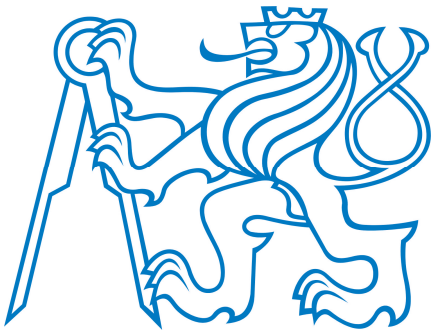
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta strojní – Ústav přístrojové a řídicí techniky



BAKALÁŘSKÁ PRÁCE

AUTOMOBILOVÉ KOMUNIKAČNÍ STANDARDY

AUTOMOTIVE COMUNICATION STANDARDS

Denisa Kaisrlíková

2022

Prohlašuji, že jsem tuto práci vypracovala samostatně s použitím literárních zdrojů a informací, které cituji a uvádím v seznamu použité literatury a zdrojů.

Datum: . . . .

Podpis

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kaisrlíková** Jméno: **Denisa** Osobní číslo: **493566**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídící techniky**

Studijní program: **Teoretický základ strojního inženýrství**

Studijní obor: **bez oboru**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Automobilové komunikační standardy

Název bakalářské práce anglicky:

Communication standards in Automotive

Pokyny pro vypracování:

1) Vypracujte rešerši na téma používaných komunikačních standardů v automobilovém průmyslu.
2) Seznamte se s programováním mikrokontrolerů od společnosti STMicroelectronics. Jako vývojovou desku zvolte např. NUCLEO-F767ZI.
3) Na základě rešerše naprogramujte odesílání zpráv po sběrnici CAN mezi 2 zařízeními, např. typu NUCLEO-F767ZI.

Seznam doporučené literatury:

[1] KIM, Dong-Seong a Hoa TRAN-DANG. Industrial Sensors and Controls in Communication Networks: From Wired Technologies to Cloud Computing and the Internet of Things. 1. Cham, Switzerland: Springer Nature Switzerland, 2019. ISBN 978-3-030-04927-0.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Zdeněk Novák, Ph.D. U12110.1

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **29.04.2022** Termín odevzdání bakalářské práce: **09.06.2022**

Platnost zadání bakalářské práce: _____

Ing. Zdeněk Novák, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	doc. Ing. Miroslav Španiel, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Studentka bere na vědomí, že je povinna vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ Datum převzetí zadání

_____ Podpis studentky

Abstrakt

Bakalářská práce se zabývá automobilovými komunikačními standardy. V teoretické části je uveden popis fyzické a linkové vrstvy sběrnice a její základní princip fungování. Dále je popsána zpráva sběrnice CAN a její části. Na konci této části jsou popsány další používané protokoly a jejich základní vlastnosti a využití. V praktické části je podrobný popis prostředí, ve kterém je vyvíjen kód pro odesílání a příjem zpráv pomocí sběrnice CAN. V závěru jsou zobrazeny probíhající zprávy mezi zařízeními, pomocí odposlouchávacího softwaru.

Klíčová slova: CAN, sběrnice, komunikace, zpráva, mikrokontrolér, vývojová deska

Abstract

The bachelor thesis deals with automotive comunication standards. The theoretical part describes the physical and layer line of the bus and its basic principle of operation. Than the CAN bus message and its parts is described. At the end of this section, other protocols used and their basic properties and uses are described. The practical part is a detailed description of the development environment in which the code for sending and receiving messages using the CAN bus is developed. Finally, ongoing messages between devices are displayed, using eavesdropping software.

Keywords: CAN, bus, communication, message, microcontroller, development board

Poděkování

Chtěla bych tímto poděkovat Ing. Zdeňku Novákovi, Ph.D. za odborné vedení, cenné rady a trpělivost, kterou mi v průběhu zpracování bakalářské práce věnoval.

Dále děkuji mé rodině za podporu při psaní této práce a při studiu.

Obsah

1 Úvod

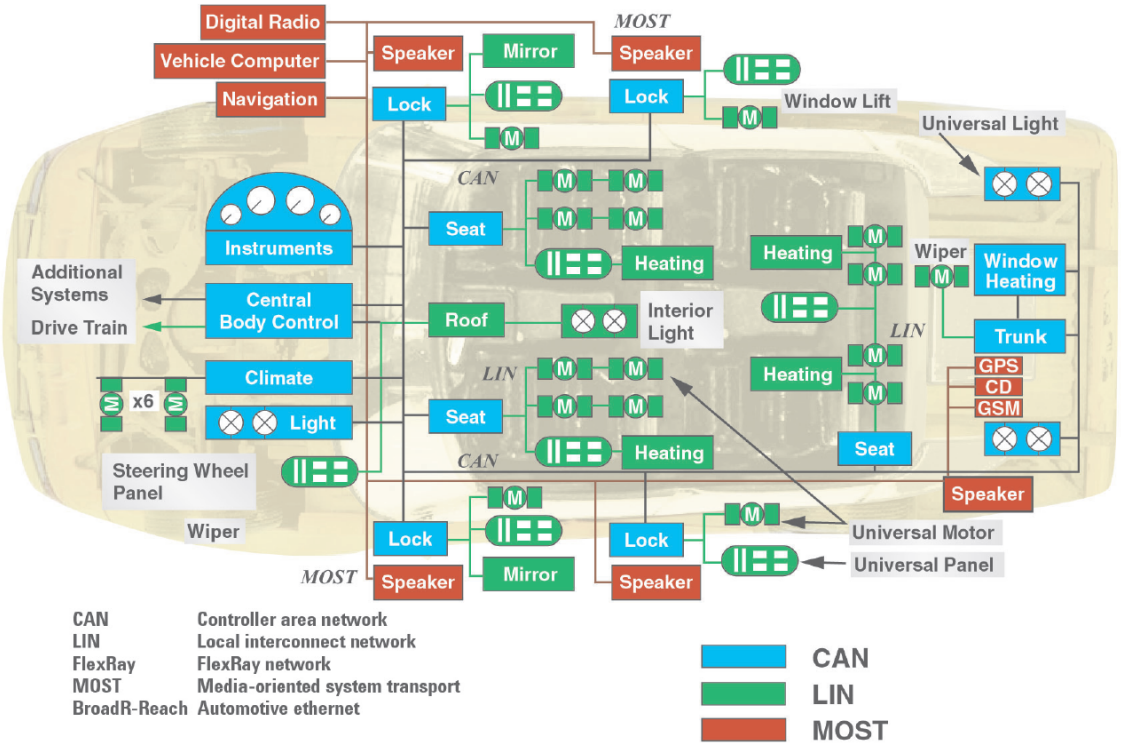
V dnešní moderní době najdeme okolo sebe spousty senzorů, řídících jednotek a dalších komponentů, které mezi sebou potřebují komunikovat. Jejich komunikace je důležitá k jejich vzájemnému fungování a správnému načasování. Je velice důležité, abychom se na tyto systémy mohli spoléhat a důvěřovat jim, jelikož nás všude obklopují a mnohdy na nich závisí i náš život. Vzájemné předávání zpráv zprostředkovávají zejména kabely - vodiče. Každý materiál a jejich zpracování přináší rozdílné možnosti a kvalitu přenosu. Dále je tento faktor ovlivněn použitou technologií výroby. V minulém století se ke komunikaci v automobilovém průmyslu používaly sériové sběrnice. Byli schopny předávat informace po jednotlivých bitech. Každá dráha byla opatřena svým vlastním vodičem. Dnes se tento způsob přenosu dat uplatňuje v počítačových sítích. Zde je toto řešení praktičtější. V automobilovém průmyslu došlo k inovaci a byla vyvinuta paralelní sběrnice, která byla schopna přenášet několik bitů najednou. Začali se utvářet datové rámce a došlo k vývoji převaděčů a komunikátorů, které jsou schopny detekovat např.: původ zprávy, její určení, obsah či její celistvost a zabezpečení. Toto řešení se v praxi jeví efektivně v oblastech, kde je potřeba komunikace mezi mnoha prvky a jejich komunikace je náročná na objem přenášených dat. Dnes, v době průmyslové automatizace se s těmito sběrnicemi lze setkat téměř na každém rohu. Vyskytují se také v lodním průmyslu, jsou součástí výtahů, lékařského vybavení či v domácích spotřebičích jako jsou pračky nebo sušičky. V průmyslu se díky těmto sběrnicím automatizují výrobní linky, které jsou schopny mezi sebou komunikovat a tím pádem dochází ke zefektivnění výroby. Dále tyto komunikační sběrnice lze nalézt i v leteckém průmyslu. Ve většině odvětví se vychází z primárně navržené sběrnice CAN. Její modifikací vznikají komunikační protokoly pro specifická odvětví, která mají své konkrétní nároky a sběrnice je jim přizpůsobena. Například v letectví se užívá sběrnice CANaerospace.

Pokud se zaměříme na automobilový průmysl. V každém dopravním prostředku ať už v osobním či nákladním autě nebo v motorce nalezneme datovou sběrnici paralelního typu. Byli primárně vyvinuty pro automobilový průmysl. Firma Roberta Bosche na konci osmdesátých let představila datovou sběrnici CAN. Další automobilky se poté snažily vynalézt své originální sběrnice. To ale v mnoha případech selhalo. Dnes, se zvyšujícími se nároky na bezpečnost, jízdní komfort či šetření paliva a úspoře emisí, roste počet řídících jednotek ve voze. Jednotky sbírají data z několika čidel ve voze. V minulosti se ke komunikaci využíval konvenční přenos dat skrz analogové kabely vedoucí napětí či proud. Každá cesta mezi jednotkami musela být opatřena jedinečným kabelem. V praxi toto řešení vede ke zvýšení hmotnosti, výrobních nákladů a nepřehlednosti kabeláže. Došlo tedy ke snížení hmotnosti a snížení nákladů. Další výhodou těchto sběrnic je jejich snadná dostupnost. Jsou plně centralizované, což znamená jeden vstupní bod do celé komunikační sítě. Zefektivněním celé páteřní sítě bylo vyvinutí datové sběrnice, která propojuje několik jednotek současně. Není zde omezení na určitý čas vyslání zprávy a řídící jednotka tedy

může odeslat zprávu v jakýkoliv čas. K zamezení přetížení sítě se používá řízení hodnoty zprávy pomocí prioritního identifikátoru. Datové sběrnice také zajišťují vysokou míru bezpečnosti, která je zde velice žádoucí. Zprávy obsahují pole s kontrolou celistvosti a správného přijetí zprávy.

V automobilových komunikačních standardech využíváme mnoho odvozených jednotek. Bit je základní jednotka dat. Nabývá hodnoty 0 nebo 1. Ty mohou interpretovat hodnoty, nebo jsou často používané jako označení pravdy a nepravdy nebo pro označení stavu zapnuto/vypnuto. Dále se také využívají násobné jednotky kilobite nebo megabite. V řádu těchto jednotek se pohybují parametry sítí. Pokud seskupíme 8 bitů, označuje se tím jednotka byte. Také se lze setkat s odvozenými jednotkami jako je bit/s. Pokud bit vydělíme časem, získáme jednotku pro popis rychlosti přenosu dat. V tomto případě jednotka udává rychlost přenosu dat na sběrnici.

Schéma propojení řídích jednotek a senzorů v autě je uvedeno na obrázku č.:1. Modré názvy a linie spojují prvky pomocí CAN sběrnice. Jsou to členy základní výbavy, například klimatizace nebo výhřev zadního okna. Dále je v tomto voze užita sběrnice MOST, která je díky své konstrukci vhodná pro použití v komunikačních a informačních technologiích vozu. Najde využití pro palubní počítač či navigaci. Nejlevnějším řešením je sběrnice LIN. Používá se pro prvky, které nevyžadují vysoké rychlosti přenosu a jsou to většinou prvky komfortní výbavy. Například naklápění zpětných zrcátek či topení.



Obr. 1: Schéma propojení komunikačních uzlů v automobilu[5].

2 Teoretická část

V dnešní době lze na trhu sehnat mnoho typů datových sběrnic které se liší svojí konstrukcí, výrobní technologii či rozdílným přenosem dat. Některé sběrnice jsou lépe uzpůsobené k přenosu audio či video signálu a tím zajišťují vysoký komfort výstupních dat. Některé sběrnice jsou vyvinuty pro zajištění rychlého a přesného doručení zprávy v reálném čase. S vývojem hybridů poptávka po sběrnicích roste. Nové pohonné jednotky v elektromobilech vyžadují větších přenosových rychlostí. Dalšími prvky jsou například měniče proudu DC/DC nebo prodlužovače výdrže baterií. Díky těmto pokročilým technologiím dochází k vývoji sofistikovanějších datových sběrnic. Komunikaci sběrnice vysvětlím na základní sběrnici CAN.

2.1 Datová sběrnice CAN

Sběrnice CAN (Controller Area Network) je tvořena dvěma vodiči označovanými CAN_H a CAN_L. Obvykle je tvořena měděnými dráty chráněnými PVC obalem. Propletené vodiče mezi sebou nevytvářejí velké elektromagnetické pole, které se následně chová jako rušení. Dále jsou na obou koncích rezistory 120 Ω k zamezení rušení a případnému odražení zprávy. Oba vodiče jsou propojeny s CAN Transceiver, CAN Controller a mikrokontrolérem. Tyto součásti dohromady tvoří uzel sítě. Transceiver, česky budič má dva piny, kterými je připojen ke sběrnici. Toto provedení je z důvodu symetrie signálu. Lze rozlišit dva druhy, vysokorychlostní a nízkorychlostní. Při výběru jsou důležitými parametry celková rychlost sběrnice a požadavek na zabezpečení sítě. Nízkorychlostní se vykazuje vyšším zabezpečením. Pokud nastane porušení jedné z linek, nedojde k úplné ztrátě komunikace. Jeho úkolem je převádět logický signál do diferenciální podoby. Takto zpracovanou informaci předá Controlleru. Dále slouží jako ochrana před elektrostatickým výbojem nebo elektrickým předpětím. Funguje na principu rozlišení dvou logických stavů sběrnice - recesivní a dominantní. Can Controller, neboli česky řadič čte a odesílá zprávy. Může být integrovaný v mikropočítači či mikrokontroleru. Filtruje přijaté zprávy tak, aby přijímal pouze ty, pro něj určené. Dále obsahuje paměťové registry k uchování zpráv. Důležitá je také časová kompaktnost systému, kterou určuje časová synchronizace celé sběrnice. Komunikace probíhá v protokolu multi-master. Není tedy určen hlavní uzel řídící přístup na sběrnici, ale jednotlivé uzly se mohou na sběrnici připojovat libovolně. V případě kolize je na sběrnici odeslán rámec s nejvyšší priroritou. Aby tento systém správně fungoval, je nutné celou sběrnici časově synchronizovat, neboli nastavit její vhodnou arbitráž. V případě špatného načasování by mohlo dojít ke zúžení průchodnosti pásma na sběrnici nebo k poškození rámců. Jednotlivé zprávy jsou přijímány všemi uzly, a ty je podle identifikátoru ignorují nebo zpracovávají. Norma ISO 11898 definuje datovou sběrnici CAN a její parametry. Nízkorychlostní sběrnice jsou velice odolné chybám, přenos dat probíhá rychlostí maximálně 125kb/s. Využívají jednodušší kabeláž a lze je najít v ovládacích prvcích displeje

palubní desky či v ovládání oken. Vysokorychlostní se liší ve vyšší přenosové rychlosti až 1 Mbit/s. S tím jsou spojeny i vyšší požadavky na aktualizaci systému a vysokou přesnost dat. Najdeme jej například v airbagových jednotkách nebo v protiblokovacích brzdných systémech. Dále lze dle normy rozdělit sběrnici na vrstvu fyzickou a linkovou. Norma také definuje 4 základní typy rámců zpráv. Každou zprávu utváří řídící jednotka. Aby nedošlo k jejímu znehodnocení či poškození, je definován jednotný rámec. K sběrnicím se lze externě připojit pomocí konektoru OBD2. Ten je většinou ve voze umístěn na spodní straně palubní desky u řidiče. Nebo případně na jiném, dobře přístupném místě. Pomocí tohoto portu se může kdokoliv se správným hardwarem a softwarem připojit k vozu a provést jeho diagnostiku či analýzu vad. Port se vyskytuje v šesti, devíti nebo šestnácti pinové konfiguraci. [6, 13]

Pokud potřebujeme sledovat informace o stavu vozidla nebo například průmyslové linky v reálném čase, lze využít zaznamenávání dat v CAN systému. Tato data jsou shromažďována prostřednictvím portu OBD2, pomocí kterého se lze připojit na sběrnici a komunikovat s jednotkami ve vozidle. Dnes je na trhu mnoho typů datových záznamníků do vozidel, přenosných či vestavěných. Tyto skříňky také v autech označujeme jako blackboxy. Data se ukládají přímo do přenosného záznamníku, nebo na SD kartu, kterou lze následně vyjmout. Například záznamník TMU CAN-FD nahrává data na SD kartu, dále tyto data lze kódovat pomocí softwaru. Tento typ má i zabudovanou WiFi a data lze tedy automaticky nahrávat na vlastní server. Toto řešení ocení firmy s velkým vozovým parkem. Pomocí těchto informací lze provádět analýzy jízdních stylů personálu nebo provádět plány na údržbu vozového parku.[6, 13]

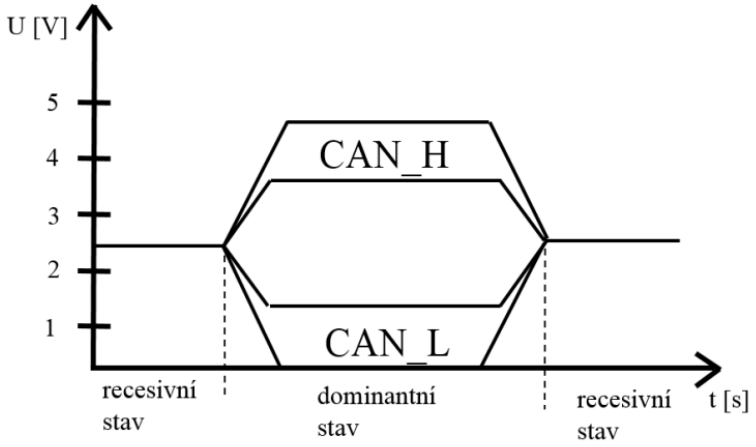
2.1.1 Fyzická vrstva

Digitální signál využívá ke komunikaci binární soustavu. Čtecí zařízení rozpoznává pouze 0 a 1. Logický stav 0 se považuje za aktivní (dominantní) a logická 1 se označuje jako pasivní (recesivní) stav. Primární funkcí systému je operace logický součin. Pokud všechny uzly soustavy vysílají pouze recesivní bity, celkový stav sběrnice označujeme jako recesivní. Začne-li vysílat nějaký uzel dominantní bity, změní se celá sběrnice na dominantní. Recesivní bit nastane, pokud napětí na CAN_H je menší než na CAN_L a dominantní bit naopak. Pokud je rozdíl napětí na sběrnici menší než 0,5 V, jedná se o recesivní bit. V opačném případě, že rozdíl je větší než 0,9 V, jedná se o dominantní bit. Tento rozdíl lze vypočítat také rovnicí:

$$V_{diff} = V_{CANH} - V_{CANL} \tag{1}$$

Dominantní stav na vodiči CAN_L je definován normou ISO 11898 v rozsahu 0 až 1,5V, na vodiči CAN_H v rozsahu 3,5 až 5V. Z obrázku č.2 lze vyčíst napěťové stavy pro recesivní

a dominantní stavy. Rychlost přenášeného signálu je závislá na délce vodiče. Standartní rychlost je 1Mbit/s pro délku 40 m. Přenosová rychlost se dá měnit změnou frekvence. Ta je ale omezena frekvencí procesoru. S rostoucí délkou sběrnice klesá rychlost přenášeného signálu. [7, 8]



Obr. 2: Schéma datové sběrnice v recesivním a dominantním stavu[7].

