TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Programové vybavení pro pokročilou analýzu sběrnic CAN a FlexRay

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika
Studijní obor: 3906T001 – Mechatronika

Autor práce: **Bc. Petr Vošta**
Vedoucí práce: Ing. Petr Pfeifer, Ph.D., MSc, MBA

Liberec 2016

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

CAN bus and FlexRay advanced analysis framework

Diploma thesis

Study programme: N2612 – Electrical Engineering and Informatics
Study branch: 3906T001 – Mechatronics

Author: **Bc. Petr Vošta**
Supervisor: Ing. Petr Pfeifer, Ph.D., MSc, MBA

Liberec 2016

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2015/2016**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Petr Vošta**
Osobní číslo:	**M14000207**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Mechatronika**
Název tématu:	**Programové vybavení pro pokročilou analýzu sběrnic CAN a FlexRay**
Zadávající katedra:	**Ústav informačních technologií a elektroniky**

Zásady pro vypracování:

1. Nastudujte dostupné informace a především standardy sběrnice CAN (ISO 11898).
2. Nastudujte dostupné informace a především standardy sběrnice FlexRay (ISO 17458).
3. Nastudujte možnosti osciloskopu Agilent řady 9000 a alespoň jedné další řady osciloskopů s velkou hloubkou záznamu jiného výrobce. Vhodně zakomponujte jejich podporu do vyvíjeného systému.
4. Nastudujte možnosti kitu Freescale S12XF a vytvořte program pro testy komunikace na sběrnici FlexRay.
5. Navrhněte a realizujte program pro analýzu navzorkovaných dat komunikace na sběrnicích CAN a FlexRay pomocí osciloskopů s velkou hloubkou záznamu. Do programu zakomponujte detekci a analýzu signálů těchto sběrnic, dekódování komunikace a chybových stavů především na nižších vrstvách dle modelu ISO/OSI a příslušných norem, spolu s detailním výpisem a časovými značkami ve formátu HTML.
6. Program vhodně zakomponujte do již existujícího systému inteligentního měřicího pracoviště se systémem dálkové správy a zpracování dat.

Rozsah grafických prací: **Dle potřeby dokumentace**

Rozsah pracovní zprávy: **cca 40-50 stran**

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

Seznam odborné literatury:

[1] **Normy ISO 11898:1-6 (CAN) a ISO 17458:1-5 (FlexRay)**

[2] **Dokumentace k osciloskopům Agilent řady 9000 a GW Instek 2000A.**

[3] **Dokumentace k vývojovému kitu FlexRay Freescale Semiconductor S12XF a mikroprocesoru řady M9S12XF.**

[4] **PRATA, Stephen. 2005. C++ primer plus 5th. Indianapolis : Sams, 2005. ISBN 0-672-32697-3.**

[5] **KERNIGHAN, Brian W a Dennis M RITCHIE. Programovací jazyk C. Vyd. 1. Brno: Computer Press, 2006, 286 s. ISBN 80-251-0897-x.**

[6] **DOMES, Martin. 2005. Tvorba internetových stránek pomocí HTML, CSS a JavaScriptu. Kralice na Hané : Computer Media s.r.o., 2005. ISBN: 80-86686-39-6.**

Vedoucí diplomové práce: **Ing. Petr Pfeifer, Ph.D., MSc, MBA**
Ústav informačních technologií a elektroniky

Datum zadání diplomové práce: **14. září 2015**

Termín odevzdání diplomové práce: **16. května 2016**

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
vedoucí ústavu

V Liberci dne 14. září 2015

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

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

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

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 16.5.2016

Podpis:

Poděkování

Tímto děkuji svému vedoucímu diplomové práce panu Ing. Petru Pfeiferovi, MSc, MBA, Ph.D. za vedení mé diplomové práce, konzultace, cenné rady a odbornou pomoc. Dále bych chtěl poděkovat za podporu své rodině.

Abstrakt

Diplomová práce se zabývá pokročilou analýzou sběrnic CAN a FlexRay. Začíná popisem obou sběrnic CAN a FlexRay a vývojovým kitem Freescale S12XF. Dále se zabývá popisem osciloskopů ze série Agilent 9000, GW Instek 2000A a Lecroy Wavesurfer 400 s ohledem na využití pro měření a následnou analýzu sběrnic CAN a FlexRay. V diplomové práci jsou také popsané jednotlivé funkce pro pokročilou analýzu sběrnic. Rovněž je řešena otázka optimalizací pro efektivní analýzu s maximální možnou rychlostí bez zásahu uživatele. Ke konci práce je uvedeno porovnání její rychlosti při spuštění na různých počítačích a porovnáním s komerčně dostupnými řešeními a implementací do existujícího inteligentního měřícího pracoviště.

Klíčová slova

Sběrnice CAN, sběrnice FlexRay, analýza sběrnice, osciloskop Agilent 9000, kit Freescale S12XF

Abstract

This diploma thesis deals with advanced analysis of CAN bus and FlexRay bus. CAN bus and FlexRay bus using Freescale S12XF development board is described. Further developed software processing the data from oscilloscope Agilent 9000 series, GW Instek 2000A series and Lecroy Wavesurfer 400 introduced and described as well, especially with regards to its use for measurements and analysis of the CAN bus and FlexRay bus. Functions to advanced analysis both the bus in the final application have been implemented and described in the thesis. The analysis has been optimized in order to maximize the speed and effective data processing without user interaction. In the final part of the thesis compares the speed of the final application on different computers applying the presented solution on selected commercially available platforms and implementation to the intelligent measurement system and AP9 framework.

