ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA APLIKOVANÉ ELEKTRONIKY A TELEKOMUNIKACÍ

DIPLOMOVÁ PRÁCE

Rozšíření Univerzální řídící jednotky USG2 o CAN FD

Bc. Ondřej Urban **2018/2019**

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2018/2019**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Ondřej URBAN**
Osobní číslo:	**E17N0036P**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Elektronika a aplikovaná informatika**
Název tématu:	**Rozšíření Univerzální řídicí jednotky USG2 o CAN FD**
Zadávající katedra:	**Katedra aplikované elektroniky a telekomunikací**

Zásady pro vypracování:

1. Proveďte rešerši stávajícího stavu CAN FD (zastřešující normy, podporující mikrokontroléry).
2. Seznamte se s řídicí jednotkou USG2 a navrhněte modul rozhraní pro CAN FD.
3. Realizujte funkční vzorek a vytvořte ukázkovou aplikaci.
4. Zdokumentujte vytvořenou nadstavbu USG2 pro její budoucí rozšiřitelnost.

Rozsah grafických prací: **podle doporučení vedoucího**

Rozsah kvalifikační práce: **40 - 60 stran**

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **ISO 11898-1:2015, Road vehicles - Controller area network (CAN) - Part 1: Data link layer and physical signalling**
2. **ISO 11898-2:2016, Road vehicles - Controller area network (CAN) - Part 2: High-speed medium access unit**

Vedoucí diplomové práce: **Ing. Jan Mráz, Ph.D.**
Katedra aplikované elektroniky a telekomunikací

Datum zadání diplomové práce: **5. října 2018**

Termín odevzdání diplomové práce: **30. května 2019**

Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

L.S.

Doc. Dr. Ing. Vjačeslav Georgiev
vedoucí katedry

V Plzni dne 5. října 2018

Abstrakt

Cílem předkládané diplomové práce je realizace rozšiřujícího modulu, podporujícího protokol CAN FD, specifikovaný normou ISO 11898-1:2015, a ISO 11898-2:2016. Tento rozšiřující modul je součástí univerzální vývojové platformy označované jako USG2. Tato platforma je určená pro prvotní ověřování koncepce nově vyvíjených ECU jednotek. USG2 disponuje modulem s mikrokontrolérem MPC5748G, pro který byl rovněž vytvořen software, realizující základní ovládání FlexCAN periferie, podporující protokol CAN FD.

Dále byla provedena rešerše norem, týkajících se tohoto protokolu, s důrazem na uvedení změn oproti protokolu CAN. Rovněž byla krátce provedena rešerše podporujícího hardware.

Pro demonstraci funkčnosti byly vytvořeny dvě ukázkové aplikace. První z nich slouží jako intuitivní ovládání FlexCAN periferie prostřednictvím uživatelského rozhraní PC. Druhá slouží k přenosu souborů mezi jednotkami USG2, a jedním z jejích účelů je demonstrace výhod protokolu CAN FD oproti původnímu protokolu CAN.

Klíčová slova

CAN FD, CAN, CANopen FD, FlexCAN, USG2, ECU, rozšiřující modul, MPC5748G, ISO 11898-1, ISO 11898-2.

Abstract

The aim of this diploma thesis is to create an expansion module, which supports CAN FD protocol, defined by ISO 11898-1:2015 and ISO 1898-2:2016 standards. This module will be part of universal development board USG2, intended for development and testing of new ECU designs. USG2 development board uses microcontroller module with MPC5748G microcontroller with FlexCAN peripheral. This FlexCAN peripheral has CAN FD capability, as well as classic CAN protocol support. Software for FlexCAN peripheral has been developed to create CAN FD library.

Also currently available standards regarding CAN FD have been summarized, with main focus on differences between CAN FD and CAN protocol. Currently available hardware with CAN FD support has been briefly summarized.

To demonstrate functionality of developed module and software, two applications were developed. First one is intended to provide interactive control of FlexCAN module using PC user interface. The second application is capable of entire file transfer. Its purpose is to highlight the advantages of CAN FD protocol over classic CAN.

Key words

CAN FD, CAN, FlexCAN, USG2, ECU, expansion module, CANopen FD, MPC5748G, ISO 11898-1, ISO 11898-2.

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této diplomové práce, je legální.

.....

podpis

V Plzni dne 27.5.2019 Bc. Ondřej Urban

Poděkování

Tímto bych rád poděkoval vedoucímu bakalářské práce Ing. Janu Mrázovi, Ph.D. za cenné rady a připomínky. Poděkování rovněž patří společnosti MBtech Bohemia, za vytvoření tohoto zadání a poskytnutí prostředků pro jeho realizaci. Dále pak Katedře aplikované elektroniky a telekomunikací za výrobu prototypu desky plošných spojů.

Obsah

Seznam symbolů a zkratek

AUI	-	Attachment unit interface
CAN	-	Controller area network
CAN FD	-	CAN with flexible data-rate
CBFF	-	Classical base fame format
CEFF	-	Classical extended frame format
CiA	-	CAN in automation
ECU	-	Electronic control unit
EMI	-	Electromagnetic interference
FBFF	-	FD base frame format
FEFF	-	FD extended frame format
ISO	-	Mezinárodní organizace pro normalizaci (International Organization for Standardization)
LLC	-	Logic link control sub-layer
MAC	-	Media access control sub-layer
MB	-	Message Buffer
MDI	-	Media dependant interface
PDU	-	Protocol Data Unit
PGN	-	Page Group Number
PLS	-	Physical coding (signaling) sub-layer
SPN	-	Suspect Parameter Number
STP	-	Shielded twisted pair
TTCAN	-	Time-triggered CAN
UTP	-	Unshielded twisted pair

Úvod

Tato diplomová práce, zpracovávána pro společnost MBtech Bohemia, se soustřeďuje na relativně nově uvedený protokol CAN FD. Cílem práce je realizovat funkční rozšiřující modul podporující CAN FD pro univerzální vývojovou jednotku USG2. Tato vývojová platforma slouží pro návrh a ověřování koncepce vyvíjených ECU jednotek. Dále je potřeba vytvořit knihovnu software pro mikrokontrolér MPC5748G, použitý touto jednotkou. Realizovaný hardware i software má být následně otestován ukázkovou aplikací.

Práce je dělena do dvou částí. První část se soustředí na rešerši stávajících norem, týkajících se protokolu CAN FD, včetně krátkého shrnutí aplikace tohoto protokolu v takzvaných vyšších aplikačních protokolech, konkrétně CANopen FD a J1939. Dále bylo provedeno krátké shrnutí stavu současně dostupného hardware, podporujícího tento protokol. Tato teoretická část byla psána s důrazem na porovnání protokolů CAN a CAN FD.

Ve druhé části práce je popsán návrh rozšiřující desky pro jednotku USG2 a software pro periferii FlexCAN. Tato deska rozšiřujícího modulu používá čtyři transceivery TLE9252, společnosti Infineon, podporující bitovou rychlost až 5 Mbit/s. Dále byla vyvinuta testovací aplikace využívající propojení s uživatelským rozhraním v PC, vytvořeným v jazyce C#. Tato aplikace je schopna ovládat FlexCAN periferii a jedná se o vhodný nástroj pro monitorování dění na sběrnici v případě že není k dispozici analyzátor s podporou CAN FD.

Druhá realizovaná aplikace byla vyvinuta pro demonstraci výhod použití CAN FD. Aplikace je schopna prostřednictvím uživatelského rozhranní PC odesílat soubory mezi jednotlivými jednotkami USG2 (reálně byla testována pouze na dvou jednotkách, nicméně implementace podporuje až 32 připojených jednotek).

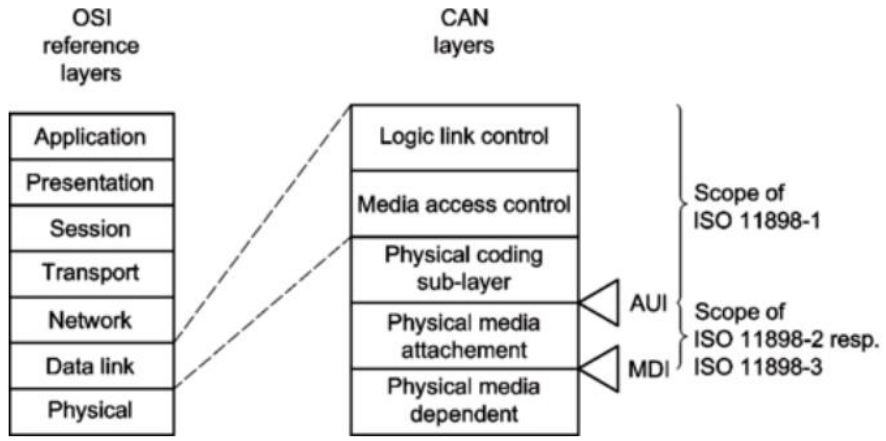
1. Zastřešující normy

Protokol CAN FD je definován v rámci skupiny norem vydávaných Mezinárodní organizací pro standardizaci, označovaných jako ISO 11898. Tyto normy standardizují datovou a fyzickou vrstvu protokolů CAN FD a klasického CAN. V následujícím textu bude uveden jejich přehled.

Norma ISO 11898 je v současnosti rozdělena na čtyři části:

- ISO 11898-1:2015
- ISO 11898-2:2016
- ISO 11898-3:2006
- ISO 11898-4:2004

Oblasti příslušnosti jednotlivých norem k jednotlivým vrstvám CAN lze snadno vidět na obr. 1.



Obr. 1: Zobrazení vrstev CAN vzhledem k ISO/OSI modelu a jednotlivým vztaženým normám. [1]

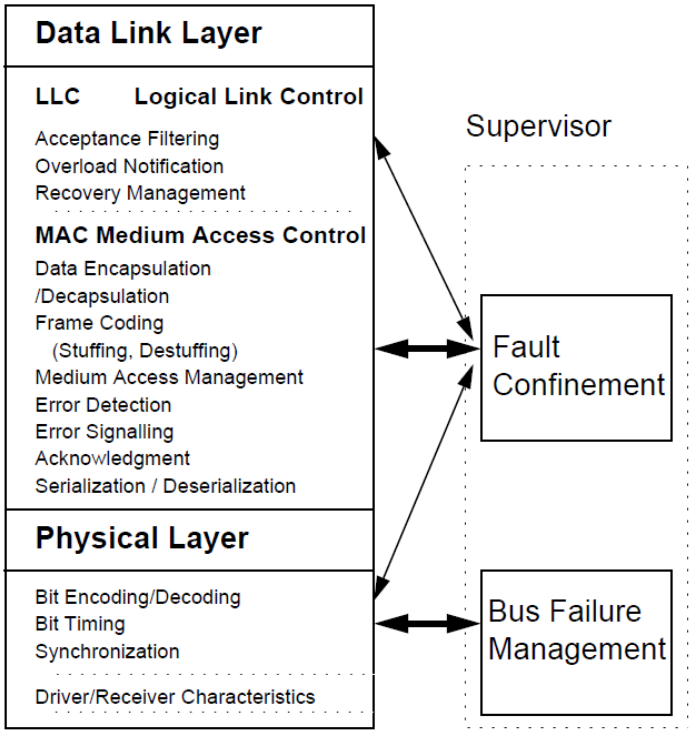
Dále existuje mnoho standardů, specifikujících nejrůznější podoby implementace fyzické vrstvy, a standardy specifikující takzvané vyšší aplikační vrstvy CAN.

Následující text bude souhrnem nejdůležitějších norem vzhledem k nově uvedenému protokolu CAN FD.

1.1 ISO 11898-1:2015

Tato norma, jejíž nejnovější podoba byla vydána roku 2015 (tedy označovaná jako ISO 11898-1:2015), specifikuje jak klasický formát rámce CAN, tak CAN FD.

Je zde popsáno hierarchické rozložení CAN do vrstev podle modelu ISO/OSI („Open systems Interconnection“) v souladu s normou ISO/IEC 7498-1. Detailně jsou zde popsány podvrstvy LLC (Logic link control sub-layer), MAC (Media access control sub-layer), a PLS (Physical coding sub-layer), jejich vazeb a chování v případě poruchy.



Obr. 2: Popis struktury a základních funkcí protokolu CAN podle ISO 11898 a ISO/OSI modelu. [1]

Norma rovněž stanovuje tři základní možnosti implementace, vzhledem k novému protokolu CAN FD, a to:

- Podpora pouze klasického CAN rámce, bez tolerance CAN FD.
- Podpora klasického CAN s tolerováním CAN FD rámců
- Podpora jak klasických CAN, tak CAN FD rámců

Podrobný popis protokolu CAN FD, specifikovaný touto normou, je rozepsán v samostatné kapitole (kapitola 2. CAN FD).

1.2 ISO 11898-2:2016

Jak lze vidět na obr. 1, tato norma specifikuje HS-PMA (high-speed media attachment) vrstvu CAN FD protokolu.

Oproti předchozí verzi z roku 2003 dochází ke značným změnám nejen technického rázu, ale i co se reorganizace dalších norem týče. Došlo k integrování dříve samostatných norem 11898-5 a 11898-6, jež se zabývaly wake-up funkcionalitou, tak aby celá vrstva HS-PMA byla specifikována v rámci jedné normy. [2]

Tato norma specifikuje požadavky na „high-speed“ budiče jak pro CAN, tak i pro CAN FD.

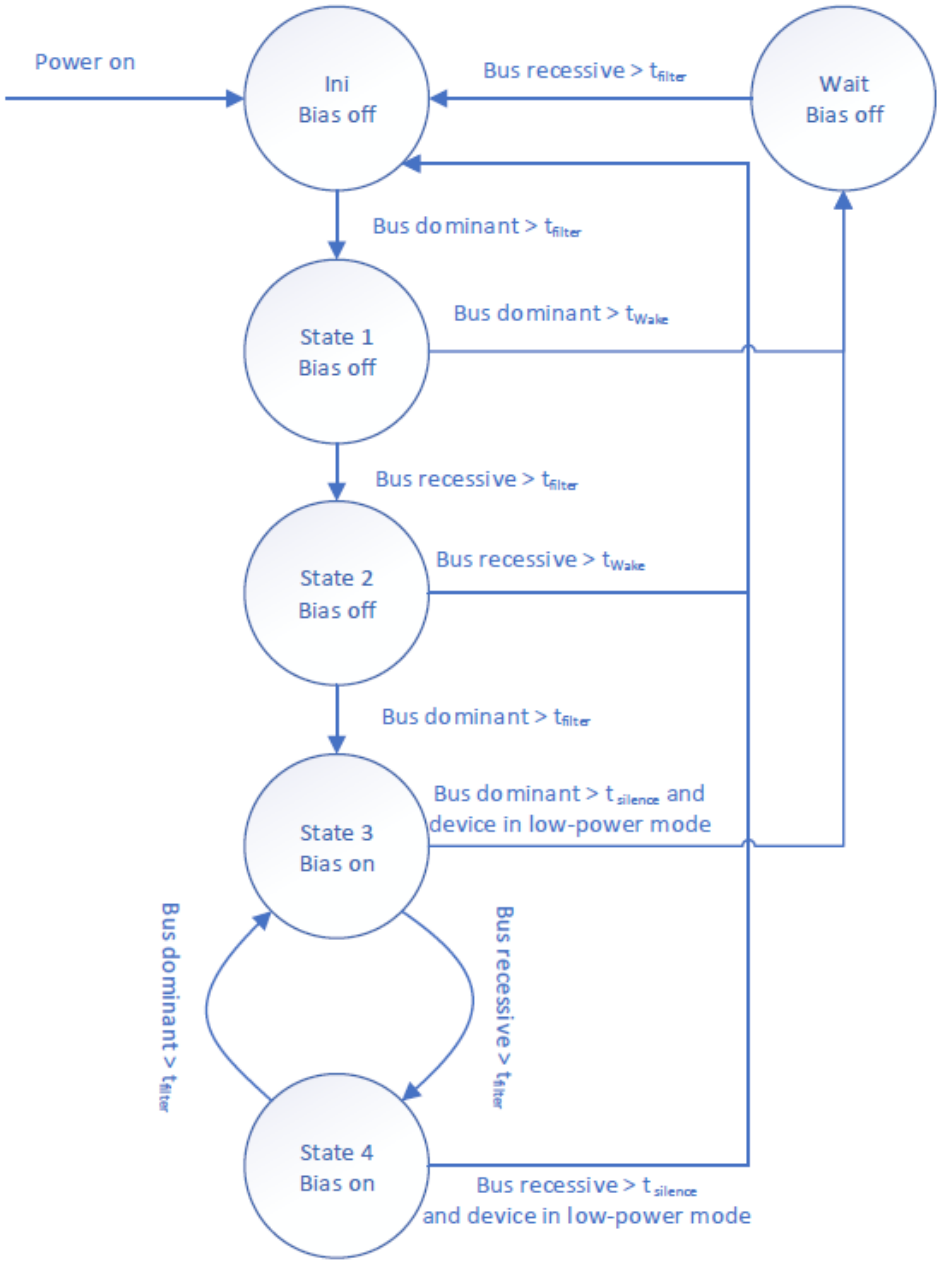
1.2.1 Wake-up pattern

Jednou ze změn, kterou norma ISO 11898-2:2016 přináší, je zavedení takzvaných „Wake-up patterns“ (WUP).

V původně platné normě 11898-5:2007 bylo podmínkou pro probuzení z nízkopříkonového režimu detekování dominantní úrovně alespoň po dobu v dokumentech označovanou jako t_{filter}. Tato doba byla normou definována jako 0,5 až 5 µs. Pro jisté probuzení bez ohledu na typ použitého transceiveru bylo tedy nutné vyvolat na sběrnici dominantní stav o trvání alespoň 5 µs. To je splněno u všech CAN rámců o bitové rychlosti 200 kbit/s a méně (trvání bitu 5 µs). V případě klasických základních rámců (CBFF) či klasických rámců s rozšířeným ID (CEFF) byla tato podmínka splněna i při 500 kbit/s (díky dominantní úrovni bitů RTR, IDE a FDF respektive RTR, FDF, r0). [3] [2]

Tento princip probouzení byl normou ISO 11898-2:2016 nahrazen za účelem zvýšení robustnosti tohoto mechanismu. [3]

Probuzení podle této normy je vyvoláno dvěma dominantními stavy o trvání t_{filter}, jež jsou odděleny recesivním stavem o trvání t_{filter}. Princip probouzení, a s tím spojené napájení sběrnice je znázorněno následujícím vývojovým diagramem.



Obr. 3: Stavový diagram WUP (wake-up pattern) podle ISO 11898-2:2016.

Jednotlivé stavy jsou ošetřeny tak, aby byl v případě poruch zajištěn přechod na výchozí stav „Ini“, či „Wait“. Pro detekci těchto poruch je zavedena doba t_{Wake} po jejíž uplynutí je vyhodnocen trvalý dominantní či recesivní stav.

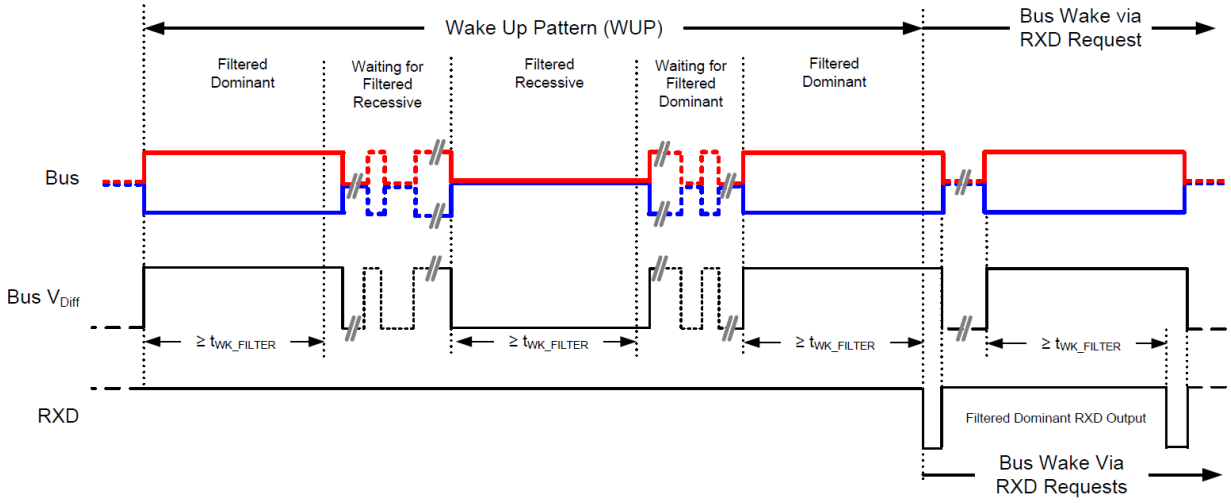
V případě, že po probuzení zařízení nedojde po době $t_{silence}$ k přijetí dominantního, resp. recesivního bitu, trvajícího alespoň t_{filter}, je zařízení vráceno do nízko-příkonového režimu přechodem do stavu „Wait“ resp. „Ini“.

Tab. 1: Definice minimálních a maximálních dob t_{filter}, t_{Wake} a $t_{silence}$ podle ISO 11898-2:2016.

	Min	Max
CAN activity filter time (t_{filter}), long	0,5 µs	5 µs
CAN activity filter time (t_{filter}), short	0,15 µs	1,8 µs
Wake time (t_{wake})	0,8 ms	10 ms
Silence time ($t_{silence}$)	0,6 s	1,2 s

Jak uvádí tabulka výše, stanovuje norma dvě možné hodnoty t_{filter}. I nadále je často užíváno delší doby, jež byla rovněž stanovena původní specifikací, jelikož disponuje vyšší odolností vůči náhodnému šumu.

Při použití protokolu CAN FD může být však vhodná kratší doba, jelikož implementace fyzické vrstvy podle některých norem (např. SAE J2284-4, SAE J2284-5, či SAE J1939-17) předepisuje bitovou rychlost v arbitrážní části 500kbit/s. Tato kratší doba t_{filter} pak umožňuje korektní probuzení.



Obr. 4: Detekce bitové posloupnosti, nutné pro probuzení u rámců CAN FD. Převzato z [4]

1.3 SAE J2284

Tento soubor norem se zaměřuje zejména na různé implementace fyzické vrstvy jak klasického CAN (J2284-1, J2284-2, J2284-3), tak i CAN FD (J2284-4, J2284-5). Jsou zde stanoveny požadavky na ECU a sběrnici tak, aby při splnění těchto podmínek systém dosahoval požadovaných parametrů.

V následující tabulce jsou zobrazeny jejich základní specifikace.

	J2284-1 revised	J2284-2 revised	J2284-3 revised	J2284-4 new	J2284-5 new
Frame Formats	Classical CAN only	Classical CAN only	Classical CAN only	CAN-FD and Classical CAN	CAN-FD and Classical CAN
CAN data bit rate [Mbit/s]	0.125	0.25	0.5	Data phase: 2 Arbitration: 0.5	Data phase: 5 Arbitration: 0.5
Number of bus nodes	32	32	24	24	2
Partial Networking (selective wakeup)	Optional	Optional	Optional	Optional	N/A
Minimum number of time quanta per bit	12	16	16	Arbitration: 40 Data phase: 10	Arbitration: 80 Data phase: 8
Stub length [m]	1.7	1.7	1.7	1.7	n/a
Publication date	NOV 2016	NOV 2016	NOV 2016	JUN 2016	SEP 2016

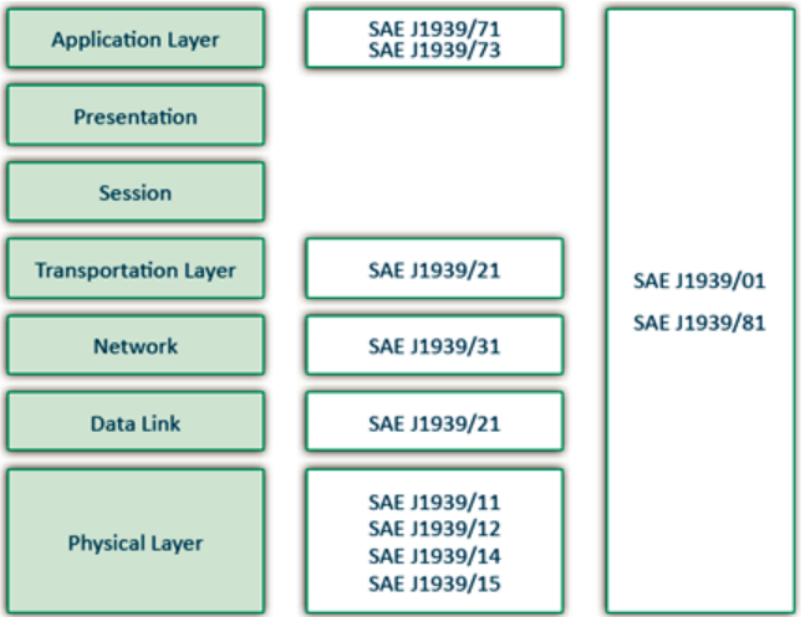
Obr. 5: Porovnání specifikací implementací fyzických vrstev podle jednotlivých standardů z řady SAE J2284. Převzato z [5].

Standard **J2284-4** z roku 2016, tedy stanovuje takové požadavky na ECU a sběrnici, aby při jejich splnění dosahoval systém arbitrážní rychlosti 500 kbit/s a datové rychlosti 2 Mbit/s. Maximální počet připojených jednotek přitom může být 24, s maximální délkou odbočky 1,7 m. Norma rovněž stanovuje minimální počet časových kvant.

Standard **J2284-5** specifikuje arbitrážní rychlost 500 kbit/s a datovou rychlost 5 Mbit/s. Povoluje však pouze topologii point-to-point (tedy dvě komunikující jednotky).

1.4 SAE J1939

Soubor norem, nesoucích označení SAE J1939, spadá pod skupinu takzvaných vyšších aplikačních vrstev. S aplikací SAE J1939 se setkáme zejména v oblasti nákladních automobilů a jiných těžkotonážních (např. konstrukčních) vozidel. Jak lze vidět z obr. 6, je tento protokol definován v několika dokumentech, zasahujících do různých vrstev ISO/OSI modelu (dle čísla normy lze rozpoznat příslušnost k dané vrstvě).



Obr. 6: Vztah některých norem SAE J1939 vzhledem k modelu ISO/OSI. (Některé z dalších příslušných norem zde nejsou uvedeny.) [6]

V následujícím textu je uveden alespoň stručný základní přehled náplně vybraných norem.

1.4.1 SAE J1939-1x

Tyto normy specifikují fyzické vrstvy různé požadavky na cílovou aplikaci. Jsou zde stanoveny bitové rychlosti a požadavky na topologii a provedení sběrnice. Rovněž specifikují maximální počet připojených jednotek.

V následující tabulce jsou stručně shrnuty některé parametry, určené vybranými specifikacemi. [7]

Tab. 2: Některé parametry specifikované v dokumentech SAE J1939-1x pro CAN protokol.

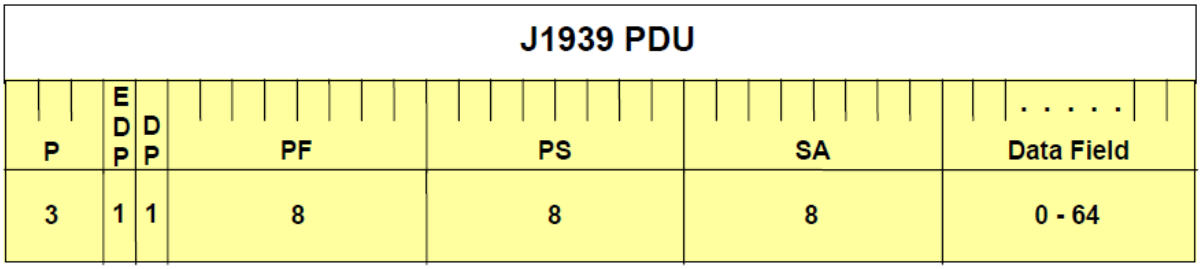
	Bitová rychlost	**Maximální délka sběrnice**	**Maximální délka odbočky**	**Maximální počet ECU**
J1939-11	250 kbit/s	40 m	1 m	30
J1939-14	500 kbit/s	40 m	1,67 m	30
J1939-15	250 kbit/s	40 m	3 m	10

1.4.1.1 J1939-17 (dosud nevydáno)

Do kategorie výše uváděných specifikací spadá i norma J1939-17. Tato norma doposud nebyla vydána, nicméně její náplní má být specifikace fyzické vrstvy pro sběrnice, využívající CAN FD. Podle dostupných informací stanovuje přenosovou rychlost na 500 kbit/s pro arbitrážní část a 2 Mbit/s pro datovou část.