2.1.2 Linková vrstva

Tato vrstva se dále rozděluje na CAN_Controller a CAN_Transceiver. Funkce CAN Controlleru neboli MAC (Medium Acces Control) je kódování dat, detekce priorit zpráv a jejich chyb. Druhá vrstva označovaná také jako LLC (Logical Link Control) rozesílá zprávy získané od MAC a dále také přijímá zprávy ze sběrnice a vyhodnocuje případné přetížení sítě. Celý proces přenosu zprávy začíná v mikropočítači řídící jednotky, který zprávu vytvoří. Dále je předána MAC, který zprávu zpracuje a předá následně LLC, která zprávu dále distribuuje na sběrnici CAN. Příjem zprávy probíhá analogicky, nejdříve zprávu přijme jednotka LLC, která ji předá MAC pro další zpracování, až do výsledné podoby, kterou zpracuje mikropočítač. Aby se zamezilo kolizi mezi jednotlivými zprávami z různých uzlů zaslaných v jeden okamžik a síť se nepřetěžovala, využívá se CSMA/CS+AMP (Carrier Sense Multiple Acces with Colision Detektion and Arbitration on Message Priority). Jedná se o nedestruktivní arbitráž s prioritou danou identifikátorem. Jde o zavedení identifikátoru zprávy, která informuje jednotku o prioritě zprávy. Pokud je zpráva důležitá, přiřadí jednotka ID nejvyšší priority a ta bude přednostně odeslána na sběrnici. V případě, že jiná jednotka ve stejný okamžik vysílá zprávu s nižším stupněm priority, přeruší vysílání a pokusí se o opětovné odeslání později.

Bitová arbitráž rozděluje každý bit přenesený na sběrnici do 4 základních segmentů. První je synchronizační segment. Jedná se o dobu, kdy dojde k přeměně z recesivní do dominantní pozice či naopak. Druhý segment kompenzuje fyzické zpoždění sítě. Jedná

se o zhruba dvojnásobný součet zpoždění šíření sběrnice, zpoždění vstupního převodu a zpoždění výstupního budiče. Třetí a čtvrté pole je určeno pro resynchronizaci. Časová arbitráž nazývaná jako tvrdá, následuje po každé nečinnosti sběrnice či po přijetí začátku rámce. Jedná se o restartování vnitřního bitového časování. Principem je, aby se hrana bitu z recesivního do dominantního stavu sešla přesně s dobou, kdy přijímač očekává hranu bitu. Tato resynchornizace se provádí i na všech ostatních takto přecházejících bitech. [7, 8]

2.1.3 Síťové chyby

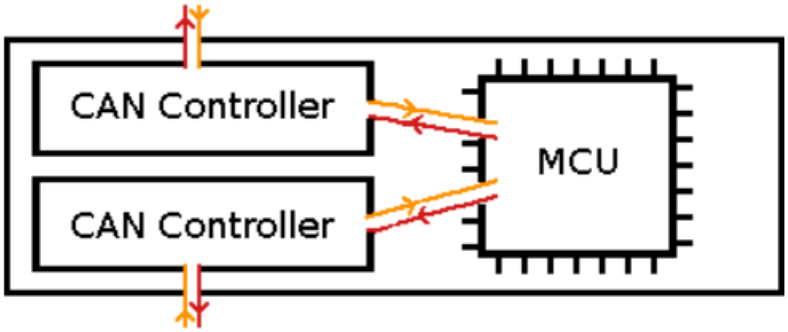
Na sběrnici může dojít k několika typům síťových chyb. Chybu může odhalit vysílací uzel. Pokud tento uzel zachytí jinou hodnotu bitu, než tu, kterou poslal, ohlásí chybu. Reakce jednotlivých systému na tuto chybu se může lišit dle její povahy. Další chybou je porušení pravidla skladby bitů. V případě, že se ve zprávě objeví série pěti po sobě jdoucích bitů stejné hodnoty, musí následně být vložen bit o opačné hodnotě. Pokud by toto pravidlo nebylo dodrženo, dojde k porušení čitelnosti zprávy. Následující chyba se nazývá chyba kontroly cyklické redundance (CRC). Tato chyba nastane v případě, že přijímací uzel přijme jinou sekvenci CRC, než očekával. CRC sekvence je vypočítávána z datového pole. Tudíž, pokud by došlo k narušení skladby bitů, jednotka by špatně vypočítala CRC, a tím se také zjistí chyba ve skladbě bitů. Pokud tento případ nastane, dojde k chybě formuláře. Nakonec chybovost ještě potvrdí ACK oblast. Vysílající uzel tedy detekuje chybové doručení zprávy. V případě, že nějaký uzel detekuje chybu sběrnice, vyšle chybový rámec. Následně bude sběrnice resetována a dojde k obnovení normální komunikace na sběrnici. Aby docházelo k omezení chyb sítě, obsahuje každý uzel čítače. CAN protokol požaduje, aby čítače byly separované na příjímací a odesílací. Při detekci chyby, čítač změní svou hodnotu o 1 nebo 8, v závislosti na typu chyby a obklopujících podmínkách. Hodnota na příjímacím čítači se zvyšuje jen tehdy, pokud dojde k chybě v přijímané zprávě. Analogicky to funguje i pro odesílané zprávy. Pokud čítač překročí hodnotu 127, tento uzel je prohlášen za pasivní s chybou. Tímto aktem dojde k omezení odesílání chybových dominantních rámců do sítě. Při překročení hodnoty 255 je uzel odstřihnut od sítě a nemůže odesílat jakákoliv data na sběrnici. Pokud ale dojde k přijetí či odeslání zprávy bez chyby, čítač vždy sníží svou hodnotu o 1. Tímto způsobem se opět vrací do aktivního stavu. I samotná síť má svůj čítač. Udržuje síť v konstantním režimu. Pokud průměrně vypočítá osm správně přijatých nebo odeslaných zpráv na každou chybu vzniklou přijetím či odesláním zprávy s chybou. Monitoring sběrnice pomocí uzlů se označuje jako globální hlídání sběrnice. Naproti tomu lokální hlídání chyb je samotná kontrola přepočtem CRC. Další mechanismus pro zabránění chyb v síti a zvýšení použitelnosti sběrnice, je detekce a vypínání vadných síťových uzlů. Díky tomu nedochází k narušování přenosu zpráv. Každý uzel, jak už bylo řečeno, obsahuje čítač a podle něj operuje s chybami. Načte tedy hodnotu chyby, ale zároveň také uvědomí sběrnici a všechny ostatní uzle odesláním chybového rámce. Ten se skládá ze šesti po sobě

jdoucích dominantních bitů. Důvodem použití šesti dominantních bitů je prosté. Jak již bylo zmíněno, existuje chyba skladby bitů. Pokud tedy máme za sebou šest dominantních bitů, signalizuje to chybu. Po tomto zjištění uzly problematickou zprávu vymažou.[20]

2.1.4 Další CAN prvky

CAN brána Zařízení, které se používá jako rozhraní mezi různými sítěmi a umožňuje připojení dalších sítí CAN. Tyto sítě mohou mít různé přenosové rychlosti a protokoly. Lze ji ale použít i pro integraci s jinou sítí, jako například automobilový ethernet. Na obrázku č.:3 lze vidět strukturu brány, neboli CAN Gateway. Brána obsahuje dva řadiče CAN a mikrokontrolér. Zprávy jsou přijímány řadičem a následně vyhodnocovány mikrokontrolérem. Odesílány jsou druhým řadičem. Vyhodnocení zprávy znamená filtrování, přemapování identifikátorů zprávy nebo změna datového pole zprávy. S touto změnou poté lze změnit i přenosovou rychlost zprávy. Tímto převodem a možnou změnou zpráv se brána liší od CAN opakovače. Je tedy flexibilnější a komunikace mezi sítěmi může být efektivnější. Samozřejmě, ale s tím také rostou náklady celé sběrnice. Naopak problémem při technickém využití brány je doba latence pro odeslání přijaté zprávy na druhou stranu. Dochází tím ke zpoždění zprávy. Pokud je síť v nečinném stavu, tato doba je stanovena jako doba šíření brány. Mohou ale nastat dva stavy, kdy čas zpoždění nelze určit. První případ je, pokud je sběrnice zatížena pouze na jedné straně. Zpoždění se zvyšuje, jelikož mikrokontrolér opakovaně musí přerušovat příjem a odeslání zpráv do doby, než budou procesorem zpracovány. Nastane-li situace, že je sběrnice zatížena z obou stran, řadič musí vyčkat na okamžik, kdy mu bude umožněn přístup ke sběrnici. Tato doba je dále ovlivněna zatížením sběrnice a výší priority zprávy. Výhoda propojení dvou sběrnice CAN/CAN je, že lze využít plnou délku sběrnice na obou stranách pro danou přenosovou rychlost a použít všechny moduly. Brány CAN/CAN se používají v případě, pokud se vyžaduje řízený tok zpráv. Další praktické využití nalezneme v situaci, kdy je potřeba připojit další ECU jednotku do sítě. ECU jednotky v systému mohou používat určité identifikátory, které jsou odlišné od přiřazených identifikátorů nové přidané jednotky ECU. Nejefektivnějším řešením, je tedy zapojení CAN/CAN brány, která ID jednotlivých zpráv dokáže přetransformovat, aby byli použitelné pro nově přidanou jednotku.[20]

CAN opakovač CAN opakovač plní stejnou funkci jako brána. Slouží k propojení dvou sítí. Skládá se z dvou CAN řadičů z vlastních logických obvodů. Opakovač tedy funguje jako kus kabelu propojující dvě strany. I zde dochází ke zpoždění přenosu. Tato doba je určena ekvivalentní délkou pro daný konkrétní opakovač. Při použití opakovače bez galvanického oddělení, je to 40 m a pro zařízení s galvanickým oddělovačem 60 m. Opakovačem nelze realizovat prodloužení sběrnice. Naopak, je možné díky němu měnit topologii zapojení sítě z linkové, například na hvězdicové zapojení. Při vhodném nastavení topologie, lze snížit délku vedení a tím snížit i náklady. Pokud zapojíme do sítě CAN opakovač, rozdělí se síť

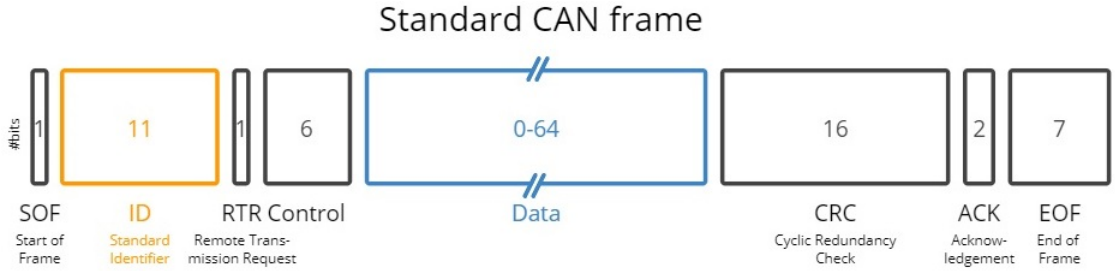


Obr. 3: Schéma CAN brány[20].

na dva segmenty. Tyto segmenty jsou fyzicky nezávislé, ale jsou považovány za jednu síť. Důvodem použití opakovače je regenerace signálu pro dlouhé CAN sběrnice, nebo jejich prostřednictvím lze zvýšit počet možných připojených uzlů k síti. Oproti bráně také dále šíří chybové rámce. Další funkcí je odpojení celého segmentu. Lze jej tedy uzamknout, a zvýšit tím bezpečnost rozlehlé sítě. Dalším využitím je galvanické oddělení dvou sítí. Jedná se o rozdělení obvodu na dva obvody, které nejsou spojeny žádným vodivým prvkem. Přesto mezi nimi dochází k přenosu energie. Většinou je galvanické oddělení řešeno pomocí relé. Na jednom obvodu máme připojenou cívku a na druhém je spínací kotva připojená ke kontaktům. Protéká-li cívkou proud, kotva je přitažena a tím uzavře obvod. Dochází k přenosu informací. [20]

2.1.5 Datový rámec

Datové zprávy se rozdělují dle délky identifikátoru na CAN 2.0A 11bitový ID a CAN 2.0B 29 bitový ID. Zpráva se skládá z několika částí, které definuje použitý rámec. Schéma zprávy je zobrazeno na obrázku č.:4. Rozdíl mezi CAN typu A a B je pouze v dodatečném 18 bitovém identifikátoru CAN typu B.



Obr. 4: Schéma datového rámce sběrnice CAN[4].

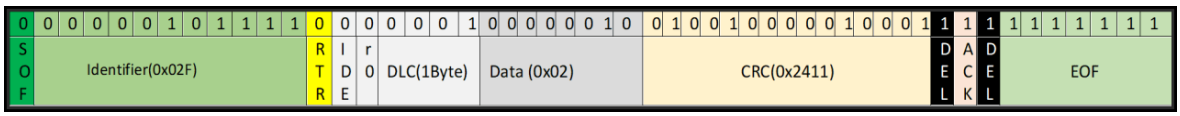
SOF a ID První pole (SOF) označuje začátek zprávy a je zde dominantní bit. V protokolu CAN je ID vyhrazeno 11 polí. Tyto pole slouží pro identifikaci zprávy. Jakému uzlu je zpráva výhradně určena a jak vysokou prioritu má. Se zmenšujícím číslem roste priorita. Dominantnější zpráva tedy obsahuje dominantní bit dříve v identifikátoru.

RTR a Control RTR nese informace, zda se jedná o datový rámec (dominantní) nebo o žádost o vysílání (recesivní) a řídí přístup na sběrnici. Je zde vyhrazeno pole o 7 bitech. Z něhož 1 bit je určený pro identifikaci zda jde o datový rámec a dalších 6 bitů je vyhrazeno pro rámec Control. Oblast řídící informace určuje počet přenášených bytů ve zprávě (0 až 8) a obsahuje 2 rezervní bity. Důležité je nastavení délky kódu. Musí být stejná jako délka očekávané zprávy. [3]

DLC a CRC Datová oblast (Data) je místo pro přenos zprávy. V základním CANu je místo pro 8 bytovou zprávu, u CAN_FD lze přenášet zprávu o velikosti až 64 bytů. Oblast CRC kontrolních 16 bitů. Jeho úloha je zabezpečení celé komunikace prostřednictvím polynomu CRC. V tomto poli je uložen výsledek kontrolního součtu přenesených bitů. Tento úkon provede vysílač. Přijímač využívá stejného polynomu pro výpočet kontrolního součtu. Tyto dvě hodnoty se porovnají a pokud se shodují, celý datový rámec je přijat. V opačném případě je zaslán chybový rámec. [1]

ACK a EOF Potvrzující pole sestává ze dvou bitů. První vyšle vysílač jako recesivní, pokud uzel přijme zprávu, změní tento bit na dominantní, čímž oznámí vysílači bezchybné přijetí zprávy. Druhý je opět recesivní s funkcí oddělovače. Následuje mezilehlé pole označující 7 recesivních bitů. V tomto momentě mohou přijímací uzly informovat vysílací uzel o možných chybách přenosu. Toto pole je ukončovací a značí konec zprávy.[2]

Na obrázku č.:5 je praktický příklad zprávy v protokolu CAN. První pole SOF je startovací pole s hodnotou 0. Následuje identifikátor o délce 11 polí nebo jej lze zapsat také jako 0x02F. Dále leží informace RTR, v tomto případě vidíme, že se jedná o datový rámec. Vedle tohoto pole se nachází dva dominantní bity, které nesou informaci o délce zprávy a dále se zde nachází kontrolní bit. Oblast DLC a data. Zde je uložena informace zprávy. Vedle se nachází pole CRC, tedy kontrolní pole. Následují dva rezervní bity, mezi kterými leží pole ACK. Zpráva je zakončena polem EOF.