Keywords

CAN bus, FlexRay bus, bus analysis, oscilloscope Agilent 9000, kit Freescale S12XF

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

CAN	Controller Area Network	Sériová sběrnice
CAN-L	Dominant Low	Signálový vodič sběrnice CAN
CAN-H	Dominant High	Signálový vodič sběrnice CAN
Idle_LP	Idle Low Power	Nečinný stav sběrnice s nízkou spotřebou
uBP		Signálový vodič sběrnice FlexRay
uBM		Signálový vodič sběrnice FlexRay
uBus		Stav sběrnice – rozdíl uBP a uBM
GND	Ground	Zemní nebo společný potenciál
ACK	Acknowledge	Potvrzení přijetí
x86	32bit system	32bitový systém
x64	64bit system	64bitový systém
GSa/s	Giga Samples per second	Miliarda vzorků za sekundu
MSa/s	Mega Samples per second	Milion vzorků za sekundu
kSa/s	Kilo Samples per second	Tisíc vzorků za sekundu
Gpts	Giga points	Miliarda bodů
Mpts	Mega points	Milion bodů
RIS	Random Interleaved Sampling	Náhodně prokládané vzorkování
TSV	Tab-Separated Values	Hodnoty oddělené tabulátorem
CSV	Coma-Separated Values	Hodnoty oddělené čárkou
HDF5		Datový model a přípona souboru
BIN		Formát a přípona binárního souboru

Html	HyperText Markup Language	Značkovací jazyk
RAM	Random Access Memory	Paměť s přímým přístupem
SSD	Solid State Drive	Pevný disk založený na flash pamětech
RMS	Root Mean square	Efektivní hodnota
PHY	Physical layer	Fyzické vrstva
TSS	Transmission Start Sequence	Sekvence začínající přenos
FSS	Frame Start Sequence	Sekvence začínající rámec
BSS	Byte Start Sequence	Sekvence začínající přenos bytu
FES	Frame End Sequence	Sekvence končící rámec
ASCII	*American Standard Code for Information Interchange*	Americký standardní kód pro výměnu informací

1 Úvod

Ke zpracování této diplomové práce na téma programové vybavení pro pokročilou analýzu sběrnic CAN a FlexRay mne přivedl můj předchozí projekt zabývající se základním systémem pro analýzu sběrnice CAN. Cílem tedy bylo tuto aplikaci vylepšit a rozšířit její funkčnost o další sběrnici, která by byla dekódovatelná bez použití specializovaných nástrojů, pouze s využitím osciloskopu a počítače. Cílem nástroje je tedy kromě samotného dekódování přenášených dat také analýza parametrů sběrnice a zjišťování chyb při přenosu a to nejen při vývoji systémů využívajících dané sběrnice, tak především při výuce. Díky zaměření na výuku jsou vybrány osciloskopy z řady Agilent 9000, GW Instek 2000A a Lecroy Wawesurfer 400, které jsou k dispozici na naší fakultě.

V úvodu práce je základní seznámení se sběrnicemi CAN a FlexRay z pohledu realizace jejich fyzické a linkové vrstvy, pro které je určená převážná část testů, a dále síťové vrstvy, díky které je možno dále dekódovat přenášená data. V další části je studium vývojového kitu Freescale S12XF, pro který je vytvořena jednoduchá aplikace pro testy komunikace na obou sběrnicích, se zaměřením pro využití ve výuce v rámci předmětů zabývajících se měřením. V neposlední řadě se zabývá výše zmiňovanými osciloskopy s ohledem na jejich využití pro analýzu zkoumaných sběrnic, především však na možnost dlouhého záznamu měřených dat.

Hlavní částí práce je výsledná aplikace, která má za cíl provést samotnou analýzu dat získaných z uvedených řad osciloskopů v co nejkratším možném čase s optimálním využitím hardwarových prostředků a bez zásahu obsluhy během analýzy. Díky těmto vlastnostem je pak integrovaná do existujícího systému inteligentního měřicího pracoviště se systémem dálkové správy a zpracování dat. Samotná analýza navzorkovaných dat se zaměřuje na detekování a měření parametrů sběrnice a z nich je možné například automatické rozpoznání, kterým kanálem osciloskopu byl vzorkován jaký vodič sběrnice, nebo správné dekódování přenášených dat včetně chybových stavů na fyzické i sítové vrstvě sběrnic.

V práci jsou popsané veškeré důležité funkce pro analýzu, kterými je například rozbor důležitých hlaviček, načítání dat a jejich komprimace, dekódování dat nebo výpis výsledků. Zároveň se práce zabývá porovnáním rychlosti výsledné aplikace na různých počítačích a porovnává ji s komerčně dostupnými řešeními.