1.4.2 J1939-21

Tato norma definuje použití klasického protokolu CAN s rozšířeným identifikátorem podle normy ISO 11898-1. Dále definuje pojem PDU (Protocol Data Units). Tyto PDU jednotky jsou tvořeny identifikátorem a datovou částí CAN rámce.



Obr. 7: Struktura PDU. Převzato z [8].

Na výše uvedeném obrázku je vidět rozdělení PDU. Pole identifikátoru se skládá se ze tří prioritních bitů (P), „extended data page“ bitu (EDP), „data page“ bitu (DP), PDU formátu (PF), PDU specifikace (PS) a zdrojové adresy (SA). Další bity, používané v CAN protokolu nejsou na obrázku uvedeny, nicméně jsou samozřejmě součástí rámce. Následuje datové pole.

Bity EDP, DP,PF a PS tvoří takzvané číslo skupiny parametrů - PGN (Parameter Group Number). Tyto skupiny parametrů sdružují spolu související signály. Jednotlivé PGN jsou uvedeny v normě SAE J1939-71, kde je rovněž uvedeno mapování k nim příslušejících signálů do zpráv.

Norma specifikuje dva formáty PDU:

Tab. 3: Dělení PDU podle PDU Format Field.

	PDU Format Field (PF)	**PDU Specific Field (PS)**
PDU 1:	0-239	Cílová adresa (Destination Address)
PDU 2:	239-255	Skupinové rozšíření (Group Extension)

Jestliže se jedná o formát PDU 1, je polem PS specifikována cílová adresa konkrétního příjemce. V případě PDU 2 specifikuje pole PS společně se čtyřmi LSB pole PF (celkem tedy 12 bitů) 4096 PGN. Protokol podporuje pět druhů zpráv:

- Command
- Request
- Broadcast/Response
- Acknowledgement
- Group Functions

Jelikož klasický CAN rámec dovoluje přenos maximálně 8 bajtů v jedné zprávě, zavádí tento dokument transportní protokol (TP). Díky tomu lze pomocí více zpráv odeslat data o maximální délce 1785 bajtů. [9] [6]

1.4.3 J1939-22 (dosud nevydáno)

Tato doposud nevydaná norma, specifikuje použití CAN FD rámců s rozšířeným identifikátorem (FEFF).

Norma je zpracovávána na základě specifikace CiA 602-2 (více v kapitole Mapování J1939 do CAN FD).

1.4.4 J1939-71

V tomto dokumentu, popisujícím aplikační vrstvu protokolu J1939, je popsáno umístění parametrů podle SPN (Suspect Parameter Number) do datových polí PGN, perioda jejich odesílání, a pravidla pro ASCII parametry.

V současné době lze nalézt všechny definice PGN a SPN v elektronickém dokumentu s označením **SAE J1939DA**.

Příklad konkrétního PGN je ukázán na následujícím obrázku.

pgn65262 - Engine Temperature 1 - ET1 -

Transmission Repetition Rate:		1 s	
Data Length:		8 bytes	
Data Page:		0	
PDU Format:		254	
PDU Specific:		238	
Default Priority:		6	
Parameter Group Number:		65262 ($00FEEE_{16}$)	
Bit Start Position /Bytes	Length	SPN Description	SPN
1	1 byte	Engine Coolant Temperature	110
2	1 byte	Fuel Temperature	174
3-4	2 bytes	Engine Oil Temperature 1	175
5-6	2 bytes	Turbo Oil Temperature	176
7	1 byte	Engine Intercooler Temperature	52
8	1 byte	Engine Intercooler Thermostat Opening	1134

Obr. 8: Příklad specifikace PGN normou J1939/71 (dříve), nyní J1939DA. [10]

Pro každý parametr SPN jsou pak popsána jeho délka, rozlišení, rozsah hodnot a příslušné PGN. Rovněž je uvedeno, zda se jedná o měřený parametr („measured“), či stavový („state“).

spn110 - Engine Coolant Temperature - Temperature of liquid found in engine cooling system.

Data Length:	1 byte
Resolution:	1 deg C/bit , -40 deg C offset
Data Range:	-40 to 210 deg C
Type:	Measured
Suspect Parameter Number:	110
Parameter Group Number:	[65262]

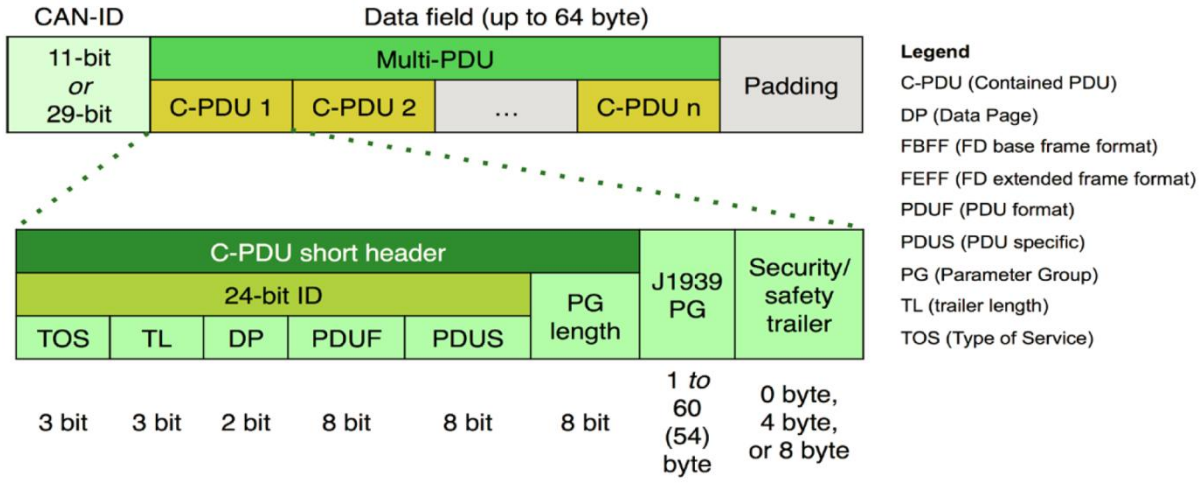
Obr. 9: Příklad definice SPN. [10]

1.4.5 Mapování J1939 do CAN FD (CiA 602-2)

Výše popisované mapování PDU do rámce CAN se vztahuje pouze na klasické rámce CAN s rozšířeným identifikátorem (CEFF). S požadavky na větší datovou propustnost a příchodem CAN FD bylo však vhodné využít výhody tohoto protokolu.

Organizace CiA (CAN in Automation) proto v dokumentu CiA 602-2 definuje způsob mapování PDU do CAN FD rámců. Pro tento účel zavádí pojmy C-PDU (Contained PDU) a Multi-PDU.

Princip je znázorněn na následujícím obrázku. Delší datové pole protokolu dovoluje přenášet více C-PDU najednou. Prakticky je v současné době využíváno 8 bajtového pole PG (aby byla zachována kompatibilita s Autosar), a je tedy možno přenést až 5 C-PDU v rámci jednoho Multi-PDU. Tento způsob mapování podporuje použití jak 11 bitového, tak 29 bitového identifikátoru, přičemž pole zdrojové adresy SA (Source Address) je v obou případech umístěno do identifikátoru. [11]



Obr. 10: Znázornění způsobu mapování C-PDU do datového pole CAN-FD. [11]

Oproti klasickému PDU rámci podle J1939-21 jsou zde navíc 3 bitová pole TOS (Type of Service) a TL (Trailer Length), nesoucí informace o volitelném poli „Security/safety trailer“ a jeho délce. Dále je v rámci třeba specifikovat délku pole PG (Parameter Group). K tomu slouží pole „PG length“. [11]

1.5 CANopen FD

CANopen FD je vyšší aplikační vrstva CAN FD, vycházející z hojně rozšířeného protokolu CANopen. Její specifikaci nalezneme v dokumentu CiA 1301 (momentálně verze 1.0.0). Tato aplikační vrstva byla vytvořena tak, aby byla zachována podobnost s CANopen a zároveň bylo využito možností protokolu CAN FD. Rámce CAN FD, použité pro přenos mohou, ale nemusejí využívat přepínání bitové rychlosti v datovém poli, Oproti CANopen však nejsou povoleny tzv. remote frames.

1.5.1 Fyzická vrstva

Fyzická vrstva je realizována podle normy ISO 11898-2:2016, umožňující bitovou rychlost až 5 Mbit/s. Volitelně je povolena podpora nízkopříkonových módů a funkce selektivního probouzení (selective wake-up). Další návrh po stránce hardware se řídí dokumenty CiA 6xx.

1.5.2 Objekty CANopen FD

V této kapitole budou popsány níže uvedené objekty CANopen FD a jejich odlišnosti od CANopen:

- USDO
- PDO
- NMT
- SYNC
- TIME write
- EMCY write
- Error control

1.5.2.1 USDO

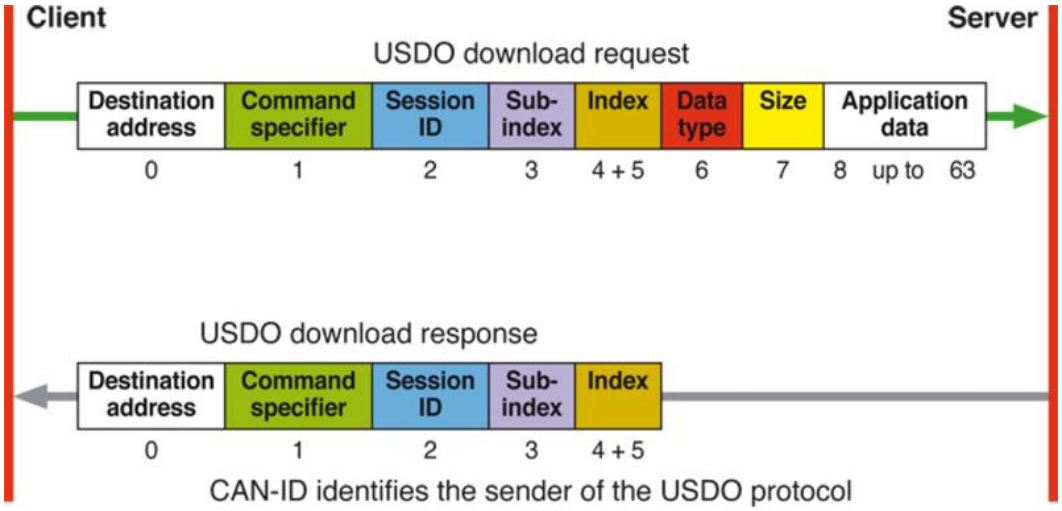
USDO (Universal Service Data Object) představuje modifikaci objektu SDO (zpětná kompatibilita však není zachována), jehož primárním účelem je přenos konfiguračních a diagnostických dat.

Původní SDO protokol slouží pro čtení či zápis do slovníku objektů (object dictionary) druhého uzlu (označovaného jako server). Jedná se tedy o „unicast“ komunikaci mezi dvěma uzly. Protokol USDO oproti tomu umožňuje kromě „unicast“ komunikace rovněž „multi-„ či „broadcast“. Další výhoda USDO spočívá v možnosti přístupu více klientů k objektu „object dictionary“ téhož serveru (kromě přístupu k témuž objektu slovníku).

Přenos dat může být realizován třemi způsoby:

- **Expedited protocol**: Umožňuje přenést maximálně 56 bajtů dat (viz obr. 11)
- **Segmented protocol**: Protokol umožňující přenesení většího množství dat, přičemž příjemce dat každý přenesený segment (blok dat) potvrzuje.
- **Bulk protocol**: Podobně jako segmentovaný přenos umožňuje přenesení většího množství dat, přičemž příjemce dat potvrzuje pouze první a poslední přenesený blok.

Jak segmentovaný, tak „bulk“ protokol pak podporují „stream“, tedy přenos dat o nedefinované délce. Takový přenos nese v poli specifikujícím délku dat hodnotu 0xFFFFFFFF a přenos pak trvá tak dlouho, dokud jej klient nepřeruší, nebo server neodešle „abort“ zprávu.



Obr. 11: Znázornění příkladu přenosu dat klient – server pomocí USDO protokolu. Převzato z [12].

Na výše uvedeném obrázku vidíme příklad „download reqest“ rámce, kde „destination address“ určuje, zda se jedná o unicast, multicast či broadcast. „Command specifier“ pak určuje typ použitého přenosu. „Session ID“ udává, číslo konkrétního přenosu a umožňuje tak například více paralelních přenosů mezi jedním klientem a jedním serverem. „Sub-index“ a „Index“ má totožný význam, jako u SDO protokolu. U USDO protokolu však navíc ještě nalezneme specifikaci dat „Data type“ a „Size“, což umožňuje, aby příjemce provedl kontrolu těchto dat. [13]

1.5.2.2 PDO

Process data object (PDO) je v síti CANopen FD velice podobný klasickému PDO využívanému CANopen. Je zde však využíváno možnosti CAN FD rámce přenést až 64 bajtů dat.

PDO přenos je vyvoláván následujícími událostmi:

- **Událostí nebo časovačem**: Přenos po nastání stanovené události, či uplynutí doby časovače.
- **Cyklický synchronní přenos**: Vyvolán přijetím synchronizační (SYNC) zprávy.
- **Acyklický synchronní přenos**: Přenos po přijetí synchronizační zprávy vyvolán pouze pokud došlo k události.
-

Oproti původnímu CANopen již nelze pro vyvolání PDO přenosu využít požadavek jiného zařízení pomocí „Remote Transmission Request“.

Stejně jako u původního PDO klasického CANopen existují tři druhy mapování proměnných: statické, variabilní a dynamické. [14]

1.5.2.3 NMT

Network Management (NMT) CANopen FD zůstává stejný, jako u klasického CANopen. Každé zařízení se může nacházet v jednom ze stavů:

- Initialization
- Pre-operational
- Operational
- Stopped

Objektem, zajišťujícím network management je **NMT** zpráva. Tato zpráva s identifikátorem 0 (tedy s nejvyšší prioritou) nese dva bajty dat, přičemž v prvním bajtu je zakódován požadavek na následující stav uzlu. Adresa uzlu, kterému je tato zpráva určena, je pak specifikována ve druhém bajtu (je-li nulová, je zpráva určena všem uzlům). [14]

1.5.2.4 Error control

Error control protokol je zajišťován pomocí Boot-up protokolu a Heartbeat protokolu. Oproti CANopen zde není implementován Node/Life guarding, jelikož zasílání „Remote Request“ rámců je zakázáno.

- **Boot-up**: Tato zpráva je odeslána při dokončení inicializace, těsně před vstupem do „pre-operational“ stavu. Je jí indikován vstup nového uzlu do CAN FD sítě a nenese žádná data
- **Heartbeat**: Jedná se o zprávu se stejným identifikátorem jako boot-up, která je periodicky odesílána. Oproti boot-up zprávě nese i informaci, ve kterém ze stavů se právě zařízení nachází.

1.5.2.5 SYNC

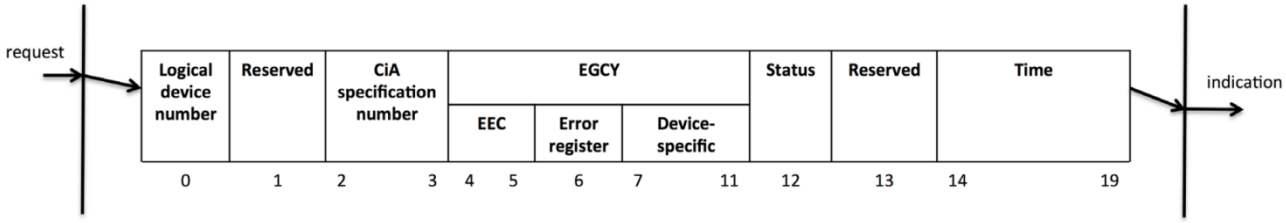
Objekt SYNC, jenž zajišťuje synchronizaci uzlů, se pro CANopen FD neliší od původního CANopen. Jedná se o zprávu, vysílanou jedním z uzlů jako „broadcast“ vysílání, které může přijmout jakákoliv připojená jednotka. Na přijetí této zprávy pak může příjemce reagovat například vysláním PDO zprávy. Tato zpráva může nést jeden bajt dat, obsahující hodnotu čítače, což platí rovněž pro CANopen podle specifikace CiA 301 verze 4.1 a novější. [14]

1.5.2.6 TIME write

TIME write objekt CANopen FD neobsahuje oproti CANopen žádné změny. Jedná se o nástroj umožňující globální nastavení času uzlů podle uzlu, jenž tuto zprávu odesílá. Zpráva nese šest bajtů dat, vyjadřujících počet milisekund uplynulých od 1. 1. 1984. [14]

1.5.2.7 EMCY write

Pro implementaci EMCY protokolu v síti CANopen FD bylo využito možnosti odeslat více dat oproti klasickému CAN. Původní rámec, nesoucí dva bajty EMCY error kódu (EEC), error registr a pět bajtů informací stanovených dle potřeby vývojáře, je stále součástí „nového“ EMCY rámce. Tato data byla dále rozšířena o další dodatečné informace (viz obr. 13).



Obr. 12: Rámec datového pole EMCY protokolu. Přejato z [14]

Přidanými poli v datovém rámci jsou „Local device number“ stanovující konkrétní část zařízení, kde k chybě došlo, „CiA specification number“, specifikující error kód, „Satus“, vypovídající to povaze chyby a „Time“, jenž udává čas, kdy k danému chybě došlo.

2. CAN FD

Výše zmiňovaná norma ISO 11898-1:2015 popisuje jak klasický CAN rámec, tak i CAN FD. V tomto oddílu práce bude popsán protokol CAN FD a zdůrazněny jeho odlišnosti oproti klasickému CAN rámci.

2.1 Základní vlastnosti

Protokol CAN FD vychází z původního protokolu CAN. Základní znaky jsou tedy pro oba protokoly společné.

Jedná se o multi-master sběrnici, kde o prioritě vysílaných zpráv rozhoduje jejich identifikátor. Konkrétně se využívá CSMA/CD+AMP (Carrier Sense Multiple Access/Collision Detection with Arbitration On Message Priority).

Oproti původnímu protokolu CAN zde došlo k zavedení dvou přenosových rychlostí - pro arbitrážní a datovou část rámce, díky čemuž lze přenášet data vyšší rychlostí (v současné době jsou k dostání budiče do rychlosti 8 Mbit/s). Přenosová rychlost v arbitrážní části je stejně jako u původního protokolu ponechána maximálně 1 Mbit/s.

Rovněž byla navýšena maximální přenášená délka dat na 64 bajtů.

2.2 Formáty datových rámců

Norma stanovuje celkem čtyři formáty datových rámců:

- CAN base format (CBFF)

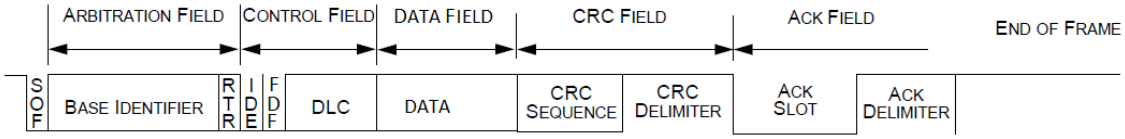
- CAN extended format (CEFF)
- CAN FD base format (FBFF)
- CAN FD extended format (FEFF)

Každý z těchto formátů se skládá z pěti polí:

- Arbitrážního pole
- Řídícího pole
- Datového pole
- CRC pole
- „Acknowledge“ pole

2.2.1 Formáty rámce klasického CAN

Pro porovnání rámců CAN FD a klasického CAN uveďme nejprve podobu „klasického“ CAN rámce.



Obr. 13: Formát klasického CAN v základním formátu.

Na začátku každé zprávy je odvysílán **SOF** (Start Of Frame) bit. Tento bit je vždy přenášen jako dominantní. Za ním následuje 11 bitů identifikátoru, odesílaného MSB napřed (je tedy odesílán 28. až 18. bit identifikátoru).

Po přenesení základního identifikátoru je přenesen bit **RTR** (Remote Transmision Request). Tím je rozlišováno mezi běžnou zprávou přenášející data (dominantní úroveň), a žádostí o zpětné zaslání dat (recesivní úroveň). Tato funkcionalita žádosti o zprávu se však v praxi využívá jen zřídka.

V následujícím řídícím poli je přenášen bit **IDE**, jež rozlišuje mezi standardním a rozšířeným formátem a bit **FDF** (ve starší specifikace CAN FD označovaný EDL – „Extended Data Length“)**.** Bit FDF (FD Format) se v rámci vyskytuje až s novou verzí ISO 11898-1:2015, a stanovuje, zda se jedná o klasický rámec, či rámec CAN FD. V kontrolním poli je rovněž přenášen nibble **DLC**, stanovující počet bajtů přenášených v datovém poli. [1] [14]

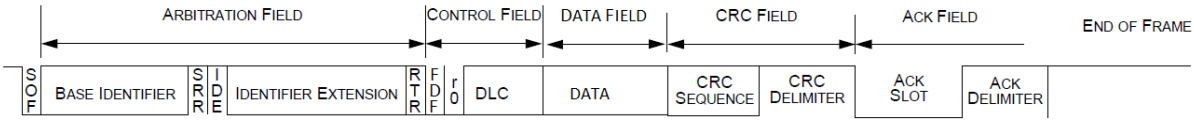
Tab. 4: Definování počtu přenášených bajtů v poli DLC pro klasický CAN rámec.

Počet datových bajtů	DLC3	DLC2	DLC1	DLC0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	-	-	-

Po datovém poli následuje pole CRC (Cyclic Redundancy Check), složené z vlastního vypočteného CRC (CRC Sequence) a recesivního bitu CRC Delimiter.

Následně je v dvoubitovém poli „ACK Field“ potvrzeno přijetí správných dat. Vysílací stanice tyto bity odesílá jako recesivní, zatímco přijímací stanice potvrdí správné přijetí dat vnucením dominantního stavu na sběrnici.

Klasický CAN rámec s rozšířeným formátem je znázorněn na následujícím obrázku.

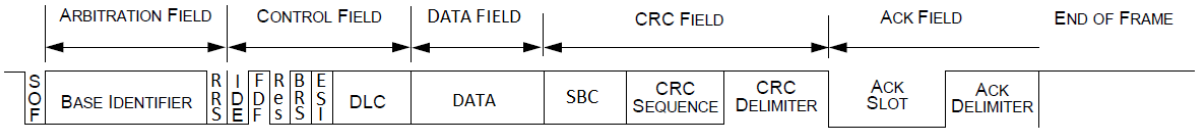


Obr. 14: Formát rozšířeného rámce CAN.

Jelikož platí, že klasické rámce mají oproti rozšířeným vyšší prioritu, je za prvními 11 bity identifikátoru namísto bitu RTR (Remote Transmision Request), přenášen bit **SRR** (Substitute Remote Request), jež je vždy recesivní, čímž je priorita standardního rámce zajištěna. Po bitu IDE pak následuje dalších 18 bitů identifikátoru (tedy 17. až 0. bit identifikátoru).

2.2.2 Formáty rámce CAN FD

Na následujícím obrázku je znázorněn rámec CAN FD se standardním identifikátorem.



Obr. 15: Rámec CAN FD se standardním 11 bitovým identifikátorem.

Jak je vidět, došlo oproti klasickému CAN rámci k několika změnám. V arbitrážní části se namísto RTR bitu vyskytuje bit **RRS** (Remote Request Substitution). Protokol CAN FD neumožňuje odesílání žádostí o zprávu (Remote frames), a tudíž je tento bit vždy odesílán jako dominantní, tedy log. 0.

Bit FDF je v případě rámce CAN FD vždy odesílán rescesivní (log. 1), a je následován bitem **Res**. Ten je naproti tomu přenášen jako dominantní a vzniká tak sestupná hrana, používaná pro takzvanou „hard synchronizaci“.

Přepínání přenosové rychlosti datové části je signalizováno recesivním stavem (log. 1) bitu **BRS** (Bit Rate Switch). Po něm následuje bit **ESI** (Error State Indicator), jež informuje o chybových stavech vysílače. V případě stavu „error active“ je bit přenášen jako dominantní (log. 0), zatímco v případě „error passive“ jako resesivní (log. 1).

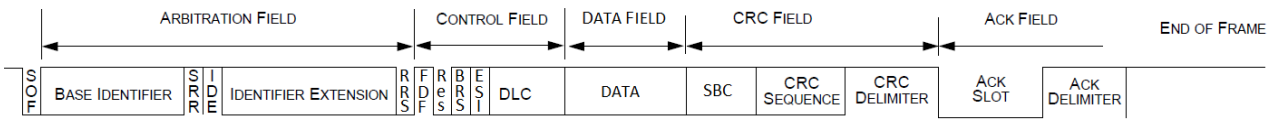
Pole DLC (Data Length Count) má stejný význam jako v případě klasického CAN. Zde je pouze rozšířené tak, aby bylo využito všech binárních kombinací a bylo umožněno odesílání až 64 bajtů. Bitové kombinace odpovídající jednotlivým délkám zpráv lze vidět v tab. 5.

Tab. 5: Tabulka závislosti počtu přenášených bajtů na bitové kombinaci pole DLC.

Počet datových bajtů	**DLC3**	**DLC2**	**DLC1**	**DLC0**
0-8	viz klasický CAN			
12	1	0	0	1
16	1	0	1	0
20	1	0	1	1
24	1	1	0	0
32	1	1	0	1
48	1	1	1	0
64	1	1	1	1

Význam zbývajících polí rámce zůstává stejný, jako u klasického CAN rámce.

Použití CAN FD rámce s rozšířeným identifikátorem je obdobně jako u klasického CAN signalizováno bitem **IDE**. Takový rámec pak rovněž používá 29 bitový identifikátor.



Obr. 16: Rámec CAN FD s rozšířeným identifikátorem.

2.3 Kódování a bit stuffing

Přenášené bity jsou stejně jako u protokolu CAN kódovány jako NRZ (Non Return to Zero). Jejich hodnota se tedy v rámci doby trvání bitu nemění. Jelikož by při po sobě následující sekvenci více bitů stejné polarity mohly nastat problémy při synchronizaci, je zaváděn bit stuffing.

CAN FD oproti původnímu protokolu rozeznává dva druhy stuff-bitů:

- Dynamické stuff-bity
- Fixní stuff-bity

Dynamické stuff-bity se řídí stejnými pravidly jako u klasického CAN protokolu. Je-li tedy posloupnost pěti po sobě následujících bitů shodná, je jako následující bit automaticky vložen komplementární bit.

Fixní bit stuffing nachází uplatnění v poli CRC, kde je komplementární bit vložen mezi každé čtyři bity i v případě, že tyto byty měly různé hodnoty.

Oba tyto druhy bit stuffingu jsou řešeny v rámci MAC a jsou důležité při synchronizaci (viz kapitola Synchronizace a kompenzace zpoždění transceiveru).

2.4 CRC

Klasický CAN rámec používá 15 bitové CRC (Cyclic Redundancy Check) tak, aby byla zajištěna Hammingova vzdálenost 6. Díky tomu by měl být tento kontrolní součet schopen detekovat pět náhodných chybových bitů.

$$g15 = x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + x^0$$

Vlivem bit stuffingu však může u tohoto CRC dojít za jistých okolností ke snížení Hammingovy vzdálenosti na 2 a tím i k možnosti, že dvojice bitových chyb zůstane neodhalena.

Nový protokol CAN FD podle ISO 11898-1:2015 tento problém řeší zavedením dvou druhů bit stuffingu. Takzvaný dynamický bit stuffing, jež je principielně shodný jako u klasického CAN, je pak zahrnut do výpočtu CRC. Před samotnou sekvenci CRC je vloženo čtyřbitové pole **SBC** (Stuff Bit Count). V tomto poli je uložen počet stuff bitů, který je zapsán jako modulo osmi skutečného počtu dynamických stuff bitů, a je kódován Greyovým kódem. Počet stuff bitů je kódován do tří bitů, přičemž zbývající čtvrtý bit SBC pole je paritní bit. [14] [1]

V celém poli CRC je pak namísto standardních dynamických stuff bitů zaveden takzvaný fixní bit stuffing (viz obr. 18). Jak napovídá název, je pozice těchto fixních stuff bitů (**FSB**) dána pevně, a to každý pátý bit, přičemž první FSB bit je na samém začátku CRC pole. Polarita těchto bitů je opačná, než je polarita předcházejícího bitu. [14]

Jelikož protokol CAN FD umožňuje odesílat relativně velká datová pole, musel být rovněž modifikován vlastní výpočet CRC tak, aby byla zachována Hammingova vzdálenost 6. Z tohoto důvodu se liší výpočet CRC pro délku datového pole do 16 bajtů od výpočtu pro délky přesahující 16 bajtů.