Obr. 5: Příklad zprávy v CAN protokolu[24].

2.1.6 Žádost o rámec

Skládá se ze stejných polí jako datový rámec, avšak RTR je v tomto případě recesivní a chybí datová oblast. Uzel tohoto rámce může využít k navázání kontaktu s jiným uzlem a následnému odeslání datového rámce. Řídící jednotka v obou rámcích nastaví identifikátor dotazovaného uzlu.

2.1.7 Chybový rámec

Pokud nějaký uzel vyhodnotí chybu zprávy, může přerušit vysílání této zprávy. Vyšle rámec sestávající z 6 dominantních a 8 následujících recesivních bitů, které informují vysílač o chybovosti.

2.1.8 Rámec přetížení sítě

Uzel tímto rámcem informuje, že je ve stavu zpracování předešlé zprávy a žádá o pozdější odeslání další zprávy. Rámec je podobný chybovému.

2.1.9 Časování bitů

Každý bit je časově rozdělen na 4 doby trvání neboli segmenty. Rozdělení je důležité pro synchronizaci přenášených dat na sběrnici. Synchronizační segment zajišťuje synchronizaci času. V tento čas se objeví hrana bitu. V propagačním segmentu probíhá kompenzace časového zpoždění na sběrnici. Fázové segmenty slouží ke kompenzaci fázových chyb. Mezi těmito segmenty leží vzorkovací bod. Pokud dojde k fázové chybě, využije se metody doplnění bitů opačné polarity.

2.2 Datová sběrnice CAN_FD

Jedná se o vylepšenou sběrnici CAN. Aplikuje se v případech, kde je potřeba pracovat s velkým množstvím dat. Díky své vyšší přenosové rychlosti až 8Mbit/s a vyšší propustnosti dokáže pracovat s objemnými daty. Přenáší 64bytová data místo 8bytových dat, které využijí hlavně náročné aplikace. Používá se pro informační systémy a pro pokročilé asistenční systémy, dále v průmyslové automatizaci či nákladní a autobusové dopravě. Tato sběrnice prošla také modifikací zabezpečení. Obdobně jako CAN obsahuje CRC část, která zabezpečuje správné obsazení rámce. Disponuje dalšími 3 bity. Detekuje chyby zkrácení či prodloužení. Zatímco v klasickém CANu obsahuje CRC 0 až 3 bity, ve vylepšeném CANu jsou vždy 4 bity fixované pro vylepšení spolehlivosti vzájemné komunikace. Je zcela kompatibilní se standartními CAN sběrnicemi. Pokud se v síti nachází oba typy sběrnice CAN, řídící jednotka je schopna si regulovat přenosové rychlosti a vybírat vhodnou velikost zprávy na základě požadavků.[2]

2.3 Vyšší protokoly

V nákladní dopravě bylo vyvinuto několik sběrnic vyšší úrovně běžících po fyzické vrstvě CAN. Jedná se o sběrnice, které mají své speciální funkce a označují se také jako nadprotokoly. Byli vytvořeny v USA, kde je velice rozvinutá těžkotonážní nákladní

automobilová doprava a je potřeba zajistit také komunikaci s jednotkami, které se v osobních vozidlech neuplatňují.

Protokol SAE J1939 Jedná se o komunikační standard používaný v nákladních automobilech, automobilech či těžké hydraulice. Je to protokol vyšší vrstvy postaven na sběrnici CAN. Fyzická vrstva se označuje J1939/11 a definuje elektrické rozhraní sběrnice. Linková vrstva J1939/21 definuje konstrukci zprávy, přístup uzlů ke sběrnici a detekuje chyby přenosu. Aplikační vrstva J1939/71 a J1939/73 se zabývá obsahem dané zprávy a těmito specifickými daty. Od klasické CAN sběrnice se liší prioritně tím, že většina zpráv je pouze vysílána na sběrnice bez určení konkrétního cílového uzlu. Tento přenos umožňuje přístup všem uzlům k těmto datům a ty s nimi mohou pracovat. Eliminuje se tím zatížení sběrnice zprávami s žádostí o přístup na sběrnici či žádostí o data. Lze ale i zprávu adresovat konkrétnímu uzlu použitím speciálních znaků v identifikátoru.

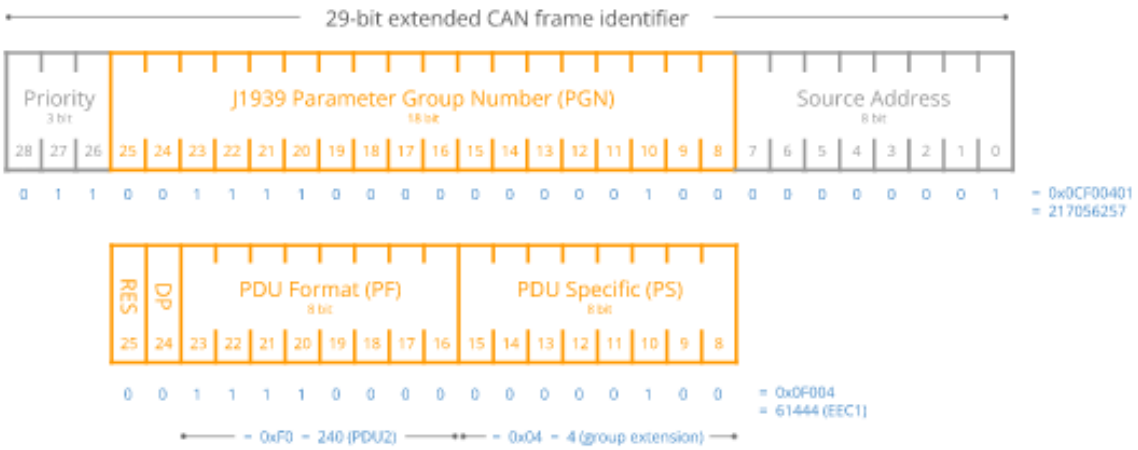
Protokol J1939 používá CAN 2.0B s 29bitovým identifikátorem. Bity v polích SOF a ID jsou definovány stejně jako u CAN sběrnice. Bit RTR je vždy u tohoto protokolu nastaven na hodnotu 0. Struktura 29bitového identifikátoru je následující. První 3 bity určují prioritu zprávy během synchronizace sběrnice, přičemž hodnota 0 má nejvyšší prioritu. Zpravidla se jedná o zprávy s řízení motoru, jako je například řízení vysokých otáček motoru. Nižší priorita se přiřazuje zprávám neobsahující data o rychlosti vozidla. Další bit má hodnotu 0 a je vyhrazen pro budoucí použití. Následuje bit zvaný selektor datové schránky. Jedná se o bit, kterým může být rozšířena parametrová skupina. Parametrová skupina čísel označovaná jako PNG je souhrn bitů rezervy, PF a PS. Pole nazývané PDU (PF) určuje, zda je zpráva konkrétně cílená, či se jedná o zprávu určenou všem. Tomuto poli je vyhrazeno 8 bitů. Další pole PDU (PS) obsahuje také 8 bitů a jeho tvar je určen polem PF. Pokud je PF na hodnotě 0 až 239, je zpráva odeslána na konkrétní nebo globální adresu. Obsahuje v sobě tedy informaci o konkrétní adrese. Pokud je PF v hodnotě 240 až 255 znamená to odeslání zprávy globálně. Posledních 8 bitů obsahuje adresu vysílajícího uzlu. Tato adresa je každému zařízení přiřazena a skrz ní je také poskytován přístup uzlu ke sběrnici. Každá adresa je tedy jedinečná. Maximální počet adres je 254. V tomto protokolu je každé ECU jednotce přiřazeno jméno (Name) skládající se z 64 bitů. Jméno jednotky se skládá z 10 polí: libovolný bit adresy, průmyslová skupina (3 bity), konkrétní systém vozidla (4 bity), obecný systém vozidla (7 bitů), rezervní bit, funkce (8 bitů), konkrétní funkce (5 bitů), konkrétní jednotka ECU (3 bity), kód výrobce (11 bitů) a v posledním poli je obsaženo identifikační číslo (21 bitů).

Hlavním účelem přiřazení jména každé jednotce je její popis. Dále se také používá pro rozeznání priority jednotky ECU. Každé zařízení v síti je spojeno s jedním jménem a jednou adresou. Toto pravidlo lze porušit. V rámci jedné ECU jednotky mohou být i stejné adresy pro dvě zařízení. Například pokud jsou dvě zařízení připojena stejným fyzickým

místem, adresa je stejná. Proto je důležité přiřadit jednotce také jméno, které identifikuje funkci zařízení a pokud v síti existuje více těchto zařízení, přiřadí mu i jedinečnou instanci. V jedné síti může tedy koexistovat pouze 254 zařízení. Adresy 0 až 127 jsou přiřazeny standartním funkcím, vyskytujícím se ve většině vozidel. 128 až 254 jsou závislé na individuálním provedení daného vozidla a vzájemně se tedy mohou lišit. Odlišnost si lze představit připojením přívěsu. Jelikož přiřazování adresy probíhá dynamicky, šetří toto základní rozdělení čas hledání a přidělení adresy. Adresa 255 je rezervována pro všesměrové vysílání neboli je nazývaná jako globální adresa. Adresa 254 se používá jako nulová a slouží pro komunikaci zařízení, která nemají svou vlastní přiřazenou adresu. Proces přiřazení adresy je následovný. Každé zařízení musí oznámit přiřazenou adresu. Získání adresy lze dvěma způsoby. Zařízení odešle zprávu s žádostí o přiřazení určité adresy. Pokud tuto určitou adresu nárokuje zařízení s vyšší prioritou, automaticky oznámí zařízení s nižší prioritou, že tato adresa je již použitá. Nebo odešle žádost o reklamaci adresy. Tato metoda se využije v případě, pokud se zařízení připojí později, jelikož už získá tabulku zabraných adres a nárokuje tedy pouze volnou adresu. Určitý řád v identifikaci zařízení zapojených v síti koriguje norma J1939/71, ve které jsou uvedeny preferované adresy jednotlivých zařízeních.

Odeslání zprávy je jednoduché. Dojde k vytvoření parametrů zprávy definovaných protokolem J1939/71. Jde například o rychlost přenosu či parametry obsahu zprávy. Pokud je nějaký parametr nulový, tento byte se nastaví na hodnotu 0xFF. Je to z důvodu, aby došlo k rozpoznání, zda se hodnota parametru ztratila, nebo je opravdu nulová. V případě zpráv, které mají větší objem dat než 8 bytů, se informace zabalí do jednotlivých paketů (definováno protokolem J1939/21). Má-li zpráva přiřazenou cílovou adresu a zařízení ji přijme, musí vysílači potvrdit správné přijetí zprávy. Jedná-li se o zprávu globální, každá jednotka zprávu zpracuje. To, zda už je obsah pro danou jednotku relevantní, již vyhodnocuje sama. Komunikace probíhá rychlostí 250kB/s. Nyní vysvětlím části konkrétní zprávy na příkladu. Schéma zprávy je vidět na obrázku č.:6. Ukázkové ID je: 0x0CF00401. Interpretace zprávy: 0C představují, prioritu zprávy a PGN. F004 je konkrétnější specifikace PGN a lze jej dohledat v normě J1939/21. F0 popisuje PF pole a 04 popisuje PS pole. 01 označuje adresu vysílače. Datová zpráva může být například v tomto formátu: 0xFF FF 82 DF 1A FF FF FF. Tato zpráva je délky 8 bytů. 1, 2, 6, 7 a 8 byt má nulovou hodnotu a je označen 0xFF. Byt 3 má hodnotu 0x82, což v tomto případě uvádí skutečné procento točivého momentu motoru. Bajty 4 a 5 reprezentují parametry rychlosti motoru. [14]

Protokol CANopen Jedná se o vyšší komunikační protokol CAN. Díky své široké konfiguraci jej lze využít v mnoha průmyslových odvětvích. Protokol je definován dle CiA DS 301. Každému objektu je přiřazen jeden či více identifikátorů, který stanovuje mimo jiné i prioritu zařízení. Pro zjednodušení návrhu a lepší orientaci se používají již přidělené ID adresy všem nezbytným objektům. Tento proces probíhá v době konfigurace sítě. V případě nutnosti lze adresy těmto objektům dynamicky měnit. Každý objekt je popsán ve

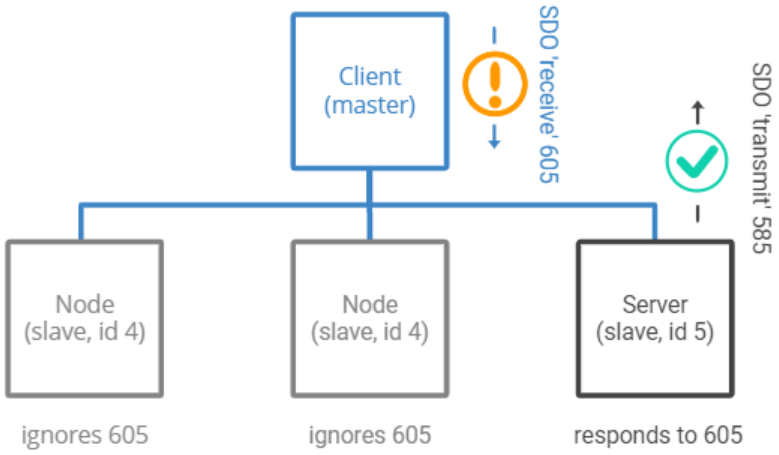


Obr. 6: Schéma zprávy protokolu J1939[15].

slovníku objektů, který je uložen v zařízení. Zde lze zjistit vlastnosti a funkce jednotlivých objektů. Tento seznam slouží jako rozhraní mezi zařízením a aplikačním programem. S každým objektem lze komunikovat skrz SDO zprávy (Service Data Objects). Provádějí čtení a zapisování dat do slovníku objektů. Přiřadí tedy objektu šestnáctibitový index. Tyto zprávy nemají stanovenou délku. Pokud nesou data objemnější než byty, data jsou rozdělená do několika CAN zpráv neboli bloků. První segment obsahuje data potřebná pro komunikaci a identifikace správného doručení segmentů. Dále obsahuje indexy objektů, které jsou uvedeny ve zprávě. Následuje zde rezervované pole pro přenos dalších uživatelských dat. Další segmenty obsahují pole s daty potřebných pro komunikaci a pole o velikosti 7 bytů, které je určené pro zapisování uživatelských dat. Přijímací uzel musí po správném přijetí zprávy odeslat informaci o stavu přijetí zprávy. Identifikátor se skládá ze dvou částí. Funkční část určuje prioritu objektu. Část označená jako Node ID nabývá hodnot 0 až 127 slouží k rozlišení dvou zařízení plnících stejnou funkci. Pokud je zpráva určena všem uzlům v síti, je Node ID na hodnotě 0.

V síti se nastaví funkce NMT Control Object, která poskytuje prostředky pro řízení podřízených zařízení v síti. Nulový identifikátor určuje tuto zprávu jako nejvíce prioritní. Hlavní řídící uzel sleduje v určitých časových intervalech stav sběrnice a stav podřízených zařízení. Odesílá jim datové dotazy, kterými zjišťuje aktivitu jednotlivých objektů. Objekt odpoví zprávou, obsahující data o jeho stavu. Na konci se používá toggle bit, který se při každém odeslání zprávy mění. Tím se stává zpráva věrohodná pro aktuální dotaz. Zároveň ale probíhá i zpětná kontrola od podřízených uzlů. Nedostane-li uzel dotaz na jeho stav do určité doby, ohlásí tuto skutečnost aplikačnímu programu. Jednotlivá zařízení mohou být v určitých stavech. Po zapnutí se objekt nachází ve stavu inicializace. Po dokončení operace automaticky přejde do stavu před operačního, kdy s ním lze komunikovat skrz SDO. Jakmile mu je vyslána zpráva NMT správcem sítě, přejde do operačního stavu. V tomto stavu může vysílat zprávy. K synchronizaci celé sítě slouží zpráva SYNC. Jedná se o zprávu periodicky

vysílanou většinou řídícím uzlem. Tato doba je buď daná ve slovníku nebo ji při přípravě na provoz lze měnit dle potřeby. Neobsahuje ID s nejvyšší prioritou a může tedy v systému dojít ke zpoždění. Aktuální čas a datum je určen časovou značkou. Na tyto značky objekty nevysílají odpověď. Pokud v síti dojde k nějaké chybě, je vyslána zpráva Emergency. Jsou ji přiřazeny vyšší hodnoty ID. Zařízení má v sobě uložen registr chyb. Dle chybové zprávy, která má v sobě vymezenou oblast pro bližší specifikaci chyby, lze lépe diagnostikovat typ erroru v síti. Každá zpráva s daty se nazývá PDO a má jedinečný identifikátor. Též musí být vysílána pouze jediným uzlem. Počet uzlů, které jí přijmou už je libovolný. Důvody odeslání zprávy mohou být různé, například pokud byl vznesen požadavek na určitá data nebo pokud byla přijata synchronizační zpráva. Rozhraní mezi sběrnicí a aplikačním programem je řešeno prostřednictvím USB slotu. V USB portu je implementován USB-CAN převodník, sloužící k ladění CAN aplikací a jejich diagnostiku. Na obrázku č.:7 lze vidět proces, kdy hlavní jednotka odeslala zprávu 605 pro objekt s identifikátorem 5. Ostatní uzly, kterým není určena tato zpráva, ji ignorují. Pouze podřízená jednotka s ID 5 ji přijme a vyšle opětovně globální zprávu o přečtení. [18]



Obr. 7: Schéma procesu odeslání a přijetí zprávy v protokolu CANopen[19].