2 Teoretický rozbor

2.1 Požadavky

Na výslednou aplikaci jsou kladeny požadavky na několik různých parametrů. Do první skupiny požadavků, které jsou nejdůležitější, patří zejména schopnost dekódovat soubory uložené zvolenými modely osciloskopů a analýza takto naměřených průběhů včetně detekce chyb především na fyzické vrstvě sběrnice, tak i na síťové vrstvě dle referenčního modelu ISO/OSI. Dalším požadavkem je detailní a srozumitelný výpis analyzovaných dat, ke kterému se pro svou univerzálnost hodí formát HTML, aby byl zobrazitelný na každém počítači pomocí webového prohlížeče. Dále aplikace musí umět data zpracovat v krátkém časovém úseku s co možná nejmenšími systémovými požadavky, aby aplikace fungovala rychle i na starších počítačích. V neposlední řadě je samozřejmá podpora x64 systémů a tím zaručená podpora zpracování i velikých naměřených souborů.

Do druhé, již trochu méně podstatné, skupiny požadavků spadá jednoduchost celé aplikace z pohledu uživatele. S tímto je kladen důraz také na autonomii celého řešení tak, aby bylo vhodné celou aplikaci integrovat do stávajícího systému inteligentního měřícího pracoviště se systémem dálkové správy, ale také aby byla jednoduše použitelná samostatně na libovolném PC. Díky autonomii není uživatel zbytečně obtěžován detaily, které se dají zpracovat přímo z naměřených dat, bez jakéhokoliv dalšího zásahu uživatele.

Pro pozvednutí výsledné aplikace ještě na vyšší úroveň oproti stávajícím řešením je nutné, aby byla co nejvíce univerzální, a tím byla dobře rozšiřitelná o podporu dalších osciloskopů, výstupních formátů dat, případně dalších sběrnic.

2.2 Sběrnice CAN

Tato kapitola obsahuje základní informace o sběrnici CAN, doplněné především o informace a údaje, které souvisejí s vyvíjeným analyzátorem a jeho požadovanými vlastnostmi.

2.2.1 Obecné informace

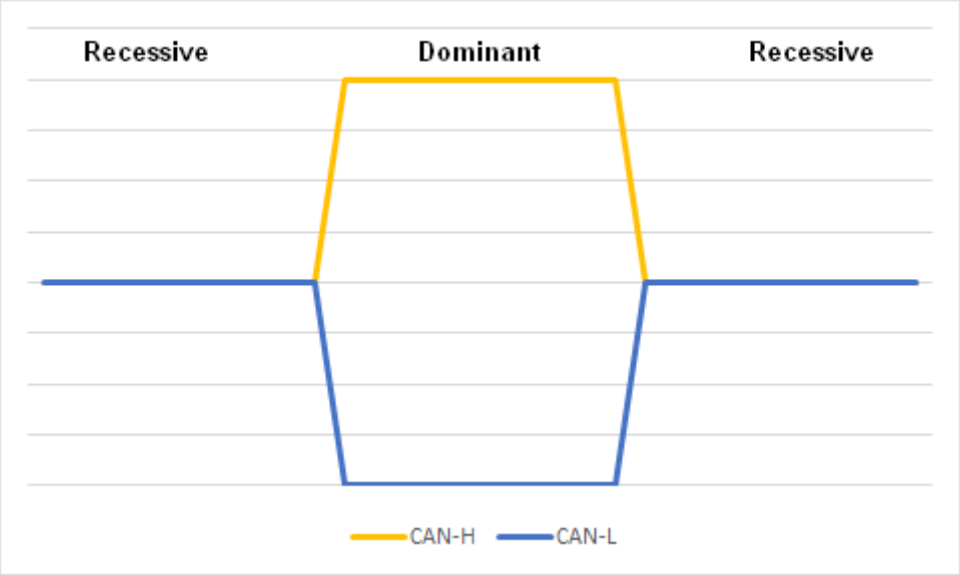
Sběrnice CAN byla vyvinuta firmou Bosch s primárním účelem nasazení v automobilovém průmyslu a nyní je definovaná normou ISO 11898. Normou je definovaná specifikace CAN 2.0A a později přibyla také specifikace CAN 2.0B. Obě specifikace definují podle modelu ISO/OSI pouze fyzickou a linkovou vrstvu. Aplikační vrstva je definovaná zvlášť několika různými protokoly, které nejsou zcela kompatibilní, například CAN Kingdom od firmy Kvaser, DeviceNet od firmy Allen-Bradlley, CANopen a další.

2.2.2 Parametry sběrnice

Fyzická vrstva je u sběrnice CAN definována odlišně, než je tomu u dalších základních sběrnic, lze použít k přenosu libovolné fyzické médium, které splňuje požadavek na logický součin. Díky tomu nejsou informace přenášeny logickými úrovněmi, jak je zvykem u ostatních základních sběrnic, ale pomocí dvou stavů sběrnice *dominant* a *recessive*. Oba stavy si lze představit jako zobecněné logické úrovně, kdy stav dominant by odpovídal logické úrovni 0 a stav *recessive* logické úrovni 1.