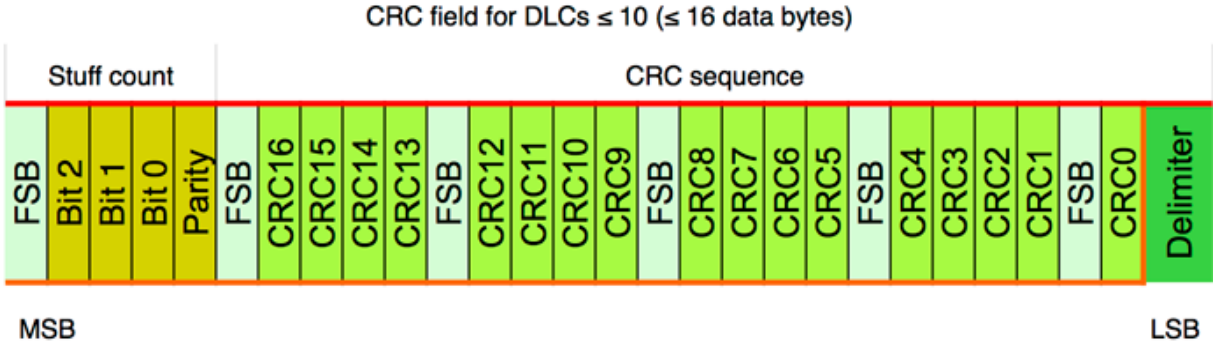
Polynom pro výpočet CRC při délce přenášených dat do 16 bajtů (včetně):

$$g17 = x^{17} + x^{16} + x^{14} + x^{13} + x^{11} + x^6 + x^4 + x^3 + x^1 + x^0$$

Polynom pro výpočet CRC při délce přenášených dat delších než 16 bajtů:

$$g21 = x^{21} + x^{20} + x^{13} + x^{11} + x^7 + x^4 + x^3 + x^0$$

Je tedy zřejmé, že se bude lišit i délka CRC pole v závislosti na délce datové části CAN FD rámce. Na následujícím obrázku je zobrazeno CRC pole pro velikost datového pole do 16 bajtů.

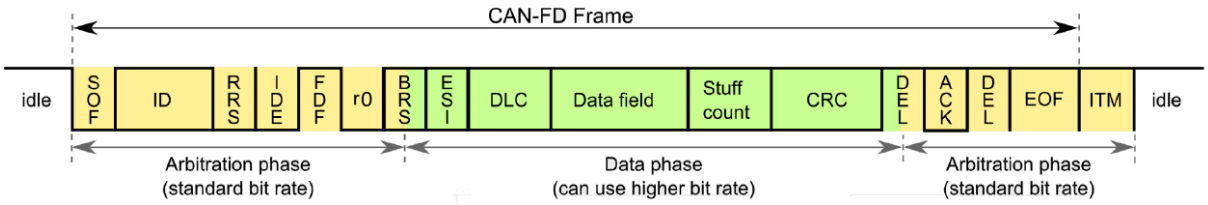


Obr. 17: CRC pole protokolu CAN FD pro počet přenášených datových bajtů do 16 bajtů. Převzato z [14].

Na konci pole CRC se nachází CRC delimiter. Jedná se o normou pevně stanovené pole, jež slouží pro potřeby kontroly chyby rámce. V případě, že přijímač nedetekuje v tomto poli očekávanou hodnotu, nastává chyba rámce (form error).

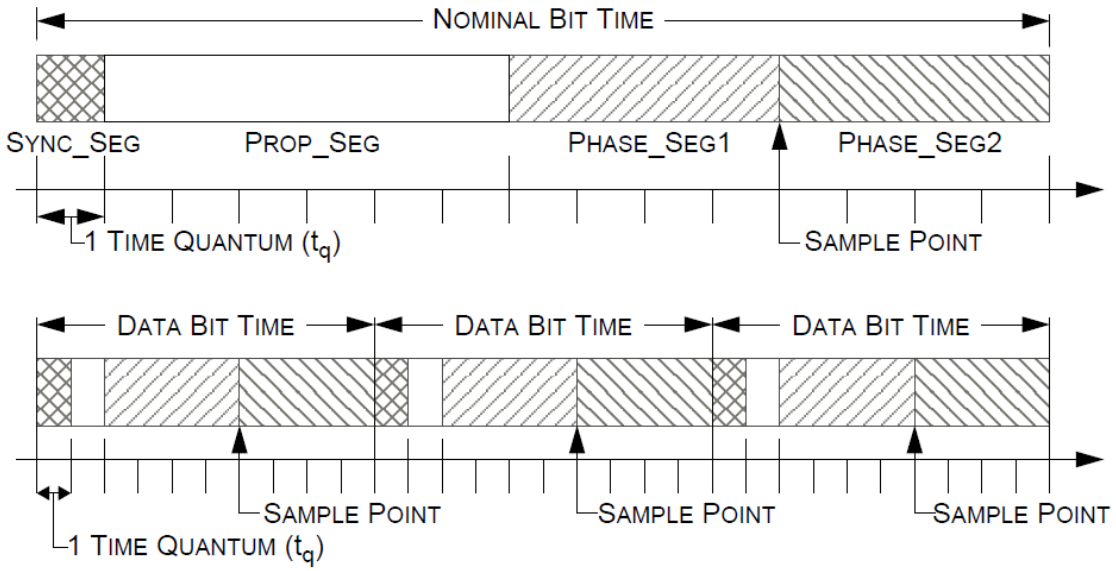
2.5 Přepínání bitové rychlosti

Protokol CAN FD umožňuje přepínání bitové rychlosti v datové části zprávy. Datovou částí je zde myšleno nejen datové pole nesoucí vlastní data, ale rovněž pole CRC (viz obr. 18). Platí přitom, že bitová rychlost v datové části musí být vždy vyšší či rovna bitové rychlosti v arbitrážní části rámce.



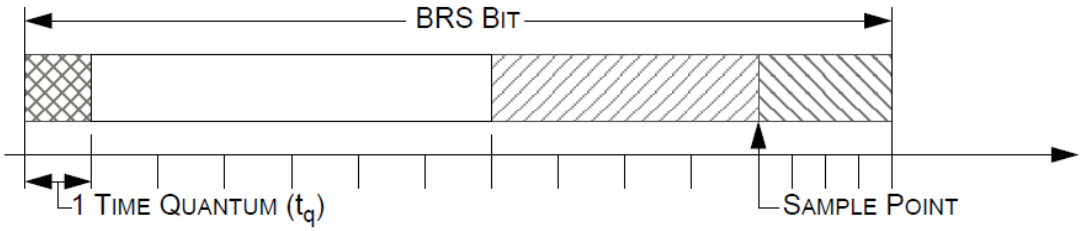
Obr. 18: Rozdělení klasického rámce CAN FD na arbitrážní a datovou fázi, podle použitelné bitové rychlosti. [14]

Změna rychlosti se může týkat jak změny počtu časových kvant, tak jejich trvání (pro časování v datové a arbitrážní části může být použit rozdílný dělící poměr hodinového signálu – viz obr. 19). V praxi je však často doporučováno volit stejné trvání časového kvanta a změnu rychlosti v datové části provádět pouze změnou počtu časových kvant.



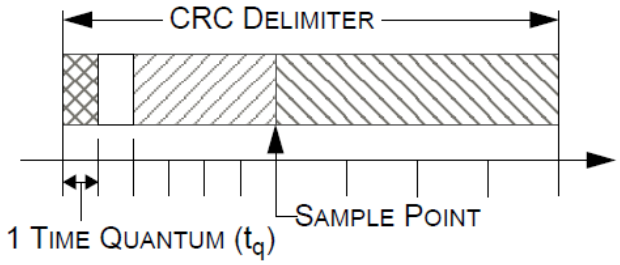
Obr. 19: Příklad, ukazující rozdílné časování v arbitrážní části (nominal bit time) a v datové části. Převzato z [15].

Přijímač tuto změnu rychlosti rozpozná pomocí bitu BSR (Bit Rate Switch). Jestliže je navzorkována jeho recesivní hodnota, je po tomto navzorkování okamžitě přepnuto na datovou rychlost, a tedy i „Fázový segment 2“ je přenesen s časováním, odpovídajícím bitové rychlosti datové části.



Obr. 20: Znázornění změny v době časového kvanta po navzorkování BRS bitu. Převzato z [15].

Konec části se zvýšenou bitovou rychlostí nastává s bitem CRC delimiter. Po navzorkování tohoto bitu přijímač přepne do původní arbitrážní bitové rychlosti.



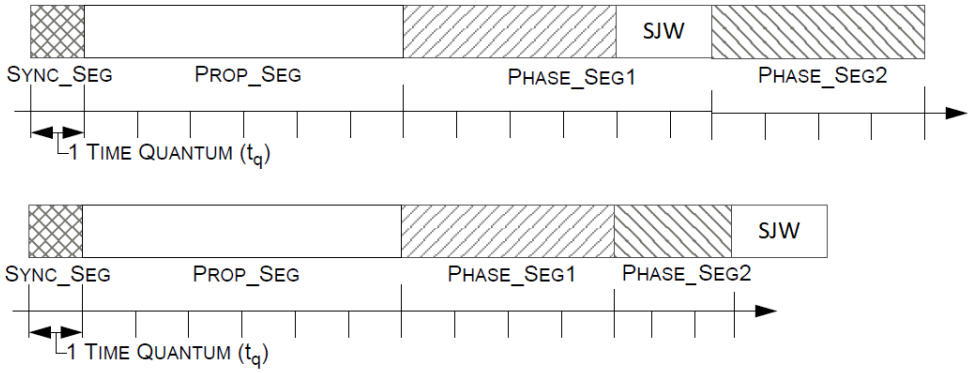
Obr. 21: Znázornění změny v době časového kvanta po navzorkování CRC delimiter bitu. Převzato z [15] a upraveno.

2.6 Synchronizace a kompenzace zpoždění transceiveru

V CAN FD protokolu rozlišujeme takzvanou „soft synchronizaci“ a „hard synchronizaci“. Jako „hard sychronizace“ je označována synchronizace podle doběžné hrany, podle které jsou hodiny receiveru synchronizovány. K tomu dochází při detekci sestupné hrany během stavů Bus Idle, Suspend Transmission a Intermission. Touto synchronizací je zaručeno, že tato sestupná hrana bude ležet v poli synchronizačního segmentu (na obr. 22 značeno SYNC_SEG).

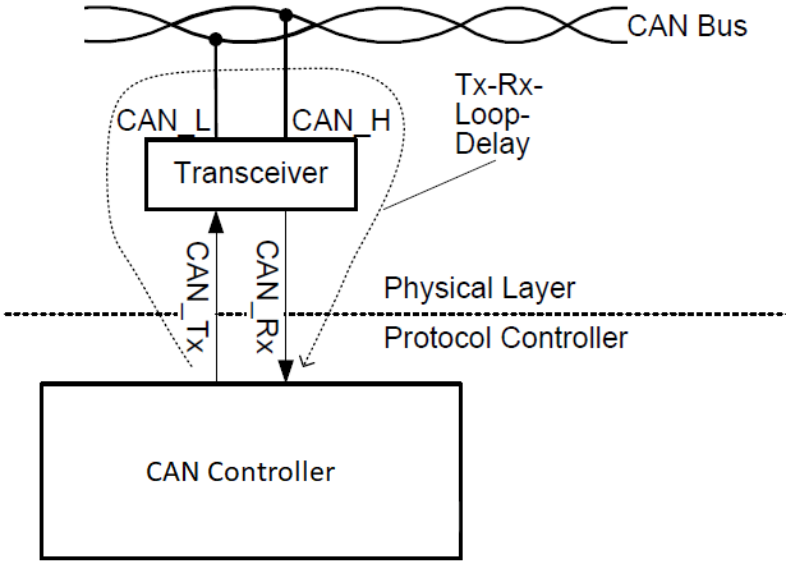
V rámcích CAN FD je požita ještě jedna dodatečná „hard synchronizace“. Důvodem je synchronizování před případným přechodem na vyšší bitovou rychlost v datové části rámce. K této synchronizaci dochází při přechodu z FDF bitu na bit r0. Jelikož je v CAN FD rámcích FDF bit vždy přenášen jako recesivní, zatímco bit r0 vždy jako dominantní vzniká hrana, jež je pro tuto synchronizaci použita.

„Soft synchronizací“ se pak rozumí synchronizace podle následujících doběžných hran přenášeného rámce. Rozdíl oproti „hard synchronizaci“ je v tom, že hodinový signál receiveru je schopen synchronizace pouze do rozsahu definovaného úseku SJW (Synchronization Jump Width). Tento proces je uskutečněn prodloužením fázového segmentu 1, či naopak zkrácením fázového segmentu 2 (viz obr. 22).



Obr. 22: Znázornění soft synchronizace pomocí SJW. [15]

Sestupná hrana mezi FDF a r0, používaná pro hard synchronizaci rovněž slouží pro určování zpoždění transceiveru (propagation delay). Toto zpoždění je naznačeno na následujícím obrázku.

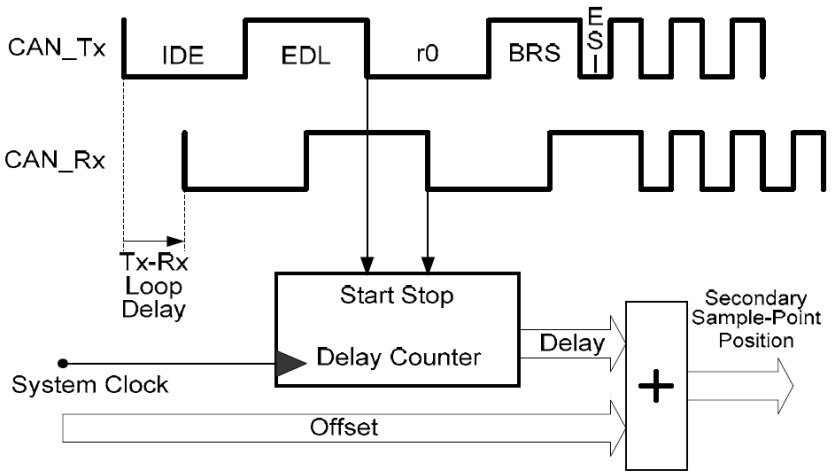


Obr. 23: Znázornění zpoždění mezi Rx a Tx piny CAN kontroléru, vlivem budiče. Převzato a upraveno z [16]

Každý transceiver musí kromě vysílání jednotlivých bitů rovněž vysílaný bit zpětně přečíst. Tento mechanismus je klíčový například při arbitráži. V arbitrážní části dochází ke čtení těchto bitů během „běžného“ vzorkovacího bodu (stejně jako u klasického CAN). To je důvod omezení maximální přenosové rychlosti.

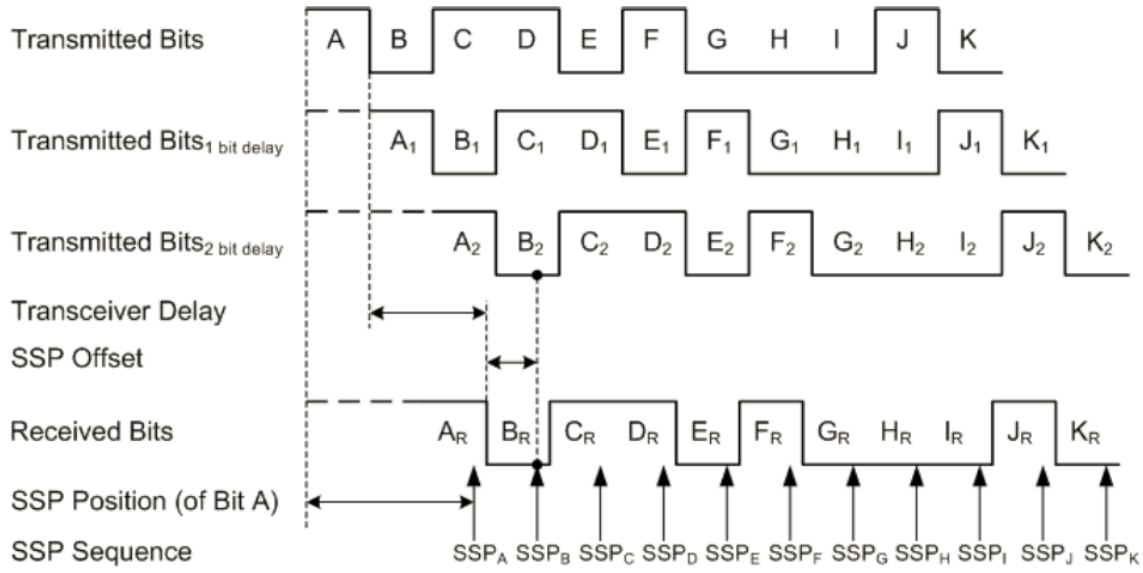
Z toho důvodu byla vytvořena funkce kompenzačního mechanismu zpoždění transceiveru (Transceiver Delay Compensation mechanism). Tento mechanismus zavádí dodatečný vzorkovací bod **SSP** (Secondary Sample Point). CAN FD kontrolér je schopen díky

hraně mezi FDF (dříve EDL) a r0 změřit prodlevu Tx-Rx, a v závislosti na této době zvolit s použitím offsetu vhodnou pozici SSP. Princip je znázorněn na následujícím obrázku.



Obr. 24: Princip tvorby SSP, pomocí měření prodlevy mezi vstupy Rx a Tx. [17]

Hodnoty odvysílaných bitů v datové části, jež jsou takto navzorkovány, jsou zpětně porovnány s již odvysílaným bitem (prodleva může být i delší než jeden bit) a v případě neshody je při následujícím vzorkovacím bodě tato informace předána konečnému stavovému automatu kontroléru.



Obr. 25: Princip vzorkování s využitím SSP v datové části rámce CAN FD. Převzato z [16].

Na výše uvedeném obrázku je zobrazen princip vzorkování odvysílaných bitů v datové části. Tento princip kontroly odvysílaných bitů umožňuje v datové fázi použití vyšší přenosové rychlosti.

2.7 Error management

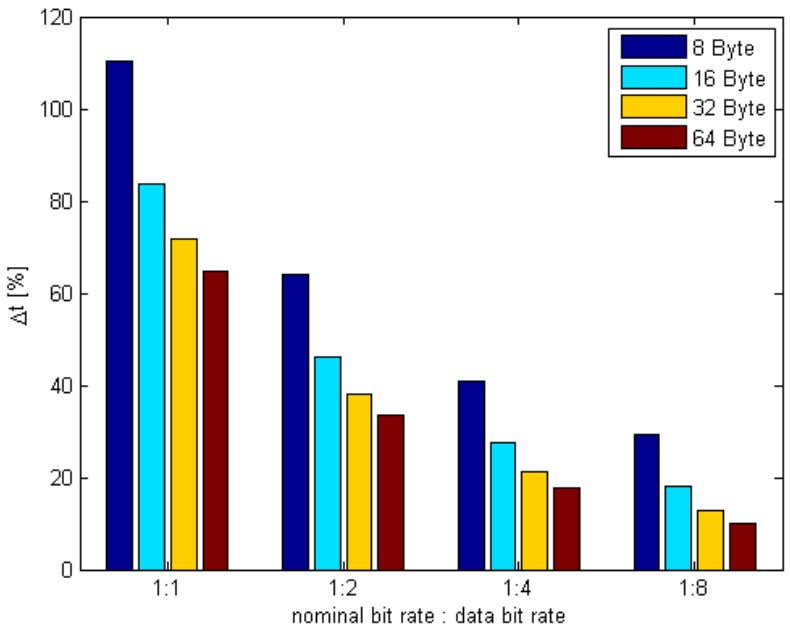
Error management je téměř shodný jako u předchozí specifikace CAN. Stejně jako u klasického CAN protokolu je rozlišováno 5 druhů chyb (Bit error, stuff error, CRC error, form error, a acknowledgment error). Rovněž je zachován přechod mezi jednotlivými stavy jednotky (error active, error passive a bus off).

Změnou v protokolu CAN FD oproti klasickému CAN je v signalizaci transmitteru, v jakém stavu se nachází. To je zprostředkováno pomocí ESI bitu (viz kapitola Formáty rámce CAN FD). Přijímač tak získá informaci, zda se odesílající uzel nachází v „error active“, či „error passive“ stavu.

2.8 Porovnání CAN FD a CAN vzhledem k rychlosti

Přínosy CAN FD rámců oproti klasickým CAN rámcům byly nastíněny v předchozích kapitolách. Pro lepší znázornění však byla vytvořena vizualizace porovnání teoretických rychlostí obou protokolů. Tato vizualizace byla vytvořena pomocí software Matlab.

Následující graf zobrazuje v procentech přibližný poměr doby přenosu daného počtu bajtů při použití protokolů CAN a CAN FD. Tento poměr je ukázán pro několik velikostí přenášených dat, a pro několik poměrů mezi arbitrážní a datovou rychlostí rámce CAN FD. Pro klasické rámce CAN bylo pro přenos více než 8 bajtů dat uvažováno maximální možné vytížení sběrnice při odesílání více rámců (tedy minimální možná prodleva mezi odvysíláním jednotlivých rámců).



Obr. 26: Porovnání rychlosti přenosu protokolů CAN a CAN FD.

Z výše uvedeného grafu je vidět, že v případě přenosu více než osmi bajtů dat je tento protokol výhodnější i pokud nevyužijeme možnosti přepnutí bitové rychlosti v datové části.

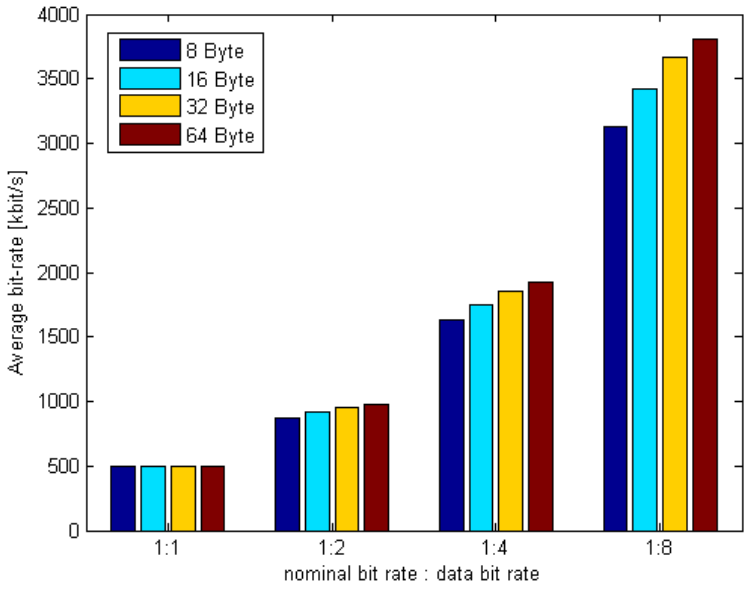
Jediným případem, kdy se tento protokol nevyplatí je případ přenosu menšího počtu bajtů bez využití přepnutí bitové rychlosti. Příčinou je větší délka rámce způsobená především prodloužením pole CRC.

Publikace [18] uvádí procentuální poměr bitů, přenášených datovou rychlostí pro různé počty bajtů přenášených dat (viz obrázek 29).

Payload (bytes)	Percent of bits in data phase (%)
8	76.98
16	84.57
32	91.02
64	94.99

Obr. 27: Podíl bitů rámce přenášených datovou rychlostí, při různých velikostech přenášených dat. Převzato z [18].

Podobných výsledků bylo dosaženo výpočtem, provedeným pomocí software Matlab. Tyto výsledky jsou znázorněny následujícím grafem, který zobrazuje průměrné přenosové rychlosti pro různé poměry mezi arbitrážní a datovou rychlostí. Pro tento případ byla volena arbitrážní rychlost 500 kbit/s.



Obr. 28: Průměrná přenosová rychlost pro různé délky přenášených dat, a různé poměry mezi arbitrážní a datovou rychlostí, při arbitrážní rychlosti 500 kbit/s.

Z výše uvedených dat vidíme, že průměrná přenosová rychlost CAN FD rámce se při velkých délkách přenášených dat pohybuje přes 90% rychlosti v datové části, což v reálném nasazení přináší nesporné výhody.

3. Podporující hardware

Protokol CAN FD byl ve své současné podobě uveden relativně nedávno (2015). Z toho vyplývá, poměrně omezená možnost volby mikrokontroléru. Naproti tomu transceiverů podporujících tento protokol je již celá řada od mnoha výrobců.

Následující část práce bude věnována stručnému shrnutí současné nabídky hardware, podporujícího tento protokol.

3.1 Mikrokontroléry s podporou CAN FD

Při výběru mikrokontroléru podporujícího CAN FD je třeba dbát na rozlišování pojmů ISO CAN FD a NON ISO CAN FD. Tyto dva termíny se vžily pro označení protokolů CAN FD podle starší, dnes již neplatné, specifikace (NON ISO CAN FD), a protokolu CAN FD podle současně platné normy ISO 11898-1:2015 (ISO CAN FD).

Výše uvedenému problému je třeba věnovat pozornost zejména z toho důvodu, že podpora ISO resp. NON ISO CAN FD se u jednotlivých mikrokontrolérů zpravidla liší u jednotlivých verzí křemíkových masek mikrokontroléru. Uvedení podpory ISO CAN FD v příslušném reference manuálu tedy nemusí platit i pro starší verze těchto masek. Oba protokoly přitom nejsou vzájemně kompatibilní.

V následující tabulce jsou uvedeny některé rodiny mikrokontrolérů, jež v současnosti podporují CAN FD.

Tab. 6: Přehled vybraných MCU s podporou CAN FD.

Výrobce	Rodina MCU
NXP	S32K, S32V, S32S, MPC57xx, LPC546xx, LPC540xx
STM	SPC58 B/C/G, STM32MP1xx
Microchip	SAM C/E5x/E40/V70/V71/A5D, dsPIC33CH
Infineon	AURIX™– TC2xx, TC3xx

Jak lze vidět v tabulce, protokol CAN FD podporují převážně mikrokontroléry určené pro automotive nasazení. Povšimnout si rovněž můžeme absence zástupce od společnosti Texas Instruments, jelikož žádný z jejich v současnosti nabízených mikrokontrolérů tento nový protokol nepodporuje.

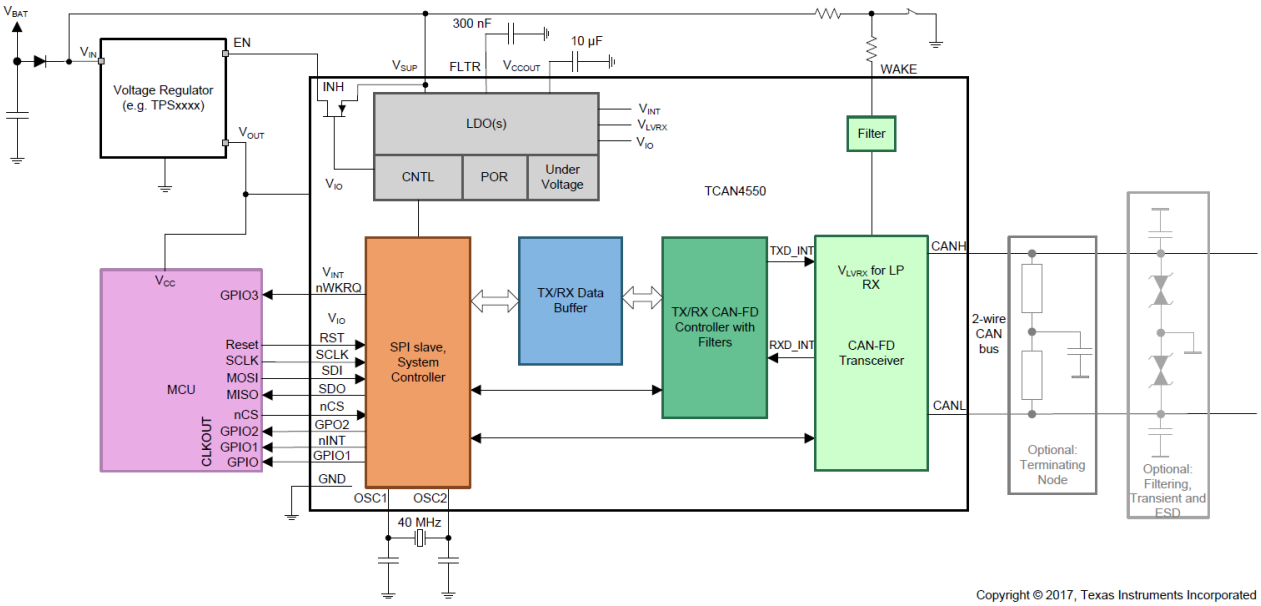
3.2 Transceivery CAN FD

Budiče podporující CAN FD, tedy splňující specifikaci ISO 11898-2:2016, pro automotive i jiné použití nabízí velké množství výrobců (NXP, Microchip, Infineon, Texas Instruments). V současné době jsou k dispozici transceivery pro maximální přenosové rychlosti až 8 Mbit/s.