2.4 Datová sběrnice MOST

Jedná se o alternativní sběrnici pro multimediální přenos dat. Byla vyvinuta společenstvím automobilek zvaným MOST Cooperation. Liší se především přenosem signálu pomocí plastových optických vláken. Je to vysokorychlostní multimediální síťová technologie využívaná pro přenos audio, video, hlasových signálů. Komunikaci zajišťuje softwarový ovladač nazývaný MOST Network Services. Je složen ze základních systémových vrstev a služeb aplikačního soketu. Tato sběrnice je schopna obsloužit až 64 zařízení, zapojených v nějaké konfiguraci. Její výhodou je funkce Plug and Play, která umožňuje snadné připojování a odpojování obsluhovaných zařízení. Sběrnice se vykytuje v mnoha

topologických provedeních jako jsou hvězda, prstenec, kruh či kaskádová hvězda. Pokud je třeba zvýšit bezpečnost pro daný okruh, lze využít redundantní dvojkruhové topologie. V prstenci se využívá časová synchronizace. V okruhu se nachází jednotka, s funkcí hlavního časovače, která vysílá rámce. Ty jsou následně zachyceny sledovači časování. Informaci, že se jedná o synchronizaci okruhu nese začátek rámce zvaný preambule, díky kterému dojde k synchronizaci. Dnes ale tato sběrnice čelí tlaku od Ethernetové sběrnice, a dá se očekávat, že ji v budoucnu nahradí. Se sběrnicemi se lze setkat ve 3 typových variantách. Sběrnice MOST25 poskytuje přenos 25 Mb/s. Lze skrz ní, přenášet data synchronně i asynchronně. Disponuje šedesáti kanály, které si uživatel může seskupovat po čtyřech bytech. Dokáže přenášet zvukové stopy v CD nebo MPEG-1 kvalitě, využitím až patnácti kanálů. Jejich výhodou je dokonalé odizolování proti elektromagnetickému poli a tím nedochází k rušení. Nepracuje v protokolu multi-master a zpráva je odeslána pouze přijímací jednotce. Sběrnice MOST50 disponuje dvojnásobnou šířkou pásma. Komunikuje rychlostí až 50Mb/s. Tato sběrnice podporuje přenos přes optické ale i elektrické fyzikální vrstvy. V dnešní době jsou ale podporovány pouze elektrické přenosy. Využívají tří měděných vodičů. Dále je připojena přídavná řídící linka ke každému uzlu v paralelním uskupení. Celkově obsahuje 5 linek. Jedna řídící linka obsluhuje data odeslané zařízením označeným Master. Další dvě jsou určeny pro vstup dat a poslední dvě pro výstup dat a odeslání do okruhu. Sběrnice MOST150 zavádí ethernetovou fyzickou vrstvu v automobilech. Velikost rámce je až 3072 bitů. Lze nastavovat šířku pásma a poskytuje izochronní přenos. Díky své šířce se jedná o multiplexní síťovou sběrnici schopnou přenášet jakékoliv formy infotainmentu.[9]

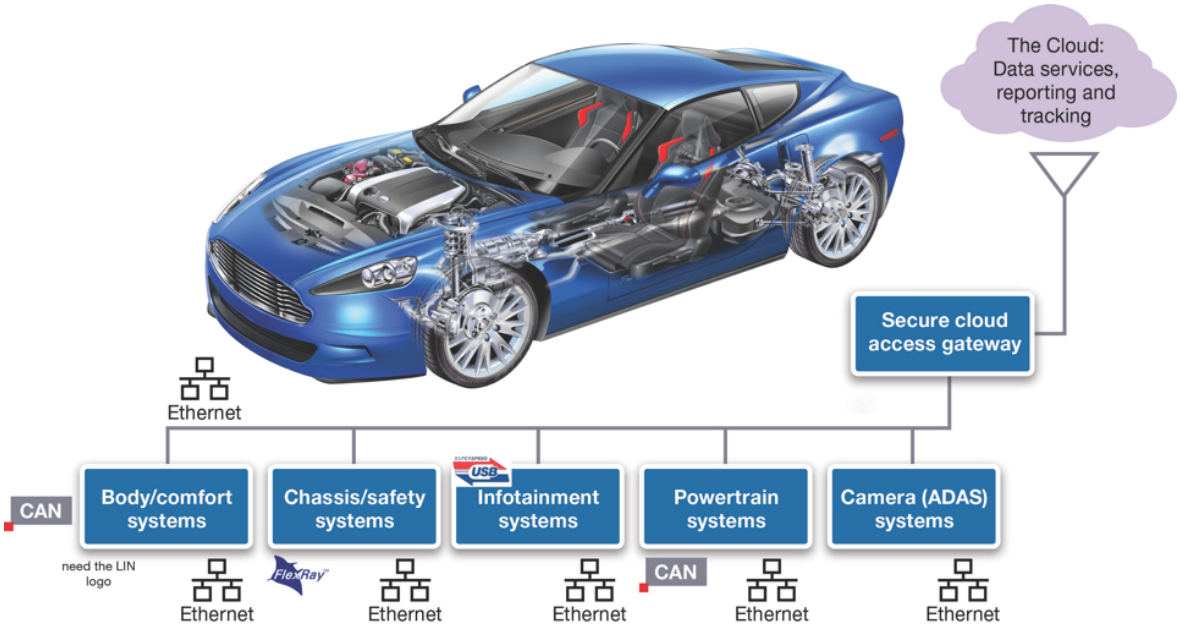
2.5 Automobilový Ethernet

Se zvýšenou poptávkou po autonomních automobilech rostou i nároky na datové sběrnice. Kamery umístěné v exteriéru vozu nebo jednotka zajišťující komunikaci mezi automobily, má již tak vysoké nároky, které by ostatní sběrnice těžko splňovaly. Rychlostním průkopníkem v přenosu dat se v roce v 2011 stal Ethernet. V tomto roce vznikla nezisková organizace Open Alliance, tvořená firmami NXP, Broadcom a BMW. Vyvinuli speciální ethernetovou fyzickou vrstvu zvanou BR, která je vhodná pro využití v automobilovém průmyslu. Na obrázku č.:8 je vidět schéma využití ethernetové sběrnice ve vozidle. Standart BroadR-Reach, zkráceně BR, je tvořen jednou kroucenou dvoulinkou. Jedná se o 15m UTP měděný kabel s dvěma konektory na obou koncích. V případě delšího spojení dvou uzlů, a to více než 15 m, napojí se pomocí spojek další segment UTP kabelu. Opět s maximální délkou 15 m. Impedance je 90-110 Ω. Konektory využívané v automobilovém průmyslu se nazývají MDI. Vyskytují se ve dvou pinové i multipinové konfiguraci. Klasický ethernetový kabel nelze využít ve vozidle, jelikož se skládá ze čtyř párů kroucené dvoulinky. Je tedy dražší a těžší v porovnání s ostatními sběrnicemi. Hlavním problémem je však nesplnění

EMC kompatibility. Každé zařízení, obsahující elektrické či mechanické součásti, musí být dostatečně odolné elektromagnetickému rušení, aby byla zaručena jeho spolehlivost a správná funkčnost. Pokud dojde ke zúžení pásma BR na 33,3 MHz, stane se vyhovujícím EMC požadavkům. Dále dochází ke snížení rizika přeslechu a zlepšuje návratovou ztrátu. Pomocí funkce echo cancellation, umí rozpoznávat a filtrovat odražený signál. Disponuje také Full-duplexním přenosem, což znamená, že využívá obousměrný přenos dat po dvou vodičích. Lze dosáhnout rychlosti přenosu až 100 Mb/s. Umožňuje vytvoření jediné stupňovité sítě místo jednotlivých uzavřených sítí. Lze tedy implementovat více elektrických systémů a zařízení. Fyzická vrstva se dále dělí na několik podvrstev. Podvrstva PCS přenáší data mezi standartním blokem řízení (označovaným MII) a podvrstvou PMA. Podvrstva PMA vysílá či přijímá signály mezi PCS a kroucenou dvoulinkou. V podvrstvě PCS je využito kódování 4B3B. Převádí data skládající se ze 4 bitů, vysílající se s frekvencí 25 MHz na data 3 bitová, s vysílající frekvencí 33,3 MHz. Dále se takto získaná data pomocí scramleru přeskupí do tzv. ternárních symbolů. Tyto symboly se spárují do dvojic a následně dojde ke zvýšení vysílající frekvence na 66,6 MHz. Metoda scramble se používá pro snížení počtu oblastí, ve kterých se vyskytuje více stejných znaků za sebou. Mohlo by tedy dojít k porušení zprávy. Druhá podvrstva zajišťuje pulzně amplitudovou modulaci signálu 3. úrovně. Díky ní je také dosaženo zúžení pásma na 33,3MHz a tím zajištěna podmínka EMC kompatibility. Celá tato síť funguje v postavení Master. Dojde tedy k nastavení hlavního řídícího a uzlu a jeho poddaných. Tato konfigurace ale není pevně stanovena, a funkci Master lze přiřadit libovolnému uzlu.

Stejně jako u ostatních sběrnic je důležité mít veškerá data v jedné časové konfiguraci. U Ethernetu se využívá protokolu PTP (Precision Time Protocol) ve verzi 2 standardizovaném IEEE 1588-2008. Tento systém je schopen generovat časové značky či časově řadit události dle okamžiku udání. Hlavní referenční čas nastavuje uzel s funkcí Master. Ostatní poddané uzly se synchronizují výměnou speciálních časových rámců. Existuje několik typů PTP hodin, které lze použít. Ordinary Clock je typ, který dokáže udávat referenční čas a nebo pracuje v režimu synchronizace dle hlavního času. Jedná se o zařízení s jedním portem. Boundary clock pracuje obdobně, jen má k dispozici více portů. Pokud je v systému zařazeno více prvků v řadě a dochází tím k časovému zpoždění, lze použít hodiny Transparent Clock. Ty měří pouze dobu, kdy rámec skrz ně prochází. Tento čas předají koncovým hodinám, které díky tomu dokáží přesnější synchronizaci. Další inovací, kterou Ethernet přináší je VLAN neboli virtuální lokální síť. Díky tomuto rozhraní lze jednu fyzicky spojitou síť rozdělit do menších celků. Takto vytvořená podsíť se nazývá LAN. Rozdělení probíhá nastavením na přepínači. Pokud je potřeba změnit hierarchii sítě, stačí pouze překonfigurovat přepínač a tím se daný uzel připojí do jiné podsítě. Celý tento systém přispívá k bezpečnosti řízení automobilu. Potencionální útočník se díky tomuto virtuálnímu rozdělení nemůže připojit skrz jednu síť k celému systému. Další výhodou je snížení počtu sběrnic. Nemusíme od každé vzdálenější součásti vést samostatnou sběrnici až do vyhodnocovací jednotky.

Ale lze ji připojit k bližší sběrnici jiného senzoru správné LAN sítě. Přes jeho vedení se informace dostane až do vyhodnocovací centrály. Automobilový Ethernet se používá pro velmi sofistikované a datově náročné aplikace. Uplatnění nachází i v přenosu audio a video signálu. Díky své přenosové rychlosti a šířkou pásma dokáže přenášet data ve vysoké rychlosti a objemu. Následný zvuk či video je pak kvalitnější než při použití jiné sběrnice. Je zřejmé, že s rostoucí poptávkou po bezpečnosti a autonomních vozidlech se bude v automobilech využívat častěji. Dnes se s ním tedy lze setkat spíše u vyšších řad automobilů. [17]

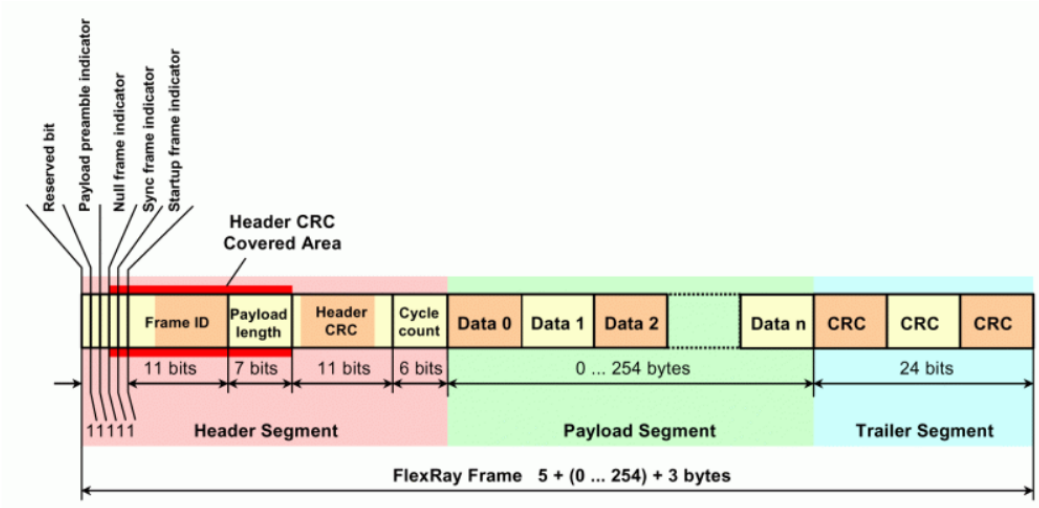


Obr. 8: Schéma použití Ethernetu ve vozidle[5].

2.6 Datová sběrnice FlexRay

Je spojení CAN a MOST sběrnice a využívá se pro dynamické automobilové aplikace. Byla vytvořena v roce 2005 koncorsiem FlexRay a dnes je standardizována dle normy ISO 17458-1 až 17458-5. Dokáže komunikovat i se sběrnicí CAN. Využívá větší průchodnosti a rychlosti. Tou se blíží ethernetovým implementacím. Dále je odolnější vlivům elektromagnetického rušení. Je složena ze dvou kanálových struktur označovaných A a B. Kanál B je využit pouze v případě poruchy kanálu A a slouží tedy jako záloha. Každý kanál disponuje přenosovou rychlostí 10 MBit/s. Pokud využijeme oba kanály, lze rychlost zvýšit na dvojnásobek. Vyskytuje se obdobně v několika typologiích. Identifikace, komu je zpráva určena je řízena dvěma identifikátory, což zjednodušuje komunikaci na sběrnici. Obdobně jako u sběrnice MOST je využita časová synchronizace celé sběrnice, která umožňuje zavádění nových koncepcí jako je metoda TDMA. Ta vymezuje kontrolovaný přístup ke sběrnici všem uzlům. Každý uzel má přesně definovanou časovou pozici, která zajišťuje správné přiřazení časové hodnoty k odeslání této zprávy. Fyzická vrstva se skládá

ze dvou nestíněných kabelů. Jsou definovány čtyři stavy vrstvy. Pokud je na sběrnici napětí 0 V, signalizuje se tím nízkoenergetický nečinný stav všech uzlů. V případě zvýšení napětí na hodnotu 2,5 V znamená, že vrstva se nachází v nečinném stavu. Ve stavu dominantní úrovně dat Data 0 je na jednom pinu napětí 3,5 V a na druhém 1,5 V. V případě dominantní úrovně Data 1 se tyto napětí na pinech prohodí. Rámec sběrnice FlexRay je tvořen třemi hlavními částmi (viz obrázek č.9). Hlavička obsahuje 40 bitů. První bit je rezervovaný, následuje bit preambule, nulový indikátor, synchronní identifikátor a startovní identifikátor. Dále pokračují bloky složené z několika bitů. Následuje oblast ID, která zahrnuje 11 bitů sloužících k identifikaci odesílatele. Payload délka udává velikost odesílané zprávy, která může být o velikosti maximálně 254 bajtů. Následuje CRC oblast, která má stejnou funkci jako u sběrnice CAN. Poslední pole hlavičky je složeno ze šesti bitů, které definují cykly, ve kterých je zpráva odesílána. Druhou významnou části je Payload. Slouží pro zapsání samotné zprávy. Poslední částí je Trailer. Opět oblast CRC, tentokráte ale sofistikovanější, zabezpečuje správné doručení zprávy a obsahuje 24 bitů. [10]

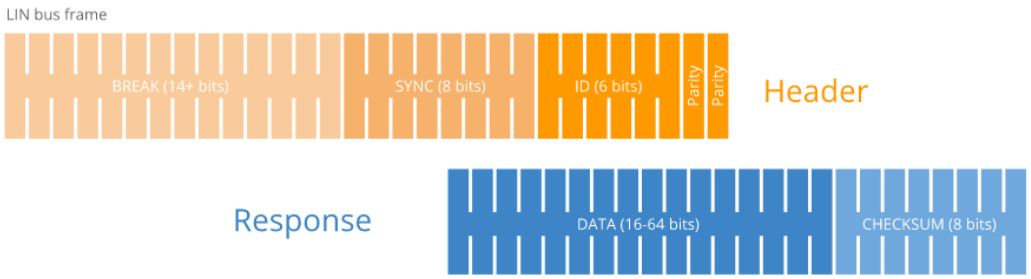


Obr. 9: Schéma datového rámce FlexRay technologie[11].

