková vrstva pouze definuje pravidla pro přenos dat [7].

Zprávy jsou zapouzdřeny do rámců s maximální velikosti datového pole 64 bitů. Vysílané zprávy v porovnání s LIN architekturou neobsahují informaci o cílovém místě doručení. Jsou přijímány všemi uzly. Každá zpráva je uvedena počátečním identifikátorem, který jí přiřazuje míru priority, což u LIN sběrnic není potřeba, tam se vysílané signály nemohou „překrýt“ či být vyslány v jeden okamžik.

V současnosti je CAN velmi používanou komunikační sběrnicí nejen pro automobilové aplikace. Ačkoliv byly později vyvinuty různé sítě v reakci na nedostatky CAN, přesto si udržuje vysokou popularitu. CAN se obvykle využívá k přenosu řídících informací mezi ECU ve vozidle.



Obrázek 3.1: D-Sub 9pin konektor

Pro rozhraní CAN sběrnice je obecně používán devíti pinový D-Sub konektor (viz *obrázek 3.1: D-Sub 9pin konektor*), který umožňuje dosáhnout maximální rychlosti sběrnice až 1 Mbit/s na délkách do 40metrů [8].

Při vzdálenostech větších, něž 40 m, dochází k zpomalení a zašumění přenášeného signálu. Při délce sběrnice 130 m je maximální přenosová rychlost poloviční, tedy 500kbit/s. Na délce 500 m klesá rychlost až na 125kbit/s a při 3300 m jen 20kbit/s [8].

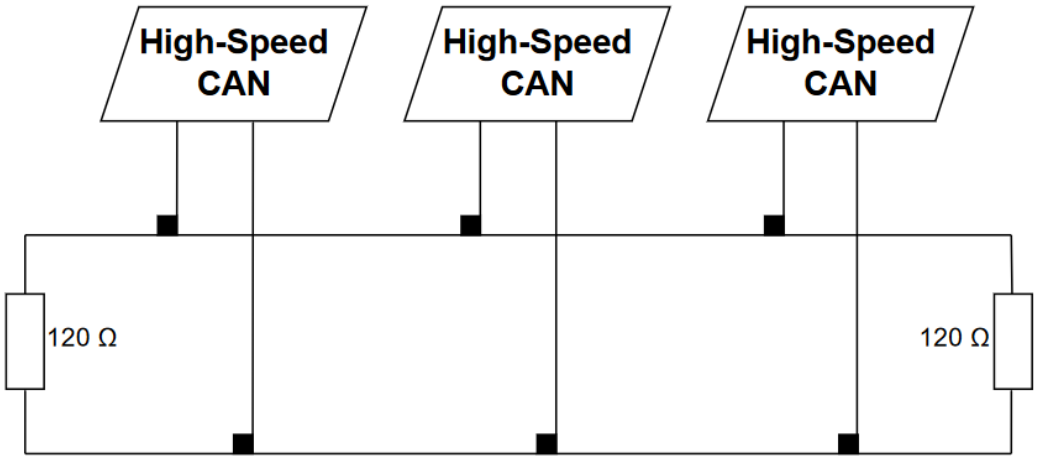
3.2 Fyzická vrstva CAN

Fyzická vrstva CAN je tvořena dvěma vodiči, tzv. kroucenou dvoulinkou.

Obrázek 3.2: Kroucená dvoulinka [21]

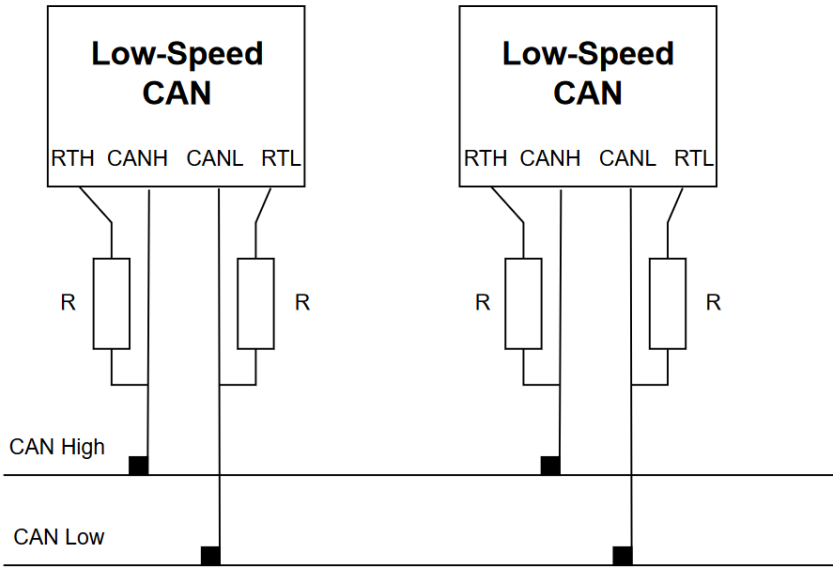
Vodiče (viz *obrázek 3.2: Kroucená dvoulinka*) jsou označeny jako CAN High (spíná k 5 V, nebo 3.3 V napěťově je tedy „nahoře“) a CAN Low (spíná k zemi). Tyto vodiče jsou diferenční – nesou stejný signál, ale opačné polarity. Jsou tvořeny měděnými dráty, obalenými PVC obalem a jsou vzájemně propleteny.

Další dělení CAN sítě je podle možností přenosových rychlostí. Konkrétně na High-Speed CAN, který umožňuje přenos datových rychlostí až 1 Mb/s a Low-Speed CAN, který má nižší přenosovou rychlost a umožňuje přenos dat pouze rychlostí do 125 kb/s.



Obrázek 3.3: High – speed architektura vedení

Soustava High-Speed je zakončená na obou koncích vodičů. Tam jsou zapojeny rezistory o hodnotě 120Ω, viz *obrázek 3.3: High – speed architektura vedení*. A to z důvodů zamezení rušení zpráv mezi sebou.



Obrázek 3.4: Low – speed architektura vedení

Oproti tomu soustava Low-Speed má na každém zakončení zařízení své vlastní rezistory [9]. Viz *Obrázek 3.4: Low – speed architektura vedení*. Hodnota těchto odporů je závislá na počtu uzlů sběrnice tak, aby dohromady dávaly při paralelním zapojení hodnotu 100Ω [23].

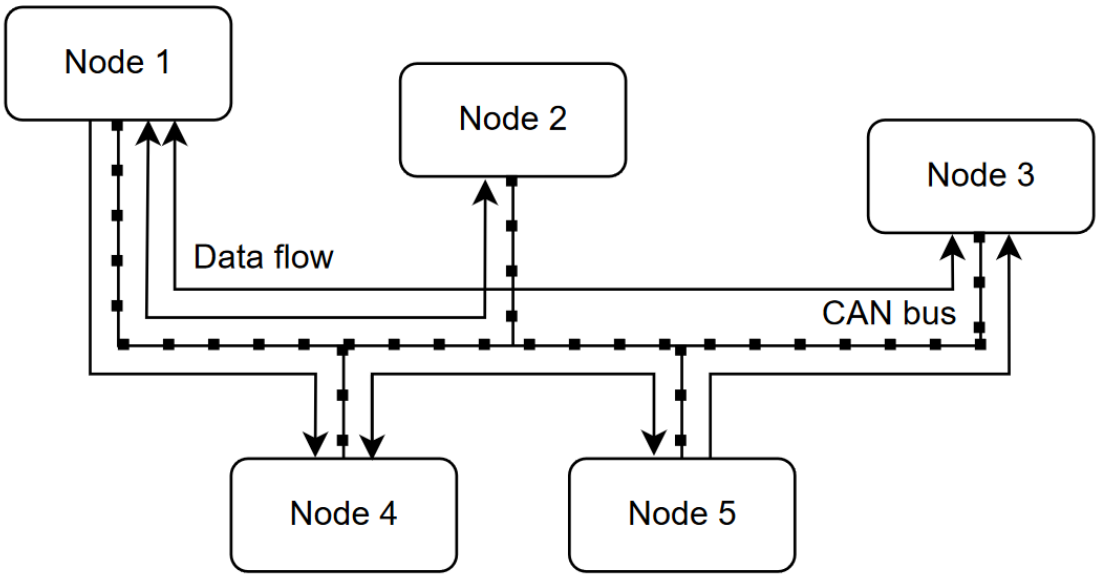
RTH se vztahuje k přijímači v transceiveru a označuje prahovou úroveň, kterou signál musí překročit, aby byl úspěšně detekován a rozpoznán jako platný signál při přenosu dat. RTL se také odkazuje na prahovou úroveň přijímače. Překročení RTL zajišťuje spolehlivé přijímání a dekódování signálů při zachování kvality zprávy [23].

Počet uzlů je teoreticky nekonečný, maximální hodnota používána v praxi je okolo 30 uzlů pro jednu dvoulinku.

Uvedené schémata propojují hardware CAN sběrnice dle normy ISO 11898 [10]. CAN může mít různé topologie v závislosti na různých fyzických vrstvách. Nejrozšířenější sítí je vysokorychlostní CAN (vysokorychlostní je taková, jež je schopná přenášet data rychleji něž 125Kbit/s). Patří do skupiny sériových sběrnicových systémů, což znamená, že všechny uzly v rámci jedné sítě jsou připojeny paralelně ke každému jinému uzlu v síti.

Uzel, který vysílá zprávu je nazýván vysílačem, ostatní uzly se v daný okamžik chovají jako přijímače. Díky podpoře filtrování zpráv, každý uzel reaguje jen na zprávu, jenž mu je určena [10]. Na rozdíl od sítí jako je USB či Ethernet, CAN neposílá velké datové bloky stylem point-to-point. Nýbrž jsou odesílána malá data do celé sítě. Toto zajišťuje vysokou míru konzistence dat v každém uzlu systému.

Existují tři způsoby, jak můžou být posílány data na sběrnici. Buď uzel vysílá svá proměnná data opakovaně, a to s danou frekvencí, nebo si uzel vyžádá odpověď na aktuálně potřebnou hodnotu z jiného snímače (uzlu) a přečte si informace pouze jako odpověď na dotaz. A poslední možností je nastavení uzlu tak, aby vysílal data pokaždé, když dojde ke změně hodnot.



Obrázek 3.5: Architektura sběrnice CAN

Uvedený *obrázek 3.5: Architektura sběrnice CAN* ukazuje obecnou strukturu zapojení sběrnice CAN s pěti uzly. Zároveň je z obrázku patrné, že oproti LIN sběrnicím (viz *obr. 2.1: Diagram použití LIN*) u této neexistuje pouze jeden hlavní, master uzel, ale všechny síťové uzly mohou fungovat jako master, nebo slave vzhledem k ostatním, a to pouze v závislosti na své funkci.

Během jednoho rámce zprávy funguje pouze jeden uzel jako master, zatímco všechny ostatní jsou v podřízené fázi a poslouchají hlavní uzel. Tato schopnost je nazývána jako Multimaster CAN. Ta je povolena prostřednictvím metodologie CSMA/CD+CR (Carrier Sense Multiple Access with Collision Detection and Collision Resolution). Je to způsob řízení přístupu k médiu (sběrnici) v protokolu CAN.

Pomocí této metody se každému uzlu umožní zkontrolovat, zda je sběrnice volná a následně odeslat svoji zprávu. Pokud více uzlů začne vysílat současně, dojde ke kolizi, a uzly přestanou vysílat. Poté následuje krátká vyčkávací doba a každý uzel čeká, než se opět pokusí vyslat svou zprávu [10].

Obecná struktura síťové architektury je specifikována v obecném OSI modelu. Pro CAN jsou definovány pouze první dvě vrstvy. Konkrétně vrstva, která propojuje fyzickou architekturu a která popisuje vrstvu datového spoje pro CAN.

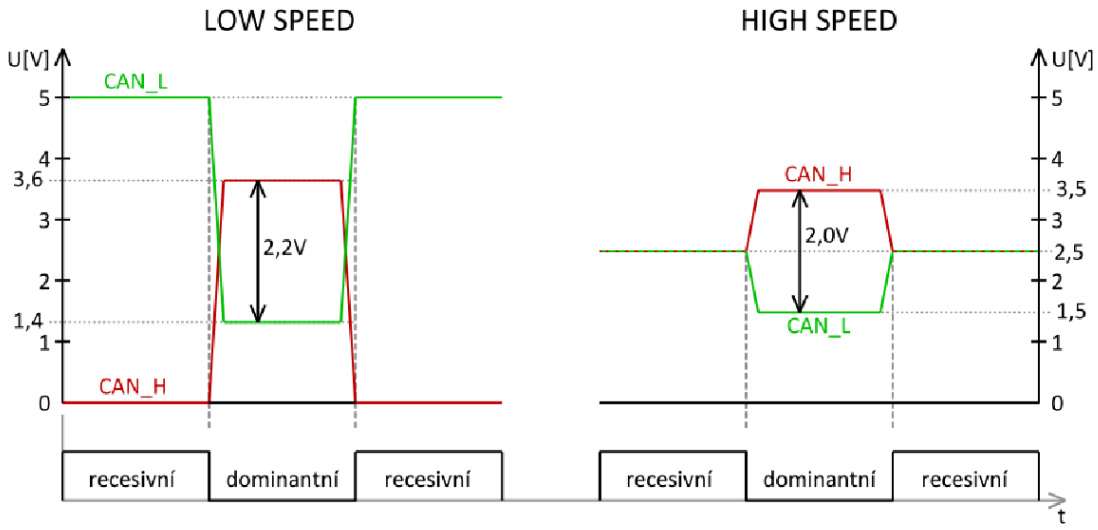
Digitální signál využívá pro síťovou komunikaci převod na binární soustavu. Koncové (přijímací) zařízení dokáže přečíst pouze hodnoty logické 1 a logické 0. Jsou definovány dvě vzájemně komplementární hodnoty. Logická 1, která se považuje za recesivní a logická 0, která je reprezentací aktivního stavu [10].

Zjistit hodnotu bitu je možné přímo z odečtení hodnot napětí. Pokud je rozdíl napětí na sběrnici:

a) menší než 0,5V jedná se o recesivní bit
b) větší než 0,9V jedná se o dominantní bit

Platí vzorec: $V_{rozdil} = V_{CAN\_High} - V_{CAN\_Low}$

Norma ISO 11898 udává schéma datové sběrnice pro recesivní a dominantní stav. Pro vodič CAN_High je udáván dominantní stav v rozmezí <3.5; 5> V.
A pro CAN_Low v rozsahu <0; 1.5> V.