Dále se již budu věnovat pouze fyzické vrstvě, která je realizovaná elektricky. U elektrické fyzické vrstvy je potřeba dvou datových vodičů, které se označují CANH a CANL a jsou zakončené rezistory s odporem 120 Ω. Napěťovým rozdílem těchto dvou vodičů jsou definovány oba stavy sběrnice. Stavu *recessive* odpovídá podle napájecího napětí napěťový rozdíl (1) mezi CANH a CANL menší než 0,5V. Stavu dominant poté odpovídá napěťový rozdíl větší nebo roven 2 V viz Obrázek 1 - CAN sběrnice - stavy.

$$U_{DIF} = CAN_H - CAN_L \tag{1}$$



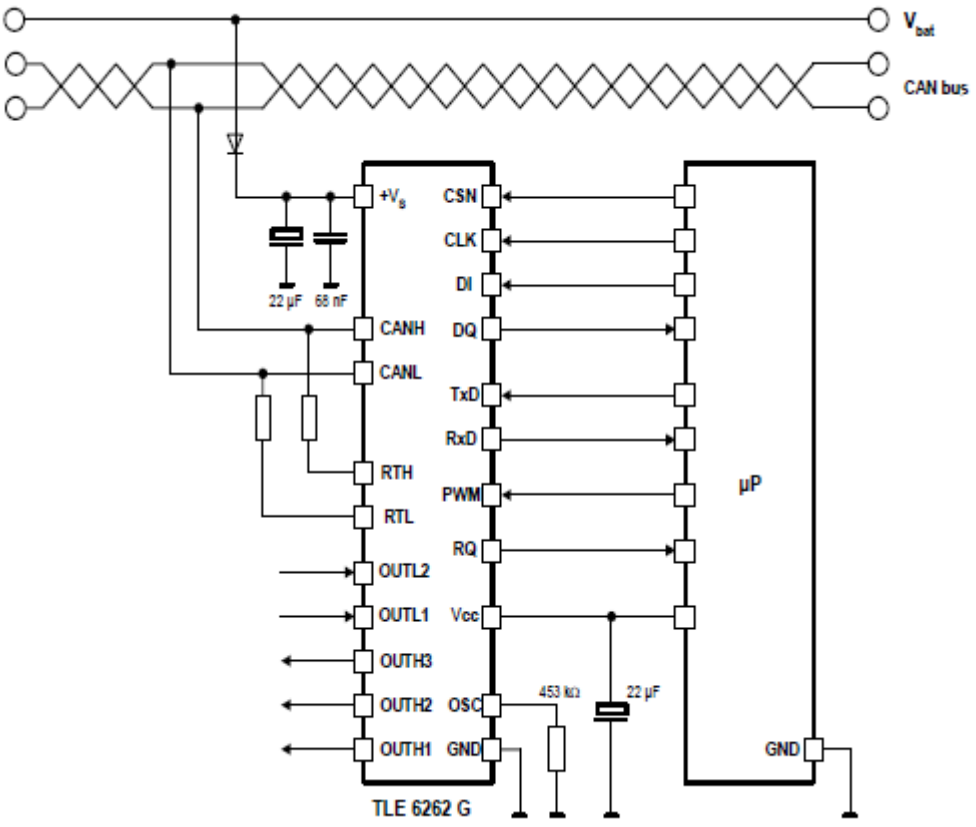
Obrázek 1 - CAN sběrnice - stavy

Nejvíce využitelná je detekce chyby na fyzické vrstvě sběrnic, která je nejčastěji využívána v automobilovém průmyslu. Při detekování jedné z možných a definovaných chyb na fyzické vrstvě, přepne řídící jednotka vozu, respektive PHY CAN sběrnice své vysílání/přijímání dat tak, aby bylo možné i přes tento chybový stav pokračovat v komunikaci, kdy se vysílač přepne do jednovodičového provozu. Seznam možných chyb, které nepřeruší komunikaci na sběrnici je vypsaná v Tabulka 1 - Popis chybových stavů fyzické vrstvy CAN. Chyby z Tabulka 1 - Popis chybových stavů fyzické vrstvy CAN se v automobilech mohou nejčastěji vyskytnout v mechanicky namáhaných částech vodičů, jako například u vodičů vedených do dveří vozu, kdy by svým chybovým stavem a znemožněním komunikace na sběrnici mohli způsobit například vyřazení z provozu centrálního zamykání, ovládání vícezónové klimatizace a dalších systémů.

Princip detekce chyb uvedených v Tabulka 1 - Popis chybových stavů fyzické vrstvy CAN je proveden pomocí připojení rezistorů mezi CANH a RTH, CANL a RTL. Zapojení je vidět na příkladu z technické dokumentace k obvodu TLE 6262 G respektive na obrázku Obrázek 2 - Příklad zapojení obvodu Infineon TLE 6262 G. Obvod je tak vybaven jedním diferenciálním přijímačem (CANH a CANL), a celkem čtyřmi samostatnými komparátory (CANH, CANL, RLH a RTL). Detekce je tak založena na připojování a měření napětí na všech čtyřech pinech obvodu.