2.7 Datová sběrnice LIN Bus

Jedná se o levnější alternativu CAN. Byla vyvinuta za účelem snížení ceny. Je to pouze jednolinková sběrnice. Používá se pro řízení prvků komfortní výbavy. Zde se nevyžaduje práce s velkým množstvím dat a přenos dat probíhá maximálně rychlostí 20kbit/s. Tato sběrnice pracuje jako single master. Jedno zařízení vykonává funkci kontroly sběrnice a ostatních zařízení. Může obsluhovat maximálně 16 uzlů. Vyslání a přijímání dat provádí mikroprocesor, který nemusí mít velký výkon, pouze musí mít funkci řadiče sériové komunikace. Dále je potřeba systém opatřit LIN Transceiverem s rozmezím 0 až 12 V. Převod signálu provádí budič. Řešení odrazu signálu lze realizovat pouze pomocí RC

oscilátoru, což také vede ke snížení výrobních nákladů. Rámec LIN je složen ze dvou hlavních polí (viz obrázek č.10). Hlavička zprávy a odpověď na zprávu. Hlavička obsahuje 34 bitů. První podpole, nazývané synchronizační pauza (Break), zajišťuje identifikaci zprávy odkud je vyslána. Většinou se jedná o 13 recesivních bitů a 1 volitelného bitu a více. Dále máme synchronizační pole (Sync). Ovládá časovou synchronizaci celé sítě, aby se všechny uzly sjednotili. Provádí se vždy před přijetím zprávy. Referenčním bodem kalibrace se užívá rezonátor a obsahuje 8 bitů. Identifikační podpole (ID) definuje délku zprávy a obsahuje unikátní identifikační kód. Skládá se ze šesti bitů. Jelikož vedlejší uzly nejsou vybaveny funkcí pro čtení ID, jsou jednotlivé ID adresy definovány a uloženy v paměti. Následují dvě pole párových bitů (Parity). Druhé hlavní pole je tvořeno datovým polem (Data) přenášejícím obsah zprávy délky 2, 4 nebo až 8 bytů a kontrolním součtem (Checksum). Ten tvoří 8 bitů. Slouží k ověření doručení celé zprávy. [10]



Obr. 10: Schéma datového rámce LIN sběrnice[12].

2.8 Mikroprocesor

K přenosu zprávy slouží sběrnice. Jaké typy a parametry sběrnice mohou mít a jaké požadavky musí splňovat bylo vysvětleno výše. K vyhodnocení zprávy a následné reakci na obsažená data slouží většinou mikroprocesor nebo mikrokontrolér. Rozdíl mezi těmito dvěma hardwary je v uložení součástí. U mikrokontroleru jsou všechny součásti umístěny v monolitickém integrovaném obalu, kdežto u mikroprocesoru mohou být některé součásti umístěny mimo hlavní obal a připojeny k základní destičce pomocí PCB spojů. Mikroprocesor je uspořádání sekvenčních logických obvodů, které dokáží provádět jednotlivé instrukce uložené v paměti dle vstupních parametrů. Tímto procesem dojde k vyhodnocení získaných dat, a výsledky interpretuje na výstup. Skládají se z téměř stejných součástí. Rychlý vývoj těchto elektrotechnických součástí proběhl v druhé polovině 20. století. Díky technologickému vývoji integrovaných obvodů, které v sobě obsahovaly veškeré nezbytné součásti k vykonávání požadovaných funkcí. Jedná se především o mnoho tranzistorů. V tomto období vznikly tedy první čipy. S dalším rozvojem začali vznikat pouzdra obsahující více prvků. Dalšími prvky, které byli přidány je například

časovač, A/D převodník či rozhraní CAN a další. V dnešní době na trhu lze najít několik typů mikrokotrolerů. Nejzákladnější a nejjednodušší je 8 bitový mikroprocesor. Jedná se dnes už o nahrazení jednoduchého sekvenčního či kombinovaného obvodu. Výkonnější 16 bitové procesory se objevují v PLC zařízeních. Dnes běžně výkonné procesory jsou 32 nebo 64 bitové. V praktické části budu pracovat konkrétně s 32 bitovým procesorem STMicroelectronics, který bude více specifikován níže. Nejdůležitější parametry, dle kterých se řídí jejich výběr jsou primárně: daná aplikace, počet vstupů a výstupů a podpora daných typů sběrnic. Sekundární paramtery jsou: pracovní frekvence, frekvence systémové sběrnice a velikost cache paměti. Pracovní frekvence je udávána v jednotkách Herz. Frekvence systémové sběrnice udává rychlost komunikace mezi procesorem a operační pamětí RAM. Poslední parametr znamená velikost mezipaměti. Ta hraje roli v získávání často používaných dat. [21]

Mikroprocesor STM32F7 V praktické části budu komunikovat s vývojovou deskou NUCLEO obsahující mikroprocesor STM32. Jedná se tedy o 32 bitové integrované obvody mikroprocesoru od společnosti STMicroelectronics. Obsahující jádro označované ARM s mnoha variantami provedení. Dále obsahuje statickou RAM, flash paměť, ladící rozhraní a další periferie. Označení F7 konkretizuje použitý typ jádra - ARM Cortex-M7F. Maximální taktovací čas jádra je 216 MHz. Flash paměť je o velikosti 2 MB a RAM paměť o velikosti 512kB/s. [22]

Typ	**CAN**	**MOST**	**Ethernet**	**FlexRay**	**LIN**
Rychlost	8 Mbit/s	150 Mbit/s	800 Mbit/s	80 Mbit/s	0.018311 Mbit/s
Sběrnice	2 vodiče	Optické vlákno	BR	2 vodiče	1 vodič
Počet zařízení	64	64		2047	16
Velikost zprávy	512 b	3072 b	12 000 b	2032 b	64 b

3 Praktická část

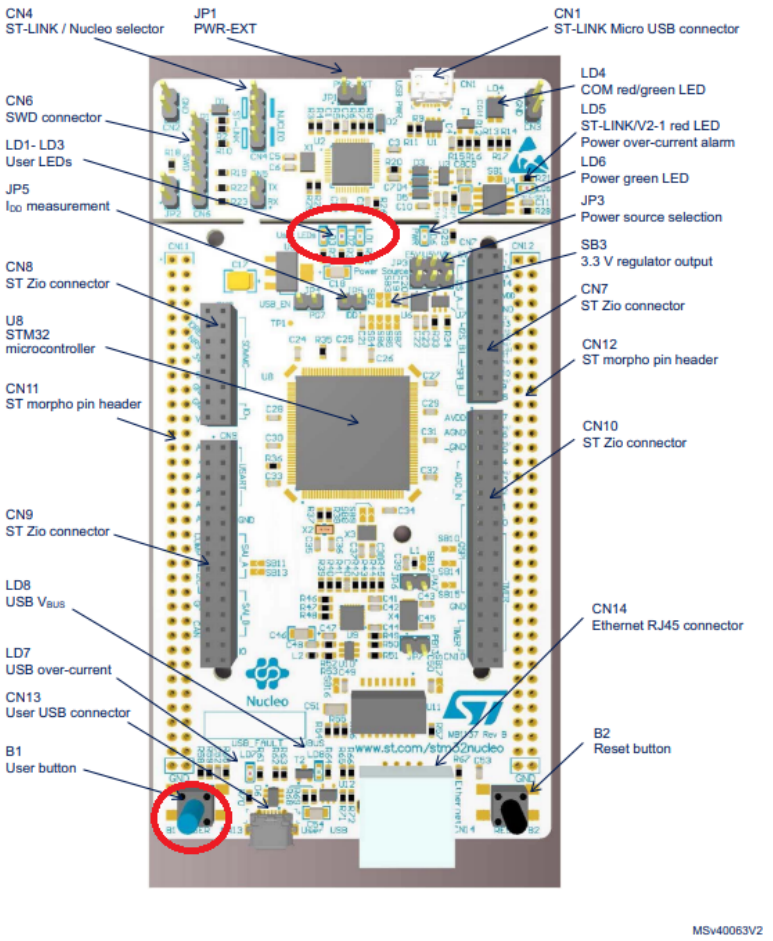
V mé bakalářské práci je komunikace pomocí CAN sběrnice realizována pomocí dvou vývojových desek. Jedná se o vývojové desky NUCLEO STM32 F767ZI osazené mikrokontroléry STM32 od firmy STMicroelektronics. Mezi těmito dvěma komponenty probíhá komunikace. Zařízení si mezi sebou vyměňují zprávu obsahující jakékoliv informace, které nastavíme, aby si zařízení předali. Tato zpráva je však omezena velikostí. Nejdříve bych ráda přiblížila použitou vývojovou desku a popsala komponenty, se kterými v práci pracuji. Dále bych ráda přiblížila použitý software od firmy STMicroelectronics. V tomto sortwaru jsem generovala a vytvořila kód. Ten je složen z několika funkčních částí. Jeho účelem je odeslání a příjem zprávy po sběrnici CAN. Tato zpráva obsahuje informace o blikání diody. Po přečtení zprávy mikrokontrolérem č. 2 vykoná akci dle přijatých parametrů. Následuje odeslání zprávy mikrokontrolérem č. 2 na sběrnici se stejnými parametry, ale odlišnými hodnotami. Tato zpráva je přijata mikrokontrolérem č. 1.

3.1 NUCLEO STM32 F767ZI

Tato vývojová deska disponuje mnoha periferiemi a příslušnými registry. Pro zjednodušení se v práci zabývám pouze periferiemi, které budu využívat.

GPIO (General-purpose I/Os) Tato periférie zprostředkovává vstupy a výstupy. Jedná se v překladu o univerzální vstupní/výstupní pin. Díky jemuž lze obsluhovat potřebné komponenty. Výhodou tohoto typu pinu je, že má přiřazenou specifickou funkci dle manuálu [28], kterou lze nastavit pomocí softwaru dle požadavků uživatele. V mé aplikaci jsem využila již předdefinovaných výstupů: LED diod a uživatelské tlačítko a další piny, které jsou potřebné k zajištění komunikace a synchronizace. Komponenty jsou vyznačeny na obrázku č.: 10. V SW prostředí STM32CubeIde lze zvolit konkrétnější režimy, ve kterém má výstup fungovat. Například lze nastavit rychlost změny hodnoty pinu, nebo režim, v jakém bude probíhající signál vyhodnocován.

RCC (Reset and clock control) Velice důležitá periferie, která nastavuje na ostatních periferiích jednotné hodiny, dle kterých je synchronizována celá komunikace a veškeré funkce mikrokontroléru. STM32 disponuje 2 interními oscilátory, 2 oscilátory pro interní krystal nebo rezonátor, a 3 smyčky fázového závěsu (PLL). Jde o elektronický obvod se stálým napětím nebo s napětím řízeným oscilátorem. A ten nepřetržitě přizpůsobuje frekvenci vstupnímu signálu. Tento typ RCC se používá pro synchronizaci výstupního signálu s referenčním. I v případě, že mají dva signály stejnou frekvenci, není zaručeno dosažení maxima u obou ve stejnou dobu. Je tedy zapotřebí synchronizovat i dosažení vrcholu ve stejnou dobu. Princip úpravy vychází z jednoduchého zpětnovazebního řízení. Systém detekuje fázový rozdíl mezi dvěma signály a tuto odchylku odešle do řízeného



Obr. 11: Schéma periferií a částí vývojové desky NUCLEO F767ZI[25].

oscilátoru, který změní napěťovou řízenou frekvenci pomocí filtrace vysokofrekvenčních vln tak, aby bylo dosaženo téměř nulové odchylky.

USART (Universal synchronous asynchronous receiver transmitter) Je to periferie, která slouží k sériové komunikaci. V mé aplikaci využívám v počítači sériovou komunikaci. Pomocí ovladače jsou data přenášena z virtuálního portu COM na USB port. Do tohoto portu je připojen kabel s USB koncovkou, který je schopen přenášet data. Na opačném konci se nachází koncovka Mini USB, kterou lze napájet a přeposílat data s mikroprocesorem. Tímto je vývojová deska propojena pomocí kabelu s počítačem. Na straně vývojové desky převádí ovladač ST-Link USB data opět na sériovou komunikaci. Data jsou tedy přeposílána v podobě USB paketů.

NVIC (The Nested Vector Interrupt Controller) Tato periferie funguje jako vektorový řadič přerušení mikroprocesoru. Fyzicky se jedná o procesor, který je propojený s jádrem. Díky tomuto spojení dokáže přerušit chod velmi efektivně a rychle. Obsahuje 150 upravitelných přerušovacích kanálů a 16 programovatelných úrovní priority. Jsou to

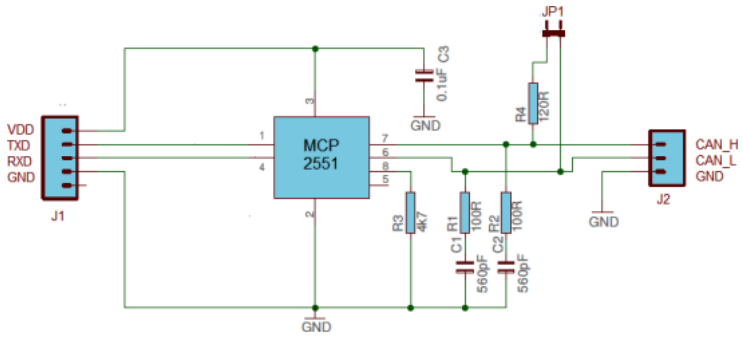
například nastavení tvrdého přerušení, přerušení kvůli výskytu nedefinovaného pokynu nebo chybě přístupu do paměti. Konkrétní možnosti nastavení přerušení a dalších parametrů lze vyčíst z příručky od výrobce.

CAN (Controller Area Network) Konektivní připojení CAN podporuje základní verze CAN 2.0 A a B. Maximální přenosová rychlost je 1Mbit/s. Je vybavena 3 odesílacími schránkami (Mailbox) s konfigurovatelnou prioritou vysílání. Dále disponuje 2 přijímacími boxy nazývanými FIFO s 28 škálovatelnými bankami filtrů. Díky těmto rozřazovacím parametrům dokáže zpracovat mnoho příchozích a odchozích zpráv s minimálním využitím CPU. Periferie pracuje ve 3 hlavních režimech: inicializace, aktivní mód a spánek.

3.2 Ostatní komponenty

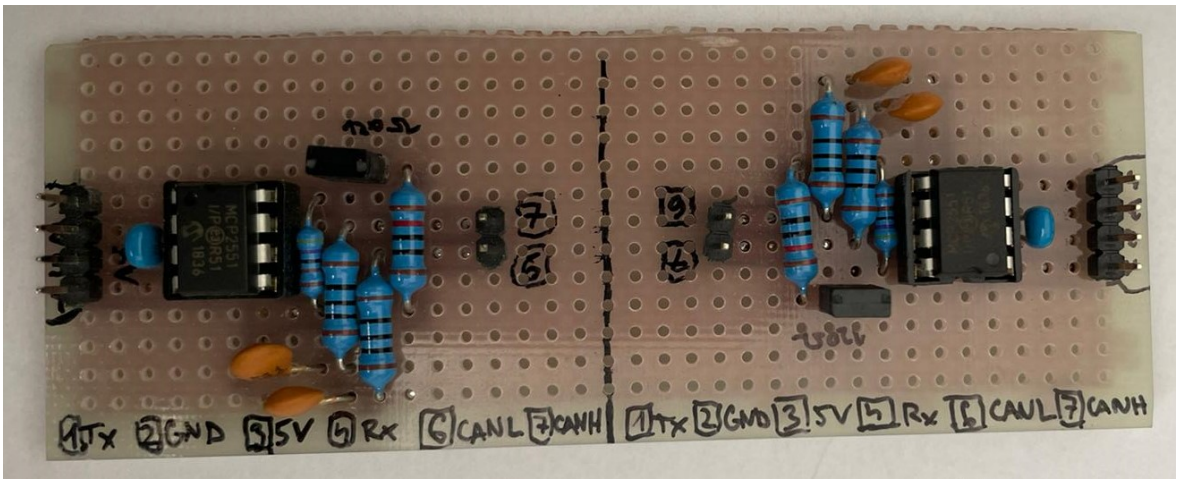
Dále využiji dvou USB kabelů, podporujících datový přenos. Následně je potřeba vyrobit kabely, které budou sloužit jako CAN sběrnice. Jedná se o jednožílový izolovaný vodič, který je na konci opatřen Dupont konektorem.