Obrázek 3.6: Schéma recesivních a dominantních stavů sběrnice CAN [22]

3.3 Vlastnosti sběrnice

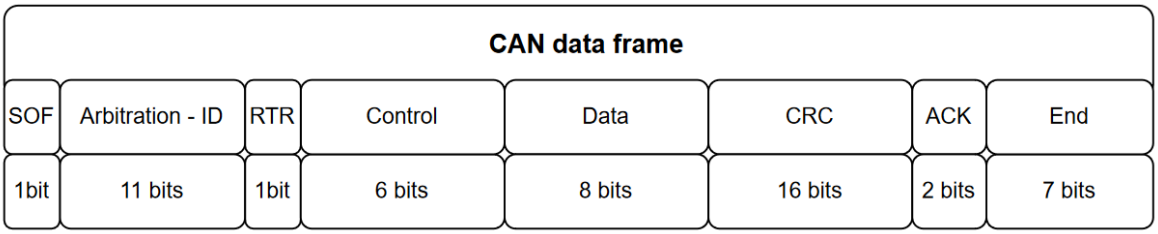
Sběrnice dosahuje velmi efektivní komunikace mezi řídícími jednotkami v reálném čase, a to s kvalitním zabezpečením [8]. Mezi nejvýznamnější vlastnosti a funkce sběrnice CAN patří:

- **Priorita zpráv** – Přiřazení priority každé zprávě; informace s důležitějšími daty (takové, které mají vyšší prioritu = nižší identifikační číslo) budou odesílány dříve, zároveň je zaručeno, že nedojde k poškození zprávy. Uzel, který tedy nedostal přístup na sběrnici musí vyčkat, až bude sběrnice opět ve volném stavu [11].

- **Latence zpráv** – Zaručená maximální latence každé zprávy.
- **Vícesměrná komunikace** – Komunikace s bitově orientovanou synchronizací.
- **Multicast přenos** – Jednu odeslanou informaci dokáže číst vícero příjemců zároveň.
- **Multimaster přenos** – Každý uzel smí přijímat cizí a vysílat vlastní datové informace na sběrnici.
- **Detekce chyb** – A k tomu příslušná signalizace chybové hlášky; chybové rámce mohou vysílat informace kdykoliv, aniž by musely čekat na ukončení přenosu jiné informace.
- **Automatické znovu odeslání** – Pro zprávy poničené chybou, za předpokladu že se jiný uzel nepokusí vyslat priorizovanou informaci.
- **Detekce trvalých poruch** – Umožňuje automatické vypnutí poškozeného uzlu.
- Flexibilita komunikace
- **Konzistence dat v celém systému** – Data jsou pevná a v okamžiku vysílání neměnná.

3.4 Datový rámec CAN

Každá datová zpráva musí být jednoznačně definovaná. Nejčastěji používaným rámcem v síti CAN ve vozidle je standardní rámec.



Obrázek 3.7: Rámec CAN zprávy

Jednotlivé částí bitové zprávy z uvedeného se dělí do následujících subkategorií:

- **SOF –** Start of Frame. Označuje začátek zprávy a nachází se zde dominantní bit.

- **ID –** Standard Identifier. Toto pole sestává z 11 bitů. Udává míru priority a cílovou adresu své zprávy. Čím více se dominantní bit blíží k SOF, tím má zpráva vyšší prioritu. Posledním bitem v rámci arbitrace je bit RTR.

- **RTR** – Remote transmission Request. Je nosičem informace, zda se jedná o žádost o zprávu (podnět k vyslání žádaných hodnot např. senzoru na sběrnici). Žádost o zprávu je podmíněna recesivním bitem. Vysílání informace poté opačným.

- **Control** – Pro rámec Control je vyhrazeno 6 bitů. Určuje množství bytů (0-8) ve zprávě. Musí odpovídat délce očekáváné zprávy [12].

- **Data** – Vyhrazeno pro přenos informace. V klasickém CAN přenosu je maximální délka stanovena na 8 Bytů. U rozšířené CAN_FD je možno přenášet data o velikosti až 64 B.

- **CRC** – Cyclic Redundancy Check. 15 bitů vyhrazených pro kontrolní součet. Více o CRC je uvedeno v následující kapitole. Poslední, šestnáctý bit je CRC Delimiter, který signalizuje ukončení pole CRC, ten musí vždy nabývat recesivní bit (1).

- **ACK** – Acknowledge. Dvoubitové potvrzující pole správně odeslané zprávy. První bit (ACK Slot) potvrdí přijmutí zprávy, druhý (ACK Delimiter) znamená ukončení ACK, jeho hodnota je nastavená neměnně na recesivní 1.

- **EOF** –End Of Frame. Obsahuje 7 recesivních bitů. Pole tímto signálem hlásí ukončení zprávy.

- **IFS** – Interframe Space. Zpravidla 3 bity. Označuje dobu mezi dvěma za sebou následujícími rámci v síti CAN. Hodnota se může lišit v závislosti na požadované rychlosti přenosu dat a počtu stanic v síti [12].

3.5 Zabezpečení a detekce chyb

Bezpečnost je pro komunikaci více než důležitá, protokol CAN má několik způsobu zabezpečení sítě a díky nim má silný mechanismus obrany. Na sběrnicích může dojít k několika typům chyb. V této kapitole jsou rozlišovány chyby při přijímání (1.-3. bod) a při odesíláni (4.-8. bod) zprávy.

Mezi ochranu komunikace patří následující způsoby [13]:

1. **Cyclic Redundancy Check (CRC) –** Vysílací zařízení provádí speciální kontrolu pomocí polynomu – bit po bitu od začátku rámce zprávy až po její konec, jedná se o posledních 16 bitů na vysílané zprávě. Následně je vyslána i část vyhrazená pro CRC kontrolu. Přijímací uzel počítá CRC sekvenci stejným způsobem a porovná ji s CRC sekvencí poslanou vysílačem. Je-li detekována chyba CRC libovolným uzlem na sběrnici, je vygenerována chyba typu error frame. Vysílání zprávy vysílačem je opakováno a hodnota čítače přijatých chyb je inkrementována. Pokud čítač dosáhne mezní hodnoty, je vygenerováno přerušení [14].

2. **Bit Stuffing –** Vkládání bitů, nastane-li situace, že je na sběrnici vysílán signál o pěti stejných bitových hodnotách totožné polarity (11111 nebo 00000) za sebou v rámci jedné zprávy, je do informace vložen jeden bit opačné úrovně. Toto opatření zamezuje špatné časové synchronizace bitů a přijímačů uzlů a přispívá k detekci chyb.

3. **Invalid Message Recieved Error** – V okamžiku, když se objeví libovolný typ chyby v průběhu přijímání zprávy, zaznamená se to do registru modulu CAN.

4. **Monitoring –** Vysílající zařízení porovná aktuální hodnotu s úrovní bitu na sběrnici. Jsou-li obě hodnoty stejné, vysílač pokračuje ve vysílání. V případě, že se porovnávané hodnoty neshodují, zařízení přestane vysílat, neboť nesmí zamezit přístupu ke sběrnici jinému uzlu, který vysílá data s vyšší prioritou. Je to primární kontrola informací. Zařízení si zkontroluje informaci, kterou samo vyšle.

5. **Acknowledge (ACK) –** Potvrzení přijetí zprávy, to musí vyslat každý uzel sběrnice, a to i ten, který danou zprávu nepotřebuje. Potvrzení je indikováno změnou prvního bitu v poli ACK Slot. Druhý bit je vyslán jako recesivní a má oddělovací funkci. Kontrola je tedy prováděna v poli ACK způsobem, že vysílací uzel vyšle recesivní bit a poté sleduje, zda některý z uzlů nezměnil hodnotu bitu na dominantní. Změní-li některý uzel svou hodnotu indikuje to výskyt chyby při vysílání a zpráva se musí odeslat znovu. Error frame se v tomto případě negeneruje.

6. **Frame error** – nastane v případě, že přijímač detekuje dominantní bit na pozici, kde očekává recesivní (EOF, ACK, CRC, mezera). Ze zjištění vyplývá, že datový rámec obsahuje chybu. Zpráva je znovu odeslána a inkrementuje se chybový rámec pro Error frame.

7. **Bit error –** Vznikne, když vysílač vyšle dominantní bit a zpět obdrží recesivní, v opačném případě, kdy vysílač vyšle recesivní bit a zpět obdrží dominantní se nejedná o chybu. Indikuje to pouze že je na síť vysílána informace s vyšší prioritou.

8. **Message Frame Check –** Kontrola zprávy, každý uzel vysílá signály dle příslušného rámce, v případě, že vyšle hodnotu bitu, která je neslučitelná s daným formátem zprávy, je vygenerována chyba rámce – formátu zprávy.

Každý uzel datové komunikace má dva interní čítače, ve kterých se sčítají chyby jak při příjmu zprávy, tak při její vysílání. Podle obsahu čítačů poté uzel přechází do následujících čtyř stavů [13]:

1. **Úplné odpojení** (Bus-off) – nastane v případě, že je uzlem generováno nepatřičné množství chyb. Dojde k vypnutí budičů uzlů a ty následně nemohou komunikovat.

2. **Aktivní** (Error Active) – uzly se podílejí na komunikaci bez problémů. Vyskytne-li se chyba ve zprávě na síti, vyšle se na síť šest po sobě jdoucích dominantních nul. Tímto způsobem nejen že dojde k poškození přenášené zprávy, ale také se poruší pravidlo Bit Stuffing.

3. **Pasivní** – obdobně jako u Aktivního stavu, nicméně při detekci chyby vysílají jen pasivní zprávu – která je zde tvořena šesti po sobě jdoucími recesivními bity. Pasivní ochrana nemá vliv na přímou destrukci zprávy.

3.6 Datová zpráva

Ve sběrnici existuje pět základních typů zpráv v závislosti na jejich obsahu a funkci:

1. **Data Frame** – Nezaměnitelný, je používán k přenosu všech zpráv mezi uzly.

2. **Remote Frame** – Používán uzlem k vyžádání přenosu datové zprávy z jiného uzlu. Odpověď může být čtena libovolným množstvím uzlů.

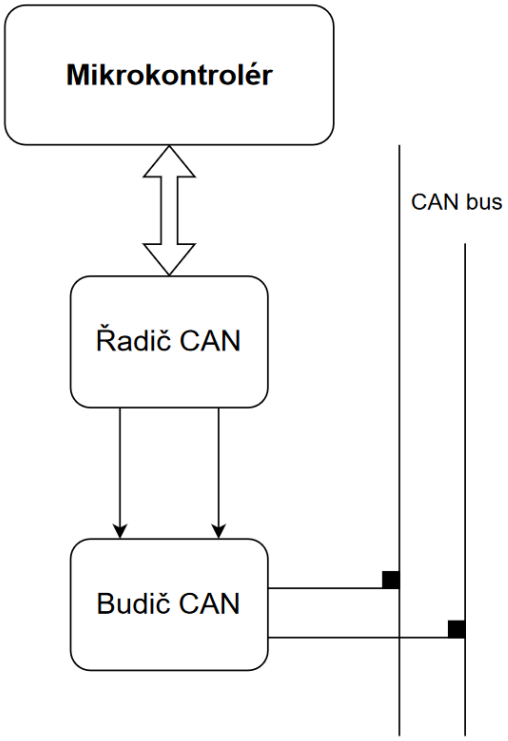
3. **Error Frame** – Je vyslán vždy, vznikne-li chyba. Po detekci chyby řídí ukončení datové zprávy.

4. **Overload Frame** – Vyžádá zpoždění před vysláním následující datové zprávy. Způsob provedení je podobný chybové zprávě, ale na rozdíl od ní původní informaci nezničí ale pouze vynutí zpoždění před jejím odesláním. Stejně jako chybová zpráva má 6 po sobě jdoucích dominantních bitů.

5. **Interframe Space** – Jedná se o tříbitový prostor mezi jednotlivými rámci zprávy, které odděluje. V každé CAN síti by měla být ihned po skončení těchto tří bitů vyslána další zpráva [8].

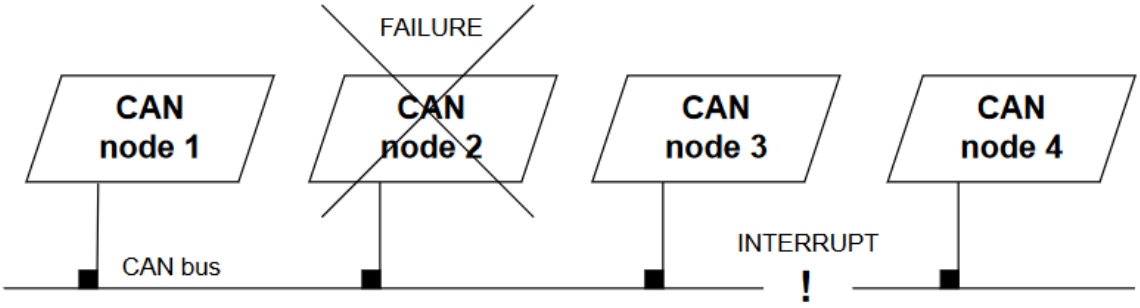
3.7 Topologie CAN

V předcházejících kapitolách bylo zmíněno, že každá jednotka (zařízení), která komunikuje po sběrnici je nazývána uzlem.



Obrázek 3.8: Komponenty uzlu CAN

Výhodou uzlového zapojení je jeho snadné připojení / odpojení ze sběrnice, jak je na *obrázku 3.8: Komponenty uzlu CAN.*



Obrázek 3.9: Poškození uzlu a přerušení linky

Zároveň, dojde-li k poruše jednoho uzlu, nenaruší to činnost samotné komunikační linky (viz *obrázek 3.9: Poškození uzlu a přerušení linky*). Každý uzel CAN sběrnice sestává z následujících komponent [9]:

- Komunikační zařízení – mikrokontrolér
- Řadič
- Budič – vysílací a přijímací

Na *obrázku 3.8: Komponenty uzlu CAN* je schéma zapojení jednotlivých komponent uzlů. V případě trvalého přerušení sběrnice dojde k utvoření dvou separátních segmentů, zařízení mohou komunikovat v jednom segmentu, data ale nemohou sdílet mezi sebou navzájem, jak je uvedeno na *obrázku 3.10: Poškození uzlu a přerušení linky*.

3.8 Rozšíření CAN-FD

V této kapitole je srovnán výkon sběrnice CAN-FD v porovnání s konvenční sběrnici CAN. Jedná se o nový protokol, rozšiřující stávající CAN. Koncovka –FD znamená Flexible Datarate, tedy flexibilní datovou rychlost. Ta umožňuje zrychlit přenos zpráv až na 8 Mbit/s pro datové pole. Což je osminásobek maximální klasické rychlosti u CAN [15]. Zároveň zvyšuje velikost datového pole až na 64 Bytů.

Hlavní motivace k rozšíření na CAN-FD:

1. Uspokojení zvýšené poptávky po zvětšení šíře komunikačního pásma.
2. Vyplnění mezery mezi CAN a FlexRay sběrnicemi, které sice dosahují rychlosti až 15-20 Mbit/s, ale jsou daleko nákladnější.

Do rozhodovacího pole v CAN-FD jsou přidány tři nové bity:

- **EDL** – Extended Data Length, definuje délku dat s nejvyšším bitem na nule.
- **BRS** – Bit-Rate-Switch, určuje, zda je zpráva přenášená klasickou rychlostí, nebo zvýšenou.
- **ESI** – Error State Indicator, dominantní bit pro vyznačení chyby stavu sítě.

3.9 Arbitrace

Každý uzel je schopen vysílat, či přijímat zprávy. Nikoliv však současně s jiným uzlem zapojeným ve stejné lince. Vyslána zpráva s větší prioritou přepíše ostatní uzly s méně dominantním ID. Tento mechanismus je nazýván jako prioritní arbitrace.

Na základě zkušeností konstruktérů jsou přiřazeny každé zprávě, kterou umí uzel vyslat kritické, či nekritické hodnoty priority. Tyto hodnoty jsou buď neměnné – statické (SCS), nebo dynamické (DCS) – u kterých jejich hodnota proměnná v čase [16].