3.3 Externí kontroléry CAN FD

Aby bylo možné zachovat současně použitý mikrokontrolér a zároveň vyhovět požadavku na CAN FD protokol, jsou k dispozici externí CAN FD kontroléry. Tyto kontroléry jsou konfigurovatelné nejčastěji prostřednictvím sériového rozhraní SPI. Některé z nich pak disponují již integrovaným transceiverem.

Mezi tyto externí převodníky patří například MCP2517FD (Microchip), či TCAN4550-Q1 (Texas Instruments).



Obr. 29: Blokové schéma a typické zapojení externího kontroléru TCAN4550-Q1. Převzato z [19].

4. Prototypová deska USG2

Prototypová deska USG2 (UniversalSteuerGerät 2) je vývojová platforma, sloužící k prvotnímu testování koncepce nově vyvíjených ECU jednotek pro automobily. Jedná se o základní desku, rozšířitelnou o nejrůznější modely pro komunikaci, vstupní a výstupní moduly a moduly pro ovládání akčních členů. Současná verze je vybavena modulem, nesoucím mikrokontrolér MPC5748G.

4.1 MPC5748G

MPC5748G je mikrokontrolér určený pro automotive a průmyslové aplikace. Mikrokontrolér obsahuje dvě jádra Power Architecture e200z4, schopné pracovat na kmitočtu 160 MHz. Tato dvě jádra jsou doplněna jádrem Power Architecture e200z2, pracujícím do kmitočtu 80 MHz. Mikrokontrolér je rovněž bohatě vybaven z hlediska jeho

periferií. Mezi ty patří například 8 FlexCAN modulů s podporou CAN FD, 18 LINFlexD modulů, využitelných pro LIN sběrnici či pro UART komunikaci, a další.

4.2 FlexCAN kontrolér

Tato periferie, obsažená v mikrokontroléru MPC5748G a jiných mikrokontrolérech výrobce NXP, podporuje jak klasické CAN rámce, tak i CAN FD. V této kapitole bude velice krátce popsán princip odesílání a příjmu zpráv CAN FD protokolu a některé problémy, se kterými se lze setkat.

4.2.1 Zásobníky zpráv (Message Buffers)

Pro příjem a odesílání zpráv slouží prostory ve vyhrazené RAM paměti. Tyto prostory jsou označovány jako „Message buffers“ (dále jen MB). Podle nastavení může být jejich velikost 16 až 72 bajtů, přičemž pro různé nastavení velikostí se mění i jejich celkový počet (velikost celkové RAM paměti pro ně je konstantní). Pro nejmenší nastavení velikosti (8 bajtů dat) lze použít až 96 těchto objektů, zatímco pro velikost 72 bajtů (64 bajtů dat) maximálně 21. Rovněž se vlivem různého ofsetu mění i počáteční adresy těchto message bufferů. Struktura těchto objektů je znázorněna na následujícím obrázku.

	0 1 2 3 4 7 8 9 10 11 12 13 14 15 16 23 24 31														
0x0	EDL	BRS	ESI		CODE		SRR	IDE	RTR	DLC				TIME STAMP	
0x4	PRIO				ID (Standard/Extended)									ID (Extended)	
0x8	Data Byte 0						Data Byte 1							Data Byte 2	Data Byte 3
0xC	Data Byte 4						Data Byte 5							Data Byte 6	Data Byte 7
0x10	Data Byte 8						Data Byte 9							Data Byte 10	Data Byte 11
0x14	Data Byte 12						Data Byte 13							Data Byte 14	Data Byte 15
0x18	Data Byte 16						Data Byte 17							Data Byte 18	Data Byte 19
0x1C	Data Byte 20						Data Byte 21							Data Byte 22	Data Byte 23
0x20	Data Byte 24						Data Byte 25							Data Byte 26	Data Byte 27
0x24	Data Byte 28						Data Byte 29							Data Byte 30	Data Byte 31
0x28	Data Byte 32						Data Byte 33							Data Byte 34	Data Byte 35
0x2C	Data Byte 36						Data Byte 37							Data Byte 38	Data Byte 39
0x30	Data Byte 40						Data Byte 41							Data Byte 42	Data Byte 43
0x34	Data Byte 44						Data Byte 45							Data Byte 46	Data Byte 47
0x38	Data Byte 48						Data Byte 49							Data Byte 50	Data Byte 51
0x3C	Data Byte 52						Data Byte 53							Data Byte 54	Data Byte 55
0x40	Data Byte 56						Data Byte 57							Data Byte 58	Data Byte 59
0x44	Data Byte 60						Data Byte 61							Data Byte 62	Data Byte 63

= Unimplemented or Reserved

Obr. 30: Struktura „Messge buffer“ objektu pro odesílání a příjem zpráv. Převzato z [20].

Klíčovou roli pro nastavení daného MB pro příjem zpráv či její odeslání je registr s ofsetem 0x00 od počátku daného MB. Tento registr je označován „Control and Status register“ a slouží jednak nastavení parametrů dané zprávy (v případě odesílání), a jednak k uložení parametrů příchozí zprávy.

Velice důležitou roli jak při příjmu, tak odesílání pak hraje pole „CODE“ jenž po zápisu příslušné hodnoty spustí proces odesílání (daná zpráva vstupuje do procesu arbitráže).

V případě příjmu pak toto pole nese informace o tom, zda byl daný MB před příjmem nové zprávy již přečten, či došlo k přepsání původní přijaté zprávy. Další významy jsou uvedeny v následující tabulce.

Tab. 7: Vybrané hodnoty pole CODE a jejich význam.

CODE	Význam	Konfigurace MB
0x0	Neaktivní (nepodílí se na příjmu)	Rx
0x04	MB je prázdný	Rx
0x02	MB je plný	Rx
0x06	Přetečení	Rx
0x08	Neaktivní (připraven k odeslání)	Tx
0x09	Přenos přerušen	Tx
0x0c	Vstup MB do arbitráže (zahájení odesílání)	Tx

4.2.2 Omezení konfigurace pro CAN FD

Při konfiguraci modulu podle protokolu CAN FD dochází k omezením v další konfiguraci oproti implementaci pouze klasického protokolu CAN. Tato omezení jsou následující:

- Nemožnost použití přijímacího FIFO
- Nemožnost použití funkce „Partial networking“
- Nemožnost využití DMA kanálu

4.2.3 Shoda s normou ISO 119898-1

Jak bylo uvedeno v kapitole „Podporující hardware“, můžeme se setkat se staršími CAN FD kontroléry, které nevyhovují současné normě ISO 11898-1:2015. To platí i pro případ zde požitého mikrokontroléru MPC5748G. Pro tento mikrokontrolér existuje několik křemíkových masek, přičemž starší verze podporují pouze takzvaný NON-ISO CAN FD.

Tab. 8: Podpora ISO CAN FD pro jednotlivé křemíkové masky

Verze masky	Podpora ISO CAN FD
1N81M	NE
0N78S	ANO
1N06M	NE

V praktickém použití to znamená, že mikrokontroléry, vyrobené podle masek 1N81M a 1N06M, mohou komunikovat mezi sebou a využívat všech výhod CAN FD protokolu, nicméně jednotka pro výpočet CRC je odlišná od jednotky podle platného standardu.

V případě propojení s jednotkou, jež implementuje ISO CAN FD, by vzájemná komunikace nebyla možná.

V tomto případě je kompletně zachována softwarová kompatibilita obou verzí modulů, a tudíž v případě nahrazení mikrokontrolérem s novou křemíkovou maskou není potřeba úprav software. Jedinou změnou u nové verze masky je možnost přepnutí mezi ISO a NON-ISO CAN FD protokolem pro zachování zpětné kompatibility.

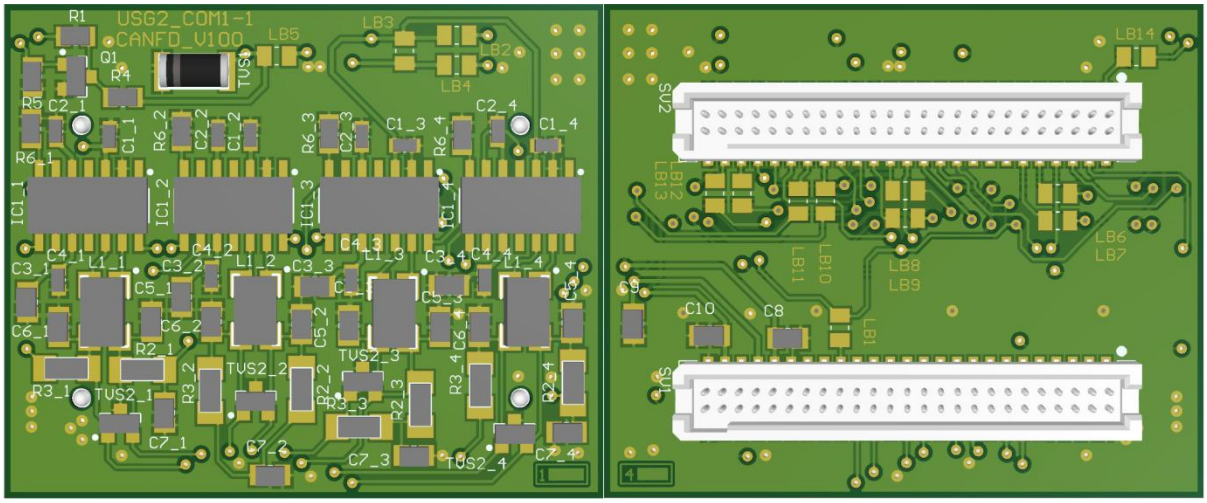
Vzhledem k cílové aplikaci jednotek USG2, nepředstavuje tato rozdílnost problém, a tudíž byl software implementován a testován podle NON ISO CANFD protokolu. V případě potřeby by bylo možno případný problém s kompatibilitou vyřešit například vytvořením jednotky fungující jako „bridge“, která by nesla mikrokontrolér s maskou 0N78S, která podporuje oba protokoly, a tudíž může zprávy přijaté jedním FlexCAN modulem dle NON ISO CAN FD odesílat druhým FlexCAN modulem jako ISO CAN FD a naopak.

5. Rozšiřující modul pro CAN FD

Pro možnost využít výše zmíněnou prototypovou desku i pro testování moderních koncepcí ECU, využívajících nového protokolu CAN FD byl navrhnut a vyroben modul, jenž tento protokol podporuje. Tento modul využívá jednoho ze dvou slotů pro komunikační periferie, tedy pro FlexCAN a LINFlexD.

Rozšiřující jednotka využívá čtyř modulů FlexCAN (FlexCAN0 až FlexCAN3), použitého mikrokontroléru MPC5748G.

Vzhledem k uspořádání mateřské desky USG2 je velikost rozšiřující jednotky CAN FD limitována na cca 5 cm x 4 cm.



Obr. 31: 3D model desky rozšiřujícího modulu, vytvořený v Altium designeru.(Pohled shora a zdola.)

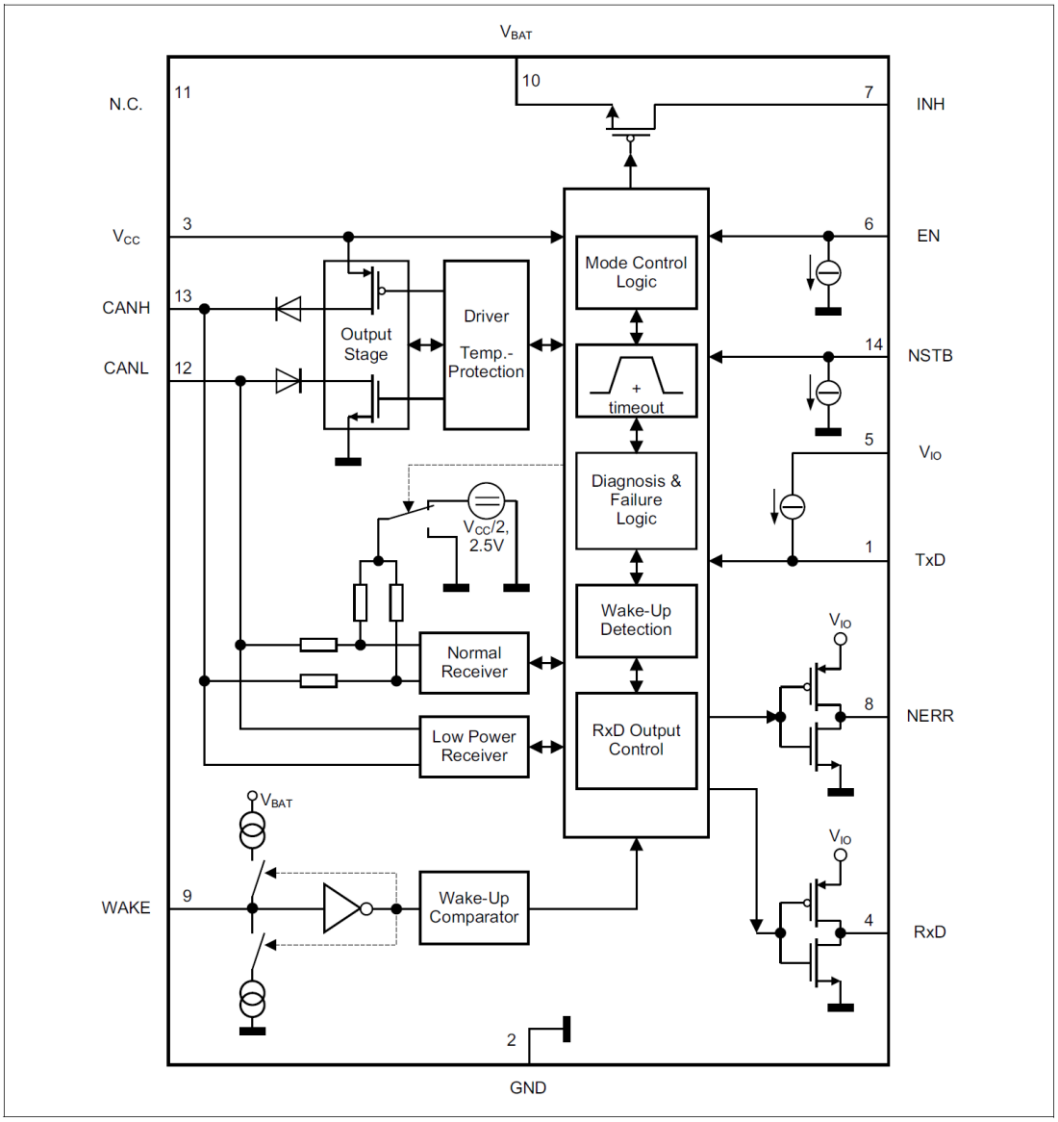
V následujícím textu bude tento rozšiřující modul podrobněji popsán.

5.1 TLE9252

Jak bylo popsáno v kapitole 4, je na trhu v současné době již množství transceiverů podporujících CAN FD. Tyto transceivery se liší jednak v maximální přenosové rychlosti, tak v dodatečných vlastnostech, jako např. schopnost selektivního wake-up, či další dodatečné výstupy.

Z důvodu neznámé aplikace, pro niž bude v budoucnu modul využíván, byl volen transceiver, který disponuje co největším množstvím těchto dodatečných funkcí, jež nejsou standardem ISO 11898-2:2016 vyžadovány.

Na základě těchto požadavků byl zvolen transceiver TLE9252, vyráběný společností Infineon.



Obr. 32: Bokové schéma transceiveru TLE9252. Převzato z [21].

5.2 Operační módy

Transceiver TLE9252 je schopen operovat v několika módech. Mezi těmito módy lze přecházet pomocí řídících vstupů obvodu, označených EN a NSTB. V této kapitole budou tyto módy krátce popsány.

5.2.1 Normal-operating mode

Jedná se o mód, ve kterém je jak vysílač, tak přijímač aktivní, tedy schopen vysílání i přijímání rámců. Vstup obvodu pro lokální probuzení (WAKE) je deaktivován. Piny připojené ke sběrnici jsou přes impedanci připojeny k polovině napájecího napětí (VCC/2). Veškeré interní kontroly stavů, jako například ochrana proti přehřátí, či monitoring poklesů napájecích napětí jsou aktivní, a chyby jsou indikované na výstupním pinu NERR.

5.2.2 Receive-only mode

Tento mód umožňuje používat transceiver pouze pro přijímání zpráv. Vysílač transceiveru je v tomto módu deaktivován a slouží pouze pro potvrzování příjmu zprávy. Během tohoto módu je rovněž neaktivní teplotní ochrana. Sběrnice je stejně jako u běžného režimu připojena přes impedanci k polovině napájecího napětí (VCC/2).

5.2.3 Stand-by mode

V tomto módu je vysílač zablokován (vstup pinu TxD je ignorován). Přijímač operuje pouze v nízko-příkonovém režimu, kdy monitoruje případný „wake-up“ na CAN sběrnici. Monitoring lokálních probuzení je rovněž aktivní. Vodiče sběrnice jsou přes impedanci připojeny k zemi.

5.2.4 Stav „Go-to-Sleep“

Jedná se o přechodný stav, během kterého je blokováno odesílání a přijímač přijímá pouze „wake-up“ zprávy na sběrnici. Po uplynutí doby, označované jako t_{SLEEP}, během které není přijat žádný požadavek na probuzení, přechází modul do stavu „sleep mode“.

5.2.5 Sleep mode

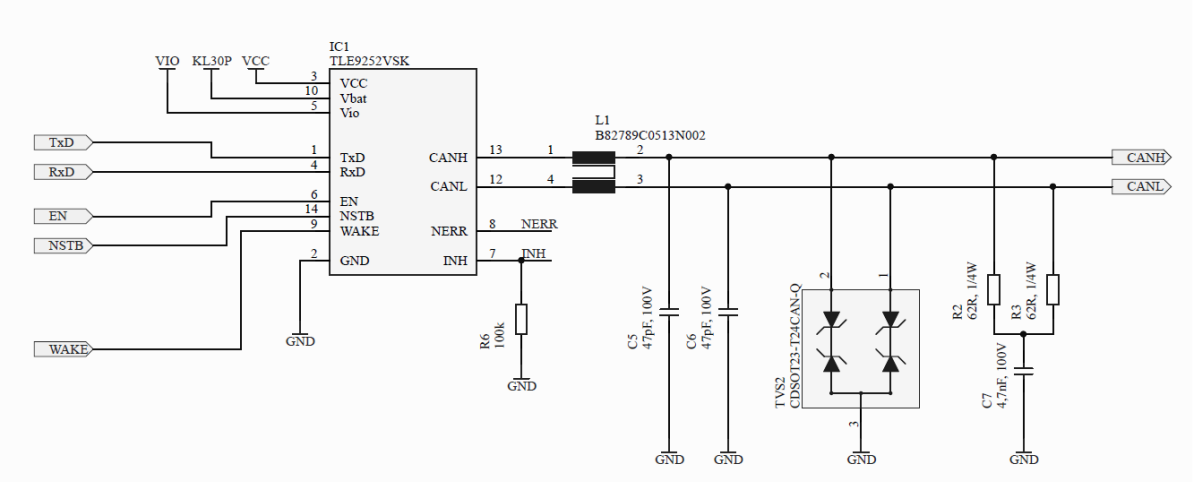
Jedná se o nízko-příkonový režim, během kterého je transceiver schopen detekovat pouze „wake-up“ zprávy, či lokální probuzení. Rovněž je vypnuta kontrola poklesu napětí u hlavního a bateriového napájení (V_{CC} a V_{BAT}).

5.2.6 Power On Reset

Do tohoto módu transceiver vstupuje pouze po náběhu napájení. Během tohoto módu není aktivní žádná funkce transceiveru.

5.3 Schéma zapojení transceiveru

Zapojení obvodů transceiverů bylo provedeno podle doporučeného zapojení výrobce s ohledem na možnosti vstupních a výstupních pinů modulu, a s ohledem na EMC.



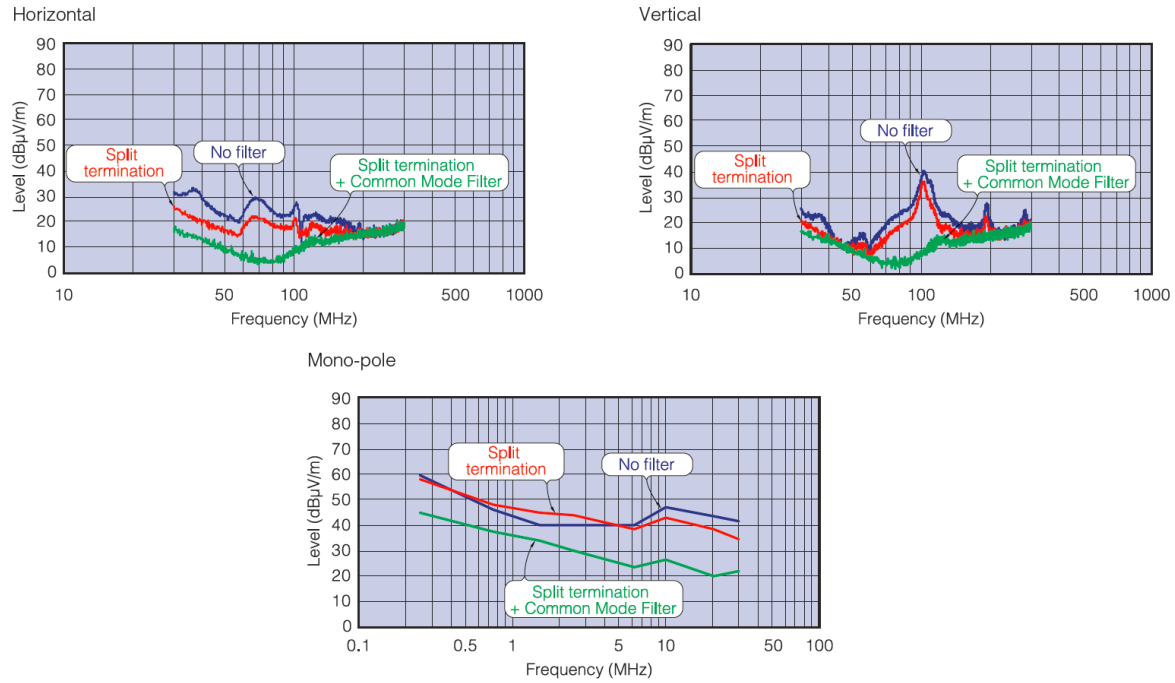
Obr. 33: Výřez schéma, zachycující zapojení použitých transceiverů.

5.3.1 Opatření pro zlepšení EMC

Na výše uvedeném schématu si můžeme všimnout zapojení tlumivky s proudovou kompenzací B82789C0513N002 o velikosti 51 µH a kondenzátorů na cestách signálů CAN H a CAN H.

Použití těchto komponent zlepšuje vlastnosti CAN sběrnice vzhledem k EMI i EMS. Dalším faktorem, jenž příznivě ovlivňuje vlastnosti sběrnice je použití takzvané „split“ terminace. Tímto termínem je označováno takové zakončení sběrnice, kdy je namísto jednoho 120 Ω terminačního rezistoru použita dvojice 60 Ω rezistorů, jejichž střed je přes kondenzátor malé velikosti (zpravidla řádově jednotky pF) spojen se zemí a tvoří tak filtr typu dolní propust.

Vliv výše uvedených opatření na výsledné naměřené rušení vyzařováním byl publikován například společností TDK (viz následující obrázek).



Obr. 34: Vliv výše uvedených opatření na naměřené hodnoty rušivého vyzařování pro horizontální, vertikální polarizaci anténa a pro měření s použitím monopólu. Převzato a upraveno z [22].

Vidíme, že podle výše uvedeného měření má na EMI sběrnice největší vliv použití tlumivky s proudovou kompenzací. Použití této tlumivky však může mít v některých situacích rovněž nepříznivé dopady. V případě zkratu na některém ze sběrnicových vodičů CAN H či CAN L může při pokusu o komunikaci docházet k přechodovým jevům, jejichž výsledkem je napěťová špička o hodnotách mnohdy přesahujících 60 V. Tyto špičky pak nepříznivě působí na použitý transceiver, a v krajním případě jej mohou i zničit. Jako ochranu proti tomuto jevu je vhodné použít prvky ESD ochrany u samotných výstupů budiče. [22]

Jako ESD ochrana sběrnice byl použit integrovaný obvod CDSOT23-T24CAN-Q obsahující dva transily pro ochranu obou CAN vodičů.

6. Základní software mikrokontroléru pro FlexCAN modul

Pro modul FlexCAN bylo potřeba vytvořit software, s nímž další případní uživatelé tohoto modulu budou moci snadno a univerzálně pracovat. Vznikl tak univerzální balík funkcí pro tuto periferii, obsahující funkce pro inicializaci libovolného z modulů, a funkce pro odesílání a přijímání zpráv jak v klasickém CAN rámci, tak CAN FD.

Tento software byl rozdělen do dvou modulů:

- CAN_HW (.h, .c)
  - Jedná se o soubor funkcí pro práci s periferií FlexCAN na úrovni registrů
  - Obsahuje deklarace struktur a globálních proměnných, používaných dalšími funkcemi
  - Podporovány jsou moduly číslo 0 až 3, přičemž byl brán ohled na možnost budoucího rozšíření pro další moduly
- CAN (.h, .c)
  - Soubor funkcí, využívajících funkce deklarované v CAN_HW.h.
  - Obsahuje funkce pro inicializaci, příjem a odesílání zpráv prostřednictvím FlexCAN modulů.

6.1 Inicializace FlexCAN

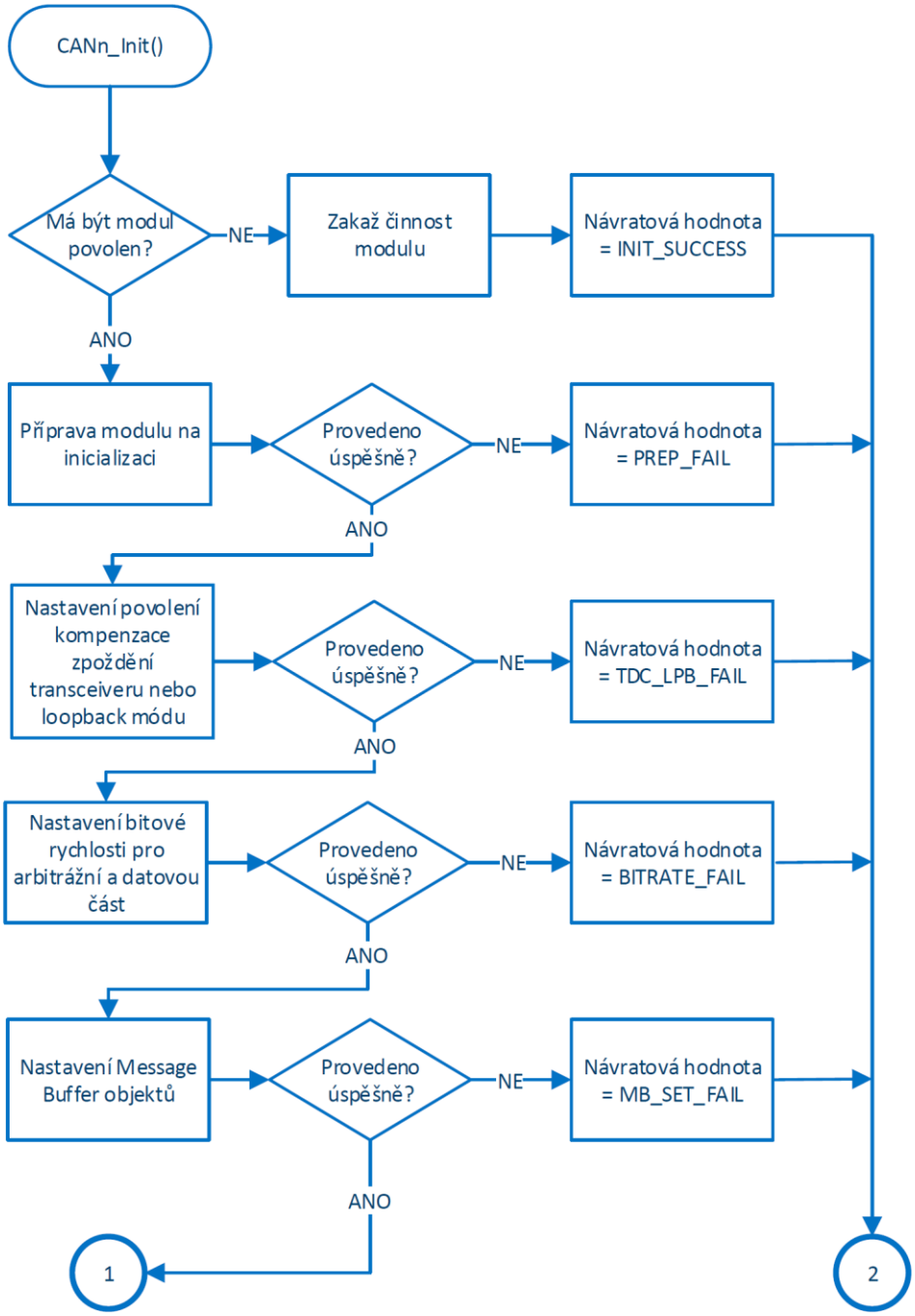
Pro co největší univerzálnost a jednoduchost aplikace v budoucích projektech využívá inicializační funkce modulu FlexCAN strukturu, obsahující veškeré potřebné parametry pro nastavení parametrů CAN FD protokolu, podporovaných tímto modulem.