Tabulka 1 - Popis chybových stavů fyzické vrstvy CAN

Číslo chybového stavu	Popis chybového stavu
1	Přerušení vodiče CANL
2	Přerušení vodiče CANH
3	Zkrat mezi vodiči CANH a CANL
4	Zkrat CANH s autobaterií, CANH > 7,2V
5	Zkrat CANH na společný potenciál GND
6	Zkrat CANH na napájecí napětí PHY, 1,8V < CANH < 7,2V
7	Zkrat CANL s autobaterií, CANL > 7,2V
8	Zkrat CANL na společný potenciál GND
9	Zkrat CANL na napájecí napětí PHY, 1,8V < CANL < 7,2V



Obrázek 2 - Příklad zapojení obvodu Infineon TLE 6262 G

2.2.3 Přenosové rychlosti

Dle normy ISO 11898-2 je maximální rychlost sběrnice CAN definovaná na 1 Mbit/s při délce sběrnice 40 m. Pro automobilový průmysl publikovala mezinárodní organizace SAE Internation specifikaci SAE J2284 [1]. Tato specifikace omezuje maximální možnou přenosovou rychlost pouze na 500 kbit/s. Pro ostatní použití bylo vydané doporučení CiA 102 [2], které počítá s rychlostmi od 10 kbit/s do 1 Mbit/s. Jak specifikace SAE J2284 tak doporučení CiA 102 definují délku sběrnice 40 metrů. Přehled rychlostí z CiA 102 viz Tabulka 2 - Přehled přenosových rychlostí sběrnice CAN dle CiA 102.

Jako referenční zařízení byl zvolený obvod SJA1000 od firmy Philips [3], který se dlouhou dobu v praxi užíval jako standard především v koncernu VW a byl ve své době považován i za průmyslový standard. Z dokumentace k tomuto obvodu vyplívá rovnice (2) pro výpočet všech možných přenosových rychlostí, pro nastavení BRP od 0 do 63.

$$f = \frac{1 \cdot 10^6}{BRP} \tag{2}$$

Tabulka 2 - Přehled přenosových rychlostí sběrnice CAN dle CiA 102

Rychlost	Čas jednoho bitu	Poznámka
1 Mbit/s	1 µs	Max. délka: 40 metrů
800 kbit/s	1,25 µs	
500 kbit/s	2 µs	
250 kbit/s	4 µs	
125 kbit/s	8 µs	
(100 kbit/s)	(10 µs)	Nedoporučováno pro nový vývoj
50 kbit/s	20 µs	Max. délka 1km
20 kbit/s	50 µs	Podporováno všemi moduly
10 kbit/s	100 µs	Minimální rychlost

2.2.4 Formát dat

Jak již bylo předesláno v kapitole 2.2.1, je formát přenášených dat definován specifikací CAN 2.0A a CAN 2.0B.

CAN 2.0A definuje základní datový rámec, který obsahuje 11 bitový identifikátor, reprezentující prioritu přenášené zprávy následovaný třemi bity. První určuje typ zprávy, zda se jedná o data nebo o vzdálený požadavek, následovaný bitem, určující o jaký typ rámce se jedná, zda základní nebo rozšířený. Poslední z této trojice je rezervní bit. Následují čtyři bity určující velikost přenášené zprávy. Následujících 0-64 bitů je datových, za kterými je 15 bitů kontrolního součtu následovaných bitem ukončující kontrolní součet, který je vždy *recessive*. V posledních devíti bitech je ACK následované bitem *ACK delimiter*, který musí být *recessive*. Posledních sedm bitů určuje konec rámce a všechny musejí mít stav *recessive*.

CAN 2.0B má začátek rozšířeného rámce stejný jako u základního. První rozdíl je po 11 bitovém identifikátoru, kde následuje bit SRR a IDE. Oba tyto bity musí být ve stavu *recessive* a bit IDE určuje, že se jedná o rozšířený rámec. Po bitu IDE, následuje druhý identifikátor, který má 18 bitů odpovídající prioritě zprávy následovaný bitem RTR, který má stejnou funkci jako bit po 11 bitovém identifikátoru ze základní zprávy. Posledním rozdílem jsou dva rezervní bity po bitu RTR a dále jsou již rámce totožné.

2.3 Sběrnice FlexRay

Tato kapitola se zabývá základním popisem sběrnice FlexRay se zaměřením na informace, které jsou potřebné pro vyvíjený analyzátor sběrnic s jeho požadovanými vlastnostmi.

2.3.1 Obecné informace

Sběrnice FlexRay byla vyvinuta konsorciem několika firem, mezi kterými byly zastoupeny firmy zabývající se vývojem a výrobou elektrotechnických součástek, tak firmy zabývající se automobilovým průmyslem. Mezi hlavní firmy podílející se na vývoji patří například NXP Semiconductors, Freescale Semiconductors, který již byl skoupen firmou NXP Semiconductors, dále firmami Robert Bosch, Daimler, Volkswagen, Ford, General Motors, BMW a dalšími. Toto konsorcium v dnešní obě neexistuje a sběrnice je tak definovaná normou ISO 17458, které je rozdělena na pět částí.

Sběrnice FlexRay byla vyvíjena s cílem vytvořit rychlou spolehlivou sběrnici, která bude využitelná pro moderní průmyslovou komunikaci, kdy je potřeba kromě již zmíněné rychlosti komunikovat s mnoha zařízeními s požadavkem na odolnost proti elektromagnetickému rušení. Využitelnost tedy kromě průmyslu měla být také v náročných aplikacích pro elektronické řízení a také jako nástupce za sběrnici CAN v automobilovém průmyslu.