Dominantní bit se prosazuje vytvořením napětí na vodičích, zatímco recesivní se nijak neprosazuje. Nastaví-li nějaký uzel rozdíl napětí, uvidí jej ihned všechny uzly a nedojde k žádnému zpoždění. Přerušený uzel se pokusí odeslat zprávu ihned poté.

Při přenosu pole identifikátoru (ID) se provádí arbitrace. Každý uzel, který začíná vysílat ve stejnou dobu, posílá ID s dominantním bitem (binární 0). Jakmile bude jejich ID větší číslo (nižší priorita), bude posílat recesivní bit (binární 1) a jakmile uvidí 0 (dominantní), tak ustoupí. Na konci ID přenosu všechny uzly kromě jednoho ustoupí a zpráva s nejvyšší prioritou projde ke všem uzlům bez omezení.

Názorný příklad arbitrace: na síť jsou vyslány současně dva signály X a Y. Jejich ID vypadají následovně:

X:	0000 0000 1111	v decimální soustavě je to hodnota 15
Y:	0000 0001 0000	v decimální soustavě je to hodnota 16

Oba mohou vyslat prvních sedm nul současně aniž by vysílače poznaly, že nevysílají samy. Při odeslání osmého bitu ale uzel Y vysílá recesivní hodnotu, zatímco uzel X vysílá dominantní bit O. Až v tomto okamžiku vysílač Y zjistí, že nevysílá sám a z komunikace odstoupí, neboť ztratil své právo vysílat.

Na síť bude vyslána informace z prvního uzlu, jehož priorita ID X (hodnota 15) je menší, než ID Y (16).

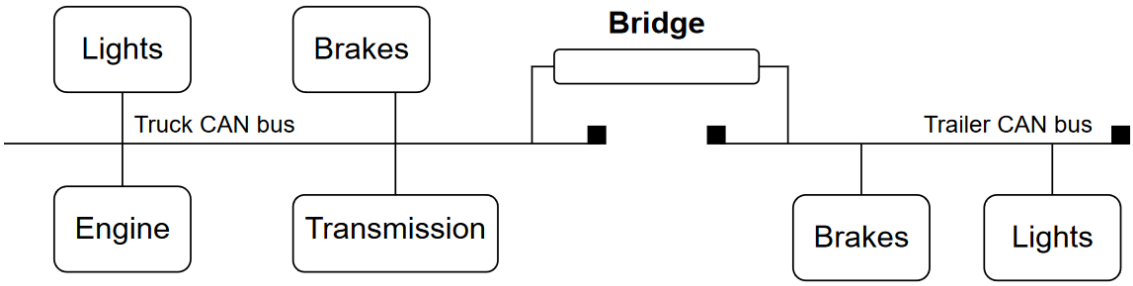
4. VYŠŠÍ PROTOKOLY

4.1 SAE J1939

Tento americký komunikační standard je používán u všech těžkých nákladních a jiných vozidel. Definuje komunikací jednotlivých mikrokontrolérů přes CAN sběrnici. Ta však definuje pouze fyzický základ (vrstva datového spojení) pro SAE J1939, nikoliv komunikaci. Tento protokol není prvním standardem pro těžká vozidla, nýbrž je nástupcem standardů SAE J1587 a SAE J1708 [17]. Norma se používá v mnoha průmyslových odvětvích, jako je například i vlaková doprava, těžební technika, vojenské transportní vozy a kombajny.

I když se standardizace této komunikace ukázala jako klíčová pro to, aby umožnila různým dodavatelům a výrobcům spolupracovat a tím také snižovat náklady, znamená to ovšem, že všechna těžká vozidla, která jsou v současnosti na silnicích v USA, od přívěsů, návěsů přes popelářské vozy a míchačky cementu až po autobusy používají stejný komunikační protokol ve svých vnitřních sítích.

Je zde používán CAN 2.0 B s 29bitovým identifikátorem. Otevřenost J1939 umožňuje každému, získat technické podrobnosti o standardních informacích proudících po sběrnici, což potenciálnímu protivníkovi umožňuje snadno získat znalosti potřebné k útoku na kritické bezpečnostní komponenty, jako například ovládání brzd, či tempomatu [17].



Obrázek 4.1: Konfigurace sítě tahač – přívěs

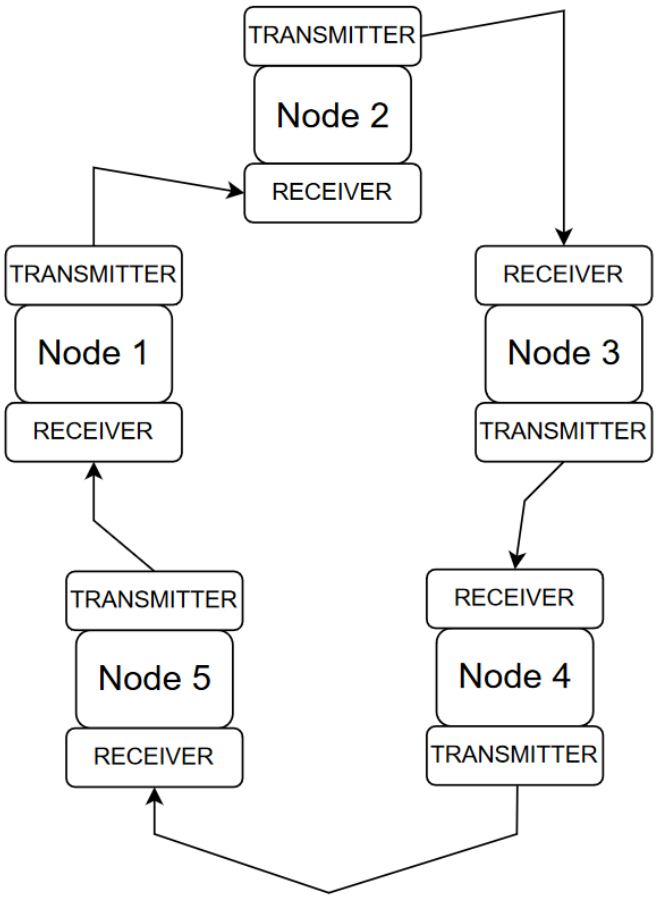
Jak je zobrazeno na schématickém *obrázku 4.1: Konfigurace sítě tahač – přívěs*. K samotné sběrnici je možné připojit se i přes piny koncovky u tažného zařízení. Automobilky se proti tomu ale brání a mají zcela oddělenou CAN sběrnici pro potřeby vyvedení a sběru dat z prostoru mimo vůz. Neposkytují tak možnost čtení a zápisu všech dat na interní síť z vnější části vozu.

Protokol je v zásadě postaven na vrstvách CAN se zavedením dalších pravidel. SAE J1939 specifikuje přenosové rychlosti 250 Kbit/s a 500 Kbit/s [18].

Kromě jedinečné identifikace každého uzlu v síti, je v datovém rámci J1939 speciální část Industry Group (IG), kde jsou označeny informace o průmyslovém odvětví, pro které je zařízení určeno, jeho funkčnosti a výrobci [18].

4.2 MOST

Síť MOST byla vyvinuta společností MOST Corporation původně z D2D. Je to vysokorychlostní, alternativní komunikační síť, která je optimalizována pro přenos multimédiálních dat a infotainmentu ve vozidle. Vyznačuje se vysokou spolehlivostí přenosu dat. V průběhu let tato síť expanduje pod stále rostoucím počtem zařízení multimediální elektroniky. Velkým mínusem je ovšem pořizovací cena sítě, což je zásadní faktor omezující její použití v průmyslu.



Obrázek 4.2: Kruhová topologie MOST

Fyzické zapojení je poté zcela odlišné od LIN i CAN sběrnice. Neexistuje něco jako hlavní, nebo vedlejší větev, na které jsou zapojeny uzly sběrnice. Jednotlivá zařízení jsou nejčastěji propojená mezi sebou do kruhu. Jak je uvedeno na *obrázku 4.2: Kruhová topologie MOST.*

Takové zapojení a její jednosměrný kruhový přenos dat je skutečně realizován i ve fyzické struktuře. Na rozdíl od jiných sítí, kde jsou vysílač a přijímač integrovány na jedné lince a přistupují k jednomu fyzickému kabelu, uzel MOST má vysílač pouze jeden. Vysílačem je zde světelná dioda s vlnovou délkou 650nm (jedná se o červené světlo) a přijímačem je křemíková fotodioda. Propojení zajišťují plastová optická vlákna.

Oproti dříve uvedeným sběrnicím zahrnuje síť MOST všech sedm vrstev modelu OSI. MOST se také odlišuje od mnoha komunikačních sítí ve vozidle díky své větší šířce pásma a univerzálnosti při přenosu různých typů dat.

Díky své univerzálnosti a rychlosti je MOST považován za perfektní komunikační síť pro ADAS [19]. Mezi hlavní důvody k tomuto tvrzení patří:

- optické kabely jsou imunní vůči elektromagnetickému rušení
- jejich komunikační chybovost je velice nízká
- Plug and Play funkce – umožňuje snadné připojování / odpojování uzlů

Podstatným problémem sítě MOST je ovšem kruhová topologie, kde dojde k přerušení celého okruhu díky jednomu vadnému uzlu, nejsou-li k dispozici náhradní kabely pro pokračování komunikace. Takové zapojení je známo jako redundantní dvojkruhová topologie. Dále je potřeba zmínit fyzické možnosti optických kabelů, které není možné používat mimo kabinu vozu.

Komunikaci umožňuje softwarový ovladač MOST Network Services.

4.3 ETHERNET

Automobilový Ethernet je určen k propojení vysokorychlostní komunikace uvnitř vozidla. Ty jsou vyžadovány podsystémy, jako jsou technologie ADAS, navigace, multimédia a další metody konektivity.

U moderních hybridních (HEV) nebo elektrických vozidel (EV) bude Ethernet výkonnou součástí komunikační architektury, která umožní propojení mezi elektronikou vozidla a internetem. Budoucí vozidla budou součástí typické aplikace IoT.

Tato síť má jedinečnou výhodu při vývoji – a to sdílení nákladů s dalšími velkými průmyslovými odvětvími, jako je IT a telekomunikace. Přestože je vysokorychlostní Ethernet na rovni Gigabitů za sekundu již používán, pro automobilové aplikace se nehodí z důvodu drahého stíněného kabelu.

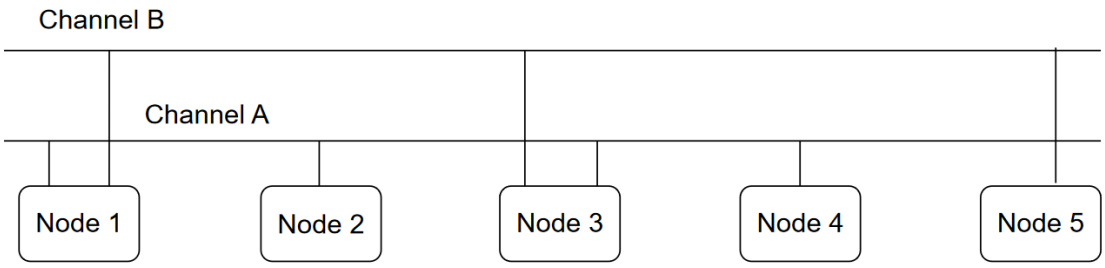
4.4 FlexRay

Protokol FlexRay byl vyvinut konsorciem FlexRay. To vzniklo při spolupráci firem BMW a Daimler. Ty spolupracovaly na vytvoření nového síťového schématu, které by vyhovovalo jejich současným i budoucím potřebám. Brzy se k nim připojily další společnosti jako Bosch a Phillips [20].

FlexRay klade, oproti CAN síti, velké omezení na dobu, kdy mohou uzly začít vysílat informace. FlexRay má pro zprávy přímo přidělené časové sloty ve kterých může zařízení vysílat data. Tím ale dochází k plýtvání času a prostoru slotu – vysílána jsou totiž i nulová data [16]. Navíc sběrnice potřebuje přesné hodiny pro časovou synchronizaci všech uzlů. Oproti tomu CAN pracuje na téměř stoprocentní šířce pásma a díky arbitraci ani neplýtvá prostorem.

FlexRay může být také použit pro nové aplikace v průmyslové automatizaci, a to díky deterministickému přístupu ke komunikaci zpráv. Tomu napomáhá použití dvoukanálové topologie, kde každý kanál může pracovat nezávisle. Oba kanály mohou být použity ke stejné komunikaci. Sběrnice je dnes standardizována dle norem ISO 17458-1 až -5.

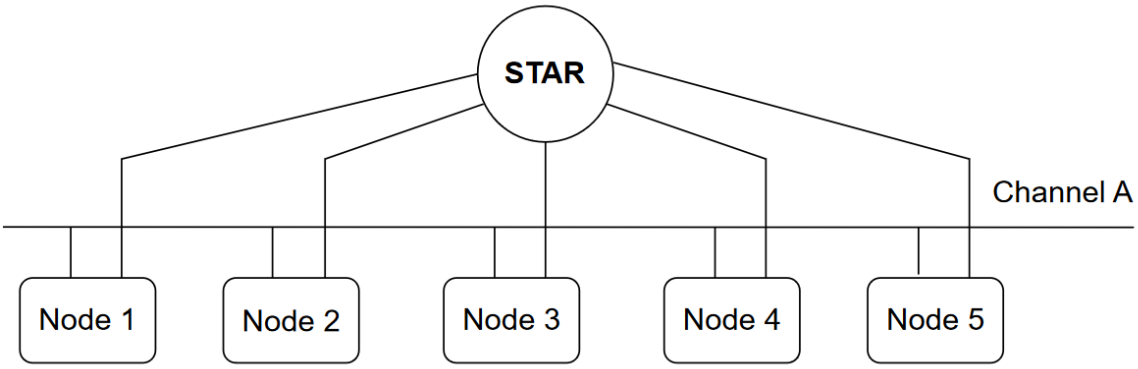
Protokol FlexRay byl navržen tak, aby přenášel informace rychlostí 10 Mbit/s každým ze svých dvou kanálů, zatímco CAN má přenosovou rychlost 1 Mbit/s. To znamená, že pokud se využijí oba kanály současně, lze dosáhnout ekvivalentní přenosové rychlosti až 20 Mbit/s, což je dvacetinásobek rychlosti systému založeného na CAN. Díky takto vysoké přenosové rychlosti jsou systémy FlexRay vhodné jako základ páteřních sítí.



Obrázek 4.3: Schéma zapojení FlexRay.

Fyzická vrstva se skládá ze dvou nestíněných kabelů, FlexRay definuje dvoukanálovou síť, kanál A a kanál B. Uzel může být připojen k jednomu nebo oběma z těchto kanálů. Pokud je uzel připojen k jednomu kanálu, nezáleží na tom, zda se jedná o kanál *A* nebo kanál *B*. Zapojení je schematicky zobrazeno na *obrázku 4.3: Schéma zapojení FlexRay.*

Z tohoto zapojení je patrné, že kanál B může být v případě vhodného zapojení použit jako náhrada kanálu A, například v případě poruchy, nebo může sloužit jako záloha. Toto rovněž přispívá k celkové bezpečnosti protokolu [20].