Tato inicializační struktura umožňuje nastavit:

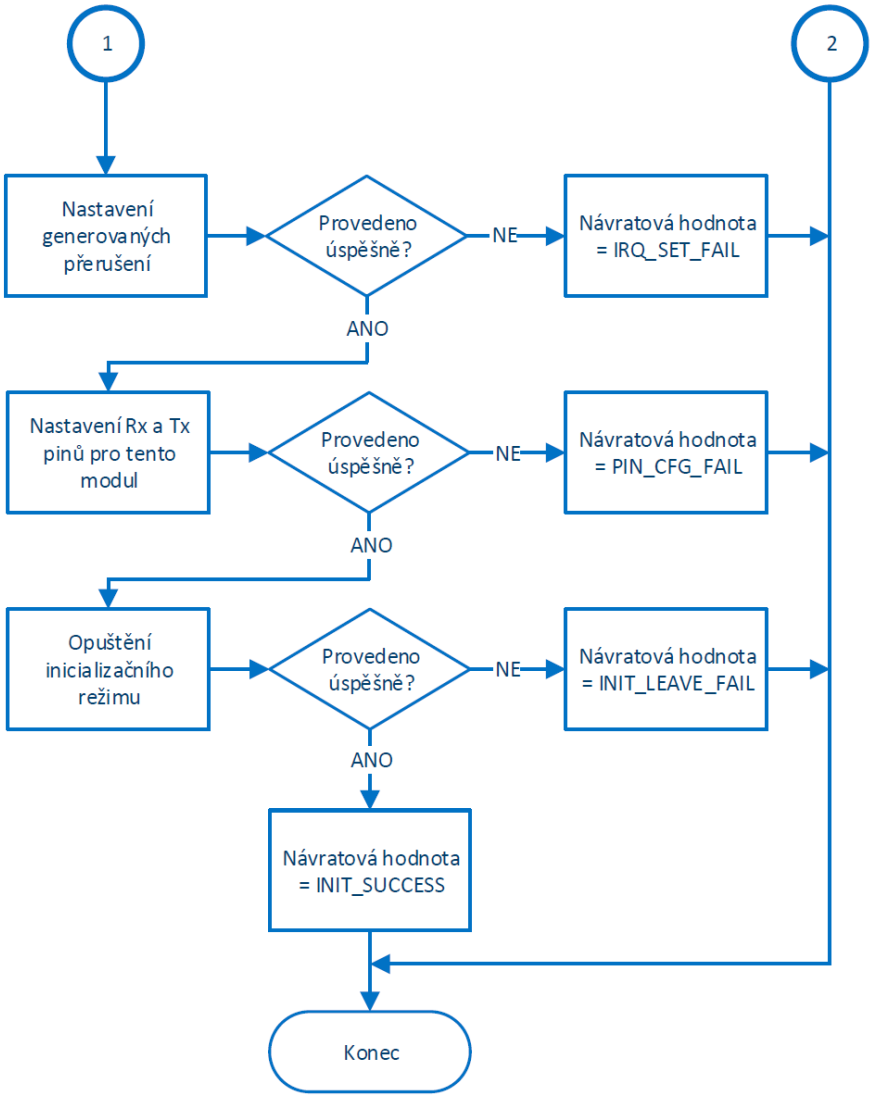
- Povolení/zakázání daného modulu
- Číslo nastavovaného modulu
- Bitovou rychlost v arbitrážní části
- Bitovou rychlost v datové části
- Zapnutí/vypnutí loopback módu
- Zapnutí/vypnutí TDC
- Nastavení jednotlivých identifikátorů a jejich masek, včetně možnosti povolit/zakázat příjem zpráv s rozšířeným identifikátorem
- Volba zdroje hodinového signálu
- Volba jader, jež mají přijímat IRQ daného modulu
- Nastavení priorit IRQ

Inicializace modulů je zajištěna funkcí ***FlexCAN_init***, jež následně volá funkci ***FlexCAN0_init***, respektive ***FlexCAN1_7_init***. Modul FlexCAN0 je inicializován samostatně, jelikož jeho sada registrů se od ostatních modulů mírně odlišuje (jako jediný z nich podporuje „pretended networking“). Tyto funkce pak volají jednotlivé funkce, jež mají za úkol nastavit určitou část inicializovaného modulu.

Inicializační sekvence je znázorněna na následujícím vývojovém diagramu, který byl pro svou velikost rozdělen na dvě části.



Obr. 35: Inicializační sekvence FlexCAN modulu. (První část)



Obr. 36: Inicializační sekvence FlexCAN modulu. (Druhá část)

Jak je z vývojového diagramu patrné, je inicializace rozdělena do několika kroků, jejichž případné selhání je indikováno návratovou hodnotou. Tato návratová hodnota pak může být dále softwarově využívána a uživatel má přehled o příčině selhání inicializace i bez hlubšího hledání místa selhání.

6.2 Odesílání zpráv

Pro odeslání zprávy, ať už s využitím rámce CAN FD, či klasického CAN byla vytvořena funkce ***TransmitMsg***. Parametrem této funkce je struktura ***CAN_Tx_msg***, jež obsahuje veškeré nastavované parametry zprávy.

Těmito parametry jsou:

- Identifikátor zprávy
- Zda se jedná o rámec podle CAN či CAN FD

- Povolení/zakázání přepínání bitové rychlosti v datové části rámce
- Použití rozšířeného/standardního identifikátoru
- Až 64 bajtů vlastních přenášených dat
- Číslo modulu, použitého pro odeslání zprávy

6.3 Příjem zpráv

Jak již bylo zmíněno v části týkající se inicializace modulů, je možné využít pro signalizaci příchozí zprávy přerušení.

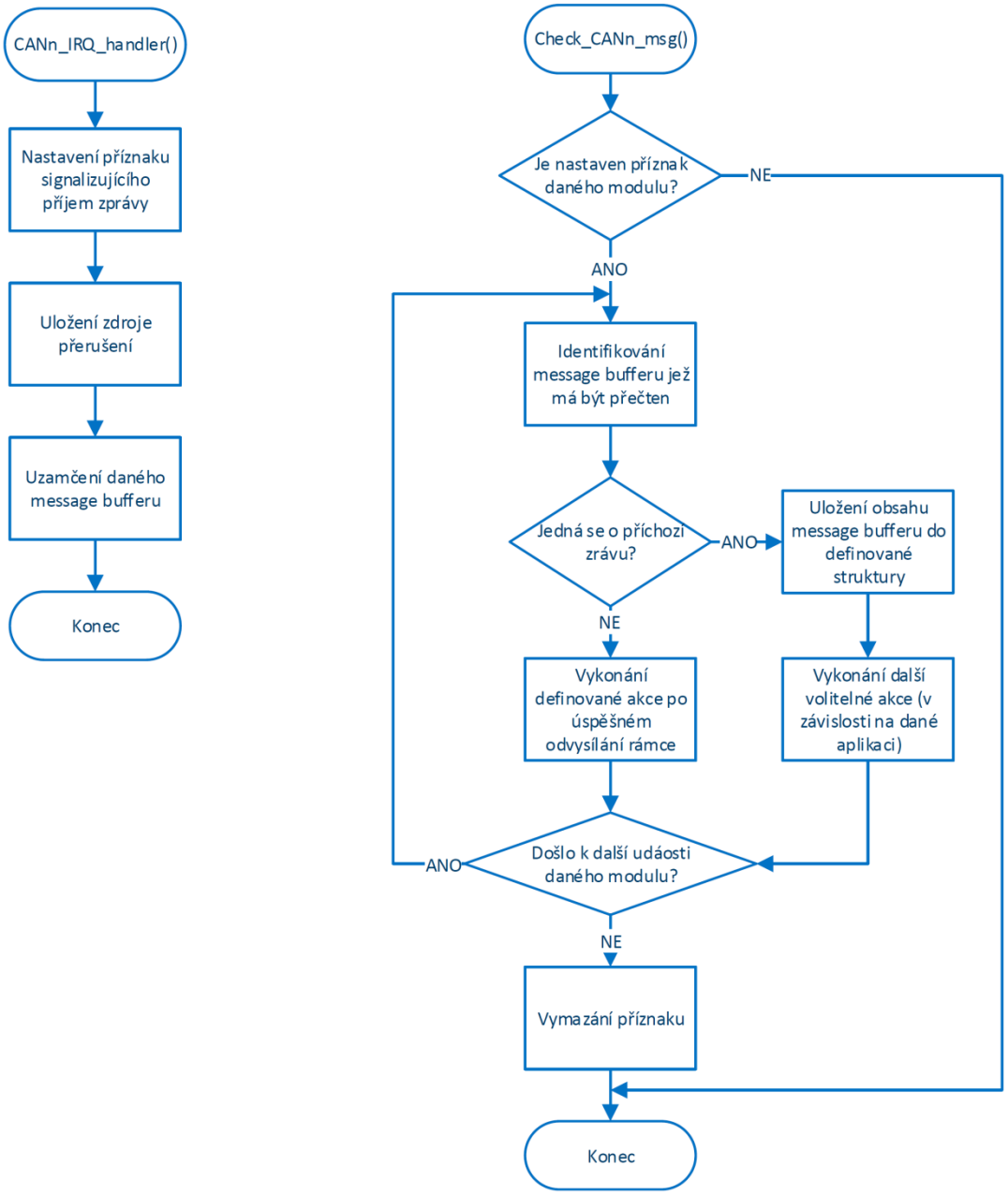
V tomto přerušení je identifikován zdroj (příslušný message buffer), jež toto přerušení vyvolal, a tato informace je uložena do pomocné proměnné. Společně s tím je do globální proměnné uložena informace signalizující, že příslušný modul obdržel zprávu. Program následně periodicky (typicky v hlavní smyčce) kontroluje, zda došlo k přijetí zprávy daným modulem, a vykoná obslužnou funkci.

V případě, že nechceme využívat přerušení, je nutné periodicky kontrolovat, zda přišla zpráva. K tomu byly napsány funkce ***get_FlexCANn_Rx_Flags***. Tyto funkce pro moduly 0 až 4, zkontrolují příznakové bity jednotlivých message boxů a nastaví příslušné globální proměnné stejně jako je tomu při vykonání ISR.

V obou případech je vhodné co nejdříve po zaregistrování přijaté zprávy uzamknout příslušný message box, aby nebylo možné jej přepsat jinou příchozí zprávou. To je v obou případech zajištěno přečtením takzvaného „Control and Status“ registru daného message bufferu.

Pro vlastní přečtení zprávy pak slouží funkce ***CAN_Rx_handler***. Parametrem této funkce je číslo modulu, jehož zprávu chceme vyčíst. Pro daný modul je pak zkontrolován stav globálních proměnných indikujících příjem zprávy pro všechny message buffery, a v případně je tento message box přečten.

Ukázka příjmu příchozí zprávy s využitím přerušení je znázorněna na následujícím diagramu.



Obr. 37: Průběh příjmu zpráv s využitím obsluhy přerušení, generované při příjmu a úspěšném odeslání.

6.4 Software ovládání transceiverů

Pro ovládání transceiverů byla vytvořena skupina funkcí, organizovaná v souborech TLE9252(.h a .c). Tyto funkce obstarávají přechody mezi jednotlivými stavy daného transceiveru a stejně tak umožňují zjistit jejich současný stav.

7. Testovací software pro FlexCAN

Pro testování chování CAN FD komunikace bylo vytvořeno uživatelské rozhraní pro PC. Program byl napsán v jazyce C#, ve vývojovém prostředí Visual Studio 2017. Toto grafické uživatelské rozhraní komunikuje s deskou USG2 prostřednictvím UART rozhraní, podporovaného LINFlexD modulem.

K tomu byla vytvořena nadstavba k základním funkcím pro ovládání FlexCAN modulů, která zajistí komunikaci s aplikací v PC.

Díky tomuto propojení je možné nastavovat klíčové atributy komunikace prostřednictvím přehledného uživatelského rozhraní. Mezi funkce, jimiž tento program disponuje, patří:

- Možnost inicializace libovolného ze čtyř podporovaných FlexCAN modulů
    - Nastavení přenosové rychlosti v arbitrážním i datovém poli rámce
    - Nastavení filtrů ID, jež budou přijímány
    - Nastavení masek jednotlivých filtrů
    - Možnost povolení příjmu zpráv s rozšířeným identifikátorem
    - Zapnutí/vypnutí loopback módu
- Možnost nastavení periodicky odesílaných zpráv pro libovolné moduly
    - Nastavení periody periodických zpráv
    - Nastavení ID a přenášených dat periodických zpráv
    - Volba mezi klasickým CAN a CAN FD rámcem
    - Nastavení standardního/rozšířeného ID
- Možnost odeslání jednorázové zprávy
    - Volba FlexCAN modulu pro odeslání zprávy
    - Volba CAN/CAN FD rámce
    - Volba standardního/rozšířeného ID
- Tabulka pro zobrazení přijatých CAN zpráv
    - Zobrazení čísla modulu, pořadového čísla zprávy, času přijetí, ID, přijatých dat
    - Možnost filtrování zpráv podle příslušného FlexCAN modulu a podle ID zprávy
    - Možnost řazení zpráv vzestupně/sestupně podle libovolného ze sloupců
    - Možnost exportu do .csv souboru
- Tabulka pro zobrazení odeslaných CAN zpráv
    - Zobrazení čísla modulu, pořadového čísla zprávy, času přijetí, ID, přijatých dat
    - Možnost filtrování zpráv podle příslušného FlexCAN modulu a podle ID zprávy
    - Možnost řazení zpráv vzestupně/sestupně podle libovolného ze sloupců
    - Možnost exportu do .csv souboru
- Správa stavů jednotlivých transceiverů
    - Zobrazení současného stavu transceiveru
    - Možnost změny stavu transceiveru
    - Indikace případných chybových stavů transceiverů

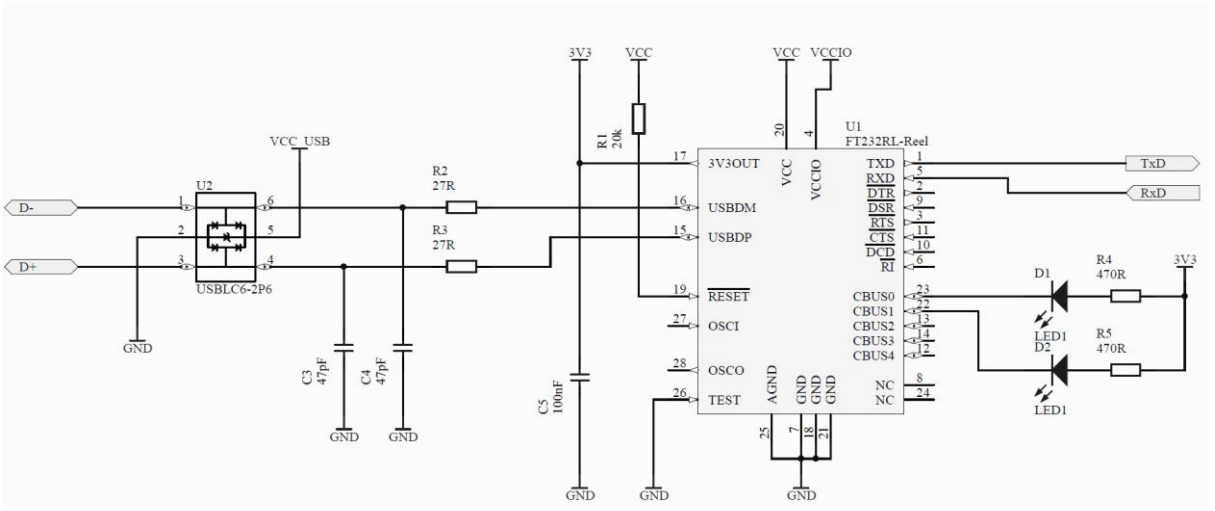
- Zobrazení chybových stavů jednotlivých FlexCAN modulů
  - Vypsání konkrétní chyby
  - Zobrazení chybových čítačů
- Možnost zobrazení současné hodnoty libovolného FlexCAN registru a její přepsání
  - Možnost vybrání příslušného modulu a následně jeho registru
  - Při zadání konkrétní adresy možnost přístupu k libovolnému registru (z důvodu bezpečnosti se nedoporučuje)

7.1 Modul USB-UART převodníku

Aby byla komunikace s uživatelským rozhraním možná, bylo potřeba jednotku USG2 rozšířit o modul převodníku USB-UART. Tento modul využívá LINFlexD periferie, dostupné z expanzního slotu USG2 pro obecné vstupní a výstupní moduly (značené I/O2).

Modul je osazen převodníkem FT232RL, jehož zapojení je provedeno tak, aby bylo pomocí nulových rezistorů možno volit různé úrovně napětí pro Rx a Tx signály.

Obvod vlastního převodníku je doplněn o ESD ochranu signálů D+ a D-. Tato ochrana je zajišťována integrovaným obvodem USBLC6-2P6, společnosti STM.



Obr. 38: Zapojení převodníku FT232RL a požitých opatření z hlediska EMC.

7.2 Komunikační protokol

Za účelem bezproblémové komunikace mezi mikrokontrolérem a PC bylo potřeba zvolit vhodný způsob výměny dat. Požadavkem na tento protokol je zejména možnost kontroly správnosti přijetí celého rámce dat a snadná rozšiřitelnost o další nové příkazy.

Formát rámce tohoto protokolu byl volen tak, aby v případě chybného přijetí některého z bajtů (např. z důvodu okolního rušení) bylo možno tuto skutečnost detekovat a patřičným způsobem na ni reagovat.

Bajt:	0	1	2	3...N	N+1
Význam:	Začátek rámce	Kód příkazu	Délka rámce	Vlastní data	CRC8

Obr. 39: Formát komunikačního rámce pro výměnu dat a příkazů mezi jednotkou USG2 a demonstrační aplikací pro PC.

7.2.1 Začátek rámce

Začátek každého rámce začíná pevně daným „start-bajtem“. V tomto případě byla zvolena hodnota 0x55. Jakmile je stavový automat zajišťující příjem rámce ve výchozím stavu, znamená přijetí této hodnoty začátek komunikace.

7.2.2 Kód příkazu

Každý druh rámce, ať už vysílaný ze strany mikrokontroléru či PC aplikace, má pevně stanovený kód. Podle tohoto kódu pak příjemce rozhodne, jak zpracovat přijatá data a jak s nimi dále naložit.

7.2.3 Délka rámce

V tomto bajtu je přenášena celková délka rámce. To znamená, že maximálně může být přenesen rámec o délce 255 bajtů (tedy 251 bajtů vlastních dat). Podle velikosti tohoto pole příjemce rozpozná dokončen přenosu tohoto rámce.

7.2.4 Vlastní data

V rámci tohoto pole jsou přenášena vlastní data, tedy například informace potřebné pro inicializaci modulu, či stavy jednotlivých transceiverů.

7.2.5 CRC8

Pro kontrolu správnosti přijetí rámce je jako poslední byte odesílán kód CRC8. V tomto případě bylo zvoleno CRC8 podle specifikace SAE J1850. Tato specifikace požívá polynom 0x1D, tedy:

$$g8 = x^8 + x^4 + x^3 + x^2 + 1$$

Tento typ CRC byl zvolen kvůli hardwarové podpoře CRC jednotky použitého mikrokontroléru MPC5748G.

7.3 Komunikační rámce

Na základě výše popsaného komunikačního protokolu byly implementovány příkazy pro ovládání FlexCAN modulu a vizualizaci zpráv zasílaných prostřednictvím těchto modulů.

- **Inicializace (Kód: 0x01):**
  - Zpráva sloužící pro inicializaci zvoleného FlexCAN modulu.
  - Rámec o proměnné velikosti, kde datová část nese informace o čísle inicializovaného modulu, nastavení bitové rychlosti v arbitrážní a datové části rámce, nastavení filtrů a příslušných masek filtrů.
  - Touto zprávou je rovněž možno nastavit loopback mód, či deaktivovat vybraný modul.

- **Nastavení periodických zpráv (Kód: 0x02):**
  - Zpráva o proměnné délce, sloužící pro nastavení periodicky odesílaných zpráv pomocí daného modulu. (Každý modul v současnosti umožňuje nastavení pouze jedné periodicky odesílané zprávy).
  - V datové části této zprávy jsou neseny informace o identifikátoru dané zprávy, o použitém rámci (CAN/ CAN FD), nastavení periody odesílané zprávy (v ms) a vlastní data, jež budou zprávou odesílána.
- **Odeslání zprávy (Kód: 0x04):**
  - Rámec sloužící pro předání požadavku na odeslání jednorázové zprávy prostřednictvím zvoleného modulu.
  - Datová část rámce nese informace o identifikátoru zprávy, čísle požadovaného modulu a typ rámce, jež má být použit (CAN/ CAN FD).
- **Nastavení módu transceiverů (Kód: 0x05):**
  - Tento povel slouží pro nastavení zvoleného transceiveru do požadovaného operačního módu.
- **Přijatá zpráva (Kód: 0x11):**
  - Tato zpráva, odesílaná mikrokontrolérem, slouží pro získání informací o přijaté zprávě.
  - Tato zpráva nese identifikátor přijaté zprávy, číslo modulu, jež zprávu přijal, a vlastní data zprávy.
- **Odpověď na inicializaci (Kód: 0x21):**
  - Tato zpráva je odesílána mikrokontrolérem, jakmile je proveden pokus o inicializaci.
  - V rámci této zprávy je nesen výsledek inicializace a aplikace v PC pak může rozhodnout, zda byla provedena správně, či nikoliv.
- **Odpověď na nastavení periodické zprávy (Kód: 0x22):**
  - Toto potvrzení je odesíláno v reakci na požadavek na nastavení periodického odesílání zpráv (0x02).
  - Podle datové části může aplikace PC rozhodnout o správnosti provedení nastavení.
- **Odpověď na požadavek odeslání CAN zprávy (Kód: 0x24):**
  - Tato zpráva dává uživateli informace o úspěšném/neúspěšném pokusu o odvysílání zprávy, tedy o úspěšném/neúspěšném vykonání funkce ***TransmitMsg***.
  - Kladná odpověď touto zprávou však ještě neznamená úspěšné odvysílání zprávy na sběrnici.
- **Odeslaná zpráva (Kód: 0x31):**
  - Tato zpráva je odeslána po úspěšném odvysílání CAN rámce.
  - V datové části nalezneme informace o modulu, jež byl k odeslání použit, ID zprávy, a data, jež tato zpráva nesla.

  - Díky této zprávě má uživatel k dispozici databázi reálně odvysílaných CAN rámců na sběrnici.
- **Odpověď na nastavení módu transceiveru (Kód: 0x28):**
  - Tato odpověď slouží jako potvrzení o nastavení požadovaného operačního módu transceiveru.
  - V datové části zprávy je nesen výsledek operace nastavování tohoto požadovaného módu.
- **Heartbeat (Kód: 0x30):**
  - Tato zpráva je periodicky odesílána stanicí za účelem indikace správné funkce mikrokontroléru a transceiverů.
  - V datové části jsou neseny základní informace o současném stavu transceiverů.
  - Výpadek v příjmu této zprávy po dobu delší než 5s je indikován uživateli.
- **Stav chybových čítačů (Kód: 0x40):**
  - Zpráva odesílaná jednotkou mikrokontroléru, obsahující současné stavy chybových čítačů obsažených v registru CAN_ECR daného modulu FlexCAN.
  - Uživateli umožňuje získání přibližné informace o stavech čítačů:
    - „Transmit error counter“
    - „Receive error counter“
    - „Transmit error counter for fast bits“
    - „Transmit error counter for fast bits“
- **Stav chybového registru (Kód: 0x41):**
  - Tato zpráva nese kompletní stav chybového registru CAN_ESR1 daného FlexCAN modulu.
- **Požadavek na získání stavu chybových čítačů (Kód: 0x60):**
  - Tato zpráva je odesílána uživatelem prostřednictvím PC aplikace za účelem získání stavu chybových čítačů daného FlexCAN modulu.
  - V datové části je specifikováno konkrétní číslo modulu, kterého se tento požadavek týká.
- **Požadavek na získání stavu chybových registrů (Kód: 0x61):**
  - Tato zpráva je odesílána uživatelem prostřednictvím PC aplikace za účelem získání hodnoty chybového registru daného FlexCAN modulu.
  - V datové části je specifikováno konkrétní číslo modulu, kterého se tento požadavek týká.
- **Požadavek na čtení registru (Kód: 0x62):**
  - Tato zpráva je odesílána uživatelem za účelem získání stavu konkrétního registru použitého mikrokontroléru.
  - V datové části je přenášena adresa konkrétního registru.
- **Požadavek na zápis do registru (Kód: 0x63):**
  - Tato zpráva slouží pro zápis do konkrétního registru.
  - V datové části je nesena adresa registru a zapisovaná hodnota.

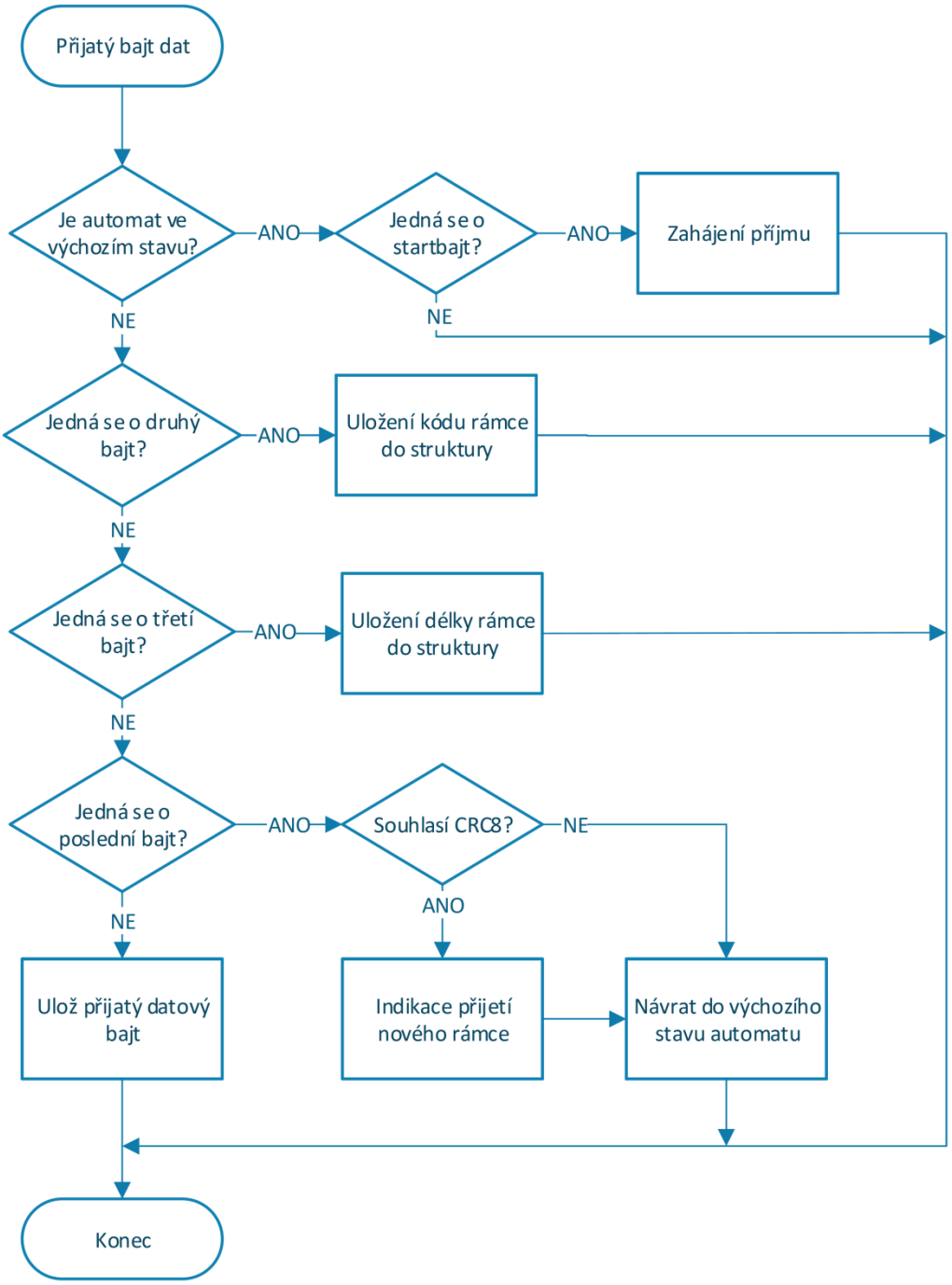
- **Odeslání obsahu registru (Kód: 0x64):**
  - Tuto zprávu odesílá mikrokontrolér jako reakci na požadavek čtení registru (0x62).
  - V datové části je nesena hodnota požadovaného registru.
- **Potvrzení zápisu do registru (Kód: 0x65):**
  - Tato zpráva je odesílána po pokusu o vykonání příkazu „Požadavek na zápis do registru“ (0x63).
  - V datové části je nesen výsledek operace (0 pokud operace proběhla úspěšně).