V automobilovém průmyslu se však využívá především sběrnice CAN a nově i. Ethernet, který, se v automobilech využívá především v multimediálních centrech a pro chod automobilu v relativně nedůležitých periferiích. Hlavním konkurentem, a de facto i nutným základem pro sběrnici FlexRay je tedy sběrnice CAN, kdy v porovnání parametrů a ceny je pro mnoho automobilek samozřejmě stále výhodnější upřednostnit sběrnici CAN, která je levná a výrobci s ní mají mnoho zkušeností. Proto je sběrnice FlexRay u automobilů využívána především v případě nových řídicích systémů a v dražších a luxusnějších vozech, například od firmy Audi jsou to modely A6, A7, A8 Q7 nebo TT, BWM 5 a BWM7, případně vozy Laborgini Huracán, Bentley Mulsanne, Rolls-Royce Ghost, Volvo XC90, nebo Mercedes-Benz třídy S, C a E.

Výrobců elektrotechnických součástek pro sběrnici FlexRay pomalu přibývá a integrují její podporu do svých integrovaných obvodů. Mezi již zmiňované zástupce Freescale Semiconductor, NXP Semiconductor, přibylo několik dalších, například Fujitsu Microelectronics, NEC Electronics, Renesas, Texas Instruments, Xilinx nebo Infineon. Přehled některých integrovaných obvodů s podporou sběrnice FlexRay viz Tabulka 3 - Přehled vybraných integrovaných obvodů

Tabulka 3 - Přehled vybraných integrovaných obvodů

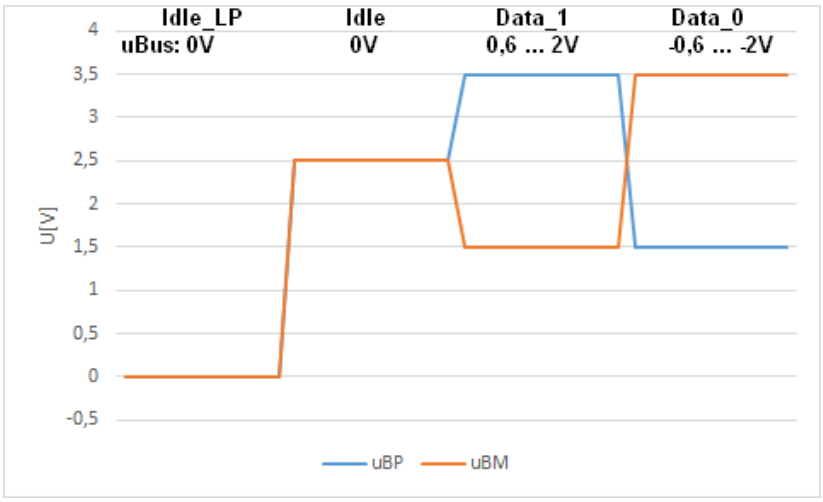
Výrobce	Integrovaný obvod	
Fujitsu Electronics	MB88121C	FlexRay řadič s podporou v2.1
	MB91F465XA	32bit 100MHz CPU
NEC Electronics	V850E/PHO3	32bit 128MHz MCU
	V850E/CAG4-M	32bit 80MHz MCU
Freescale Semiconductors (NXP)	MC9S12XF Series	16bit 50MHZ MCU
	MPC5510	32bit 80MHz MCU
	FRCC2100	FlexRay řadič
	TJA1080A	PHY, Ucc = 5V, Vbat = 60V
Texas Instrumens	TMS570LS	16/32bit 300MHz Cortex-R MCU
Xilinx	LogiCORE	Knihovny maker pro FPGA
Infineon	CIC-310	FlexRay řadič
Renesas	SH7450 Series	32bit 240MHz MCU
OnSemiconductor	NCV7381	FlexRay PHY
	NCV7383	FlexRay PHY

2.3.2 Parametry sběrnice

Fyzická vrstva sběrnice FlexRay je stejně jako u sběrnice CAN definovaná na libovolném fyzickém médiu, které splňuje požadavek na logický součin. Data jsou tak primárně přenášena ne jako logické úrovně signálu, ale určitými stavy sběrnice. V případě elektrického propojení se k tomu využívají dva vodiče s názvy *uBP* a *uBM*. U těchto dvou vodičů jsou definované celkem čtyři stavy sběrnice s názvy *Idle_LP*, *Idle*, *Data_0* a *Data_1*. Stavy *Data_0* a *Data_1* jsou definovány pomocí *uBus,* která na základě rozdílu napětí na vodičích *uBP* a *uBM* viz rovnice (3) tyto stavy reprezentuje. Pro lepší znázornění Obrázek 3 - Sběrnice FlexRay - stavy. Pro kladnou hodnotu *uBus* mezi hodnotami 0,6V a 2V odpovídá stav logické hodnotě jedna tj. stavu Data_1 a pro napětí *uBus* mezi hodnotami -0,6V až -2Vodpovídá daný stav logické hodnotě nula tj. stavu Data_0. Díky takto definovanému stavu *uBus* odpadá část problémů s nedodržením napěťových úrovní obou vodičů, respektive jejich rozdílu.

$$uBus = uBP - uBM \tag{3}$$

Na obou stranách sběrnice je nutné připojit mezi vodič uBP a uBM ukončovací rezistor, jehož hodnota je závislá na impedanci použité kabeláže a pohybuje se mezi 80 až 110 Ω. Zároveň je také nutné dodržet maximální délku vodičů mezi jednotlivými uzly sběrnice, která je stanovená na 24m a nesmí dojít k většímu zpoždění signálu než je 250ns. K přístupu na sběrnici se místo protokolu CSMA/CR jako u sběrnice CAN využívá TDMA, zajišťující rovnoměrný přístup ke sběrnici všem připojeným zařízením. Podrobněji elektrickou fyzickou vrstvu popisuje norma ISO 17458, viz [4].