Obrázek 4.4: Hvězdicové zapojení FlexRay

FlexRay zároveň umožňuje různé topologie zapojení. Viz výše uvedený *obrázek 4.4: Hvězdicové zapojení FlexRay.* Kanál A zde byl implementován jako sběrnice, zatímco kanál B je zapojen do hvězdicové topologie. Možné kombinace topologií FlexRay z něj činí velmi přizpůsobivý a flexibilní systém, který lze navrhnout tak, aby na míru vyhovoval různým aplikacím [20].

Rámec zprávy FlexRay je rozdělen do tří sekcí: sekce záhlaví, přenášených dat a závěrečné sekce.

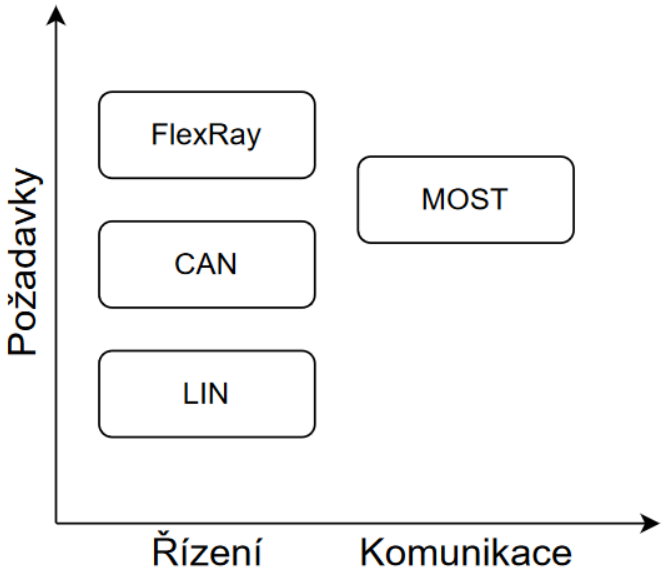
Sekce záhlaví obsahuje stavové informace. Jsou zde také bity indikující délku přenášeného dat a bity CRC. Datové pole obsahuje data, která mají být přenášena po síti. Jeho délka se může lišit od 0 do 254 Bytů. Ve srovnání s 0 až 8 Byty v rámci CAN je to významné zlepšení. Závěrečná sekce je tvořená 24bitovým CRC, které je vypočítané z pole dat a sekce záhlaví [20].

Nedostatky protokolu tkví v chybějícím mechanismu upozorňujícím na chyby. To znamená, že přijímač pouze zahodí chybovou zprávu, aniž by o tom informoval vysílač. Není také zajištěno opakování, v případě nedoručení zprávy.

Společnou vlastností FlexRay a MOST sběrnice je využití časové synchronizace celé sběrnice.

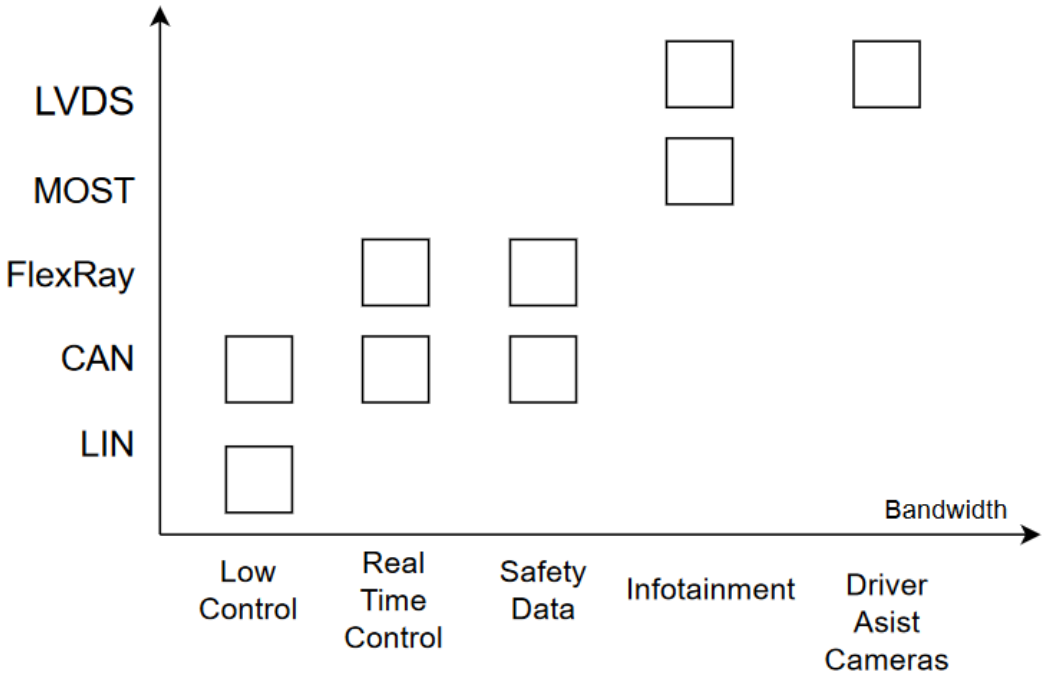
5. SROVNÁNÍ AUTOMOBILOVÝCH SBĚRNIC

V této kapitole jsou shrnuty základní rozdíly mezi zmíněnými sběrnicemi.



Obrázek 5.1: Srovnání sběrnic

Jako první je uveden přehled na *obrázku 5.1: Srovnání sběrnic*. Ten znázorňuje vhodnou volbu komunikační sběrnice v závislosti na způsobu využití.



Obrázek 5.2: Srovnání provozu automobilových sběrnic

Obrázek 5.2: Srovnání provozu automobilových sběrnic zobrazuje detailnější přehled sběrnic, v závislostí na způsobu využití v automobilové komunikaci.

LVDS je zkratka pro metodu přenosu digitálních signálů s nízkým napětím a vysokou rychlostí. Používá se především pro přenos dat mezi zařízeními na krátké vzdálenosti, například v automobilových sítích, displejích nebo výstupních zařízeních pro video a audio signály. LVDS využívá dva dráty pro přenos signálu s diferenciálním způsobem kódování, což zajišťuje lepší odolnost proti rušení a snižuje spotřebu energie.

Dalším podstatným rozdílem sběrnic jsou maximální rychlosti přenosu zpráv jednotlivých protokolů a jejich fyzické rozhraní. Tedy materiál, z čeho jsou tvořeny a k tomu je navázán rovněž způsob (protokol) jakým jsou schopny informace vysílat. Tyto rozdíly jsou shrnuty v následující tabulce.

Tabulka 2: Přehled sériových automobilových komunikačních protokolů

Sběrnice	**Bitová rychlost (max)**	**Médium**	**Komunikační protokol**
LIN	20 kb/s	Jednolinka	Sériový
CAN	1 Mb/s	Kroucená dvoulinka	CSMA/CR
CAN_FD	8 Mb/s	Kroucená dvoulinka	CSMA/CR
SAE J1939	500 kb/s	Kroucená dvoulinka	CSMA/CR
MOST	150 Mb/s	Optické vlákno	TDMA
Ethernet	100 Gb/s	Dvoulinka	TDMA
FlexRay	20 Mb/s	Kroucená dvoulinka	TDMA

6. KOMUNIKACE S HARDWAREM

Šestá kapitola je věnována první polovině praktické části diplomové práce, konkrétně úvodním realizovaným činnostem s hardwarem, který byl pro tuto práci poskytnut, nebo zapůjčen. Zároveň jsou zde uvedeny všechny použité přístroje a zařízení. Dále jsou shrnuty jejich základní informace a vlastnosti. Na tuto část navazuje kapitola sedmá, která se již opírá o znalosti použitého hardwaru.

6.1 Vector VN1630A

Prvním zařízením, které zde je popsáno je převodník VN1630A. Jde o profesionální nástroj pro analýzu a simulaci automobilových sběrnic, jako jsou CAN, LIN a FlexRay. Vyrábí jej společnost Vector Informatik GmbH a je široce používán v automobilovém průmyslu pro testování elektronických řídicích jednotek a síťové komunikace. Zařízení podporuje vysokorychlostní i nízkorychlostní CAN sběrnici. Dokáže zachycovat a analyzovat síťová data v reálném čase. VN1630A je univerzální zařízení a je kompatibilní s programy MATLAB a Simulink prostřednictvím nástroje Vehicle Network Toolbox, který poskytuje podporu pro odesílání a příjem zpráv CAN.

Vehicle Network Toolbox obsahuje editor databází CAN pro vytváření a úpravu databází, které popisují signály a zprávy používané sítí CAN. Toolbox také podporuje import a export standardních databázových formátů, jako je CANdb. Pomocí toolboxu mohou uživatelé prostředí MATLAB vyvíjet a testovat komunikaci CAN, trasovat a analyzovat datový tok v síti.

Obrázek 6.1: Vector VN 1630A

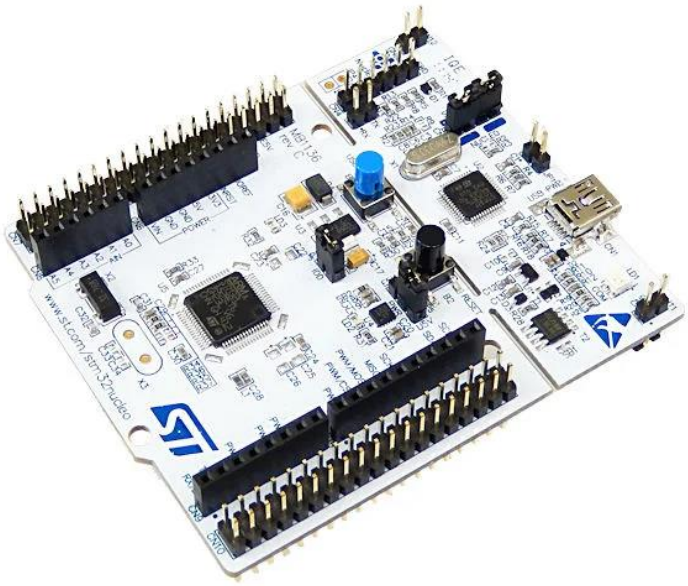
6.2 STM Nucleo F446RE

Vývojová deska STM Nucleo F446RE umožňuje stavět prototypy pro komunikaci softwaru s vnějším světem. Deska má zabudované vstupní / výstupní piny připravené na zapojení propojovacích drátů s pinovými konektory. Což zjednodušuje a zrychluje práci na projektu. Dále umožňují rozšíření základní funkcionality o široký výběr specializovaných zařízení a shieldů.

Na desce je zabudován debugger. Dále se na stránkách výrobce nachází spousta knihoven a příkladů použití desky v praxi. Také oficiální software STM32 Cube IDE k programování kitu je bezplatně ke stažení.

Na kitu se nachází:

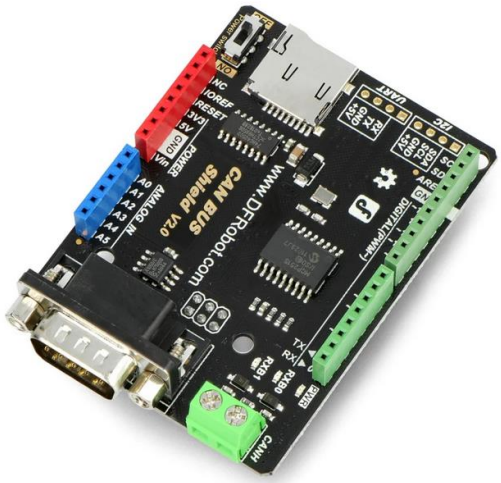
- Jedna uživatelská LED dioda
- Jedno resetovací a jedno uživatelské tlačítko
- 32.768kHz oscilátor
- Napájecí konektor USB mini – B
- Procesor ARM Cortex-M4F



Obrázek 6.2: STM Nucleo F446RE

6.3 DFRobots CAN Bus Shield

CAN-BUS Shield od DF Robots je zařízení, které je používáno ke komunikaci mezi mikrokontroléry, nebo jinými zařízeními pomocí protokolu CAN. Tento shield má integrovaný čip MCP2515, který má funkci CAN-BUS transceiveru. CAN-BUS shield také obsahuje konektory pro připojení k hostitelskému zařízení a umožňuje ukládání dat na MicroSD kartu, což je ideální pro aplikace s datovým záznamem.



Obrázek 6.3: DFR CAN BUS Shield

6.4 CEM jednotka

Zkratka pro centrální elektronický modul ve společnosti Volvo. Jednotka bývá zpravidla umístěná pod schránkou v palubní desce.

Po zapojení CEM jednotky na napájení (12 V) a připojení pinů sběrnice k hardwaru Vector VN1630A, se v Matlabu po spuštění aplikace CAN Explorer zobrazí uživatelské rozhraní pro nastavení připojeného hardwaru.

Dále je umožněno snímat a odečítat signály proudící na připojené sběrnici. A to v reálném čase. Aplikace umožnuje jednoduché třídění a zobrazení signálů. Buďto jak jsou v čase za sebou přijímány, nebo mohou být vyfiltrovány (pomocí Unique Messages) a dále jsou zobrazovány pouze změny signálů, seřazených dle svých ID.

Toto zobrazení je výhodné pro svou jednoduchost a přehlednost sledování informací, neboť některé signály jsou vysílány s různou frekvencí. A tím pádem je na obrazovce vždy nejaktuálnější hodnota každého signálu.

Po dvou zobrazených CAN linkách byly čteny periodické zprávy. Pro CAN Low-Speed (29bit ID, 125 kb/s) a CAN High-Speed (29bit, 500 kb/s).

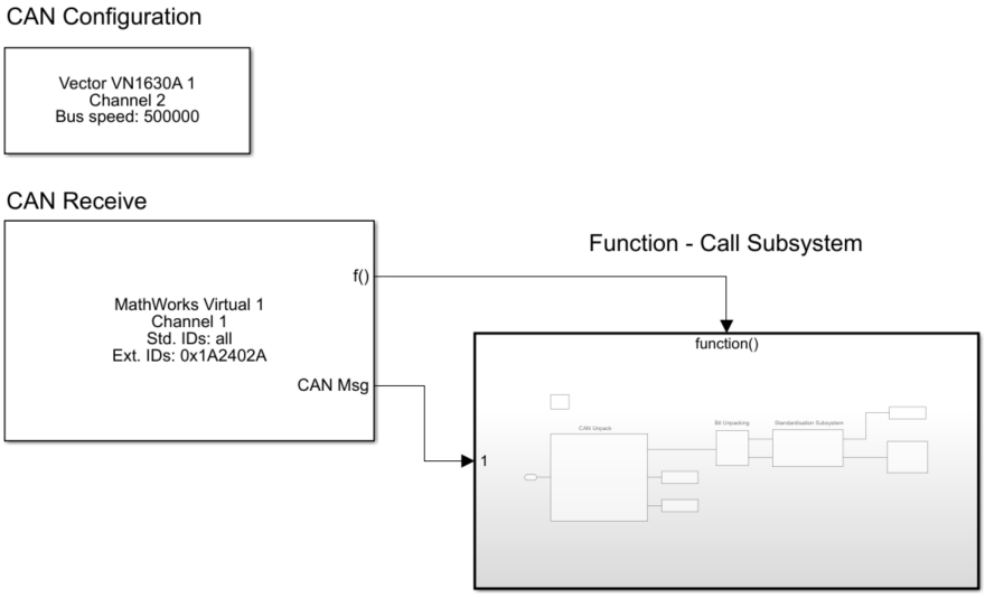
6.5 SAS senzor

Snímač úhlu natočení volantu. Zodpovídá za měření úhlu, úhlové rychlosti a rychlosti změny natočení volantu. Data jsou odesílány do jednotky CEM, která je zpracovává a dále rozesílá do vozidla. Informace jsou posléze používány při provozu dalších systému, jako je posilovač řízení (EPS), stabilizační systém (ESP), asistent pro udržení vozidla v jízdním pruhu (LKA) a další.