7.4 Balík funkcí mikrokontroléru pro testovací software

Pro testovací uživatelské rozhranní je samozřejmě nezbytná podpora na straně mikrokontroléru. Funkce zajišťující správnou komunikaci a dekódování obdržených příkazů jsou organizovány do souborů ***uart (.h, .c)*** a ***FSM (.h, .c)***.

Pro komunikaci prostřednictvím sériového rozhraní byla vytvořena knihovna, obsahující funkce pro příjem a odesílání zpráv. Tyto funkce jsou použity jak samostatně pro příjem rámců ze strany PC, tak v rámci funkcí, jež na základě událostí vytvoří definovaný rámec, jež je následně odeslán.

Správný příjem rámce příkazu je zajištěn funkcí ***FSM_handler***. Tato funkce, jejíž argumentem jsou jednotlivé přijaté bajty je schopna rozpoznat začátek rámce, a přepnout se do režimu jeho příjmu. Přijatá data jsou pak ukládána do definované struktury. Jakmile je příjem rámce dokončen, je ověřena jeho správnost porovnáním CRC. V případě správného přijetí je pak na základě kódu rámce přečten datový obsah, a vykonána požadovaná akce. O tuto akci se stará funkce ***execute_cmd.***



Obr. 40: Vývojový diagram stavového automatu, pro příjem komunikačního rámce po jednotlivých bajtech a kontrolu jejich správnosti.

7.5 Aplikace pro PC

Uživatelské rozhraní bylo vytvořeno vývojovým prostředím Visual Studio 2017. Jedná se o Windows Form aplikaci, napsanou v jazyce C#.

Tato aplikace je podle funkce rozdělena do několika formulářových oken tak, aby uživatelské rozhraní bylo co nejpřehlednější.

7.5.1 Hlavní okno aplikace

Toto hlavní okno je spuštěno s otevřením aplikace. Skrze něj je možno za použití menu otevřít ostatní pomocná okna aplikace.

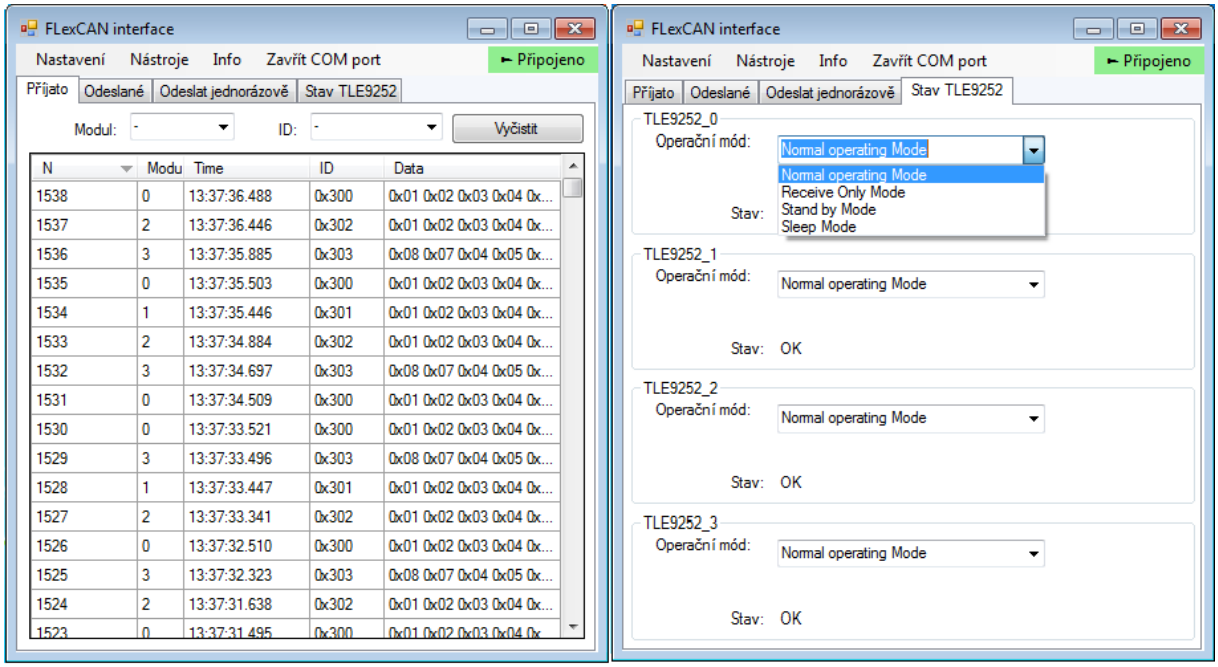
Rovněž zde může uživatel zobrazit a procházet seznamy přijatých a odeslaných zpráv prostřednictvím všech čtyř podporovaných modulů FlexCAN. Tyto seznamy zpráv lze filtrovat jednak podle čísla modulu, ke kterému náleží a též podle konkrétních ID. Tyto dva filtry lze libovolně kombinovat, a zvolit si tedy například zobrazení zpráv určitého modulu, nesoucí určité ID.

Tyto seznamy zpráv je dále možno řadit vzestupně či sestupně, podle libovolných položek sloupce. Toto řazení je nezávislé na nastavení filtrů.

Dále bylo pro další zpracování dat přijatých po CAN sběrnici vhodné implementovat možnost uložit tyto seznamy zpráv do .csv souboru. Tyto seznamy lze uložit s již aplikovanými filtry, a tudíž může uživatel například uložit pouze příchozí či odchozí zprávy konkrétního modulu.

Otevře-li uživatel záložku „Stav TLE9252“, dostane se mu možnosti vidět aktuální operační mód jednotlivých transceiverů a jejich případné chybové stavy. Toto okno rovněž umožňuje tyto stavy změnit a nastavit tak požadovaný operační mód.

Další funkcí tohoto hlavního okna aplikace je možnost odeslat jednorázovou zprávu. Uživatel vybere požadovaný modul a zadá parametry zprávy. Aplikace umožňuje zvolit mezi klasickým CAN rámcem a CAN FD.



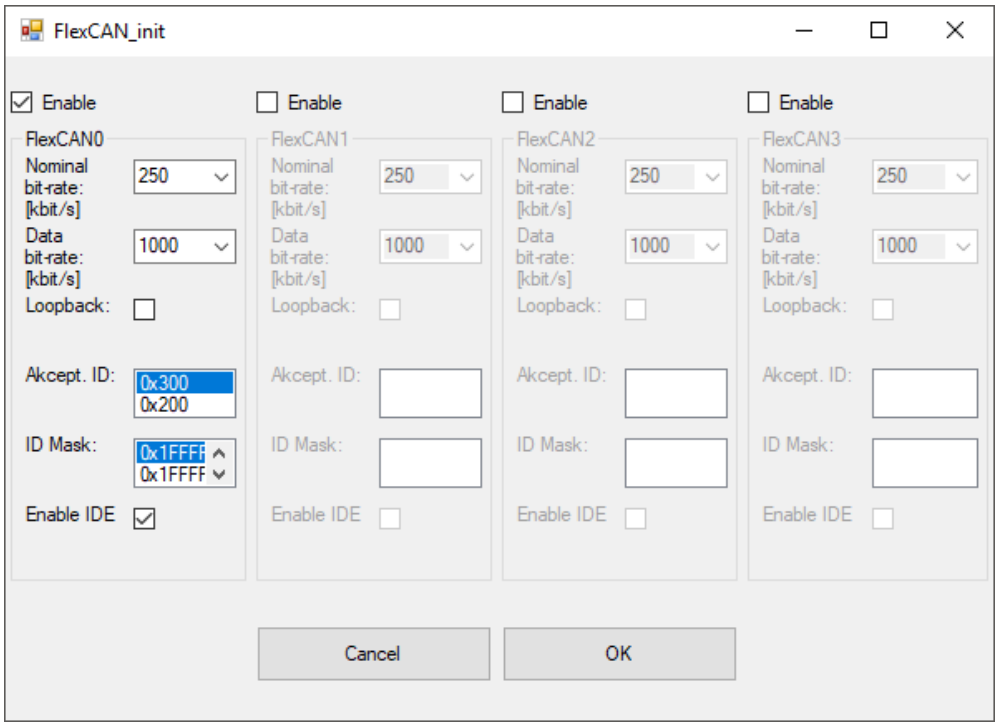
Obr. 41: Náhled na hlavní obrazovku aplikace. Záložka s přijatými zprávami zobrazena vlevo, záložka s operačními módy transceiverů vpravo.

7.5.2 Inicializační okno

Toto okno slouží pro inicializaci (případně zakázání) jednotlivých FlexCAN modulů. To je realizováno uživatelským rozhraním, umožňujícím nastavit přenosové rychlosti v arbitrážním i datovém poli, a zadání přijímaných identifikátorů a jejich masek. Z důvodu zlepšení uživatelského komfortu je při nastavování ID a jejich masek kliknutím na textové pole otevřeno nové okno pro zadání těchto parametrů ve větším a přehlednějším textovém editoru.

Dále lze nastavit podporu rámců s rozšířeným identifikátorem a zapnutí/vypnutí loopback módu.

Pro uživatelský komfort je samozřejmě nastavovaná volba zapamatována, tak, aby při potřebě úpravy nastavení nebylo nezbytné veškeré parametry nastavovat znovu. (Totéž platí i pro všechna ostatní okna, jež slouží k nastavení.)



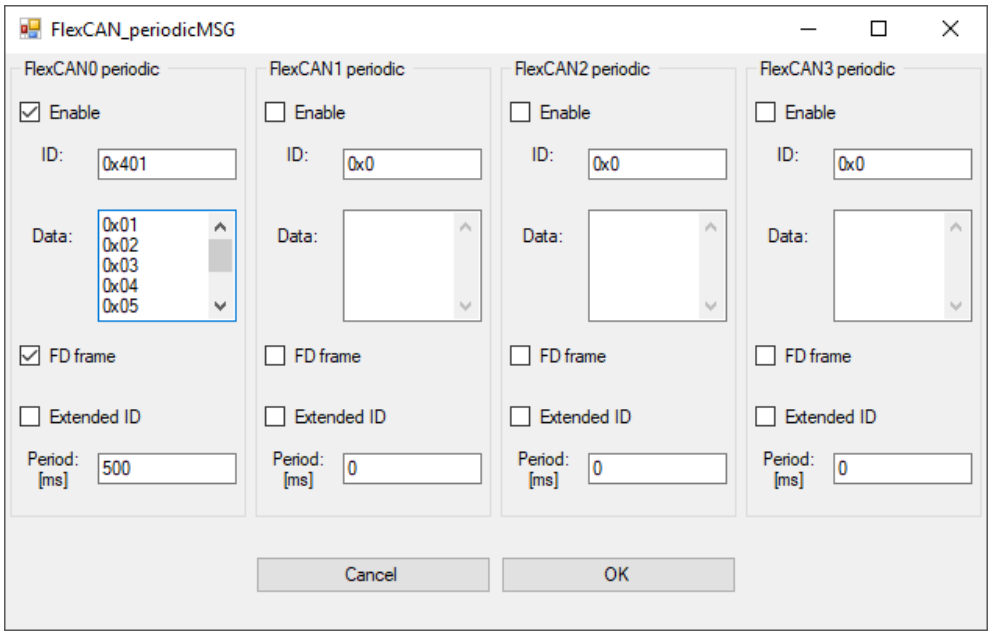
Obr. 42: Okno pro inicializaci všech čtyř FlexCAN modulů. (Zvolen modul 0)

7.5.3 Okno nastavení periodických zpráv

Okno pro nastavení periodických zpráv umožňuje pro každý modul nastavit periodicky odesílanou zprávu. (Tato možnost je na straně mikrokontroléru implementována v rámci souborů testovací aplikace, nikoliv vlastní FlexCAN knihovny.)

Pro každou takovou zprávu lze nastavit její identifikátor a data, jenž tato zpráva ponese. Dále se nastavuje perioda odesílání. Toto je hodnota v milisekundách mezi 10 a 65535 ms.

Rovněž lze nastavit, zda má být tato zpráva odeslána v rámci CAN či CAN FD.

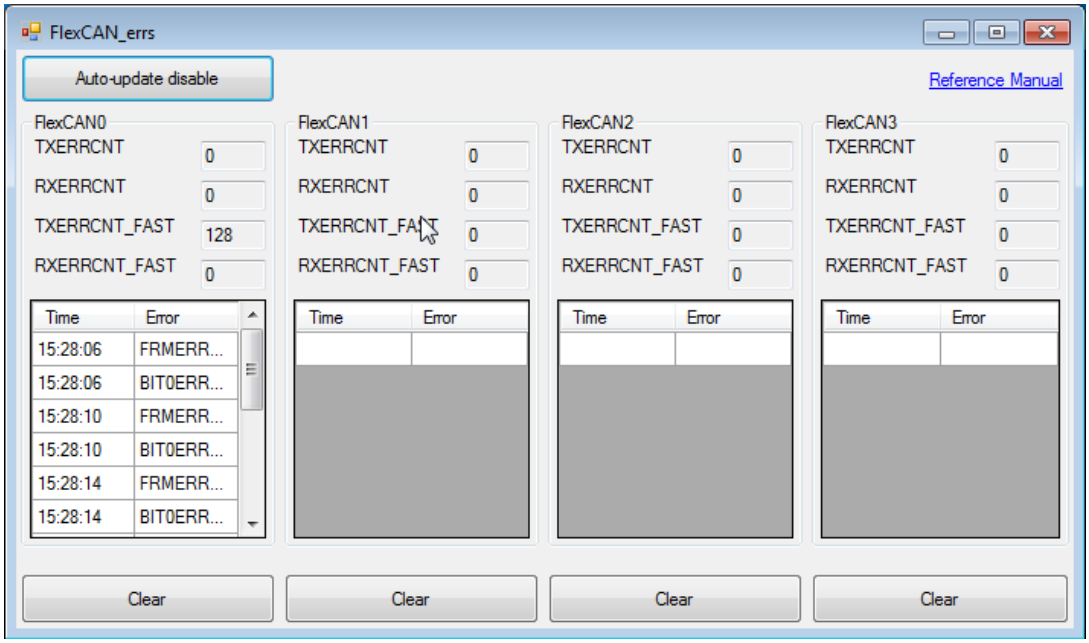


Obr. 43: Okno pro nastavení odesílání periodických zpráv pro všechny čtyři moduly.

7.5.4 Okno chybových hlášení

Toto okno slouží pro zobrazení stavů chybových registrů jednotlivých FlexCAN modulů.

V horní polovině lze vidět stavy chybových čítačů registru CAN_ECR, a ve spodní polovině pak výpis jednotlivých chybových bitů registru CAN_ESR1, včetně času přijetí této chyby. Aplikace umožňuje zapnutí/vypnutí automatické aktualizace těchto ukazatelů.



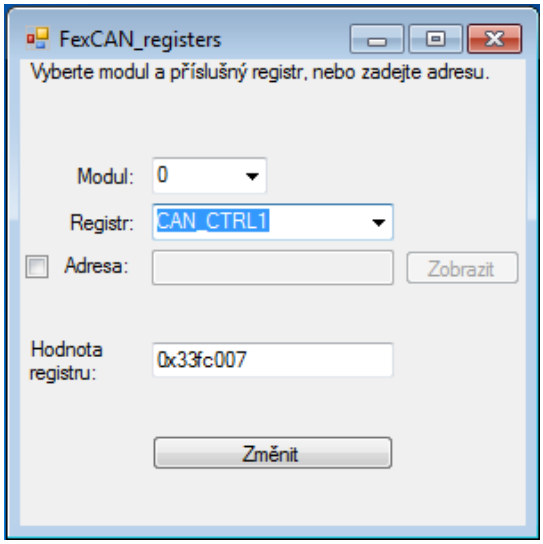
Obr. 44: Okno aplikace zobrazující jednotlivé chyby, včetně stavů chybových čítačů. (Pro tento příklad byl rozpojen jeden z vodičů kroucené dvojlinky modulu 0)

7.5.5 Okno přímé práce s registry

Tato funkcionalita byla vytvořena pro lepší možnost odladění software mikrokontroléru pomocí přímého čtení obsahu jednotlivých registrů.

Rozhraní umožňuje vybrání konkrétního registru FlexCAN periferie, či přímého zadání adresy paměti.

Dále je možné toto rozhraní využít k zápisu do některého z registrů. Tuto funkci je však třeba využívat velice obezřetně, neboť může vést k vyvolání neošetřené výjimky.



Obr. 45: Okno přístupu k registrům.

7.5.6 Okno připojení ke COM portu

Pro navázání komunikace s jednotkou mikrokontroléru slouží okno připojení ke COM portu. V tomto okně lze vyhledat příslušný COM port a navázat s ním spojení.

Navázání spojení je indikováno ukazatelem v hlavním oknu aplikace. Tento ukazatel může nabývat tří hodnot: „**Nepřipojeno**“, „**Připojeno**“ a „**Modul neodpovídá**“. Stav „Nepřipojeno“ signalizuje, že aplikace není připojena k žádnému COM portu. Stav „Modul neodpovídá“ signalizuje, že aplikace je připojena k některému z COM portů, nicméně po dobu delší než 5 sekund nebyla obdržena „heartbeat“ zpráva (kód 0x30).

7.5.7 Okno ostatního nastavení

Dále bylo vytvořeno okno pro volbu dodatečného nastavení uživatelského rozhraní. K tomuto nastavení patří možnost volby mezi zobrazením dat v desítkové, či šestnáctkové (hexadecimální) soustavě.

Rovněž je umožněno zakázat či povolit výběr buněk jednotlivých tabulek s daty (jak přijatými a odeslanými zprávami, tak tabulek s chybovými bity FlexCAN modulů).

8. Užití CAN FD pro přenos souborů

Pro demonstraci výhod CAN FD oproti klasickým CAN rámcům byl navržen software, umožňující přenos celých souborů mezi jednotlivými PC, prostřednictvím této sběrnice.

Za tímto účelem byla vytvořena aplikace v jazyce C# (v prostředí Visual Studio 2017) a rovněž byl vytvořen nezbytný software mikrokontroléru.

Pro umožnění odesílání velkých souborů byl vytvořen speciální protokol, jímž je zajištěno předávání dat mezi jednotkami.

V následujících kapitolách budou tyto části podrobněji popsány.

8.1 Protokol pro přenos souborů

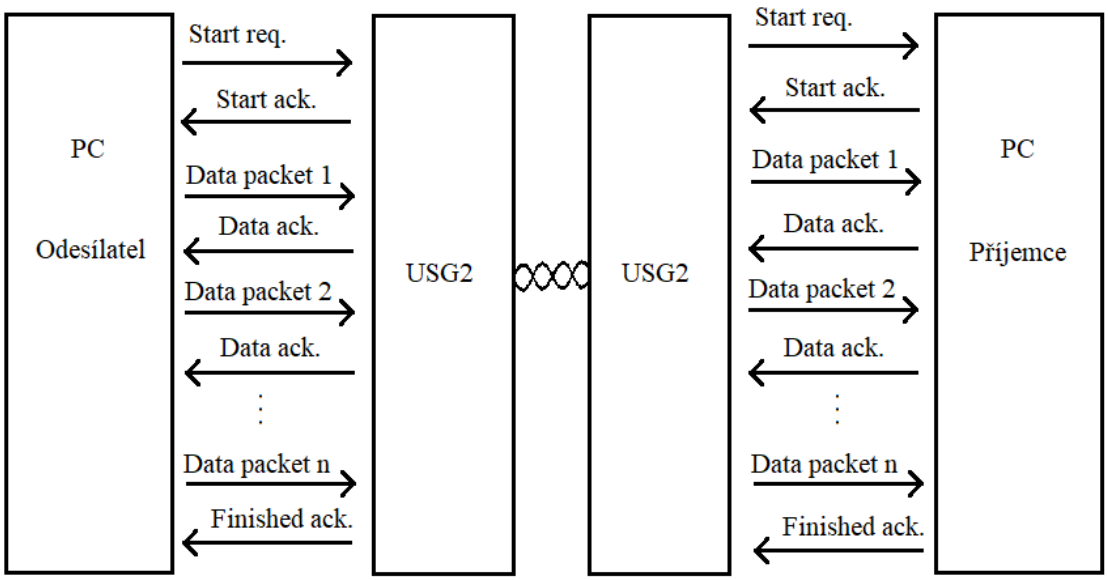
Je zřejmé, že při odesílání souborů je nezbytné nejprve rozdělit tyto soubory na jednotlivé menší pakety, jež pak lze předávat mezi PC a USG2 jednotkami.

V dané konfiguraci hardware dochází při přenosu jednoho paketu souboru, přenášeného mezi dvěma PC, ke třem datovým přenosům:

- Přenos datového paketu z prvního PC do jednotky mikrokontroléru, připojené k PC pomocí USB-UART rozhraní.
- Přenos paketu či jeho fragmentů z jednoho mikrokontroléru do druhého prostřednictvím CAN sběrnice.
- Přenos paketu z cílové jednotky mikrokontroléru do cílového PC pomocí UART-USB rozhraní.

Vzhledem k výše uvedenému způsobu přenosu paketů, je nezbytné řádně zajistit správnost předávaných dat. Je tedy vhodné, aby PC, jež data přijímá, měl možnost tato data ověřit a v případě detekování chyby adekvátně reagovat.

Pro naplnění stanovených požadavků byl navržen protokol komunikace znázorněný na následujícím obrázku.



Obr. 46: Grafické znázornění průběhu komunikace mezi dvěma jednotkami, kdy první jednotka (vlevo) předává soubor druhé jednotce (napravo).

8.1.1 Přenos paketů mezi PC a jednotkou USG2

Pro přenos jednotlivých paketů mezi PC a jednotkou USG2 byl v lehce pozměněném tvaru použit protokol popsaný v kapitole 7.2. K přenosu je využíván vytvořený USB-UART převodník, prostřednictvím kterého je odesílán rámec, do něhož jsou zabalena vlastní přenášená data. To umožňuje kontrolu úplnosti dat a jejich správnosti.

Oproti původní verzi tohoto protokolu je pozměněn význam třetího přenášeného bajtu. Tento bajt se, stejně jako tomu bylo v předchozím případě, vztahuje k délce přenášených dat. Pro tuto aplikaci, kde je očekáván přenos delších rámců, je však nevhodné přenášet délku celého rámce. Tento bajt nově vyjadřuje počet bajtů v datové části rámce snížený o jedna (např. při délce přenášených dat 64 bajtů bude hodnota tohoto bajtu 63). Tato, ačkoliv poměrně nepatrná změna se ukázala jako poměrně praktická.

Další změnou je použití CRC32 namísto CRC8. Tato změna byla provedena z důvodu nutnosti spolehlivě provádět kontrolu mnohem delších rámců, než tomu bylo v případě předchozí testovací aplikace. Pro výpočet CRC32 byl použit polynom 0x04C11DB7, jež je podporován hardwarovou jednotkou mikrokontroléru.

8.1.2 Typy rámců pro přenos souboru

Pro realizaci komunikace byly implementovány rámce, umožňující jak přenášení vlastních paketů dat, tak podpůrné rámce pro zajištění bezproblémového přenosu. V této kapitole je uveden jejich seznam a krátký popis.

- **Transmission start request (Kód: 0x11)**
  - Tento rámec odesílá iniciátor výměny dat.

  - V tomto rámci je přenášen název souboru pro odeslání, jeho velikost v bajtech, typ souboru, a identifikátory odesílatele a příjemce.
  - Příjemce má možnost se na základě údajů nesených v tomto rámci rozhodnout, zda soubor přijme, či nikoliv.
- **Transmission start acknowledge (Kód: 0x21)**
  - Příjemce souboru odesílá tento rámec jako kladnou odpověď požadavek pro začátek přenosu (Transmission start request).
  - Jednotka příjemce i odesílatele je po odeslání tohoto rámce v „uzamčeném módu“, kdy nepřijímá další požadavky na začátek přenosu.
- **Data paket (Kód: 0x12)**
  - Tento rámec nese část dat přenášeného souboru.
  - Velikost dat v paketu se může pohybovat od 1 až do 256 bajtů.
- **Data paket acknowledge (Kód: 0x22)**
  - Tento potvrzovací rámec je odsílán příjemcem souboru po přijetí paketu.
- **Transfer finished (Kód: 0x23)**
  - Tento rámec je odesílán příjemcem souboru v případě, že je přijat celý soubor.
  - Přijetím/odesláním tohoto rámce je přenos souboru ukončen.
- **Transmission abort (Kód: 0x41)**
  - Tento rámec může odeslat jak odesílatel souboru, tak jeho příjemce.
  - Přijetí tohoto rámce signalizuje ukončení přenosu z různých důvodů.
  - Přijetím, resp. odesláním tohoto rámce se jednotka vrací do výchozího režimu a je připravena k další činnosti.
- **Frame repeat request (Kód: 0xE0)**
  - Tento příkaz je určen k odeslání žádosti o opětovné odeslání naposledy odeslaného paketu.
  - Příkaz může odeslat jak odesílatel souboru, tak jeho příjemce v případě detekování chyby v přeneseném rámci.

8.1.3 Další komunikační rámce

Rovněž bylo vhodné zajistit řádné chování v případě neočekávaných chyb, a rovněž rámce, zajišťující network management. K tomu byly implementovány následující příkazy.

- **Software reset (Kód: 0xFF)**
  - Tento rámec není přenášen dalšímu modulu.
  - Jednotka po přijetí tohoto rámce okamžitě vyvolá softwarový reset a tím návrat do výchozího stavu.
- **Name set (Kód: 0x40)**
  - Jedná se o rámec předávající jednotce uživatelem zvolené jméno a tím umožňuje inicializaci jednotky

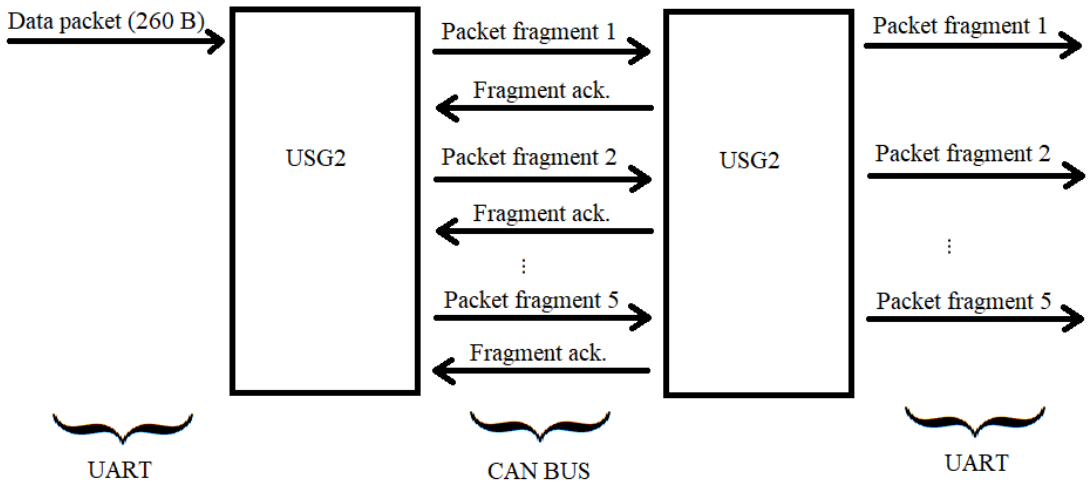
- **Name taken (Kód: 0x45)**
  - Tento rámec je odesílán jednotkou v případě odmítnutí zvoleného jména jinou jednotkou.
  - Uživatel je pak pobídnut k volbě nového jména.
- **Claim success (Kód: 0x46)**
  - Jednotka tímto rámcem informuje PC aplikaci o úspěšném získání ID.
  - Nese získaný identifikátor a zvolené jméno jednotky.
- **Collision (Kód: 0x4a)**
  - Tento rámec je odesílán jednotkou po zjištění konfliktu ID.
  - Aplikace po obdržení tohoto rámce umožní opětovnou inicializaci jednotky.
- **ID list update (Kód: 0x49)**
  - Rámec je odesílán jednotkou za účelem aktualizace seznamu připojených stanic.
  - Nese seznam identifikátorů a jmen aktivních jednotek.