V poslední řadě je potřeba vytvořit CAN Transciever, který bude převádět logický signál na diferenciální a druhý zpět. Tento Tranciever byl vytvořen dle schématu obr. č.: 12. Jedná se především o mikročip MCP2551 vysokorychlostní CAN budič. Tento čip je velice odolný chybám, které se mohou vyskytnout na sběrnici CAN nebo na řadiči CAN implementovaného v mikrokontroleru. Dle katalogového listu je schopen pracovat až rychlostí 1 Mb/s. Ve schématu zapojení je vyznačen konektor J1, který obsahuje 4 piny: napájení 5V (VDD), 2 CAN piny (Tx, Rx) a poslední zem (GND). Výstupní konektor označen J2 obsahuje výstupní CAN High a CAN Low. Tento pin je také připojen na zem (GND). Propojka JP1 slouží jako zakončovací odpor sběrnic (120 ohmů). Praktická realizace tohoto obvodu je na obr. č.: 13



Obr. 12: Schéma zapojení CAN Transcieveru[26].

Nyní bych ráda přiblížila použitý software od firmy STMicroelectronics. Jedná se o integrované vývojové prostředí, ve kterém lze generovat kód, který lze následně nahrát do mikrokontroléru. Existuje však mnoho dalších programů, které lze použít. Jako



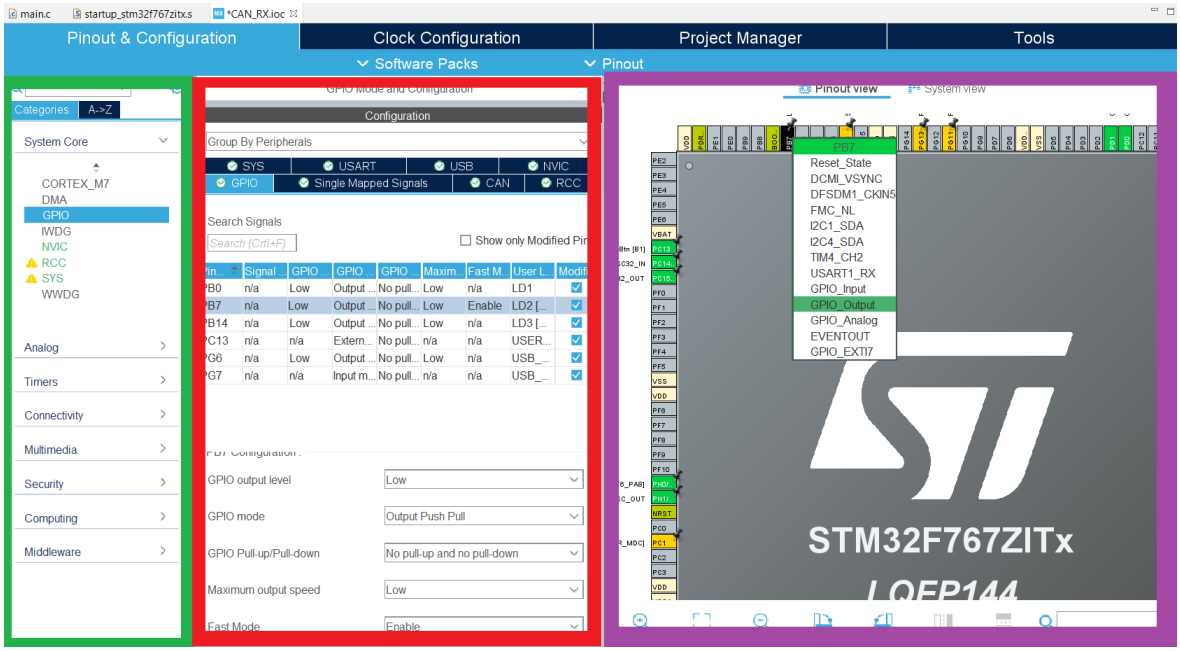
Obr. 13: Fotografie vyrobeného CAN Transceiveru.

například: IAR Embedded Workbench. Programy jsou grafickým prostředím velmi podobné a analogické. Zmíněný program (IAR Embedded Workbench) ale není volně dostupný a je potřeba k jeho používání zakoupit licenci.

3.3 STM32cubeIDE - Grafické nastavení mikrokontroléru

Software, který jsem si vybrala pro programování mikrokontroléru se nazývá STM32CubeIDE. Jedná se o volně dostupný program od společnosti STMicroelectronics. Program je rozdělen na dvě sekce. V jedné sekci se nachází grafický model mikrokontrolerů, kde lze nastavit a aktivovat požadované periferie. Následně lze zde také nastavit konfiguraci hodin a vytváří se zde automaticky kód dle nastavení periferií. Dále je zde možné vyčíst energetickou náročnost desky.

Nastavení diody Na obr. č.: 14 je ukázáno nastavení pinu PB7 jako výstup pro uživatelskou modrou LED diodu. V obrázku je využito tří barevných rámečků (zelený, červený a fialový) pro popis jednotlivých částí integrovaného vývojového prostředí. Z uživatelské příručky lze vyčíst, na kterém pinu je zapsána modrá LED dioda. V tomto případě se jedná o pin s označením PB7. Obecně lze v zeleném rámečku najít všechny periferie, které lze nastavit. Periferie, které jsem nastavila jsou USART, GPIO, RCC, SYS, NVIC, Single Mapped Signals a CAN (detailněji popsáno v následující kapitole). Pro nastavení diody volím sekci System Core a periferii GPIO. Dále v červeném rámečku lze nastavit konkrétnější funkce daného výstupu jako je například již zmíněná rychlost reakce na změnu signálu či výběr modu, dle kterého se bude vyhodnocovat přicházející signál. U LED diody máme několik dalších konfigurací, například je výhodné pro blikání diody zvolit mód Output Push – pull, který rozlišuje mezi dvěma stavy signálu, a podle toho rozbliká diodu. Ve fialovém okně najdeme grafický model mikroprocesoru. Zde zvolím pro použití diody GPIO výstup (GPIO_Output).



Obr. 14: Nastavení periferie GPIO a aktivace pinu s uživatelskou modrou diodou.

Nastavení CAN Rx a Tx Na obr. č.: 15 je znázorněna aktivace a inicializace pinu pro CAN. Z uživatelského manuálu lze opět zjistit na kterém pinu leží příslušné piny CAN Rx a CAN Tx. V tomto případě se jedná o piny PD0 a PD1. Následně je důležité nastavit ostatní paramenty. Abychom dosáhli přenosové rychlosti 500 kbit/s, je potřeba nastavit hodnotu předděličky (Prescaler) na hodnotu 24. Výpočet provádíme dle vzorce:

$$CANClock = CoreRate/Prescaler = 48000000/24 = 2000000Hz \tag{2}$$

kde hodnota frekvence jádra je 48 MHz. Dostáváme hodnotu CAN hodin, která je 2MHz. Pomocí této hodnoty vypočteme časové kvantum dle vzorce:

$$TimeQuantum = 1/CANclock = 1/2000000 = 500ns \tag{3}$$

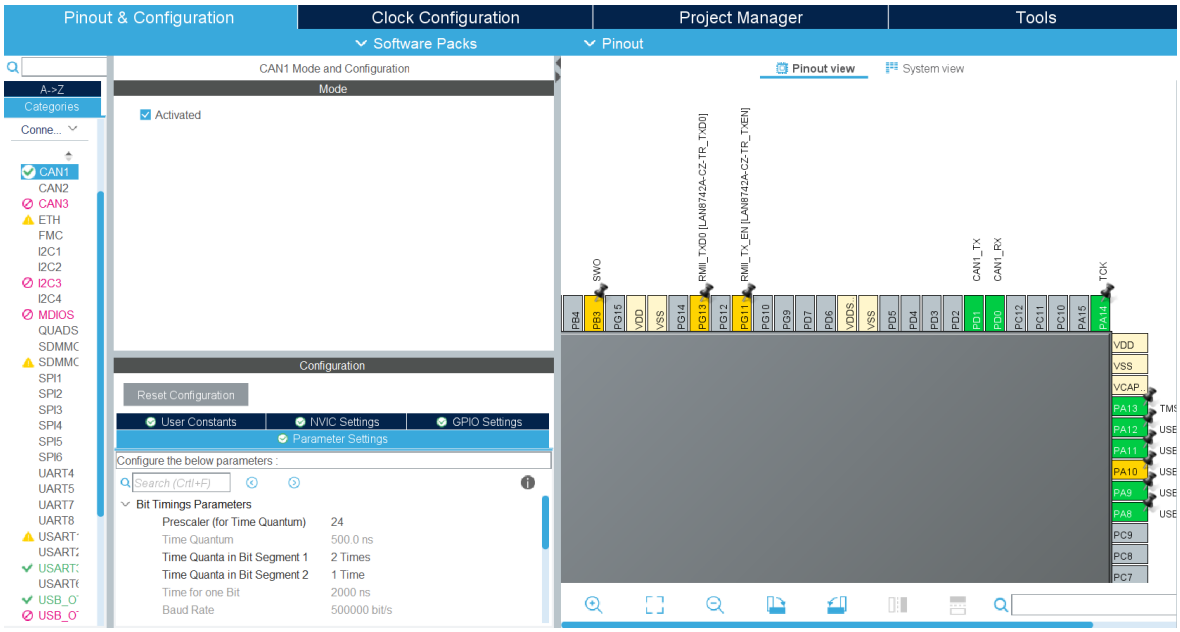
Hodnotu časového kvanta jsem tedy určila jako 500 ns. Již zbývá jen určit hodnotu časového kvanta v 1. a 2. bitovém segmentu. K výpočtu je potřeba ještě zjistit hodnotu PROP_SEG, kterou lze určit pomocí vztahu:

$$PROP_{SEG} = TimeforoneBit/TQ = 2000/500 = 4 \tag{4}$$

Kde za hodnotu Time for one Bit dosadím 2000 a TQ je zkratka pro Time Quantum. A jako poslední je potřeba výpočítat PHASE_SEG:

$$PHASE_{SEG} = NBT - (PROP_{SEG} + SYNC_{SEG}) = 8 - (4 + 1) = 3 \qquad (5)$$

PHASE_SEG je počet volných segmentů, které je potřeba rozdělit mezi časová kvanta v 1. a 2. bitovém segmentu. Vypočítá se z délky bitu (NBT), což je v tomto případě 8. SYNC_SEG se nastavuje vždy fixně na hodnotu 1. Nakonec zbývá tedy rozdělit 3 časová kvanta mezi 2 sekce. Volím pro první časové kvantum hodnotu 2 a pro 2. časové kvantum hodnotu 1.

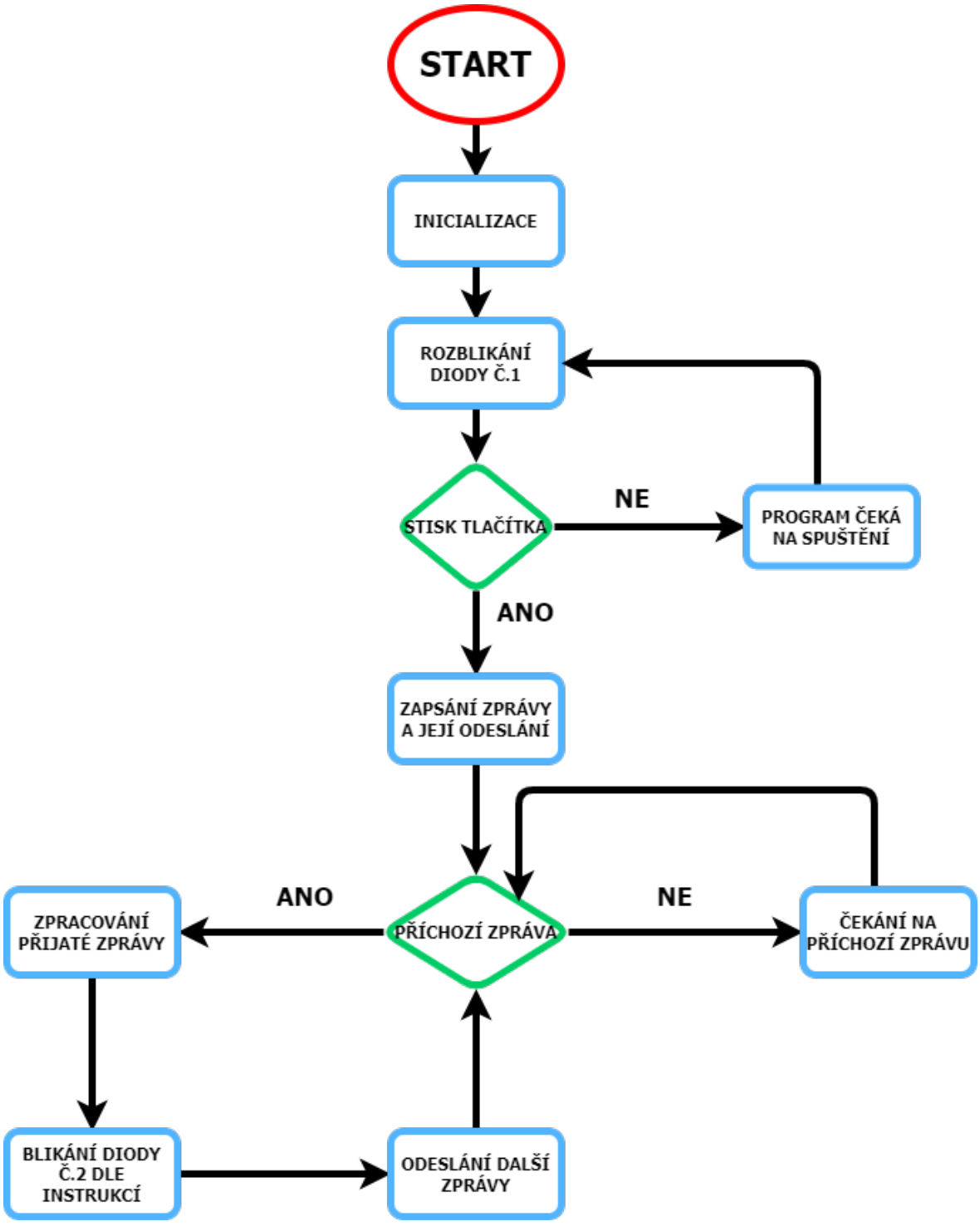


Obr. 15: Nastavení připojení CAN a aktivace pinu.

3.4 STM32cubeIDE - Tvorba kódu

Nyní se v softwaru STM32cubeIDE přesuneme do druhé sekce. V této sekci se automaticky vygeneruje kostra kódu a aktivují se příslušné periferie, které jsem nastavila. Programovat budu pomocí Hal knihovny. Jedná se o soubor již předdefinovaných příkazů. Tyto příkazy jsou popsány v dokumentu Hal knihovna [28], který je volně přístupný. Lze v něm najít jaké proměnné, v jakém pořadí je potřeba přiřadit, aby příkaz správně fungoval. Díky využití těchto příkazů se kód velice zjednoduší a zpřehlední pro další úpravy. Níže popsaný kód je pro odesílací (zahajovací) mikrokontrolér. Druhý kód je strukturně stejný, pouze v něm chybí nastavení spouštěcího tlačítka.