Obrázek 6.4: Snímač SAS

6.6 Vlastní program

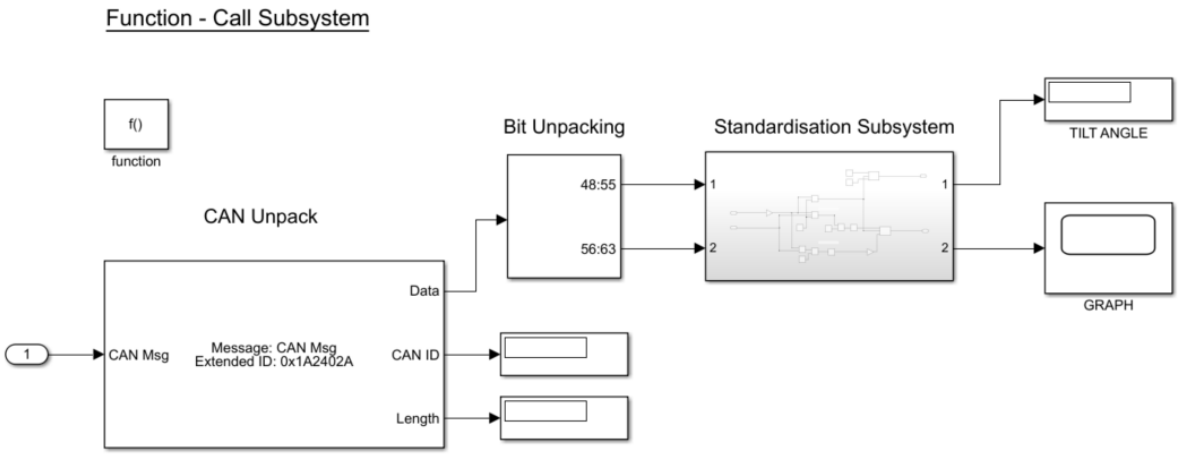
Obrázek 6.4: Snímač SAS zobrazuje konkrétní zařízení, se kterým se pracovalo ve spojení s jednotkou CEM a se Simulinkem. V kapitole je přehled odvedené práce s tímto snímačem.



Obrázek 6.5: Načítání dat v Simulinku

Po propojení převodníku VN1630A společně s jednotkou CEM, do které byl připojen i SAS, se v Simulinku vytvořil jednoduchý program pro načítáni dat, viz *obrázek 6.5: Načítání dat v Simulinku.*

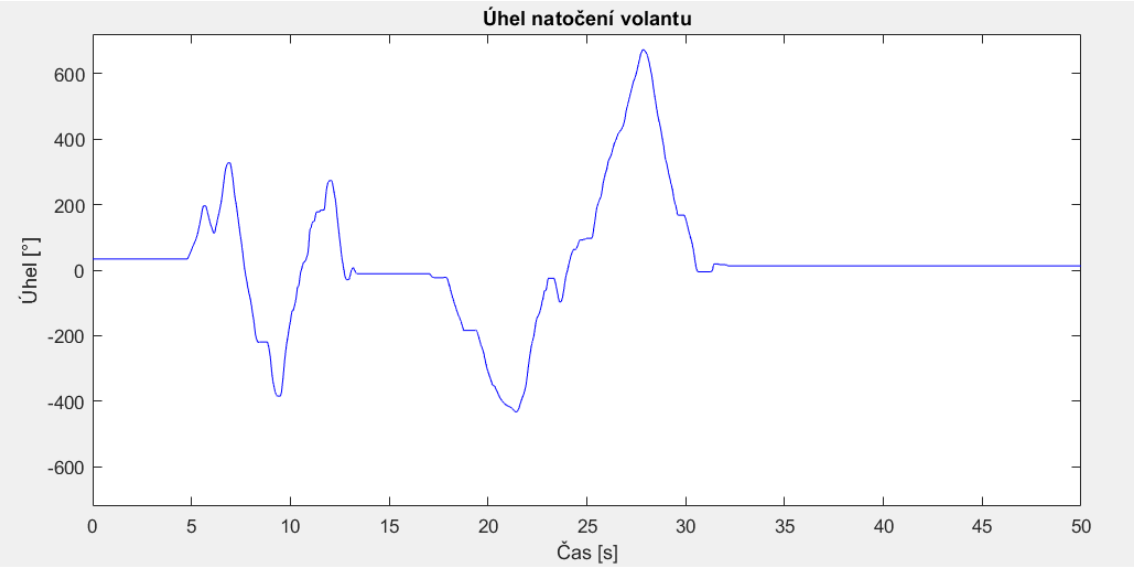
Prvně bylo potřeba nastavit blok CAN Configuration. V něm byl nastaven kanál a jeho požadovaná rychlost. Dále se pracovalo s blokem CAN Receive, kde bylo opět potřeba nastavit kanál doplněný o ID chtěné zprávy. Snímač vysílá rozšířená data s hlavičkou ID: 01A2402A.



Obrázek 6.6: Function Call Subsystem

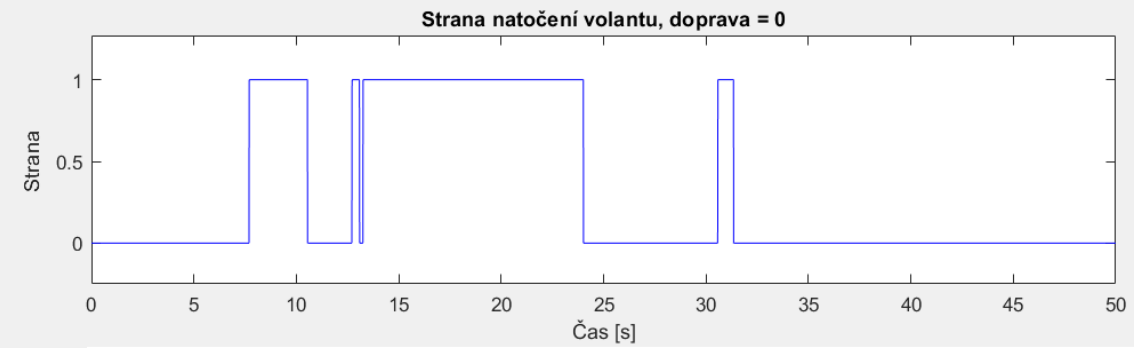
V další části programu bylo potřeba vybrat vhodné bity z těla zprávy (viz *obrázek 6.6: Function Call Subsystem)*, které přenáší zprávy o hledaném úhlu natočení. Za pomocí bloků CAN Unpack a dále Bit Unpacking.

Hledaná data jsou uložena na posledních dvou Bytech zprávy. Z nich byla provedena standardizace hodnot a následný převod na úhlové natočení. Přičemž maximální úhel, v absolutní hodnotě, kterou dokázal SAS vysílat, byl 720°. Při překročení této hodnoty se senzor octl v chybě a bylo jej potřeba fyzicky odpojit a znovu připojit k CEM modulu.



Obrázek 6.7: Graf získaných hodnot úhlů natočení volantu

Na *obrázku 6.7: Graf získaných hodnot úhlů* je výsledný graf natočení volantu.



Obrázek 6.8: Graf směru otáčení volantu

Tento *obrázek 6.8: Graf směru otáčení volantu* doplňuje předešlý graf a ukazuje, ve kterém směru je volant natočen. Zda doprava (v grafu symbolizuje hodnota 0), či doleva (hodnota 1).

6.7 Komunikace přes OBD 2 konektor

V rámci práce byl proveden experiment s načítáním dat z nastartovaného vozidla do prostředí Matlab pomocí OBD2 konektoru (viz *obrázek 1.1: Konektor OBD2*) a zařízení Vector.

Výsledky experimentu na vozidle Volkswagen Golf z roku výroby 2012 naznačily, že prosté čtení dat prostřednictvím OBD2 není možné. Tato konkrétní CAN sběrnice byla chráněna a fyzicky oddělena od ostatních sběrnic ve vozidle, což může být důvodem nedostupnosti dat. Data jsou v takovém případě k dispozici pouze po zavolání, jak uvádí ISO norma 15765.

Tentýž pokus byl proveden na vozidle Volvo S80 z roku 2007, kde bylo úspěšně zachyceno a pasivně odposlechnuto 47 různých identifikátorů a jejich datových zpráv. Na této Volvo platformě bylo oproti VW možné přímo přistupovat k datům, včetně chybových kódů a proměnných dat ze snímačů. Zobrazeny byly díky matlabovské aplikaci CAN Explorer.

Tento rozdíl mezi vozidly naznačuje, že existují odlišnosti v architektuře a implementaci sběrnic CAN v různých automobilech a také v závislosti na době výroby vozu. Některá auta mohou mít omezenou podporu nebo dodatečná opatření, která ovlivňují přístup k datům skrze OBD2 konektor.

Tyto poznatky mají praktický význam při práci s automobilovou diagnostikou a monitorováním sběrnice pomocí OBD2. Aby bylo možné efektivně a spolehlivě získat potřebná data pro další analýzy a diagnostiku, je nutné porozumět specifickým vlastnostem jednotlivých vozidel a jejich implementaci OBD2 protokolu.

7. VLASTNÍ ZAŘÍZENÍ

Pro diplomovou práci bylo stěžejní vytvořit MATLAB Simulink model komunikující po vybrané sběrnici, a to prostřednictvím podporovaného komunikačního zařízení.

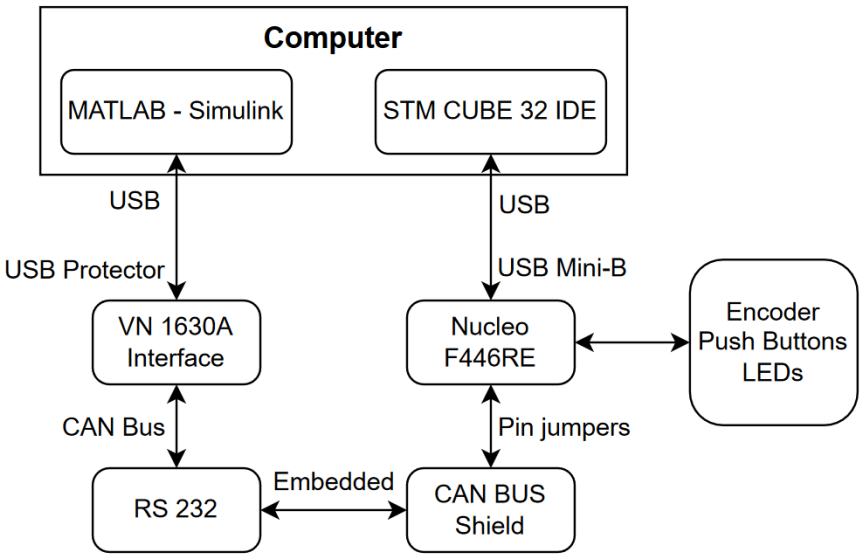
Dalším požadavkem bylo používat minimálně jedno řídící a jedno řízené zařízení. A vytvořit k nim stručný databázový soubor.

Vlastní projekt, který je řešen na následujících stránkách se zaobírá předáváním dat po CAN sběrnici mezi STM kitem Nucleo F664RE a prostředím Matlab. Programovací kit byl naprogramován ve vývojovém prostředí STMcube32. Důležitou periférii pro mikrokontroler F446RE byl v této práci enkodér. Jeho hodnoty slouží jako vstupní proměnné programu. Pomocí enkodéru byla zjišťována poloha, směr otáčení a indikace stlačení enkodérového tlačítka.

Data byla načítána do mikrokontroléru a následně vnitřně zpracována. Výsledná hodnota natočení byla odesílána na připojenou CAN sběrnici. Dále bylo ještě připojeno externí tlačítko a LED diody. Ty jsou na základě přijatých zpráv ze Simulinku rozsvěcovány na čelním panelu vlastního zařízení.

Aby bylo možné posílat data z kitu na sběrnici, bylo nezbytné zapojení dalšího zařízení. Proto bylo na samotný STM kit připojeno rozhraní CAN BUS Shield od společnosti DFRobots. To obsahuje funkci CAN bus transceiveru. Zároveň má v sobě zabudován devíti pinový konektor D-Sub, ten je vhodný pro připojení na CAN sběrnici.

Je vhodné zmínit, že toto rozhraní umožnuje být napájeno buďto přímo z vývojové desky, ke které je připojeno, nebo může být napájeno z připojené CAN sběrnice. K volbě režimu napájení slouží přepínací tlačítko umístěné přímo na desce. V této práci je transceiver napájen z vývojové desky STM.



Obrázek 7.1: Schéma propojení zařízení

Na opačné straně CAN sběrnice bylo připojené rozhraní VN1630A společnosti Vector. Tento CAN/LIN převodník byl dále připojen přes USB k PC a v reálném čase odesílal data ze sběrnice do prostředí Matlab – Simulink. Tímto je okruh přenosu dat mezi Matlabem a vlastním zařízením dokončen. Zjednodušené zapojení lze vidět na *obrázku 7.1: Schéma propojení zařízení.*

7.1 Architektura projektu

Tato podkapitola má za cíl zpřehlednit kroky a funkce obou programů. Je zde popisován *obrázek 7.2: Architektura programu*, který je z důvodů své velikosti umístěn na následující straně.

V horní třetině jsou znázorněny vstupy do programu pro kit F446RE. Konkrétně datový tok z enkodéru (*encoder_position* a *encoder_button*), dále se zjišťuje stav pěti LED diod použitých pro vizualizaci chodu programu a posledním vstupem je funkce *HAL_GetTicks(),* ta vrací aktuální počet milisekund od spuštění aplikace (tzv. tick count). Funkce je přímou součástí HAL knihovny. Důvod použití této funkce je dvojí – jednak bylo potřeba vymyslet proměnná data která budou proudit z kitu ven.

A zadruhé je funkce *HAL_GetTicks()* typicky používána v RTOS systémech, kde je důležité měřit a řídit časování událostí s vysokou přesností a spolehlivostí. V projektu je tímto ověřeno, že přes sběrnici nikde nedochází k zásadnímu zpoždění aktivity.