8.1.4 Přenos paketů mezi jednotkami USG2

Poté co jednotka USG2 přijme prostřednictvím sériového rozhraní některý z výše uvedených rámců, je tento rámec odeslán další vybrané jednotce na sběrnici. V případě použití rámce CAN FD je možné většinu přenášených paketů odeslat pomocí jedné zprávy (maximální délka dat protokolu CAN FD je 64 bajtů). V případě datových paketů, které mohou mít velikost až 263 bajtů, či v případě použití klasického CAN rámce CAN je nutné odesílaný paket rozdělit na několik fragmentů.

Pro zajištění bezproblémového přenosu je přijetí a následné zpracování (předání PC pomocí sériového rozhraní) signalizováno odesláním potvrzující zprávy „fragment acknowledge“. Tím je zabráněno případnému problému s přepsáním předchozího fragmentu v případě pomalejší činnosti jednotky jenž tyto fragmenty přijímá.

Tento proces je znázorněný na následujícím obrázku.



Obr. 47: Příklad rozdělení datového paketu o velikosti 260 bajtů do pěti zpráv (čtyř nesoucích 63 bajtů dat a páté nesoucí 8 bajtů dat).

8.1.5 Rozdělení přenášených zpráv podle významu paketu

Pro zajištění správné funkce a umožnění filtrace je nezbytné, aby měl každý paket přiřazený identifikátor CAN FD zprávy, pod kterým je jeho fragment přenášen.

Totéž platí i pro zprávy, které jsou přenášeny interně mezi jednotkami, bez zásahu uživatele. V následující části jsou vypsány jednotlivé identifikátory zpráv a stručný popis jejich významu.

8.1.5.1 Zprávy interní komunikace

Tyto zprávy CAN (FD) jsou přenášeny pouze mezi jednotlivými jednotkami USG2, bez následného předání PC aplikaci. Aplikace uživatelského rozhraní tedy nemá k daným zprávám přístup. Na jejich základě však může dojít k odvysílání některého z rámců, informujících uživatele o některém stavu (např. informace o odmítnutí zvoleného jména jednotky).

- **Claim ID: ID = 0x100 + dočasné ID jednotky**
  - Tato zpráva je odesílána jednotkou, jež se pokouší získat vlastní identifikátor (ID jednotky) a vlastní jméno.
  - Obsahuje jméno jednotky (7 bajtů) a požadovaný identifikátor jednotky (1 bajt).
  - Jako dočasné ID jednotky je pro první žádost použito číslo, získané z požadovaného jména pomocí CRC8, aby byla minimalizována hrozba kolize s jiným ID. Pro následující pak ID navržené odpovídající jednotkou.
- **Refuse claim: ID = 0x600 + ID jednotky**
  - Tato zpráva je odesílána jednotkou detekující kolizi s požadovanými parametry obdrženými prostřednictvím „claim ID“ zprávy.
- **Heartbeat: ID = 0x700 + ID jednotky**
  - Tato zpráva, jež může nabývat identifikátorů 0x701 až 0x720 je periodicky odesílána každou jednotkou.
  - Součástí zprávy je i jméno jednotky, jež zprávu odesílá.
- **Fragment acknowledge: ID = 0x200 + ID jednotky**
  - Jak je patrné z obr. 47, jsou tyto rámce odesílány automaticky jako oznámení o zpracování přijatého fragmentu.

8.1.5.2 Zprávy pro přenos paketů PC aplikace

Tyto zprávy jsou odesílány jako prostředek pro přenos rámců vyvolaných uživatelem pomocí grafického rozhranní PC aplikace.

- **Transmission start: ID = 0x400 + ID jednotky**
  - Fragmenty zpráv odesílaných pro požádání o začátek přenosu souboru (zpráva nesoucí kód 0x11) mají identifikátory od 0x401 do 0x420.
- **Abort/Finish transmission: ID = 0x500 + ID jednotky**
  - Jak požadavky pro zastavení přenosu, tak zprávy pro oznámení dokončení přenosu (zpráva nesoucí kód 0x41 a 0x23), jsou odesílány prostřednictvím zpráv s identifikátory od 0x501 do 0x520.
- **Data fragment: ID = 0x300 + ID jednotky**
  - Fragmenty zpráv, přijatých pod kódem 0x12 (datový paket) jsou přenášeny jako CAN FD zprávy s identifikátorem 0x301 až 0x320.
  - Pod tímto ID jsou odesílány i zprávy sloužící pro potvrzení přijetí paketu.

8.1.5.3 Rozlišení cílové jednotky

Komunikace byla navržena tak, aby byla umožněna komunikace více jednotek. V takovém případě je zapotřebí realizovat mechanismus rozlišení příjemce daných paketů. Jelikož může každá jednotka vysílat zprávy pouze s vlastním ID jednotky, je nemožné rozlišit adresáta dané zprávy pouze podle tohoto ID, a je třeba zajistit jiný způsob určení.

Z tohoto důvodu je v datové části vyhrazen jeden bajt (první) pro přenos ID cílového uzlu. Podle tohoto bajtu pak jednotky, jež zprávu přijmou rozhodnou, zda je zpráva určena právě této jednotce.

Tab. 9: Význam bajtů přenášených v datové části zpráv.

Č. bajtu:	**0**	**1 - 64 (7)**
Význam:	**ID adresáta**	**Fragment přenášeného paketu**

8.2 Network management

V praxi je předpokládána existence vícero jednotek na téže CAN sběrnici. Tato skutečnost musí být zohledněna při volbě identifikátoru jednotky, jelikož v případě existence dvou jednotek se stejným identifikátorem by mohlo docházet ke konfliktům na sběrnici a nemožnosti tyto jednotky od sebe rozlišit.

Z tohoto důvodu je nezbytné, aby byl každý uzel jednoznačně rozpoznatelný, tedy měl jedinečný identifikátor. Vhledem k požadavku na variabilitu celého systému, kdy může libovolná jednotka k síti přistoupit či odstoupit je vhodné, aby tyto identifikátory byly rovněž variabilně jednotkám přidělovány.

Rovněž je vhodné, aby bylo možno ke každé jednotce kromě identifikátoru přidělit jméno, pod kterým bude tato jednotka v síti vystupovat. Díky tomu má uživatel možnost přehledného zobrazení odesílatele, či příjemce daného souboru.

8.2.1 Proces přidělení identifikátoru a jména jednotky

Pro získání identifikátoru jednotky byl navržen proces přidělování ID, jež je schopen zajistit přidělení identifikátoru jak v případě přítomnosti více již aktivních uzlů na sběrnici, tak v případě, že se daný uzel připojil jako první. Tento proces je znázorněn na obr. 48.

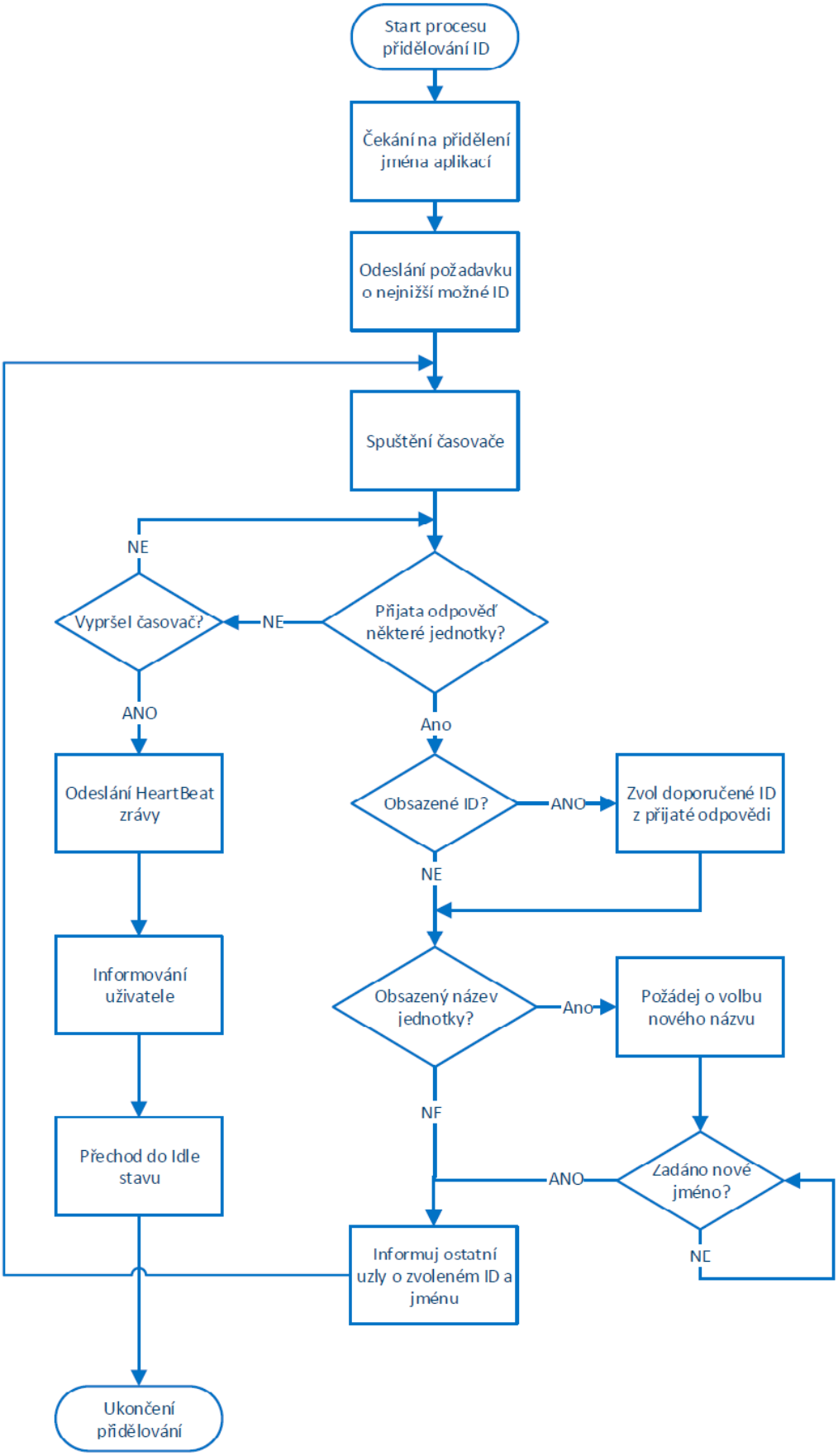
Nejprve jednotka čeká, dokud není přijat rámec, nesoucí zvolené jméno jednotky. Toto jméno je voleno uživatelem v rámci uživatelského rozhraní. Po přijetí jména jednotka odesílá požadavek na přidělení identifikátoru a tohoto jména ostatním jednotkám na sběrnici (tzv. „ID Claim“ zpráva). Jako požadovaná hodnota identifikátoru je přitom použita nejnižší možná hodnota, tedy 0x01.

Po odvysílání této zprávy jednotka spustí časovač, a čeká na případné přijetí „claim reject“ zprávy. Tato zpráva je odesílána jednotkou na sběrnici, která detekuje konflikt požadovaného ID či jména jednotky se svým vlastním. Aby bylo zaručeno, že je tato zpráva odeslána vždy jen jednou, prověří jednotka, která detekuje konflikt ve jménu rovněž konflikt ID jiného uzlu, a případně odeslání zprávy přenechá na jednotce s konfliktem ID. V případě konfliktu v ID pak vysílaná zpráva obsahuje hodnotu nejnižšího volného ID. Dále je přenášena informace o tom, zda je zvolené jméno volné či nikoliv.

V případě, že po vypršení časovače není obdržena „claim reject“ zpráva, oznámí jednotka prostřednictvím „heartbeat“ zprávy ostatním jednotkám svou přítomnost a vstoupí do „Idle“ režimu, kdy je připravena k činnosti.

Pakliže je přijata „claim reject“ zpráva, jednotka podle obsahu této zprávy vyhodnotí příčinu odmítnutí - tedy zda je obsazen požadovaný identifikátor, jméno jednotky, či oboje. V případě, že došlo k odmítnutí kvůli konfliktu jmen, je nezbytné nejprve zvolit jméno nové. Poté je opětovně odeslána „ID claim“ zpráva, a celý proces se opakuje.

I v případě uzlu, který nemá přidělený vlastní identifikátor, je nezbytné zajistit co možná nejlépe jedinečnost ID, se kterým odesílá zprávu „ID Claim“. K tomu je využito jméno, které uživatel volí. Toto jméno o velkosti sedm bajtů je zakódováno pomocí CRC8, a tato hodnota je použita jako dočasné ID jednotky. Možnost konfliktu v tomto případě je možná, ale velice málo pravděpodobná, jelikož by muselo dojít k připojení dvou různých jednotek se stejným CRC jména v rámci relativně krátkého časového rozmezí (1s).



Obr. 48: Proces přidělování identifikátoru a jména jednotky.

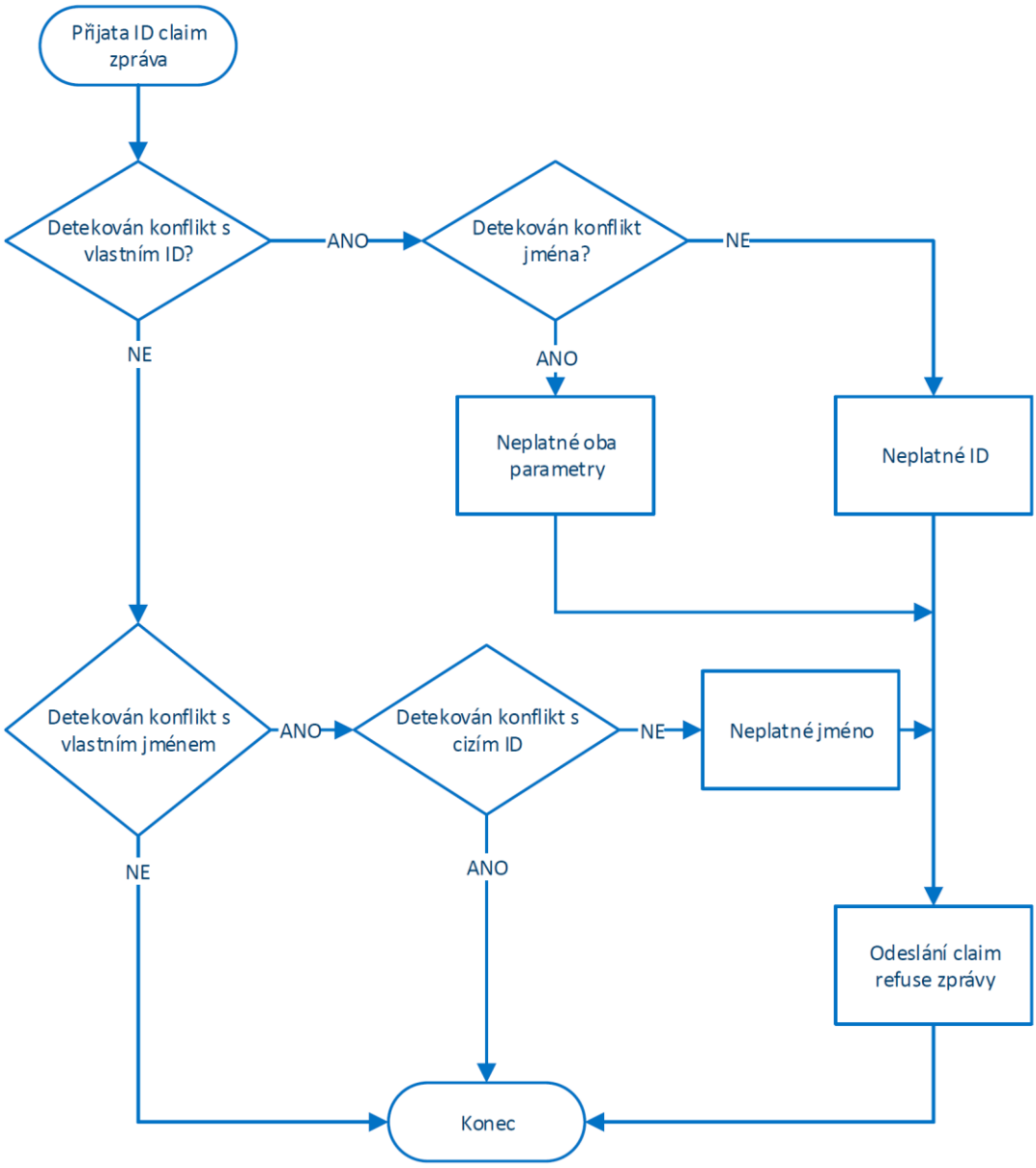
8.2.2 Proces ověření požadovaného identifikátoru a jména

Při přijetí požadavku na přiřazení adresy („ID claim“) je nezbytné jednoznačné určení odesílatele případného odmítnutí požadavku („claim reject“). Tím je zaručeno, že i při větším počtu uzlů na sběrnici nedojde k situaci, kdy budou na jeden požadavek o přidělení ID odpovídat dva, či dokonce více uzlů najednou, a nedojde tedy k případné nejednoznačnosti.

Pokud nový uzel odešle „ID claim“ zprávu, obdrží tuto zprávu každá jednotka na sběrnici. Jelikož však má každá jednotka k dispozici seznam připojených jednotek (identifikátorů i jmen), může snadno rozhodnout o tom, zda je právě tato konkrétní jednotka povinna odeslat případnou zápornou odpověď „claim reject“.

Existuje tedy pravidlo, že nejvyšší prioritu při odesílání „claim reject“ zprávy má jednotka, u níž nastal konflikt identifikátoru. Pokud není detekován konflikt v identifikátoru, odesílá zprávu jednotka, která má shodné jméno s požadovaným jménem.

Celý proces je realizován funkcí *FileSender_NMT_ClaimID_Response_Handler,* jež využívá funkce *FileSender_NMT_CheckForName* a *FileSender_NMT_CheckForID*. Princip je znázorněn na následujícím diagramu.



Obr. 49: Průběh vyhodnocení přijaté zprávy „ID claim“.

8.2.3 Objekt identifikátorů uzlů

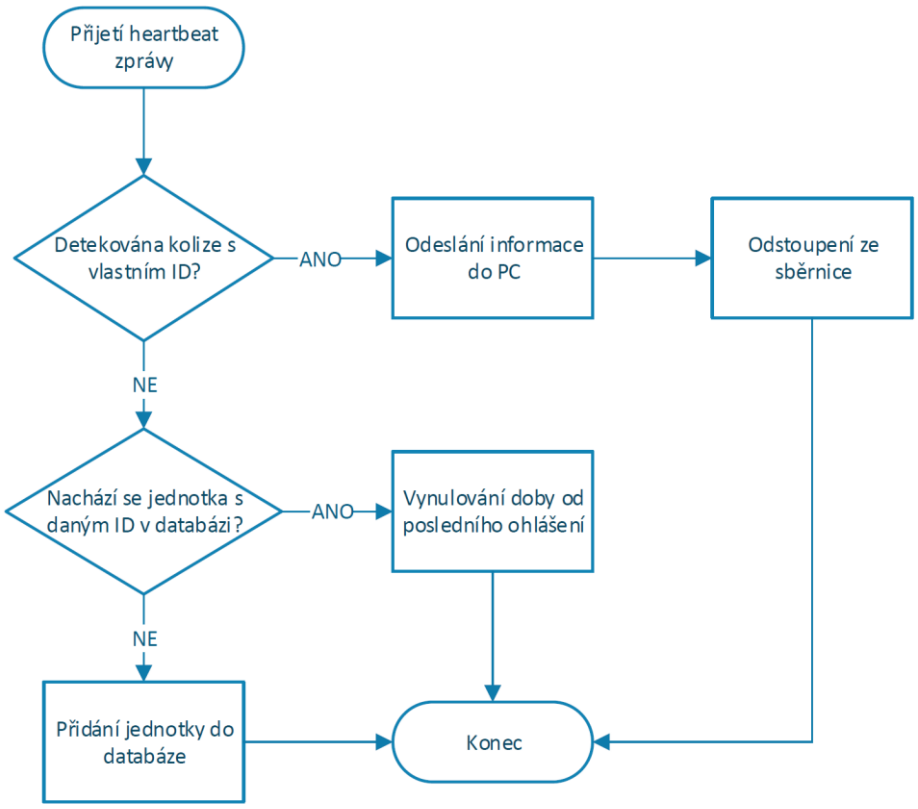
Aby byl celý network management (a potažmo celá koncepce sdílení souborů) aplikovatelný, je nezbytné, aby každá jednotka uchovávala informace o právě připojených aktivních uzlech. Tyto informace jsou obsaženy ve struktuře *FileSender_ID_obj*.

V tomto objektu jsou obsaženy informace jak o identifikátoru uzlu, jeho zvoleném jméně, tak o době (v milisekundách) od posledního přijetí heartbeat zprávy.

Tento objekt je pak používán při přidělování identifikátoru jednotky novému uzlu na sběrnici, či v případě odstoupení některého uzlu ze sběrnice. Tento objekt je rovněž sdílen s aplikací PC, díky čemuž má uživatel přehled o stanicích dostupných pro komunikaci.

8.2.4 Aktualizace objektu ID

Udržování objektu *FileSender_ID_obj* je nezbytné pro správnou funkci celého systému. K aktualizování tohoto objektu slouží „heartbeat“ zprávy, odesílané každým aktivním uzlem na sběrnici. Jakmile dojde jednotkou k přijetí „heartbeat“ zprávy je její identifikátor porovnán se současným obsahem *FileSender_ID_obj* struktury. Pokud zde identifikátor odesílatele není nalezen, je jednotka přidána do této databáze aktivních uzlů. Tento proces je obstaráván funkcí *FileSender_NMT_HB_Handler* a je znázorněn na následujícím diagramu.



Obr. 50: Proces vyhodnocení přijaté heartbeat zprávy.

Stav čítačů posledního ohlášení jednotky je periodicky inkrementován a po dosažení maximální povolené hodnoty je tato jednotka odebrána ze seznamu připojených zařízení. Rovněž je periodicky předáván seznam připojených zařízení PC aplikaci, která má tedy přehled o všech jménech aktuálně aktivních jednotek a jejich identifikátorů.

8.2.5 Prevence kolizí při přenosu

Při přenosu souboru je vhodné, aby tento proces probíhal bez vměšování dalších jednotek. K tomu by mohlo dojít například v případě, že jiná jednotka požádá o přenos souboru jednotku, která se již nějakého přenosu účastní. K zabránění takovéto situace lze využít softwarovou ochranu jak na straně software mikrokontroléru, tak v aplikaci realizující uživatelské rozhraní.

V tomto případě bylo zvoleno opatření již v software mikrokontroléru. Tento způsob byl zvolen z důvodu časové náročnosti přenosu rámce pomocí sériové komunikace, která by byla zapotřebí ve druhém případě.

Software mikrokontroléru po zahájení komunikace přijímá pouze vybrané zprávy, týkající se daného přenosu a network managementu. V případě přijetí jiných zpráv, které nejsou při přenosu relevantní, jsou tyto zprávy ignorovány. Jakmile je přenos souboru dokončen, vrátí se software mikrokontroléru do stavu, kdy tato filtrace zpráv není aktivní.

8.3 Uživatelské rozhraní

Uživatelské rozhraní pro přenos souborů bylo vytvořeno v prostředí Visual Studio 2017. Jedná se o Windows form aplikaci, složenou z několika oken. Aplikace neslouží jen jako grafický ovládací panel, ale řídí celý proces přenosu souborů. Software mikrokontroléru tedy ve své podstatě pouze filtruje a předává pakety přijímané prostřednictvím sériového rozhraní další jednotce přes CAN FD protokol a kontroluje správnost obdržených rámců.

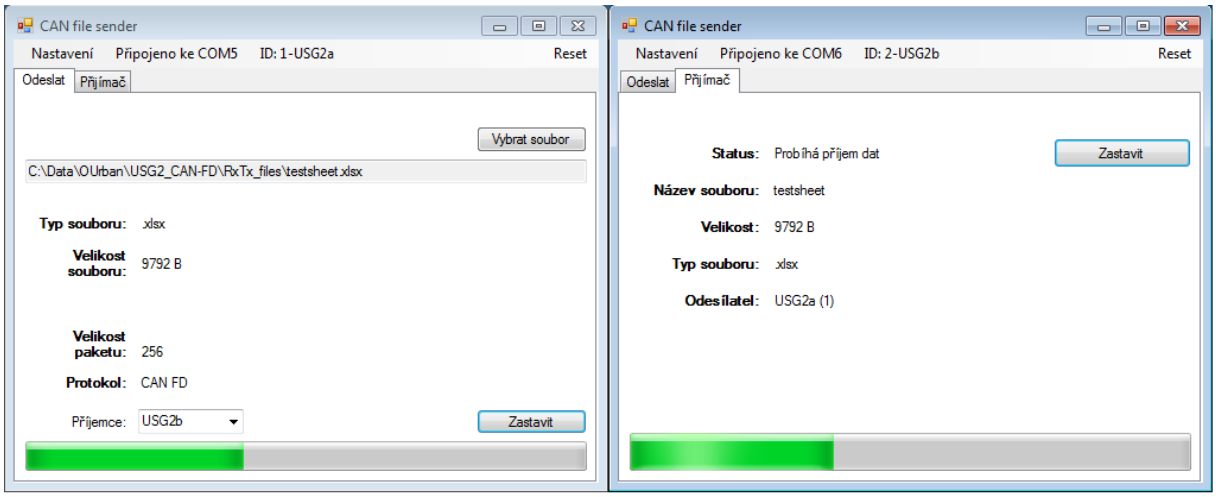
V následujícím textu bude tento software podrobněji popsán.

8.3.1 Hlavní okno aplikace

V hlavním okně aplikace nalezneme menu, umožňující otevřít další okno s nastavením přenosu. Dále je zde možnost otevření či uzavření vybraného COM portu (rovněž skrze nové okno), a v neposlední řadě možnost otevření okna, umožňujícího inicializaci jednotky.

Hlavní okno rovněž obsahuje dvě záložky. První nese označení „Odesílání“ a slouží k výběru požadovaného souboru a jeho odeslání.

Uživatel kliknutím na tlačítko „Vybrat soubor“ otevře dialogové okno, ve kterém vybere soubor k odeslání. V tabulce pod tlačítkem se poté zobrazí cesta k vybranému souboru. Následně uživatel vybere adresáta zprávy. K tomu slouží výběrový prvek v levé dolní části obrazovky, který obsahuje seznam všech dostupných stanic. Uživatel podle jména (jelikož jména jednotek na sběrnici jsou jedinečná) cílovou jednotku. Kliknutím na tlačítko odeslat je pak zahájen proces odesílání souboru.



Obr. 51: Uživatelské rozhraní při odesílání a přijímání souboru.

Panel označený „Příjem“ je automaticky přepnut v případě, že je od jiné jednotky přijat požadavek na zahájení přenosu souboru. V takovém případě jsou v tomto panelu zobrazeny informace o přijímaném souboru a název jednotky, jenž se pokouší tento soubor odeslat. Rovněž se zobrazí tlačítko, umožňující přijetí zprávy.

8.3.2 Okno inicializace jednotky

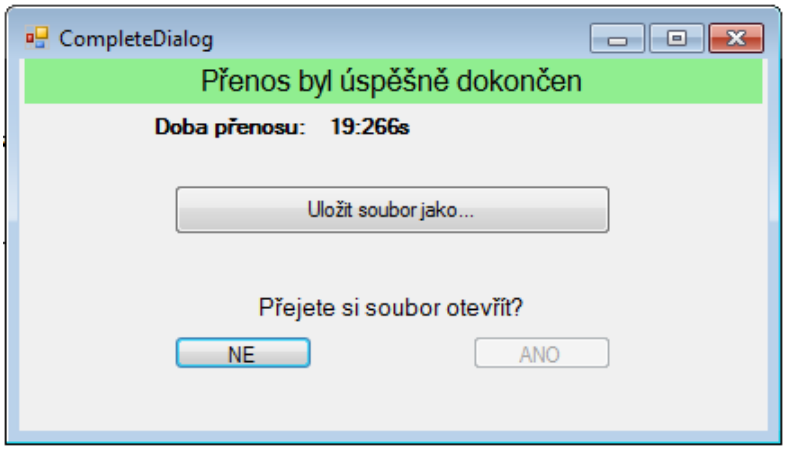
Prostřednictvím tohoto okna uživatel volí jméno, pod kterým bude daná jednotka na CAN sběrnici vystupovat. Po obdržení potvrzení o úspěšném přiřazení identifikátoru jednotky a jména je toto okno automaticky zavřeno, a uživatel již nemá možnost tuto inicializaci znovu provést. Jediným případem, kdy je to možné, je vyvolání resetu jednotky (tlačítko „Reset“ v hlavním okně), či při detekování kolize v identifikátoru či jméně a následném obdržení zprávy, požadující opětovnou inicializaci.