Obrázek 3 - Sběrnice FlexRay - stavy

2.3.3 Přenosové rychlosti

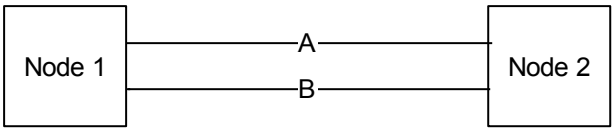
Tabulka 4 - Přehled přenosových rychlostí sběrnice FlexRay

Rychlost	Čas jednoho bitu
2,5 Mbit/s	400 ns
5 Mbit/s	200 ns
10 Mbit/s	100 ns

2.3.4 Topologie

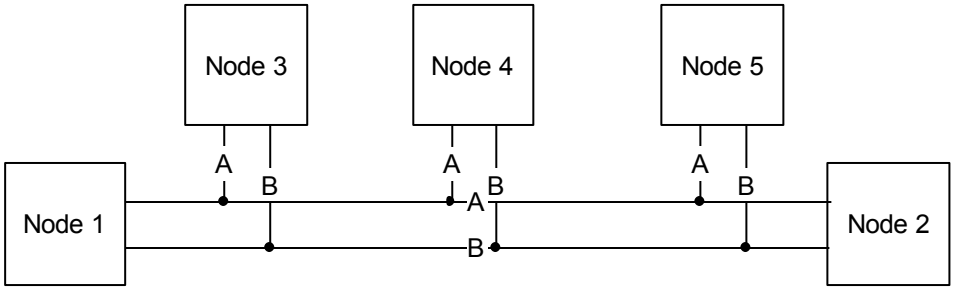
Každý uzel (zařízení) připojené ke sběrnici FlexRay má dva nezávislé kanály A a B, kterými může být do sítě připojen. Každý připojený uzel ke sběrnici může komunikovat s ostatními uzly na stejném kanálu, a není možné, aby jeden uzel na kanálu A komunikoval s jiným uzlem na kanálu B. Díky tomu, že na obou kanálech lze komunikovat nezávisle lze dosáhnou na sběrnici FlexRay celkového datového toku až 20Mbit/s.

Sběrnici lze zapojit do několika topologií, kdy nejjednodušší, a ve školních podmínkách zatím také jedinou, smysluplně realizovatelnou topologií je přímé zapojení dvou zařízení (Point to Point) viz Obrázek 4 - Topologie Point to Point.



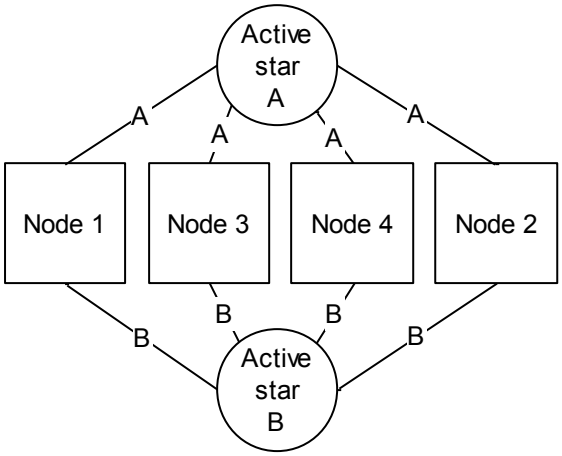
Obrázek 4 - Topologie Point to Point

Další základní a jednoduchou topologií, která se dá aplikovat je pasivní sběrnice, kdy jsou všechny uzly sběrnice připojeny ke společným vodičům, viz Obrázek 5 - Topologie pasivní sběrnice.