Kód, který navrhuji bude fungovat dle vývojového diagramu na obrázku č.: 16



Obr. 16: Vývojový diagram algoritmu

```c
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */


/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */

uint32_t timeout_set=10;
uint8_t data_output[8];
/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
CAN_HandleTypeDef hcan1;

UART_HandleTypeDef huart3;
DMA_HandleTypeDef hdma_usart3_rx;
DMA_HandleTypeDef hdma_usart3_tx;

PCD_HandleTypeDef hpcd_USB_OTG_FS;

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_USART3_UART_Init(void);
static void MX_USB_OTG_FS_PCD_Init(void);
static void MX_CAN1_Init(void);
```

Zdrojový kód 1: Inicializace a načtení požadovaných periferii

Inicializace a načtení funkcí Na zdrojovém kódu č.: 1 je vidět první část kódu. Tato část je sama generovaná dle aktivace příslušných periferií. Mezi zelené řádky lze vepsat další příkazy. Ovšem se musí dbát na to, aby se uživatelský kód nacházel mezi dělícími

řádky BEGIN a END. Pokud by se kód nacházel mimo tyto řádky, po každé nové inicializaci kódu by se automaticky smazal. Ve své práci nepotřebuji do inicializace a volání periferií zasahovat a nechávám tedy kód takový, jaký jej program vygeneroval. Je zde vidět nastavení příslušných proměnných pro příslušné periferie. První je CAN (ř. 42) a je mu přiřazena proměnná hcan1, dále jsou zde periferie podporující komunikaci s dalšími zařízeními pomocí USB kabelu. Jsou to UART, přiřazeno huart3 a takto bych mohla dále analogicky pokračovat. Dále v kódu, pod linkou Privátních funknčních prototypů (ř. 54) najdeme inicializaci základních funkcí pomocí příkazu void. Jedná se o deklaraci funkce, která nevrací žádné hodnoty. Například systémové hodiny, aby se sběrnice mohla sesynchronizovat. Následně probíhá inicializace GPIO periferii, CAN periferie a komunikační periferie s dalšími zařízeními.

```
64 /* Private user code ---------------------------------------------------*/
65 /* USER CODE BEGIN 0 */
66
67 CAN_TxHeaderTypeDef TxHeader;
68 CAN_RxHeaderTypeDef RxHeader;
69
70 uint8_t TxData[8];
71 uint8_t RxData[8];
72 uint32_t TxMailbox;
73
74 int datacheck = 0;
75
76 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
77 {
78   if (GPIO_Pin == GPIO_PIN_13)
79   {
80     TxData[0]= 100;
81     TxData[1]=40;
82
83     HAL_CAN_AddTxMessage(&hcan1, &TxHeader, TxData, &TxMailbox);
84   }
85 }
86
87 void HAL_CAN_RxFifo0MsgPendingCallback(CAN_HandleTypeDef *hcan)
88 {
89   HAL_CAN_GetRxMessage(hcan, CAN_RX_FIFO0, &RxHeader, RxData);
90   if (RxHeader.DLC == 2)
91   {
92     datacheck = 1;
93   }
94 }
```

Zdrojový kód 2: Vložení zpráv a aktivace jejich příjmu

Vložení zpráv a aktivace jejich příjmu Na zdrojovém kódu č.: 2 již začíná můj kód. Jak je vidět, je nutné, aby byl vepsán mezi části BEGIN a END. Nejdříve definuji odeslanou (Tx) a přijatou (Rx) zprávu jako TxHeader a RxHeader. Dále je definováno místo TxData, kam se data zapisují, aby mohly být dále předána. Odeslaná zpráva se po vytvoření uloží do sekce Mailbox, kde je připravena k odeslání. Na řádku 74 je inicializována deklarace int, datacheck. Jeho hodnota je nastavena na 0. Od řádku 76 nastavuji uživatelské tlačítko, po jehož stisknutí se odešle zpráva obsahující parametry blikání diody. Ty lze najít na řádcích 80 a 81. První informace je o délce svitu diody v ms a druhá informace nese opakovací cyklus. Pomocí funkce na ř. 83 je zpráva vložena do Mailbox. Na řádku 87 se nachází kód zajišťující příjem zpráv. V případě, že transciever obdrží zprávu a uloží ji do schránky FIFO0, změní se datacheck na hodnotu 1. Nastavení této funkce začíná deklarací funkce z Hal knihovny [28], která umožňuje přijetí a uložení zprávy.

```
98  /**
99    * @brief  The application entry point.
100   * @retval int
101   */
102 int main(void)
103 {
104   /* USER CODE BEGIN 1 */
105
106   /* USER CODE END 1 */
107
108   /* MCU Configuration--------------------------------------------------------*/
109
110   /* Reset of all peripherals, Initializes the Flash interface. */
111   HAL_Init();
112
113   /* USER CODE BEGIN Init */
114
115   /* USER CODE END Init */
116
117   /* Configure the system clock */
118   SystemClock_Config();
119
120   /* USER CODE BEGIN SysInit */
121
122   /* USER CODE END SysInit */
123
124   /* Initialize all configured peripherals */
125   MX_GPIO_Init();
126   MX_USART3_UART_Init();
127   MX_USB_OTG_FS_PCD_Init();
128   MX_CAN1_Init();
129   /* USER CODE BEGIN 2 */
```

```
130
131   HAL_CAN_Start(&hcan1);
```

Zdrojový kód 3: Inicializace periferií a zahájení CAN přenosu

Inicializace a spuštění CAN periferie Na zdrojovém kódu č.: 3 se dostávám do hlavní části kódu, která je označovaná jako main. Zde je opět část, do které nebylo potřeba nijak zasahovat, jelikož jsem správně nastavila periferie v předchozí části programu a automatickým generováním kódu se vše nastavilo správně. V kódu lze vidět, že po nastavení proměných a volání funkcí se vše resetuje (ř. 111), aby došlo ke správné synchronizaci času na všech periferiích. Následně dochází k jejich postupné inicializaci (od ř. 125). Jakmile je tento start hotov, dochází ke spuštění, pro mě nejdůležitější periferie, CAN (ř. 131). Jedná se o mnou vložený příkaz, který uvede sběrnici do aktivního stavu. Opět jsem využila příkazu z knihovny Hal [28].

```
133     // Activate the notification
134     HAL_CAN_ActivateNotification(&hcan1, CAN_IT_RX_FIFO0_MSG_PENDING);
135
136     TxHeader.DLC = 2;  // data length
137     TxHeader.IDE = CAN_ID_STD;
138     TxHeader.RTR = CAN_RTR_DATA;
139     TxHeader.StdId = 0x445;  // ID
140
141     TxData[0] = 100;
142     TxData[1] = 40;
143
144   /* USER CODE END 2 */
145
146   /* Infinite loop */
147   /* USER CODE BEGIN WHILE */
148
149   while (1)
150   {
151     /* USER CODE END WHILE */
152
153     /* USER CODE BEGIN 3 */
154     HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_0);
155     HAL_Delay (200);
156
157         if (datacheck)
158         {
159           // blink the LED
160           for (int i=0; i<RxData[1]; i++)
161           {
162             HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_7);
163             HAL_Delay(1000);
```

```
164             }
165             datacheck = 0;
166 
167           TxData[0] = 100;
168           TxData[1] = 40;
169 
170             HAL_CAN_AddTxMessage(&hcan1, &TxHeader, TxData, &TxMailbox);
171         }
172   }
173   /* USER CODE END 3 */
```

Zdrojový kód 4: Odesílání zpráv a aktivace diod

Odesílání zpráv a aktivace diod Na zdrojovém kódu č.: 4 se nachází kód programu, který jsem musela vepsat. Na předchozím obrázku byl zadán příkaz startu celé periferie. Následně aktivuji příjem zpráv. Od této chvíle je mikrkontrolér připraven přijímat zprávy a dále je zpracovávat. Následně definuji veškeré údaje o odesílané zprávě. Nastavuji délku dat DLC, identifikátor zprávy a nastavení hodnoty RTR. Obsah zprávy je definován následně od ř. 141. TxData 0 obsahují údaj o rozsvícení diody na 100 ms, TxData 1 definují, že dioda bude cyklus opakovat 40x. Následuje cyklus while. V tomto cyklu je nastavována dioda, nacházející se na pinu 0 (ř. 154). Funkce TogglePin definuje blikání diody. Přičemž dioda bude svítit a zhasínat po 200 ms. Tato dioda mi signalizuje, zda se program na daném mikrokontroléru nachází a funguje. Dále jsem definovala moment, kdy se má rozblikat dioda na pinu 7. Tato dioda blikáním signalizuje příjem zprávy. Po přijetí zprávy nastaví hodnotu datacheck opět na 0, aby byl program schopen přijmout další zprávu. Po příjmu zprávy kontrolér odešle Tx zprávu na sběrnici.

```
260 /* USER CODE BEGIN CAN1_Init 2 */
261   CAN_FilterTypeDef canfilterconfig;
262 
263     canfilterconfig.FilterActivation = CAN_FILTER_ENABLE;
264     canfilterconfig.FilterBank = 1;
265     canfilterconfig.FilterFIFOAssignment = CAN_FILTER_FIFO0;
266     canfilterconfig.FilterIdHigh = 0x446<<5;
267     canfilterconfig.FilterIdLow = 0;
268     canfilterconfig.FilterMaskIdHigh = 0x446<<5;
269     canfilterconfig.FilterMaskIdLow = 0x0000;
270     canfilterconfig.FilterMode = CAN_FILTERMODE_IDMASK;
271     canfilterconfig.FilterScale = CAN_FILTERSCALE_32BIT;
272     canfilterconfig.SlaveStartFilterBank = 10;
273 
274     HAL_CAN_ConfigFilter(&hcan1, &canfilterconfig);
275   /* USER CODE END CAN1_Init 2 */
```

Zdrojový kód 5: Konfigurace filtrů CAN sběrnice

Konfigurace filtrů CAN sběrnice Na zdrojovém kódu č.: 5 lze vidět kód, kde se nastavuje filtrování zpráv. Každá zpráva dle příslušného ID může být mikrokontrolérem přijata. V praktickém využití, kde se ovšem na sběrnici vyskytuje mnoho zpráv je efektivnější nastavit filtr, který bude dané zprávy filtrovat pouze na ty, které jsou pro danou větev důležité. Jedním typem tohoto filtru je tzv. maska. Její princip funkce lze vidět na obrázku č.: 17. Jedná se o nastavení porovnávacího ID, které obsahuje informaci, na kterém místě identifikátoru se musejí hodnoty shodovat. První sloupec tabulky nám označuje hodnotu masky, druhý hodnotu filtru a třetí hodnotu bitu identifikátoru příchozí zprávy. Na prvním řádku je zobrazen případ, kdy není uvedena maska. Dochází k příjmu všech zpráv. Na dalším řádku lze vidět, že pokud má na daném místě maska hodnotu 1, toto místo v ID bude porovnáváno. Dle filtru požaduji na daném místě hodnotu 0 a ta se nachází i na daném místě v ID přijaté zprávy. Zpráva tedy projde skrze filtr. Na třetím řádku ale vidíme, že se liší hodnota filtru a hodnota přijaté zprávy. Zpráva je tedy filtrem blokována. Například, pokud máme ID, které má 11 znaků. Dle nastavení mají všechny členy sběrnice prvních 5 znaků stejných. Větve se tedy rozlišují následujícími hodnotami v ID. Větev, od které chceme zprávy přijímat má 6, 7 a 8 hodnotu identifikátoru stejnou, ostatní hodnoty se liší. Nastavíme tedy masku tak, aby filtrovala a přijímala pouze zprávy, které budou mít na 6,7 a 8 místě stejnou hodnotu jako maska. Tím zaručíme nezahlcení paměti v mikrokontroléru nepotřebnými zprávami. Odesílací mikrokontrolér má nastavenou hodnotu ID 0x446. Pokud bych tedy na řádku 268 změnila hodnotu z 0x455 na například 0x106, zpráva odeslaná z druhého mikrokontroléru by nebyla přijata, protože by neprošla přes filtr zpráv.

FILTER/MASK TRUTH TABLE

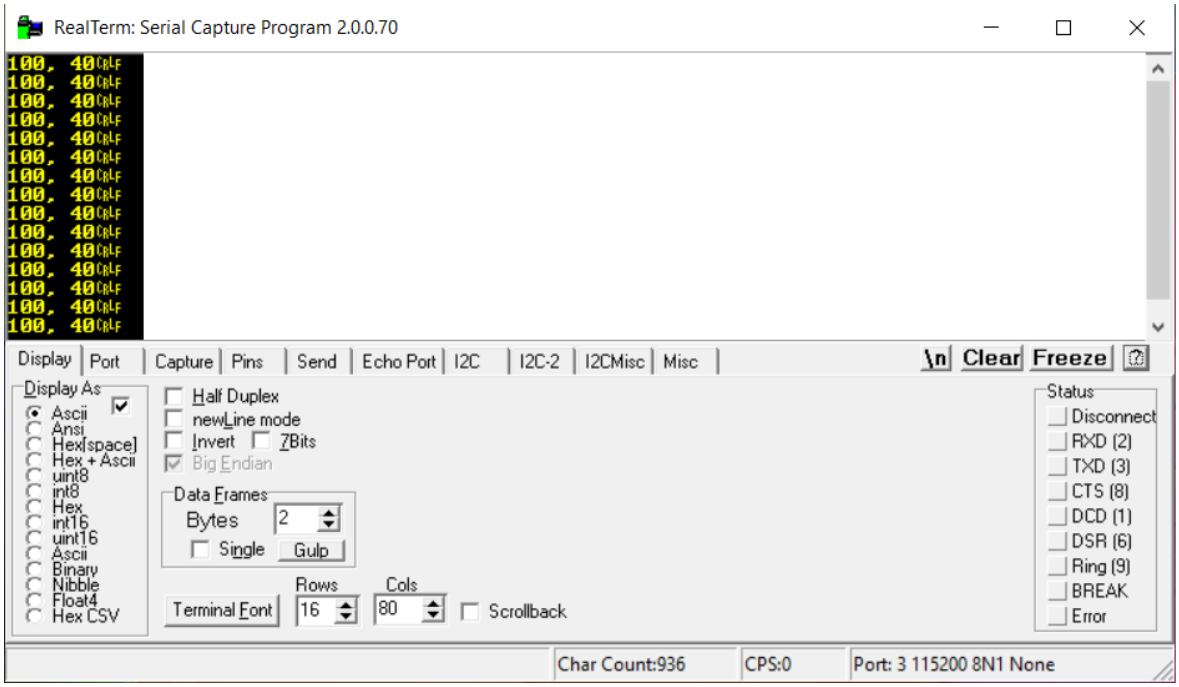
Mask Bit n	Filter Bit n	Message Identifier bit	Accept or Reject bit n
0	x	x	Accept
1	0	0	Accept
1	0	1	Reject
1	1	0	Reject
1	1	1	Accept

Note: x = don't care

Obr. 17: Pravdivostní tabulka masky a filtru CAN sběrnice[27].

3.5 Zobrazení zpráv pomocí softwaru

Na závěr jsem si zprávu odesílanou jedním mikrokontrolérem zobrazila v programu RealTerm. Aby se výsledky odesílali pomocí USB kabelu do PC, musela jsem nastavit periférii USART. Potřebné části kódu jsem vložila do příslušných sekcí. Lze vidět ve zdrojových kódech č.: 6, 7, 8 i s příslušným popisem přidaných částí. Výsledek je vidět na obrázku č.: 18. Jsou zde opakovaně vidět hodnoty 100 a 40. Což jsou informace pro druhý mikrokontrolér o parametrech blikání diody.



Obr. 18: Odposlouchávání sběrnice pomocí sériové linky a programu Realterm

```
20 #include "main.h"
21 #include <stdio.h>
22 #include <string.h>
```

Zdrojový kód 6: Aktivace příslušných knihoven s funkcemi

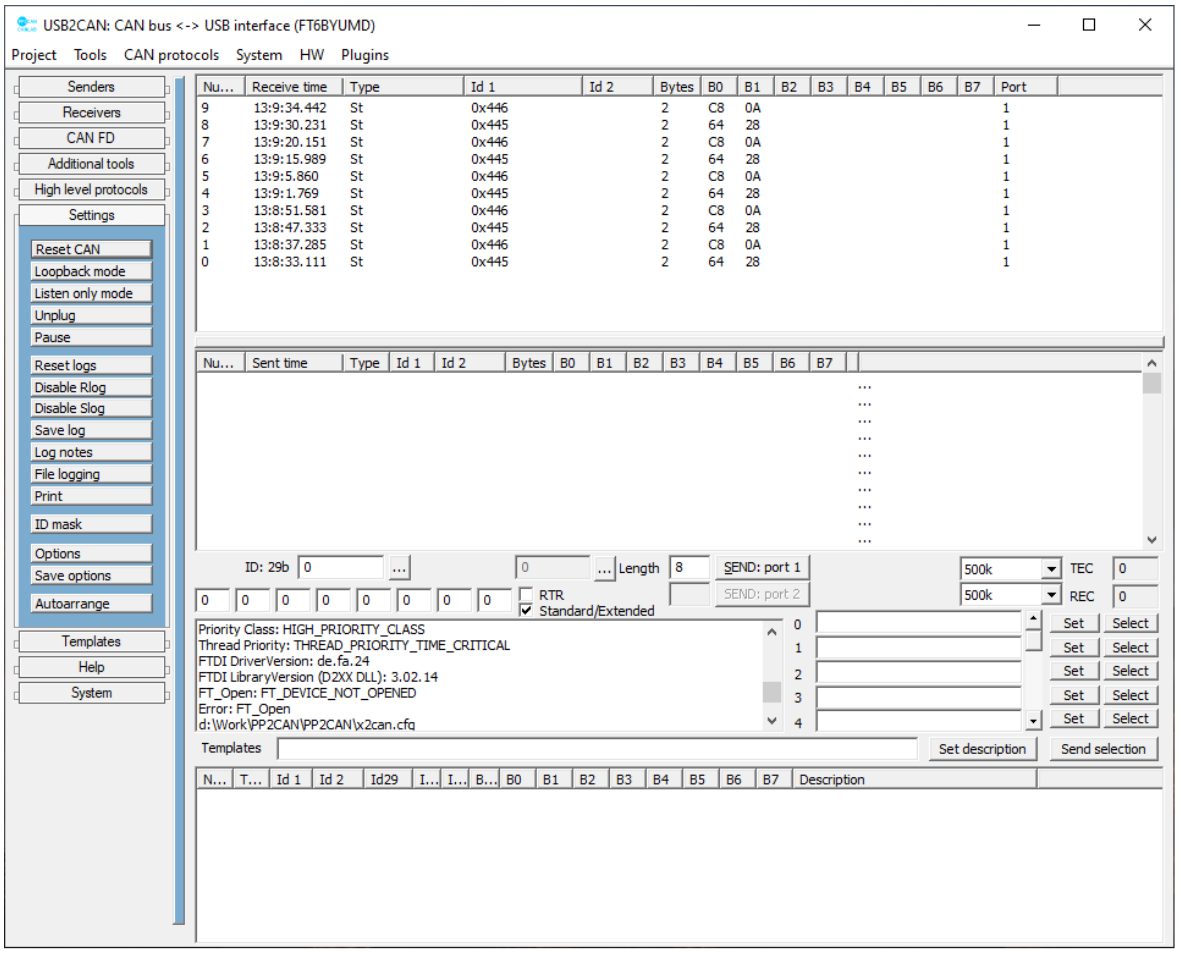
```
36 /* USER CODE BEGIN PD */
37
38 char data_output[20];
39 uint8_t number1;
40 uint8_t     number2;
41 uint32_t timeout_set=10;
42
43 /* USER CODE END PD */
```

Zdrojový kód 7: Vytvoření potřebných proměnných a nastavení základních parametrů

```
166 for (int i=0; i<RxData[1]; i++)
167         {
168           HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_7);
169           HAL_Delay(RxData[0]);
170         }
171
172        number1=RxData[0];
173       number2=RxData[1];
174         sprintf(data_output,"%d, %d\r\n", number1, number2);
175           HAL_UART_Transmit(&huart3, (uint8_t *)data_output,
176       strlen(data_output),timeout_set);
177
178        HAL_CAN_AddTxMessage(&hcan1, &TxHeader, TxData, &TxMailbox);
```

Zdrojový kód 8: Přiřazení příslušných hodnot do proměnných a odeslání po sériové lince

Dále jsem vyzkoušela odposlouchávání sběrnice pomocí převodníku, který byl napojený na sběrnici. Toto zařízení sloužilo pouze jako odposlech a nijak do sběrnice nezasahovalo. Odesílalo informace do PC, a zde byli pomocí speciálního softwaru vyobrazovány výsledky, které jsou na obrázku č.:19. Čísla, které jsou posílána jsou ovšem zobrazována v Hexadecimálním systému dle převodní tabulky ASCI [29]. Jedná se o obsah zpráv, kde (C8, 0A)h = (200, 10)d a (64, 28)h = (100, 40)d. Lze si dle obrázku ověřit, že zprávy obsahují ID 445 a 446, což odpovídá komunikaci mezi oběma použitými zařízeními.