8.3.3 Oznámení o úspěšném přenosu

Jakmile je dokončen přenos (je odeslán, respektive přijat „Transfer finished“ rámec), je tato skutečnost signalizována uživateli.

V případě, že aplikace byla odesílatelem souboru, je zobrazena prostá zpráva s informací o dokončení přenosu a dobou přenosu.

Pokud aplikace soubor přijala, je otevřeno nové okno, oznamující tuto skutečnost, a nabízející uložení souboru. Po kliknutí na tlačítko „Uložit jako“ se otevře dialog s doporučovaným jménem souboru. Po úspěšném uložení se uživateli zpřístupní možnost tento soubor otevřít prostřednictvím výchozího programu, díky čemuž nemusí uživatel vyhledávat adresář s právě uloženým souborem.

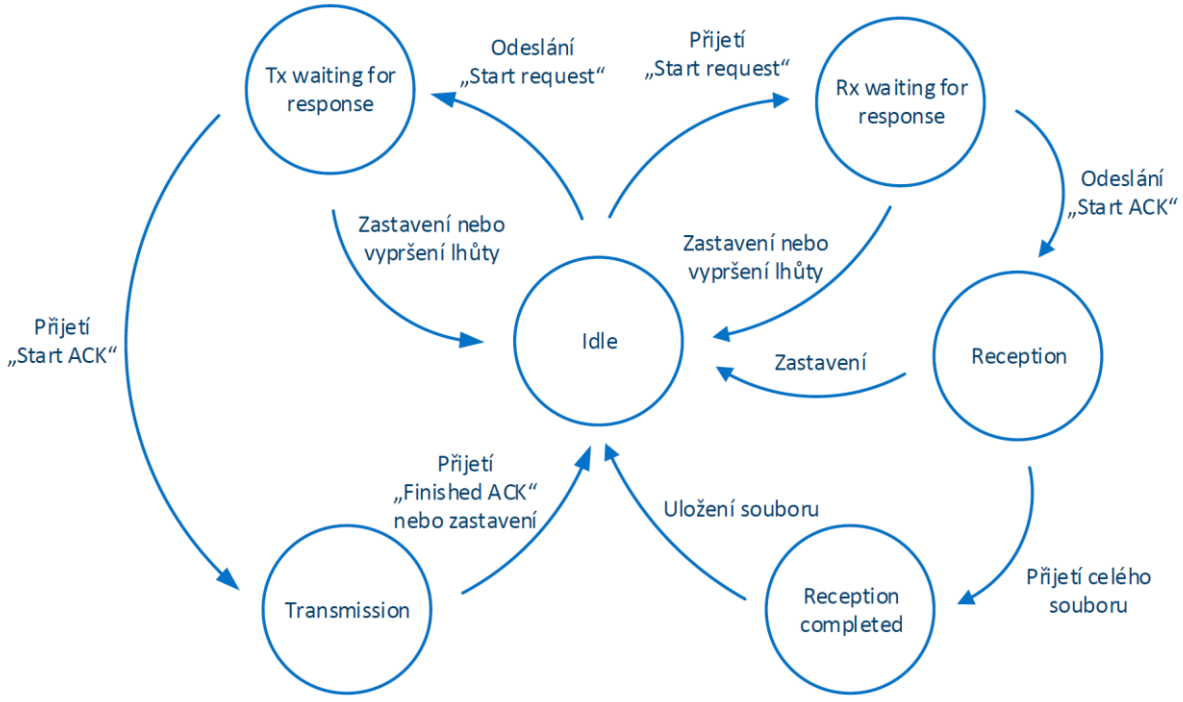


Obr. 52: Okno s informací o úspěšném přijetí souboru, nabízející uložení a následné otevření přijatého souboru.

8.4 Řízení přenosu souborů

Jak již bylo nastíněno, přenos souborů je řízen prostřednictvím PC aplikace. V této aplikaci je sestaven stavový automat, realizující příjem, zpracování a přenos jednotlivých paketů.

Na následujícím diagramu je znázorněn přechod mezi jednotlivými stavy.

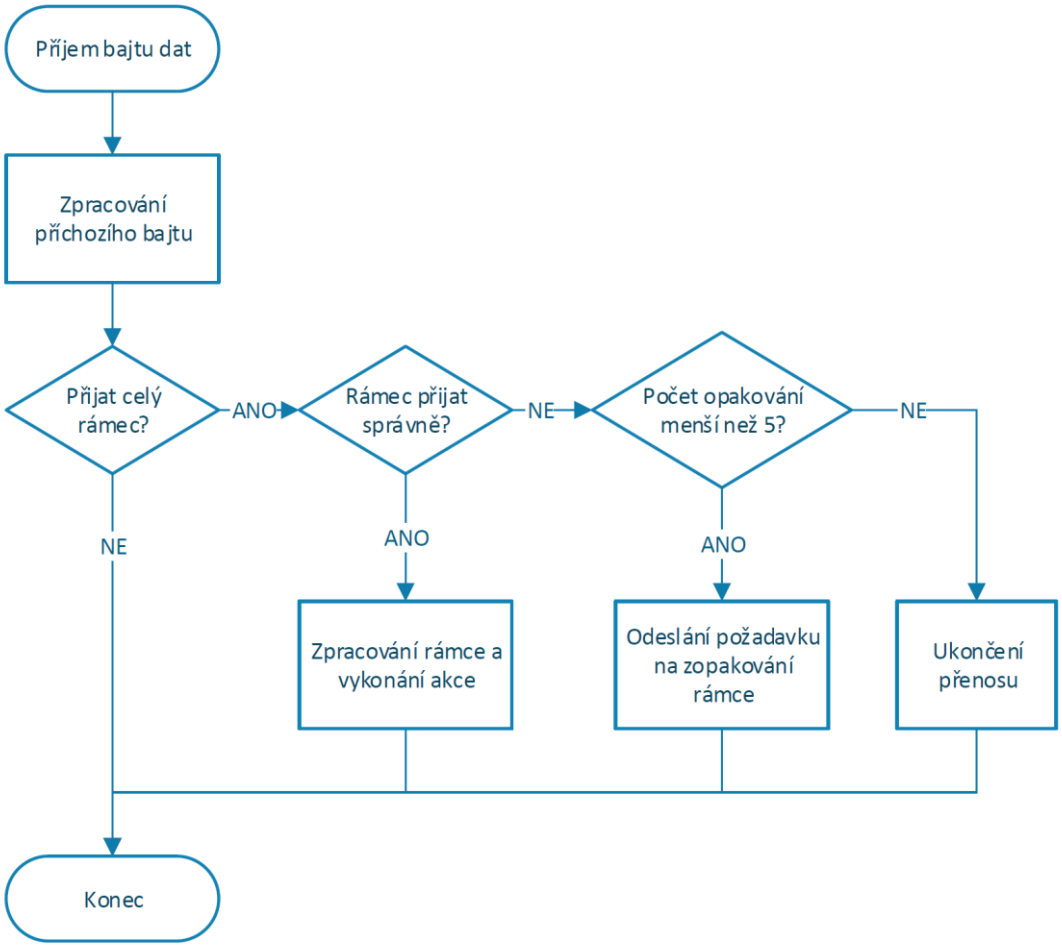


Obr. 53: Stavy aplikace pro odesílání souborů, a přechody mezi nimi.

Výchozím stavem je stav „Idle“. Aplikace v tomto stavu setrvává, dokud není odeslán požadavek na přenos souboru, nebo není tento požadavek přijat. V prvním případě se aplikace dostává do stavu „Tx waiting for response“. Aplikace nyní čeká na potvrzení od druhé stanice, čímž započne stav „Transmission“, kdy je možné odesílat pakety souboru. V případě zamítnutí, či vypršení časovače dochází k návratu do výchozího stavu. Pokud dojde v průběhu přenosu datových paketů k zastavení, nebo pokud je přenos dokončen, přechází stanice opět do výchozího stavu.

V případě, že aplikace ve výchozím stavu přijme požadavek na zahájení přenosu souboru, dostává se do stavu „Rx waiting for response“. Z tohoto stavu je možný návrat do výchozího stavu odmítnutím přenosu, nebo lze potvrzením přenosu přejít do stavu „Reception“, ve kterém jsou přijímány datové pakety. Příjem souboru ve stavu „Reception“ lze zastavit, a navrátit se tak do výchozího stavu. Po přijetí celého souboru dochází ke vstupu do stavu „Reception complete“, kdy je uživateli nabídnuto uložení souboru a jeho otevření. Po dokončení této akce je aplikace navrácena do výchozího stavu.

Proces vyhodnocování příchozích zpráv je znázorněn na následujícím vývojovém diagramu.



Obr. 54: Princip zpracování příchozích rámců zpráv a vykonání odpovídající reakce.

Aplikace periodicky kontroluje stav bufferu sériového přijímače, a v případě příchozího bajtu je volána funkce pro příjem rámce. Při příjmu celého rámce je spočteno CRC a porovnáno s přijatou hodnotou. V případě souhlasu obou hodnot je rámec postoupen jeho dalšímu zpracování. V případě, že se vypočtená hodnota liší od hodnoty přijaté, je odeslán požadavek na zopakování rámce. V případě, že se tato situace zopakuje pětkrát po sobě, je komunikace vyhodnocena jako nefunkční, a je ukončen přenos.

8.5 Porovnání dosažených rychlostí přenosu

Díky podpoře jak klasických CAN rámců, tak CAN FD je možno v reálné aplikaci vyhodnotit přínos tohoto protokolu. Za tímto účelem byla provedena řada přenosů souborů, a softwarově změřeny jejich doby.

Vzhledem k použití USB-UART převodníku pro komunikaci jednotky s PC aplikací nejsou však celkové časy přenosu na první pohled příliš vhodné k porovnávání. Maximální přenosové rychlosti, reálně dosahované u UART sběrnice, dosahují typicky 115 200 bit/s. To je v porovnání s přenosovou rychlostí CAN FD protokolu velice pomalé. Většina doby přenosu je tedy způsobena právě pomalým přenosem mezi PC a mikrokontrolérem.

Pro získání relevantního rozdílu mezi rychlostmi obou protokolů je tato aplikace však dostačující, neboť dobu přenosu po UART sběrnici je možno snadno vypočítat a následně od doby celkového přenosu odečíst, a získat tak dobu potřebnou pro přenos mezi dvěma mikrokontroléry.

Tab. 10: Naměřené časy přenosu pro soubor o velikosti 10 kB při použití protokolu CAN FD a klasického CAN.

	$t_{CAN\ FD}$ [ms]	t_{CAN} [ms]
	125	577
	125	515
	203	533
	113	560
	109	531
Průměr:	**135**	**543,2**

Z výše uvedených dat vyplývá, že rychlost přenosu po odečtení času potřebného pro přenos přes UART sběrnici byla podle očekávání značně vyšší při použití protokolu CAN FD. V tomto případě byl potřebný čas na přenos pouze **24,8%** doby potřebné při použití klasických CAN rámců.

Porovnáme-li dosaženou hodnotu z těchto měření s teoreticky očekávanou hodnotou podle obr. 29 (kapitola 2.8 - Porovnání CAN FD a CAN vzhledem k rychlosti) zjistíme, že dosažený poměr doby přenosu zhruba odpovídá předpokladům.

Důvod, proč je poměr v rychlostech přenosu ještě menší, než teoretický předpoklad, je rozdílný počet takzvaných „frame acknowledge“ zpráv, odesílaných jednotkou příjemce souboru. V případě protokolu CAN je v rámci jedné zprávy možno odeslat zhruba osmkrát méně dat než je tomu v případě CAN FD, a tudíž je odesíláno osmkrát více „frame acknowledge“ zpráv.

Tato úspora může v aplikacích, kde je při použití klasického CAN vysoké vytížení sběrnice přinést značné snížení této vytíženosti.

9. Závěr

Tato práce se zabývala vývojem rozšiřujícího modulu a software mikrokontroléru MPC5748G, umožňujícího použití protokolu CAN FD. Tento modul byl navržen, vyroben a otestován. Modul umožňuje současné použití čtyř FlexCAN periferií, a to až do bitové rychlosti 5 Mbit/s v datové části.

Pro možnost hlubšího otestování celého systému byl vyvinut software, umožňující ovládání celé periferie prostřednictvím grafického uživatelského rozhraní. Uživatel může tedy testovat chování FlexCAN modulu bez nutnosti debuggingu, či analyzátoru podporujícího CAN FD protokol.

Dále byl pro demonstraci výhod tohoto nového protokolu vytvořen software, umožňující přenos celých souborů přes tuto sběrnici. Přenos je řízen prostřednictvím PC aplikace, což však přináší některé komplikace týkající se omezeného datového toku UART komunikace. Získané výsledky nicméně dokládají velký přínos CAN FD protokolu. V praktickém nasazení bylo docíleno v průměru o 75% kratší doby, potřebné pro přenos stejného objemu dat. To v praxi znamená podstatné snížení vytížení sběrnice, což je při současném trendu narůstání počtu ECU a objemu přenášených dat v automobilech velice přínosné.

Na základě zkušeností s implementací CAN FD komunikace na prototypové vývojové platformě USG2 lze říci, že má tento protokol velký potenciál nahradit současně používaný protokol CAN, a to zejména díky malým nákladům na přechod mezi těmito dvěma protokoly, vyššímu zabezpečení komunikace a vyšší datové propustnosti.

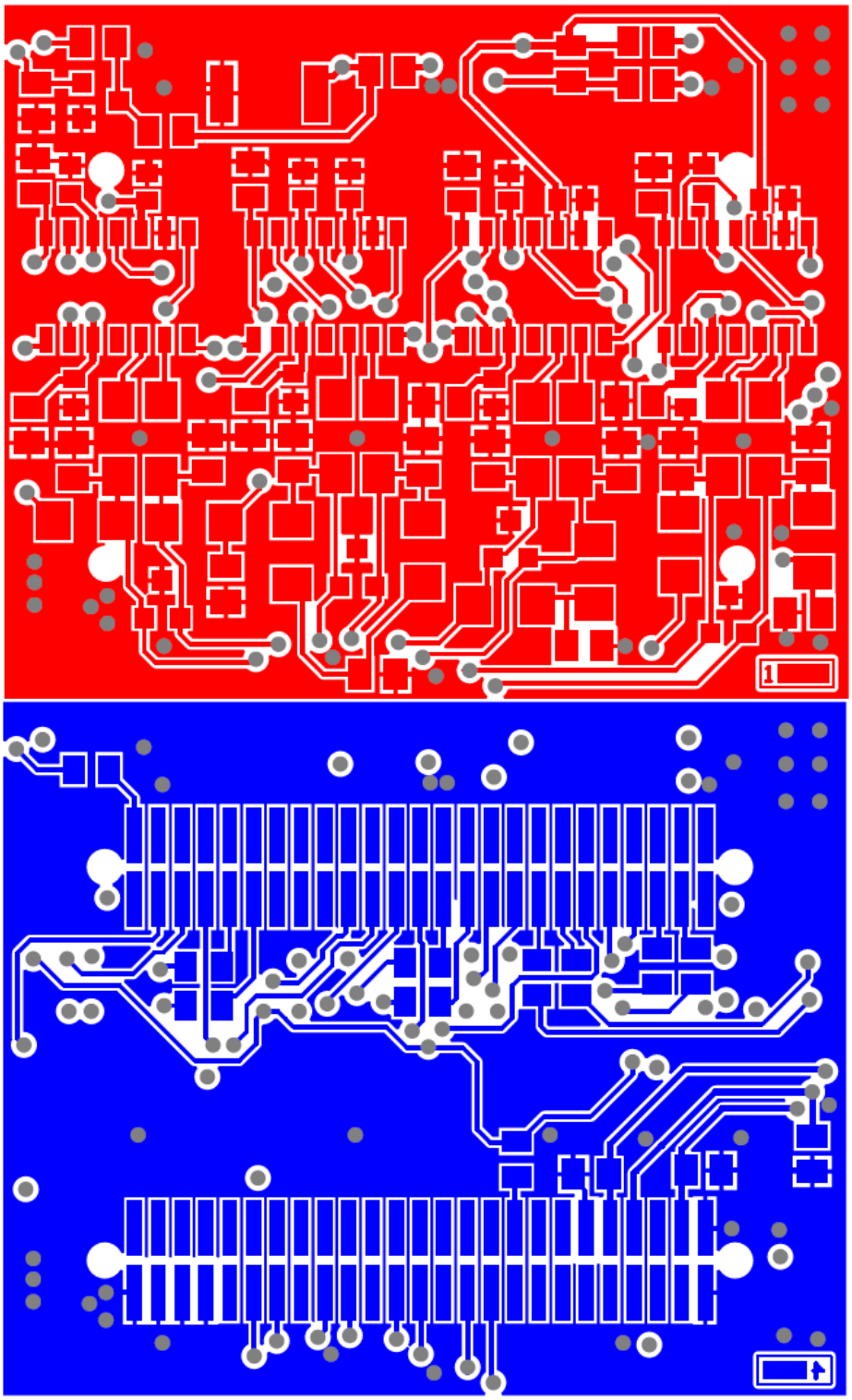
V současné době již probíhají vývojové práce na protokolu CAN XL, který by v budoucnu měl umožnit přenos dat o velikosti až 2 kB (2048 bajtů), při bitových rychlostech dosahujících i více než 10 Mbit/s. Uvedení tohoto protokolu je však stále otázka několika následujících let.

Zdroje

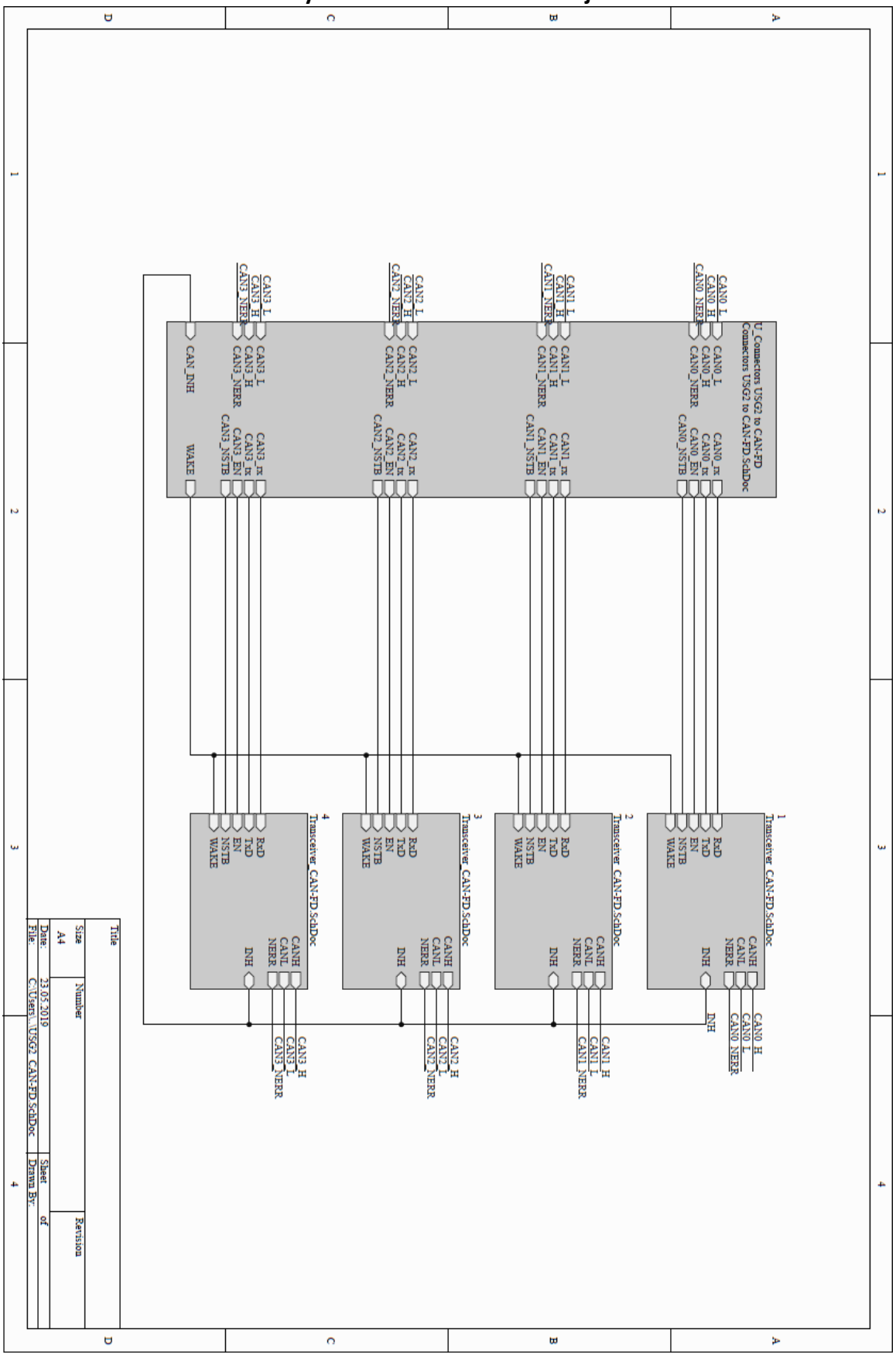
1. **International Organization for Standardization.** *ISO 11898-1:2015.* Ženeva : International Organization for Standardization, 2015.
2. —. *ISO 11898-2:2016.* Ženeva : International Organization for Standardization, 2016.
3. **Hell, Magnus Maria.** *WUP - The new bus wake up concept.* Norimberk : Infineon, 2016.
4. **Texas Instruments.** TCAN1042 Fault Protected CAN Transceiver with CAN FD. [Online] 2017. [Citace: 10. 2 2019.] http://www.ti.com/lit/ds/symlink/tcan1042h.pdf.
5. **Eisele, Harald K. a Wienckowsk, Natalie A.** *CAN FD From Concept to Production.* místo neznámé : General Motors, 2017.
6. **Ixxat.** SAE J1939 Technology. [Online] Ixxat. [Citace: 12. květen 2019.] https://www.ixxat.com/technologies/all4can/sae-j1939-technology.
7. **Simma Software.** J1939. *Simma Software.* [Online] 2018. [Citace: 2018. 10 3.] http://www.simmasoftware.com/j1939.html.
8. **SAE International.** *J1939-21.* 2006.
9. **Kvaser.** J1939 Standards Overview. *Kvaser.* [Online] [Citace: 2018. 10 5.] https://www.kvaser.com/about-can/higher-layer-protocols/j1939-standards-overview/.
10. **International, SAE.** *J1939-71.* 2003.
11. **Zeltwanger, Holger.** *Mapping of J1939 to CAN FD.* místo neznámé : CiA, 2016.
12. **Zitzmann, Reiner.** *CANopen FD – more than just higher bandwidth.* místo neznámé : CiA, 2017.
13. USDO – Universal Service Data Object. *Embeded Communication.* [Online] [Citace: 4. 12 2018.] http://www.embedded-communication.com/en/canopen/usdo/.
14. CAN in Automation. *CiA.* [Online] 2019. [Citace: 1. 5 2019.] https://www.can-cia.org/.
15. **Robert Bosch GmbH.** *CAN with Flexible Data-Rate.* Gerlingen : Bosh, 2012.
16. **Hartwich, Florian.** *Bit Time Requirements for CAN FD.* místo neznámé : Robert Bosch GmbH, 2013.
17. —. *CAN with Flexible Data-Rate.* místo neznámé : Robert Bosch GmbH.
18. **Sinha, Amit Kumar a Saurabh, Shubham.** *CAN FD: Performance Reality.* místo neznámé : IEEE, 2017.
19. **Texas Instruments.** *TCAN4550-Q1AutomotiveControlArea NetworkFlexibleData Rate (CAN FD) Controllerwith IntegratedTransceiver.* Dallas : Texas Instruments, 2019.
20. **NXP Semiconductors.** *MPC5748G Reference Manual.* místo neznámé : NXP Semiconductors, 2017.
21. **Infineon Technologies, AG.** *TLE9252V.* Mnichov : Infineon Technologies AG, 2018.
22. **Tomonari, Toshio.** *EMC Countermeasures for In-Vehicle Communication Networks.* místo neznámé : TDK Corporation.
23. **Skroppa, Ole Kristian a Monroe, Scott.** *Common Mode Chokes in CAN Networks: Source of Unexpected Transients.* místo neznámé : Texas Instruments, 2008.

Přílohy

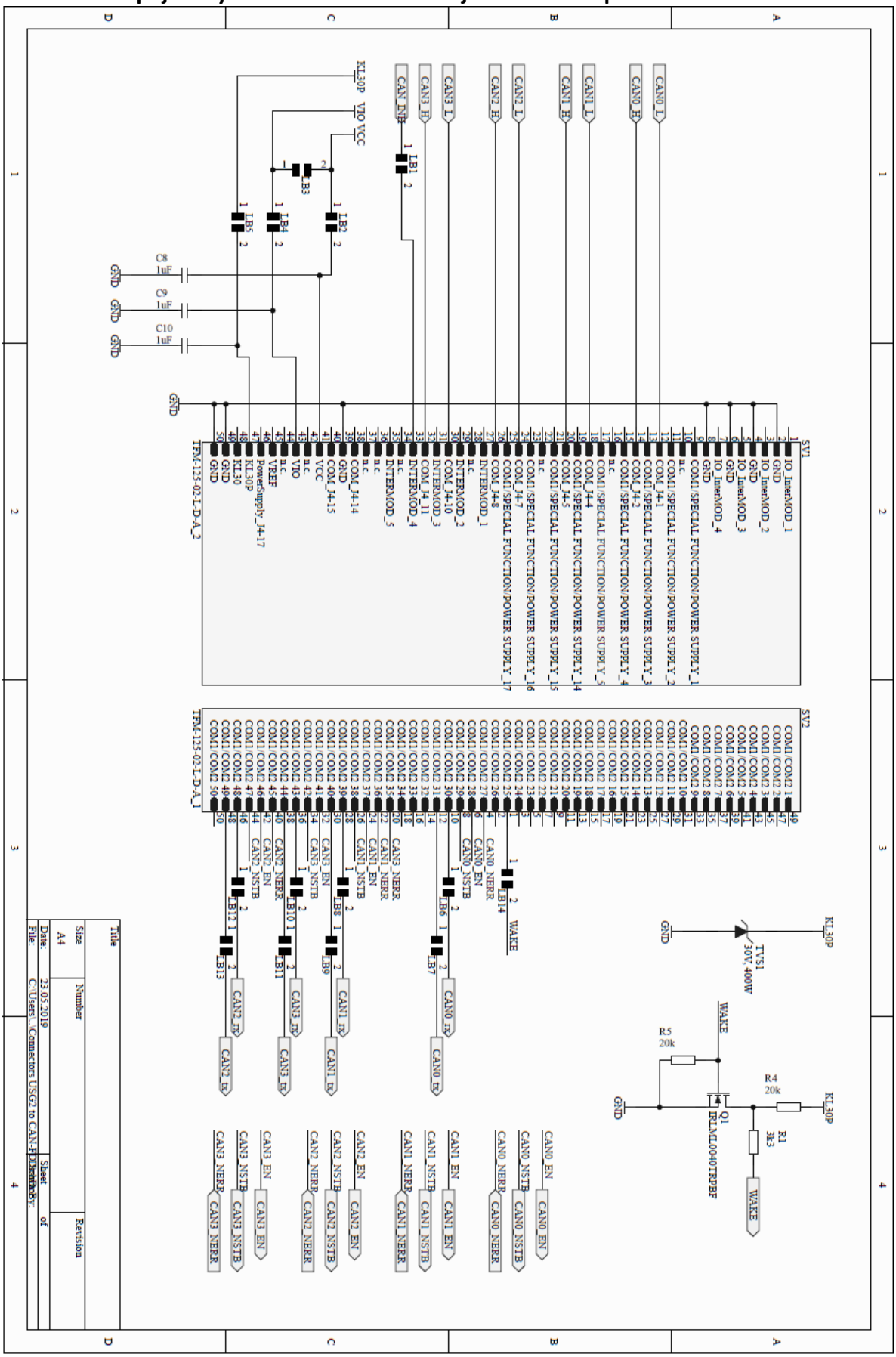
Příloha A – Horní a spodní vrstva vytvořené desky plošných spojů rozšiřujícího modulu



Příloha B – Vrchní schematický dokument CAN FD rozšiřujícího modulu



Příloha C – Zapojení vývodů konektorů rozšiřujícího modulu pro CAN FD



Příloha D – Modul USB-UART převodníku pro USG2 desku