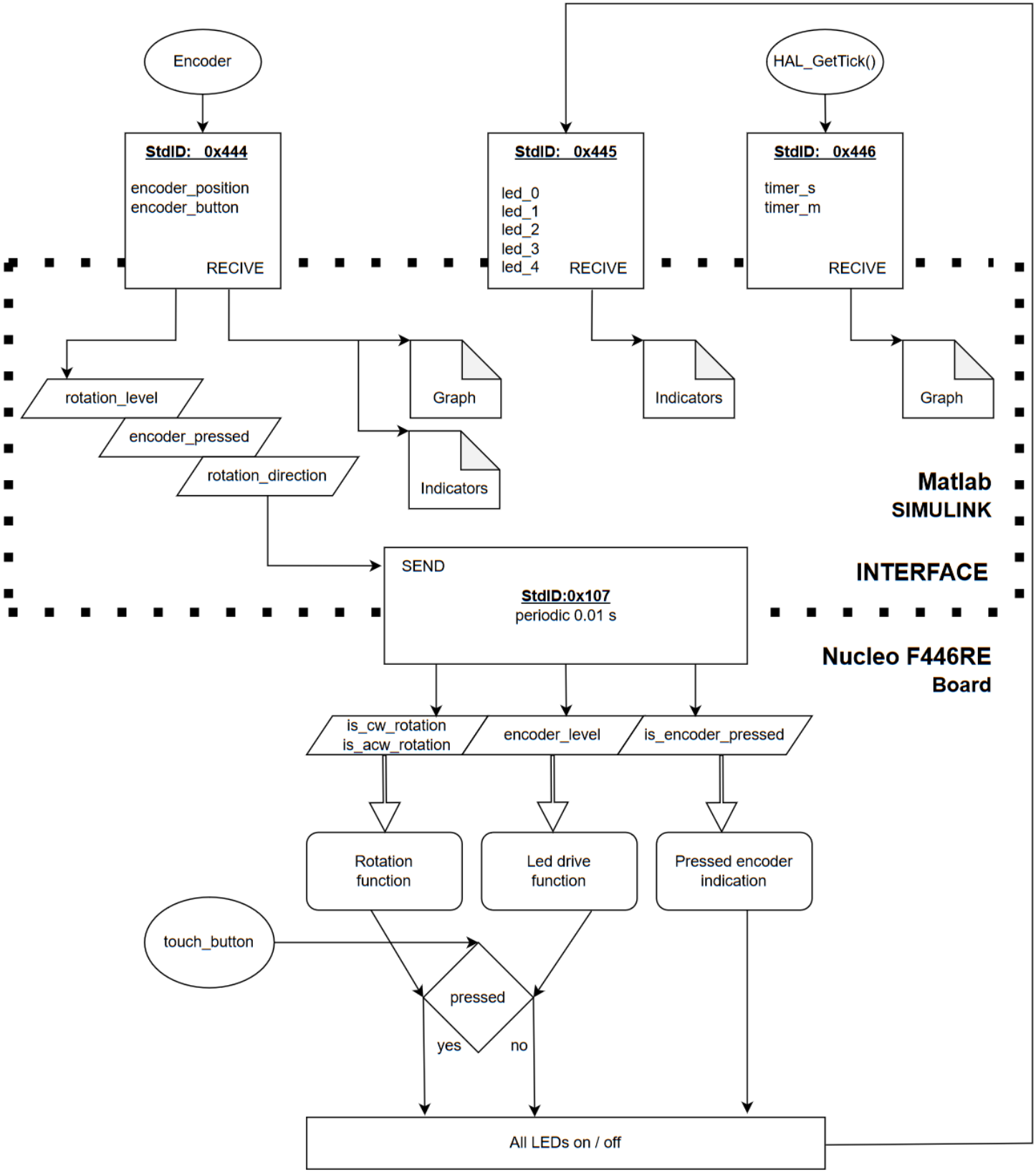
Střední část schématu je ohraničená čtverečkovaným rozhraním a zobrazuje chod programu Simulink. Skrze tři standardní ID: 0x444, 0x445 a 0x446 vnikají do Matlabu proměnné. Ty jsou následně zobrazeny do grafu, či na display v prostředí Simulink. Také jsou zde tři funkce: *rotation_level*; *encoder_pressed*; *rotation_direction*. Více o této části je v kapitole 7.5 Simulink projekt. Ze zpracovaných dat jsou výsledné hodnoty posílány skrze standardní ID: 0x107 zpět do mikrokontroléru.

Poslední část diagramu je opět o scriptu. Přijaté proměnné jsou posílány do funkcí v těle programu s následným vyhodnocováním v reálném čase, a to viditelně díky LED diodám na čelním panelu zařízení. Panel je blíže popsán až v kapitole 7.4 Čelní panel. V této části je zpracována i informace (*touch_button*) o stavu stisknutí externího tlačítka. Hodnota je jediná, která neproudí mezi rozhraními skrze CAN sběrnici. A interně rozhoduje o aktivací dvou funkcí:

a) Rotation Function, které znázorňuje směr otáčení enkodéru na čelním panelu

b) LED drive Function, kde dochází k postupnému rozsvěcování diod v závislosti na natočení enkodéru.

Hodnoty stavu pěti ledek jsou nakonec vysílány na sběrnici pod ID 0x445.



Obrázek 7.2: Architektura programu

7.2 Zapojení

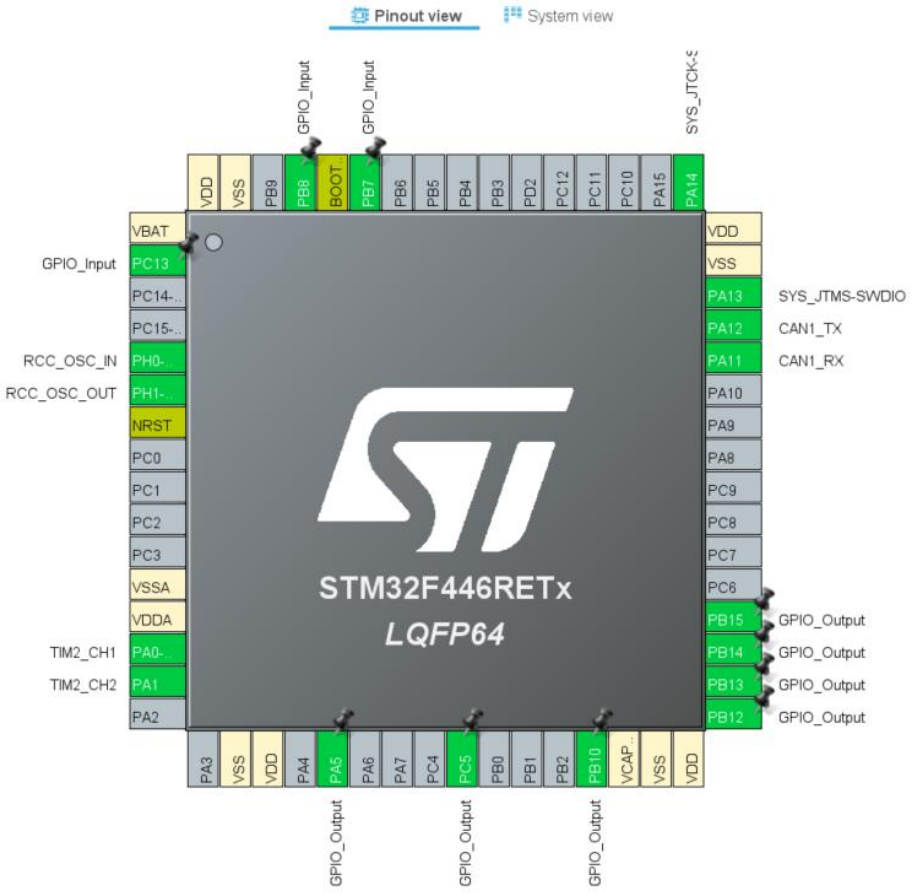
V níže uvedené *tabulce 3: Pinout zapojení* jsou informace o zapojení jednotlivých pinů nebo konektorů. Je zde uvedeno kompletní zapojení funkčního zařízení. Je třeba poznamenat, že ke každé z pěti externích LED diodě byl připojen 120 Ω odpor.

Tabulka 3: Pinout zapojení

F446RE	CAN bus Shield	LED + R 120 Ω	Enkodér	Externí tlačítko
PA_11	CAN_RX			
PA_12	CAN_TX			
PB_12		LED_12		
PB_13		LED_13		
PB_14		LED_14		
PB_15		LED_15		
PC_5		LED_10		
PA_0			DT	
PA_1			CLK	
PB_7			SW	
PB_8				+
VIN	VIN			
RESET	RESET			
5 V	5 V		+	
GND	GND	GND	GND	GND

7.3 STM Cube projekt

Následující *obrázek* 7.3: *Pinout View* v STM32Cube IDE zobrazuje grafické znázornění konfigurace pinů mikrokontroléru ve vývojovém prostředí. Je zde umožněno vývojářovi snadno a přehledně určit funkci jednotlivých pinů a způsob jejich připojení k periferiím. Tyto informace jsou na začátku projektu klíčové, protože pomáhají zajistit správný návrh hardwaru a správnou konfiguraci mikrokontroléru.



Obrázek 7.3: Pinout View

Pinout View nabízí integraci s generátorem kódu, což umožňuje automatický přenos konfigurace pinů z náhledu do inicializačního kódu pro mikrokontrolér. Tato funkce snižuje riziko chyb a urychluje proces vývoje, protože uživatel nemusí manuálně psát inicializační kód pro každý pin.

Při případné úpravě, či rozšíření konfigurace pinů v Pinout View se tyto změny automaticky promítnou do nově generovaného kódu, což zajišťuje konzistenci a přesnost pinů v celém projektu. Tímto způsobem se zjednodušuje vývojový proces, šetří se čas a úsilí programátorů.

Nutno ovšem dodržet pravidla pro psaní kódu pouze mezi vygenerované komentáře `/* USER CODE BEGIN x */` a `/* USER CODE END x */`. V opačném případě při novém generování kódu dojde ke ztrátě části vlastního kódu, který byl na nepatřičném místě.

Pin Name	GPIO mode	GPIO Pull-up/Pull-down	GPIO output level
PA5	Output Push Pull	No pull-up and no pull-down	Low
PB7	Input mode	Pull-up	n/a
PB8	Input mode	Pull-up	n/a
PB10	Output Push Pull	No pull-up and no pull-down	Low
PB12	Output Push Pull	No pull-up and no pull-down	Low
PB13	Output Push Pull	No pull-up and no pull-down	Low
PB14	Output Push Pull	No pull-up and no pull-down	Low
PB15	Output Push Pull	No pull-up and no pull-down	Low
PC5	Output Push Pull	No pull-up and no pull-down	Low

Obrázek 7.4: Nastavení GPIO pinů

GPIO piny jsou univerzální vstupy/výstupy, které umožňují komunikaci mezi mikrokontrolérem a externími zařízeními. Mohou být také nastaveny do různých režimů, v mém projektu byly piny nastaveny, jak je na *obrázku 7.4: Nastavení GPIO pinů*.

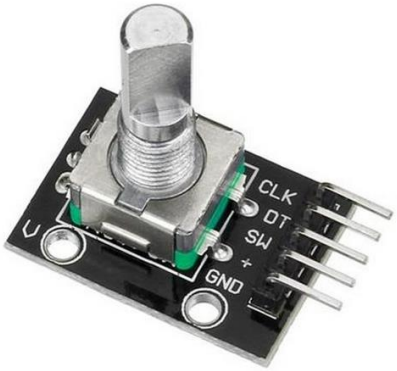
Režim pinu určuje, zda bude pin nastaven jako Push-Pull nebo Open-Drain. V režimu Push-Pull je pin nastaven jako vysoký (logická 1). V režimu Open-Drain může být pin buď připojen k zemi (logická 0), nebo "otevřený" (neaktivní stav).

Pull-up rezistory jsou používány k zajištění stabilního stavu vstupních pinů. Pokud by výstupní pin nebyl připojen k žádnému zařízení, mohlo by se stát, že napětí na pinu bude neustále kolísat, a způsobí tak nežádoucí chování. Tyto rezistory se také používají pro připojení výstupních pinů k periferiím spínačů. A tak tomu je i v této práci. Kde jsou na pinech *PB_7* a *PB_8* jsou zapojeny tlačítka.

7.1 Enkodér

Načítání dat z enkodéru bylo zjednodušené díky zabudované funkci v STM CubeIDE pro enkodéry.

Pro správnou funkčnost bylo potřeba provést několik úprav v programu. Nejprve byla povolena periferie TIM a časovač byl nakonfigurován tak, aby pracoval v režimu enkodéru. Dále bylo řešeno ošetření zákmitů tlačítka enkodéru, které mohou způsobovat rušení signálu jdoucího do mikrokontroléru při stisknutí, nebo uvolnění tlačítka. Cílem těchto úprav bylo filtrovat nepřesnosti a zajistit čistý a stabilní signál.



Obrázek 7.5: Použitý enkodér

V rámci programu bylo ošetření zákmitů realizováno následovně: Byla použita proměnná *is_button_pressed*, která slouží ke sledování stavu tlačítka enkodéru, a hodnota proměnné *touch*, která se mění v závislosti na stisku tlačítka. Tímto způsobem je dosaženo eliminace zákmitů a poskytování spolehlivého signálu tlačítka enkodéru.

```
//debounce – Handling of the read values from the encoder button (pin PB_7)

uint32_t current_time = HAL_GetTick();
button_state = HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_7) == GPIO_PIN_RESET;

          if (button_state != is_button_pressed) {
              if (current_time - last_debounce_time > 50) {
                  is_button_pressed = button_state;
                 }
                if (is_button_pressed) {
                       touch ^= 1;
                              // Rotate the 'touch' based on the button press
                    }
                last_debounce_time = current_time;
          }
```

Níže je uvedený kód, který představuje odeslání zprávy přes sběrnici CAN v mikrokontroléru pomocí funkce *HAL_CAN_AddTxMessage()* z knihovny HAL. Kód nastavuje různé parametry pro zprávu, jako je délka dat, typ identifikátoru (v tomto případě standardní), typ rámce, a ID zprávy (0x444).

Poté je aktivována notifikace pro příjem zpráv na sběrnici CAN pomocí funkce *HAL_CAN_ActivateNotification()*. Následně jsou data z *encoder_position* uložena do proměnné TxData. A zpráva je odeslána.

Proměnná *mode* je použitá k odesílání různých datových zpráv s jinými ID po sobě. Tímto je zaručeno že se zprávy nebudou mezi sebou mísit. Tato zpráva je zde uvedena jako demonstrace funkčnosti kódu v real time aplikaci.

```
if (mode==0) {

       TxHeader.DLC = 1;               // data length
       TxHeader.IDE = CAN_ID_STD;      // standard ID
       TxHeader.RTR = CAN_RTR_DATA;    // type of frame
          TxHeader.StdId = 0x444;      // ID

       HAL_CAN_ActivateNotification(&hcan1, CAN_IT_RX_FIFO0_MSG_PENDING);

       TxData[0] = encoder_position;          // position of encoder
          HAL_CAN_AddTxMessage (&hcan1, &TxHeader, TxData, &TxMailbox);
          HAL_Delay(5);
          mode++;
   }
```

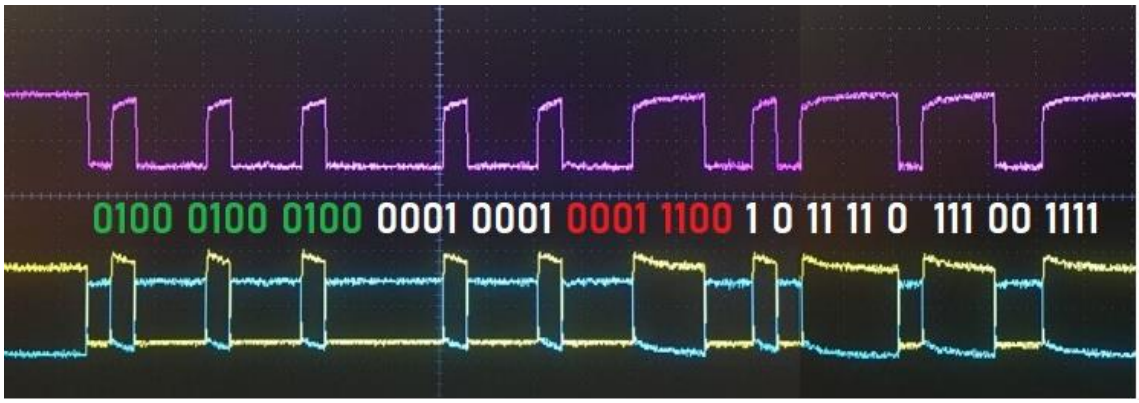
Jako ID zprávy, která tuto proměnnou nese bylo zvoleno 0x444. Protože každý symbol hexadecimální soustavy reprezentuje čtyři bity, celková hodnota 0x444 reprezentuje binární číslo 0100 0100 0100.