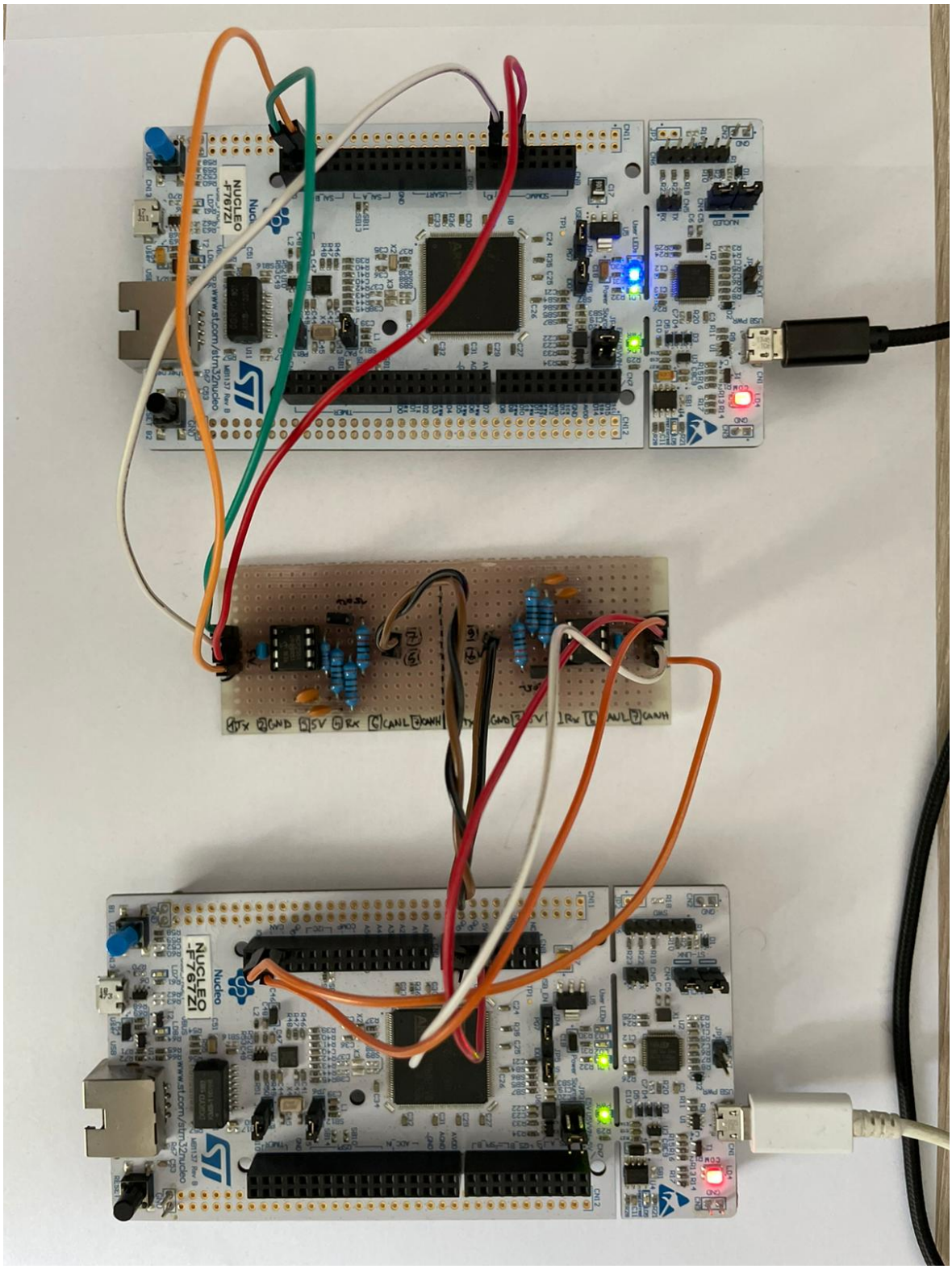


Obr. 19: Odposlouchávání sběrnice pomocí třetího zařízení a speciálního softwaru od firmy CANLAB

3.6 Realizace komunikace

Realizaci komunikace mezi dvěma vývojovými deskami pomocí sběrnice CAN lze vidět na obrázku č.:20.

Na přiloženém CD je nahrané video, na kterém lze vidět komunikaci mezi oběma vývojovými deskami. Nejdříve vidíme nahrání kódu do mikrokontroléru. Rozblikání zelené diody na desce. Následně po stisknutí levého tlačítka se spustí odesílání a komunikace. Na videu lze vidět rozblikání modré diody, které je realizováné odeslanou zprávou, která obsahuje údaje o rychlosti jejího blikání a délky blikání.



Obr. 20: Výsledné zapojení všech součástí

4 Závěr

Tématem této práce byla rešerše automobiových komunikačních standardů, jejíž součástí byla realizace komunikace mezi dvěma mikrokontroléry pomocí datové sběrnice CAN. V první, teoretické části, jsem se zabývala datovou sběrnicí. Konkrétně její fyzickou vrstvou, z jakých materiálů se sběrnice skládá a další její nutné prvky pro správný a bezpečný provoz. Dále jsem rozebrala linkovou vrstvu. Následně jsem se věnovala CAN zprávě a jejímu formátu. Na závěr jsem se zabývala dostupnými sběrnicemi a nakonec shrnula jejich parametry do přehledné tabulky. V té lze rychle porovnat sběrnice mezi sebou a získat náhled do celé problematiky.

V praktické části jsem realizovala komunikaci pomocí dvou vývojových desek a CAN Transevieru spojení mezi dvěma mikrokontroléry. Spojení mezi nimi jsem realizovala pomocí několika vodičů s příslušnými koncovkami. Napájení mikrokontroléru bylo realizováno pomocí USB kabelu, který sloužil také jako datový kanál pro komunikaci miktorkontroléru s počítačem. V programu STM32 Cube IDE jsem aktivovala a definovala potřebné periferie a vytvořila kód. V něm jsem pomocí Hal knihovny a jejích příkazů definovala spuštění periferii a nastavila mikrokontrolér pro příjem a odesílání zpráv. Dále jsem nastavila zprávy odesílané mezi dvěma mikrokontroléry. Obsahem zpráv byla realizace blikání diody. Kód jsem před nahráním zkontrolovala programem a nechala zkonvertovat pro odeslání do mikrokontroléru.

Následně jsem ověřila jeho funkčnost. Dle přiložených výsledků lze říci, že se kód choval dle očekávání a odesílání zpráv bylo relizováno úspěšně.

Seznam použitých značek a symbolů

CAN Controller area network

MOST The Media Oriented Systems Transport

LIN Local Interconnect Network

DC Stejnosměrný elektrický proud

CAN H CAN High

CAN L CAN Low

CAN FD CAN with Flexible Data-Rate

PVC Polyvinylchlorid

OBD2 On-board diagnostics

SD karta Secure Digital card

Wifi Wireless Ethernet Compatibility Aliance

MAC Medium Acces Control

LLC Logical Link Control

CSMA/CS+AMP Carrier Sense Multiple Acces with Colision Detektion and Arbitration on Message Priority

ID Identifikace ve výpočetní technice

CRC Cyklický redundantní součet

ACK Acknowledgement code

ECU Electronic control unit

SOF Start-of-frame

RTR Rámec vzdáleného vyžádání

DLC Data Length Code

EOF End of frame

PGN Parameter Group Number

PF Protocol Data Unit Format

PS Protocol Data Unit Specific

PDU Protocol Data Unit

SDO Service Data Objects

NMT The network management

SYNC Synchronization

PDO Process data objects

MDI Medium Dependent Interface

BR BroadR-Reach

UTP Nestíněná kroucená dvojlinka

EMC Elektromagnetická kompatibilita

PCS Physical Coding Sublayer

PMA Physical Medium Attachment

PTP Precision Time Protocol

LAN Local Area Network

VLAN Virtual Local Area Network

TDMA Time Division Multiple Access

PCB Printed Circuit Board

PLC Programovatelný logický automat

RAM Random Access Memory

LED Elektroluminiscenční dioda

GPIO Univerzální vstupní/výstupní pin

SW Software

PLL Fázový závěs

RCC Reset and clock controller

USART Universal synchronous asynchronous receiver transmitter

COM Hardwarové rozhraní

NVIC The Nested Vector Interrupt Controller

VDD Označení pro napájecí pin

GND Zem

Seznam použité literatury a zdrojů

[1] CiA. *Cyclic redundancy check (CRC) in CAN frames.*. CiA [online]. CAN in Automation, 2021 [cit. 2021-10-15]. Dostupné z: https://www.can-cia.org/can-knowledge/can/crc/

[2] CSS Electronics. *CAN FD Explained - A Simple Intro* CSS Electronics [online]. CCS Electronics, 2021 [cit. 2021-10-15]. Dostupné z: https://www.csselectronics.com/pages/can-fd-flexible-data-rate-intro

[3] KVASER. *The CAN Protocol Tour* [online]. Kvaser, 2021 [cit. 2021-10-15]. Dostupné z: https://www.kvaser.com/about-can/the-can-protocol/can-messages-23/

[4] CCS Electronics. *CAN Bus Explained - A Simple Intro.* CSS Electronics [online]. CSS Electronics, 2021 [cit. 2021-10-15]. Dostupné z: https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial

[5] Tech Design Forum. Using Ethernet in automotive networks [online]. Mountain View [cit. 2022-05-09]. Dostupné z: https://www.techdesignforums.com/practice/technique/using-ethernet-automotive-networks/

[6] KRATOCHVÍL, Michal. *Datová sběrnice.* ELUC [online]. Olomouc: MŠMT, 2015 [cit. 2021-10-12]. Dostupné z: https://eluc.kr-olomoucky.cz/verejne/lekce/1938

[7] TARABA, Radek.*Aplikování sběrnice CAN.* Vyvoj.hw.cz [online]. Redakce HW serveru, 2004 [cit. 2021-10-12]. Dostupné z: https://vyvoj.hw.cz/navrh-obvodu/rozhrani/aplikovani-sbernice-can.html

[8] HORÁK, Jakub. *Akční člen s CAN protokolem.* Liberec, 2006. Dostupné také z: https://dspace.tul.cz/bitstream/handle/15240/47414/V$_$6306$_$M.pdf?sequence=1$&$isAllowed=y.,Diplomovápráce.TechnickáuniverzitavLiberci.

[9] MOST Bus. *MOST* Bus [online]. Wikipedia, 2021 [cit. 2021-10-24]. Dostupné z: https://en.wikipedia.org/wiki/MOST$_$Bus

[10] JENDEK, Jan. *Návrh monitoru automobilových sběrnic.* Plzeň, 2017/18. Dostupné také z: https://otik.zcu.cz/bitstream/11025/32212/1/Bakalarska$%$20prace$%$20Jan$%$20Jendek.pdf.Bakalářskápráce.ZápadočeskáuniverzitavPlzni.

[11] Sandeep's Blog on AUTOSAR. *How FlexRay Works – Part 2* [online]. Sandeep, 2016, 11.6.2016 [cit. 2021-11-29]. Dostupné z: https://sandeeptiwari.com/flexray-works-part-2/

[12] CCS Electronics. *LIN Bus Explained - A Simple Intro* CCS Electronics, [online]. 2021, 2021 [cit. 2021-11-29]. Dostupné z: https://www.csselectronics.com/pages/lin-bus-protocol-intro-basics

[13] AutoPi. *CAN Bus Protocol: The Ultimate Guide* AutoPi, [online]., 2021, 18.3.2021 [cit. 2021-11-29]. Dostupné z: https://www.autopi.io/blog/can-bus-explained/

[14] KVASER *SAE J1939 Introduction*Kvaser, [online]., 2021 [cit. 2021-11-29]. Dostupné z: https://www.kvaser.com/about-can/higher-layer-protocols/j1939-introduction/

[15] CCS Electronics. *J1939 Explained - A Simple Intro* [online]. CSS Electronics, 2021, 2021 [cit. 2021-11-29]. Dostupné z: https://www.csselectronics.com/pages/j1939-explained-simple-intro-tutorial

[16] A. VARGHESE, Alan. *Automobile In-Vehicle Networks—Ethernet, SERDES, or Both?* EE Times, [online]. EE Times Asia, 2021, 29.6.2021 [cit. 2021-11-29]. Dostupné z: https://www.eetasia.com/automobile-in-vehicle-networks-ethernet-serdes-or-both/

[17] BUČKOVSKÝ, Dmitrij. *Využití sítí Ethernet v osobních automobilech.* Praha, 2016. Dostupné také z: https://dspace.cvut.cz/bitstream/handle/10467/64842/F3-BP-2016-Buckovsky-Dmitrij-Vyuziti$_$siti$_$Ethernet$_$v$_$osobnich$_$automobilech.pdf?sequence=1$&$isAllowed=y.BakalářskápráceČeskévysokéučenítechnickévPraze.VedoucípráceDoc.Ing.JiříNovák,Ph.D.

[18] ROMÁNEK, David. HW server. *Co je CANopen a jak na něj* [online]. Redakce HW serveru, 2006, 20.3.2006 [cit. 2021-11-30]. Dostupné z: https://vyvoj.hw.cz/produkty/co-je-canopen-a-jak-na-nej.html

[19] CCS Electronics. *CANopen Explained - A Simple Intro* [online]. CSS Electronics, 2021, 2021 [cit. 2021-11-30]. Dostupné z: https://www.csselectronics.com/pages/canopen-tutorial-simple-intro

[20] MARADANA, Sudhakar. *Automotive Basics.* CAN Basics [online]. Automotive Basics, 2012, 1.6.2012 [cit. 2021-11-30]. Dostupné z: https://automotivetechis.wordpress.com/2012/06/01/can-basics-faq/

[21] HAZDRA, Martin. *Návrh programovatelné pojistkové skříně pro studentskou formuli.* Praha, 2021. Dostupné také z: http://hdl.handle.net/10467/97162.BakalářskápráceČeskévysokéučenítechnickévPraze.VedoucípráceIng.LubomírMusálek.

[22] Wikipedia. *STM32* [online]. Wikipedia, 2021, 31.7.2021 [cit. 2021-11-30]. Dostupné z: https://en.wikipedia.org/wiki/STM32

[23] Farnell. *STM32F767ZIT6* [online]. Farnell [cit. 2021-11-30]. Dostupné z: https://cz.farnell.com/stmicroelectronics/stm32f767zit6/mcu-32bit-cortex-m7-216mhz-lqfp/dp/2535499

[24] EmbedClogic: *Type Of CAN message* [online]. [cit. 2022-02-11]. Dostupné z: https://embedclogic.com/can-protocol-protocol-to-broadcast-message-on-a-network/type-of-can-message/

[25] STM32 Nucleo-144 boards User manual, *STM32 Nucleo-144 boards User manual* [online]. STMicroelectronics, 28.8.2020 [cit. 2022-04-01]. Dostupné z: https://www.st.com/en/evaluation-tools/nucleo-f767zi.html$#$documentation

[26] HAREENDRAN, T.K.,*ElectroSchematics.* DIY CAN Transceiver [online]. ElectroSchematics, 17.5.2020 [cit. 2022-04-01]. Dostupné z: https://www.electroschematics.com/diy-can-transceiver/

[27] CANLAB s.r.o., 2020. *Masky a filtry* [online]. 5.7.2020 [cit. 2022-05-10]. Dostupné z: https://www.canlab.cz/cs/masky$_$a$_$filtry

[28] Description of STM32F4 HAL and low-layer drivers, 2021. *STMicroelectronics* [online]. STMicroelectronics [cit. 2022-05-12]. Dostupné z: https://www.st.com/content/ccc/resource/technical/document/user$_$manual/2f/71/ba/b8/75/54/47/cf/DM00105879.pdf/files/DM00105879.pdf/jcr:content/translations/en.DM00105879.pdf

[29] Wikipedia, 2007. *ASCI Table* [online]. [cit. 2022-05-12]. Dostupné z: https://en.m.wikipedia.org/wiki/File:ASCII-Table.svg

Seznam obrázků

Seznam obrázků

Obsah přiloženého CD

- Kompletní kód pro odesílání zprávy
- Kompletní kód pro příjem zprávy
- Celé znění práce v pdf
- Komunikace mezi vývojovými deskami pomocí CAN (video)