V rámci této fáze diplomové práce byly ke sběrnici připojeny sondy osciloskopu. Byla záměrně vybrána specifická hodnota, která umožňovala snadno pozorovat změny přenášených dat a prověřit tak funkčnost systému v reálném čase. Při otáčení enkodéru bylo možné vizuálně sledovat postupný posun bitů dat na osciloskopu bez potřeby dalšího zařízení či sofistikovaných měření.

Pro potřeby znázornění tohoto kroku a z důvodů zdokumentování správnosti kódu do diplomové práce byla nastavena pozice enkodéru na hodnotu 28. Binární hodnota tohoto čísla je 0001 1100.

Pro odečítání dat z osciloskopu bylo použito matematické funkce rozdílu dvou kanálů. Záměrně je ale odečítáno $V_{CAN\_Low} - V_{CAN\_High}$. Toto zapojení bylo provedeno z důvodu lepšího vizuálního zobrazení přenášených dat na osciloskopu. Přenášená hodnota byla zvolena tak, aby převládal dominantní stav bitu jako bitová hodnota 1 a minimalizovala se viditelnost stavu bitu 0.

Tedy obráceně, než je uvedeno v definici normy ISO 11898 a ve vzorci uvedeném v kapitole 3.2 Fyzická vrstva CAN. Tímto způsobem bylo zajištěno jasnější a výraznější zobrazení datového signálu na osciloskopu, což usnadnilo sledování a ověření správné funkčnosti systému.



Obrázek 7.6: Osciloskop I.

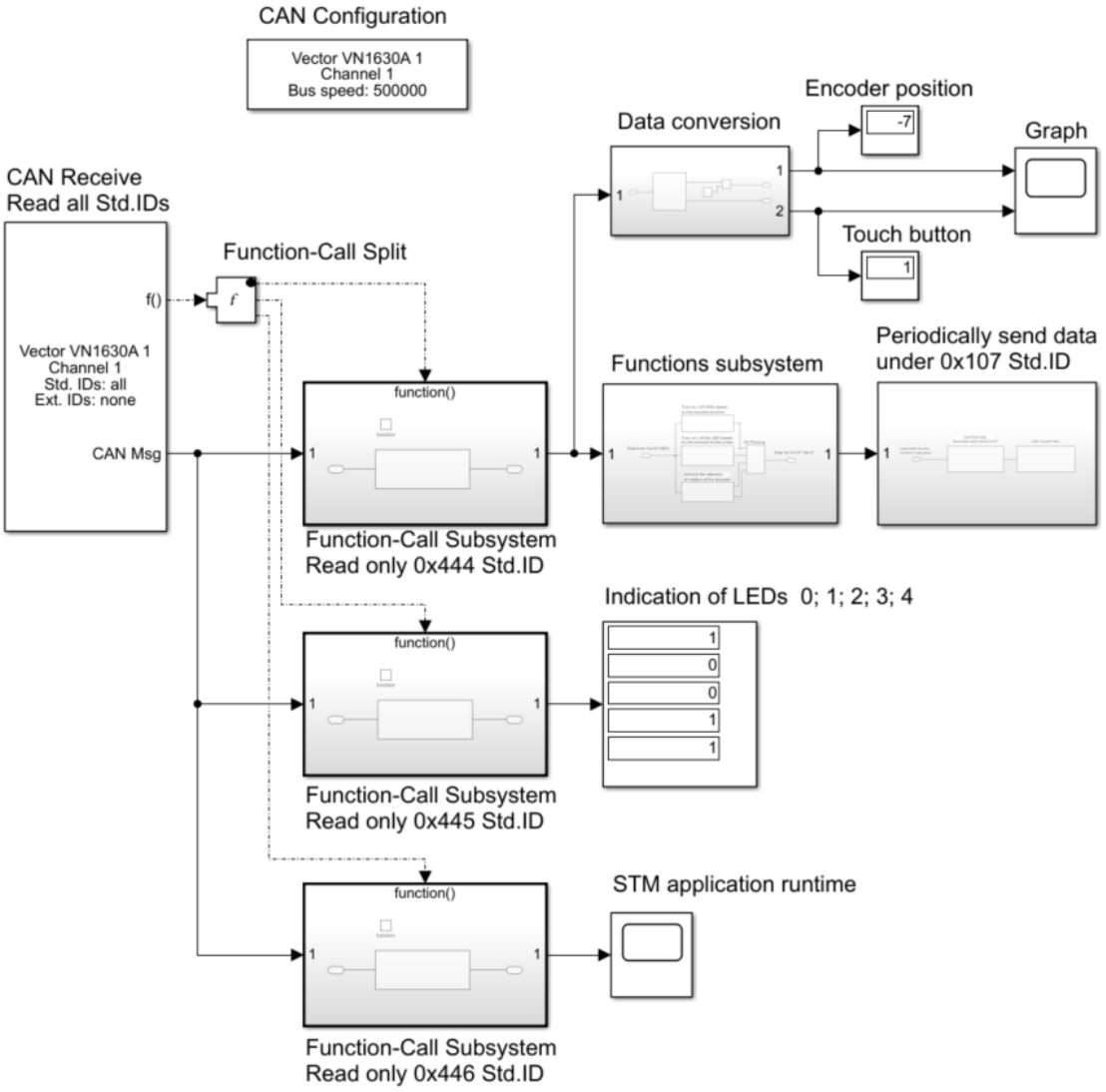
Na *obrázku 7.6: Osciloskop I.* je zobrazen display osciloskopu s vepsanou binární hodnotou signálu. Fialový signál je výsledkem rozdílu napětí na vodičích sběrnice a udává data. Datový rámec je barevně rozdělen do tři části.

Zelená část udává ID zprávy (viz 0x444 v binární soustavě), červená zvýrazňuje hodnotu přenášené informace (hodnota 28). A bity v bílé barvě, představují kontrolní část po přečtení ID zprávy a poté část CRC s několika ukončovacími bity v samotném závěru.

Tímto obrázkem bylo dokázáno, že vysílání dat z kitu F446RE a CAN BUS Shieldu na sběrnici funguje dle očekávání.

7.2 Simulink projekt

Matlab a Simulink jsou vynikající nástroje pro vývoj a testování řídících algoritmů a systémů. Jednou z výhod těchto nástrojů je, že umožňují online komunikaci s CAN sběrnicí pomocí Vehicle Network Toolboxu.



Obrázek 7.7: Hlavní část Simulink modelu

Obrázek 7.7: Hlavní část Simulink modelu znázorňuje úvodní panel v mém programu. Pro práci jsou podstatné následující funkční bloky:

- **CAN Configuration**, kde uživatel volí připojené zařízení a kanál se kterým chce komunikovat. V projektu je použitá pouze Baud Rate: 500 000.

- **CAN Receive** blok umožňuje základní filtr identifikátorů CAN zpráv. Byly povoleny všechny standardní typy ID a zakázány ty rozšířené. Zprávy pak proudí do Function-Call Subsystému, ve kterém se nachází blok CAN Unpack. Každý již čte pouze zprávu se specifikovaným ID.

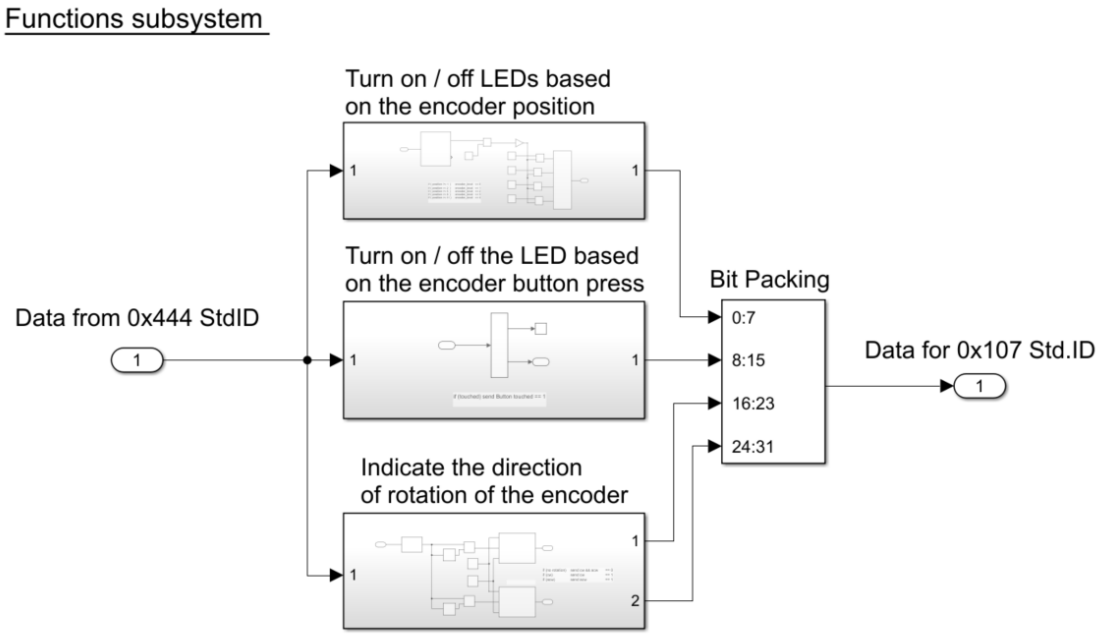
- **Subblok Data Conversion** pouze dekóduje hodnotu přijaté pozice enkodéru. Pro počáteční nulové natočení enkodéru je vysílána hodnota 127 (polovina datového typu uint8) a v tomto bloku je opět posunuta na nulovou hladinu.

Dále se zde nachází dva grafy:

- Pro natočení a indikaci zmačknutí enkodérového tlačítka
- Pro znázornění času běhu aplikace na kitu STM

A tři displaye zobrazující:

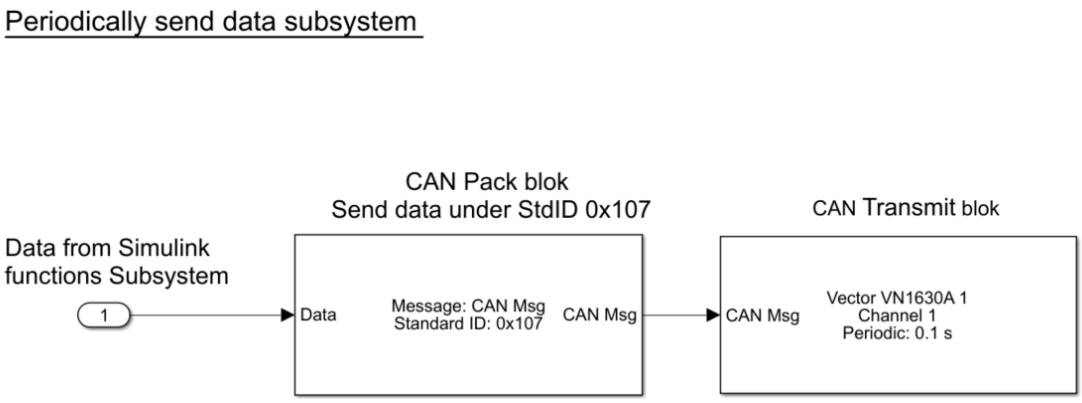
- Hodnotu natočení enkodéru
- Indikaci tlačítka enkodéru
- Pět řádků pro ukazatele stavů LED diod



Obrázek 7.8: Function Subsystém

Subsystém naplněný vlastními funkcemi je uvedený na *obrázku 7.8: Function Subsystem* a jsou zde popisovány tři funkcionality Simulink modelu.

- První navrací hodnotu *encoder_level* na základě dat o natočení enkodéru
- Další vypíná a zapíná diodu na čelním panelu, prostřednictvím proměnné *button_touched*, na základě přijatého signálu z tlačítka enkodéru
- Poslední blok vysílá dvě hodnoty: *cw_rotation* a *acw_rotation*. Ty inicializuje a vypíná v případě, že funkce odhalí natáčení enkodéru doprava, či doleva



Obrázek 7.9: Transmit data Subsystém

Posledním a významným blokem je Subblok který periodicky odesílá data. Jeho vnitřní část je na *obrázku 7.9: Transmit data Subsystem.* Přijatá data z bloku funkcí posílá pod hlavičkou standardního ID 0x107 jako CAN Message do bloku CAN Transmit blok, ten odesílá signál na sběrnici s periodou 0.100 s.

7.3 Přijaté zprávy

Podkapitola je zaměřena na přijímání zpráv z pohledu F446RE kitu. Je zde uvedený kód, který popisuje funkci, jež se aktivuje při příjmu zprávy na CAN sběrnici.

Zprávu obdrží FIFO 0. To je speciální paměťový buffer v CAN řadiči, který slouží pro příjem zpráv. Tento buffer umožňuje řazení přijatých zpráv v pořadí, v jakém byly doručeny, a zajišťuje tak, že nejstarší zprávy jsou zpracovány první.

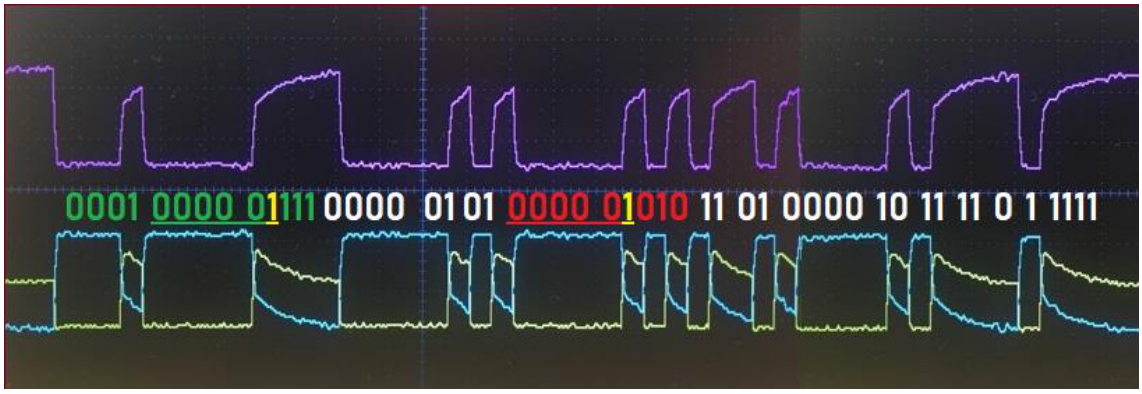
Poté kód získá zprávu a uloží data do proměnných za předpokladu shody standardního ID s hodnotou 0x107. Konkrétně tato funkce uloží data z CAN zprávy do proměnných *encoder_level, button_touched, cw_rotation* a *acw_rotation*, které jsou použity v dalších částech programu.

```
void HAL_CAN_RxFifo0MsgPendingCallback(CAN_HandleTypeDef *hcan)
   {

         HAL_CAN_GetRxMessage(hcan, CAN_RX_FIFO0, &RxHeader, RxData);

      if (RxHeader.StdId == 0x107)
            {
            encoder_level      = RxData[0];       // for LED Drive fcn
            button_touched     = RxData[1];       // for push indication
            cw_rotation        = RxData[2];       // for rotation fcn
            acw_rotation       = RxData[3];       // for rotation fcn
            }
   }
```

Pro kontrolu funkčnosti kódu bylo provedeno další testování přenosu zpráv přes sběrnici s ukázkou na osciloskopu.

Z *obrázku 7.9: Transmit data Subsystém* je patrné, že se periodicky odesílá zpráva pod standartním ID 0x107. V binárním zápise je ID: 0x107 hodnotou 0001 0000 0111. Toto ID bylo vymyšleno záměrně tak, aby v binárním přepisu mělo za sebou minimálně pět nul. Tímto jsem chtěl ověřit, zda vůbec a případně jak dojde k Bit Stuffingu uvedeného v kapitole 3.5 Zabezpečení a detekce chyb.

Zpráva ze Simulinku na osciloskop vysílá hodnotu proměnné *encoder_level*. Ta je používána v C kódu ve funkci LED Drive – ve které se postupně zapínají a zhasínají LEDky v závislosti na natočení enkodéru. Nutno podotknout, že na tomto případě je zároveň otestováno, že data o natočení enkodéru proudí v reálném čase jak z STM kitu (pošle hodnotu natočení *encoder_position*), tak ze Simulinku (pošle hodnotu *encoder_level*, která uvádí kolik LEDek má svítit). Nesoustředěný čtenář by si mohl myslet, že se diody zapínají přímo na základě natočení enkodéru bez použití CAN sběrnice. Není tomu tak. Komunikace funguje v obou směrech.



Obrázek 7.10: Osciloskop II.

Obrázek 7.10: Osciloskop II. zobrazuje data přijatá ze sběrnice a osciloskop byl nastaven úplně stejně jak tomu bylo u předešlého měření, uvedeného na *obrázku 7.6: Osciloskop I*. Zelenou barvou je opět znázorněno ID zprávy. Červenou poté samotná hodnota *encoder_level*, jež zde nabývá hodnotu dva – ve dvou Bytovém, binárním zápisu je to číslo 0000 0010. Opět záměrně zvolená hodnota pro ověření případného Bit Stuffingu.

Dále jsou na *obrázku 7.10: Osciloskop II* podtrženy nuly, kterých je pět v řadě za sebou. Bit Stuffing zde byl dokázán, a po pěti recesivních (nebo dominantních) znacích jdoucích po sobě protokol CAN zareaguje a vkládá jeden bit opačné úrovně. Na obrázku znázorněn žlutou podtrženou jedničkou. Tyto bity se nesmí počítat do celkové binární hodnoty identifikátorů ani přenášené informace! Vložený bit se při čtení zprávy musí jednoduše přeskočit. Jinak by došlo ke zkreslení přenášených dat.

Těmito kapitolami bylo ověřeno následující:

- Komunikace je oboustranně funkční.
- Hodnoty jsou přenášeny v reálném čase bez zpoždění.
- Bit Stuffing byl ověřen a zobrazen na osciloskopu.
- Sekce ID má 12 bitů (Bit Stuffing se do této hodnoty nepočítá!).
- Část Control, na obrázcích mezi zeleným ID a červenou hodnotou přenášené informace, má v základním přenosu datového rámečku osm bitů (dohromady s Bit Stuffing bitem!).
- Část CRC (16 bitů), která je zařazená po přenosu proměnné, se nepodařilo zcela objasnit a chová se vždy odlišně při vysílání z Nucleo kitu STM a jinak při vysílání ze Simulinku. Jediným zcela jasným bitem byl až poslední (Delimiter) bit, který vždy nabývá recesivní hodnoty (pro jistotu, z pohledu ISO normy znamená recesivní = 1) a je jim signalizováno ukončení CRC.

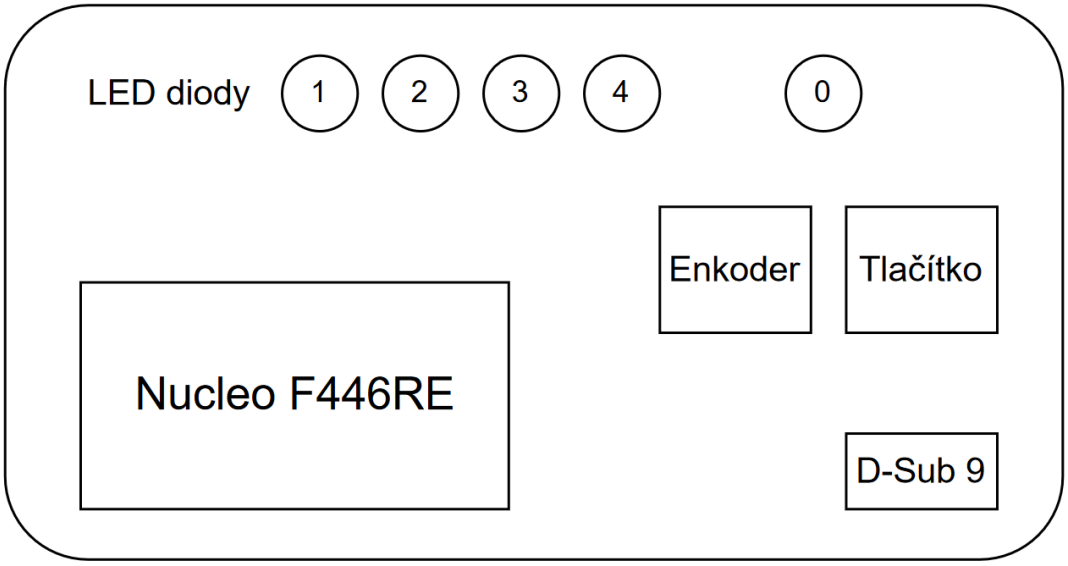
  Norma Autosar, dle které se v tomto případě řídí i Matlab, specifikuje konkrétní CRC algoritmy a jejich parametry, které jsou použity v rámci komunikace mezi komponentami v automobilovém prostředí. Tato diplomová práce si ale neklade za cíl objasnit chování CRC.
- Po poli CRC následuje pole ACK, kde jsou oba bity nastaveny do recesivních hodnot.

- Poslední část datového toku EOF byla zakončena sedmi recesivními bity. Na obrázcích z osciloskopu to ale není z důvodů rozlišení vidět.

- A v práci není možno vidět ani následující část IFS, která slouží k oddělení jednotlivých CAN zpráv od sebe. IF nabýval hodnot 3–10 bitů.

7.4 Čelní panel zařízení

Využití čelního panelu zařízení, který obsahuje různá vstupní a výstupní zařízení, je důležité pro interakci uživatele s daným systémem.



Obrázek 7.11: Schéma čelního panelu

V případě vlastního zařízení (viz *obrázek 7.11: Schéma čelního panelu*), které zahrnuje Nucleo F446RE, enkodér, tlačítko, D-Sub konektor a pět LED diod, čelní panel slouží k ovládání a vizualizaci různých funkcí a stavů systému. A představuje tak důležitý prvek, který umožňuje uživatelům interagovat se systémem.

LED dioda 0 indikuje zmáčknutí enkodérového tlačítka. Dále diody 1–4 jsou postupně rozsvíceny ve funkci *Led drive*. Funkce *Rotation* programu poté používá přístup k LED 1 pro indikaci směru otáčení enkodéru proti směru otáčení hodinových ručiček a pro signalizaci otáčení do opačného směru je použita LED 4.

8. ZÁVĚR

V uvedené práci byla prozkoumána oblast automobilových sběrnic se zvláštním zaměřením na sběrnici CAN, přičemž zde byly zmíněny i další významné protokoly, jako jsou LIN, MOST, Ethernet a FlexRay. Tato rešerše poskytla ucelený přehled o těchto komunikačních systémech a jejích využití v automobilovém průmyslu. Dále byl v práci předložen ucelený popis datových toků a struktur rámců zpráv LIN a CAN.

Studiem vlastností, možností a omezení těchto komunikačních protokolů si tato rešerše kladla za cíl podat ucelený přehled o jejích úloze v moderních vozidlech.

Jako doplňující prvek k teoretické rešerši byla do studie začleněna první praktická část. Byl vytvořen model v prostředí Matlab Simulink, který společně s převodníkem Vector VN 1630A umožňoval komunikaci po sběrnici CAN. Tento simulační model sloužil jako testovací prostředí pro získávání a analýzu dat ze snímače otáčení volantu automobilu. Kromě toho byla také získána reálná data přímo z vozu skrze konektor OBD2, což dále zvýšilo praktický přínos práce.

Poslední část práce zahrnovala vytvoření vlastního zařízení. Vlastní zařízení využívalo mikrokontrolér Nucleo F446RE pro čtení a přenos dat prostřednictvím sběrnice CAN. Tato praktická realizace ukázala proveditelnost a účinnost vybrané sběrnice pro komunikaci v reálném čase v automobilových systémech.

Celkově lze říci, že tato práce úspěšně spojila teoretický výzkum, simulace a praktickou implementaci. Tím přispěla k pochopení a využití automobilových sběrnic. Výsledky této studie nejen prohlubují znalosti o automobilových komunikačních systémech, ale také poskytují pevný základ pro budoucí studium v této oblasti.

Vyvinuté praktické zařízení slouží jako hmatatelná ukázka teoretických konceptů zkoumaných v celé práci a zdůrazňuje potenciál pro reálné aplikace v automobilových systémech.

POUŽITÁ LITERATURA

[1] Trends in automotive communication systems. *IEEE* [online]. 2005(6), 1204–1222. [cit. 2022-12-10]. Dostupné z: https://www.academia.edu/1515382/Trends_in_Automotive_Communication_Systems.

[2] Fewer wires, lighter cars. *IEEE* [online]. 2013(12). [cit. 2022-12-10]. Dostupné z: https://theinstitute. ieee.org/benefits/standards/fewer-wires-lighter-cars.

[3] H. Kimm and H. Ham, Integrated fault tolerant system for automotive bus networks. IEEE [online]. 2010, 486-490. [cit. 2022-12-10]. Dostupné z: https://dl.acm.org/doi/abs/10.1109/ICCEA.2010.100.

[4] Mark du Preez and others. *Pinoutguide.com*. [online]. [cit. 2022-12-10]. Dostupné z: https://pinoutguide.com/CarElectronics/car_obd2_pinout.shtml.

[5] Sutorý Tomáš. LIN. *Elektrorevue.cz*. [online]. [cit. 2022-12-10]. Dostupné z: http://www.elektrorevue.cz/clanky/04012/index.html?fbclid=IwAR0684WR5UFM ZWSObTXbfBz03GTbBokDaifc9JtVG0TRkHisxfqSBh6yso8.

[6] Hacklet Eric. *LIN Protocol and Physical Layer Requirements*. [online]. [cit. 2022-12-10]. Dostupné z: https://www.ti.com/lit/an/slla383a/slla383a.pdf?ts=1669021913135.

[7] Taraba, Radek. *Aplikování sběrnice CAN*. HW.cz. [online]. 2004 [cit. 2022-11-01]. Dostupné z: http://www.hw.cz/navrh-obvodu/rozhrani/aplikovani-sbernice-can.html

[8] Robert Bosch GmbH: *CAN Specification 2.0* [online]. 1991, [cit. 2022- 11-05]. Dostupné z: http://www.kvaser.com/software/7330130980914/ V1/can2spec.pdf

[9] Semenec, Pavel a Lubomír Novák. *Funkce protokolu CAN*. *Simple CAN Node* [online]. [cit. 2022-12-10]. 2002. Dostupné z: http://simple_can_node.sweb.cz.

[10] VOSS, Wilifried. *A comprehensible guide to controller area network*. Greenfield: Cooperhill Technologies Corporation, 2005. ISBN 978-0976511601.

[11] Ulsoy, A. G.; Peng, H.; Cakmakci, M. *Automotive Control Systems*. 2012. Cambridge University Press, ISBN 978-1-107-01011-6, [cit. 2022-12-10]

[12] Randt Michael. *CAN (Controller Area Network)*. [online]. [cit. 2022-12-10]. Dostupné z: http://www.carbussystems.com/pdfs/CAN.pdf.

[13] Polák, Karel. *Sběrnice CAN*. Elektrorevue [online]. 2003 [cit. 2022-11-01]. Dostupné z: http://www.elektrorevue.cz/clanky/03021/index.html.

[14] Microchip Technology, *CAN module*. Section 23. 2007 [online]. [cit. 2022-11-12] Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/70070D.pdf.

[15] Ricardo De Andrade and col., *Analytical and Experimental Performance Evaliations Evaluations of CAN-FD Bus.* [online]. [cit. 2022-11-12] Dostupné z: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8338047.

[16] Chin-Long Wey and col., *Enhancement of Controller Area Network (CAN) bus arbitration mechanism.* [online]. [cit. 2022-11-12] Dostupné z: https://ieeexplore.ieee.org/document/6799923.

[17] Yelizaveta Burakova, Bill Hass, Leif Millar, *Truck Hacking: An Experimental Analysis of the SAE J1939 Standard.* [online]. [cit. 2022-11-12] Dostupné z: https://www.usenix.org/system/files/conference/woot16/woot16-paper-burakova.pdf.

[18] Pal-Stefan Murvay, Bogdan Groza. *Security Shortcomings and Countermeasures for the SAE J1939.* [online]. [cit. 2022-11-12] Dostupné z: https://ieeexplore.ieee.org/document/8263125

[19] Rajkumar Singh Rathore and col. In-Vehicle Comunication Cyber Secuerity. [online]. [cit. 2022-11-12] Dostupné z: https://www.mdpi.com/1424-8220/22/17/6679

[20] Robert Shaw, Brendon Jackman. *An Introduction to FlexRay as an industrial network.* [online]. [cit. 2022-11-12] Dostupné z: https://ieeexplore.ieee.org/abstract/document/4676987?casa_token=i-v-qu4exo0AAAAA:W7u4IoRA3FdIK-DLhvJL4EbTKkMfRczuxDL9zr2LKuhSAg3z28hO_rf2LQbPujUx0KUeHVyVPg8.

[21] Falch Martin. *CAN Bus explained.* [online]. [cit. 2023-2-2] Dostupné z: https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial

[22] Louč Tomáš. Automatické monitorování sběrnice CAN v automobilech. [online]. [cit. 2023-2-2] Dostupné z: https://dspace.tul.cz/bitstream/handle/15240/17015/DP_2014_Tomas_Louc.pdf?sequence=1

[23] Schade Frank, Muth Matthias. Fault Tolerant CAN transceiver. [online]. [cit. 2023-2-2] Dostupné z: https://www.nxp.com/docs/en/application-note/AH0801.pdf