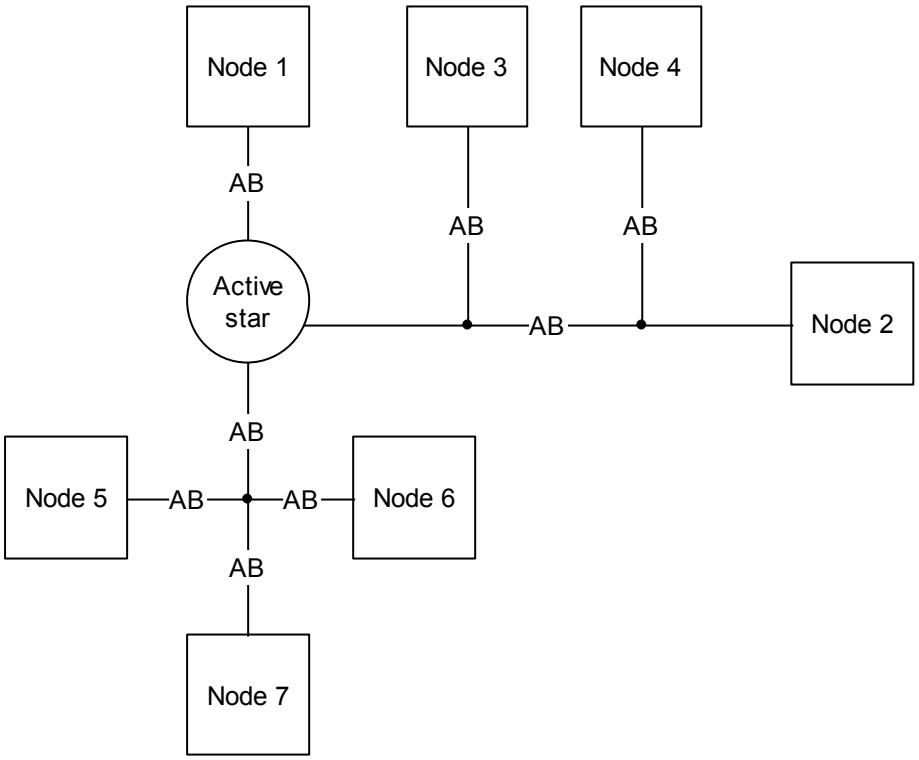


Obrázek 5 - Topologie pasivní sběrnice

Ještě existuje topologie tzv. aktivní nebo pasivní hvězda. U topologie pasivní hvězda se jedná téměř o topologii pasivní sběrnice, kde nejsou společné části vodičů, ale všechny vodiče ode všech zařízení jsou spojeny v jednom uzlu. Aktivní hvězda je odlišná v tom, že každý uzel je připojený k centrálnímu aktivnímu uzlu. Zde mohou být propojeny všechny uzly pomocí obou, nebo jen jednoho kanálů a vytvořit jednu aktivní hvězdu, nebo mohou být zapojeny každým kanálem do dvou různých aktivních centrálních uzlů a vytvořit tak dvě nezávislé hvězdicové topologie viz Obrázek 6 - Topologie aktivní hvězda. Z této topologie vznikla tzv. hybridní topologie, která je založená na složení předešlých topologií viz Obrázek 7 - Hybridní topologie.



Obrázek 6 - Topologie aktivní hvězda



Obrázek 7 - Hybridní topologie

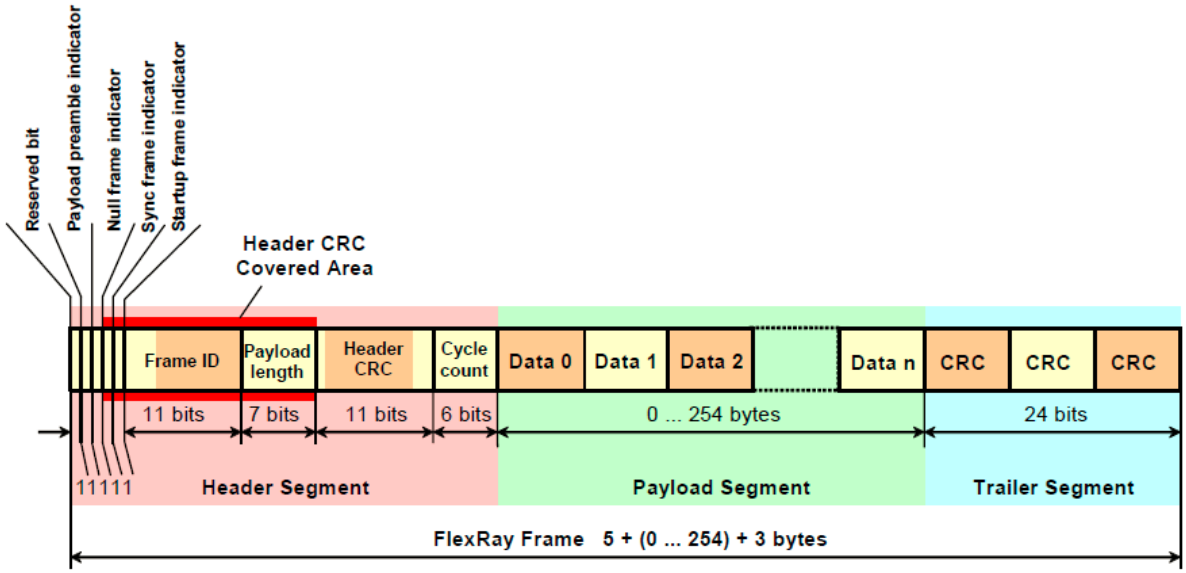
2.3.5 Formát dat

Samotná data přenášená pomocí sběrnice FlexRay jsou přenášena v tzv. rámcích. Na sběrnici FlexRay je normou [5] a [6] definovaný rámec, který se skládá ze tří segmentů.

V prvním segmentu datového rámce je hlavička přenášených dat, která začíná pěti jednotlivými bity s názvy Reserved bit, který zatím není využíván, ale je definován pro budoucí využití. Payload preamble indicator značí přítomnost vektoru pro správu sítě v případě statického rámce a v případě dynamického rámce indikuje přítomnost ID zprávy. Null frame indicator v případě že je nulový značí, že rámec nenese platná data a v případě že je roven logické 1 tak rámec obsahuje platná data. Sync frame indicator v logické 1 značí, že daný rámec je synchronizační pro synchronizaci hodin zařízení. Poslední bit je Startup frame indicator značící že daný rámec je spouštěcí. Následuje 11 bitů představující ID přenášeného rámce, 7 bitů délky přenášených dat, kdy toto číslo po vynásobení 2 představuje počet přenášených bytů daným rámcem. Předposlední v hlavičce je CRC součet hlavičky, který má délku 11 bitů a posledních 6 bitů je vyhrazeno pro čítač, který čítá vždy se začátkem přenosu.

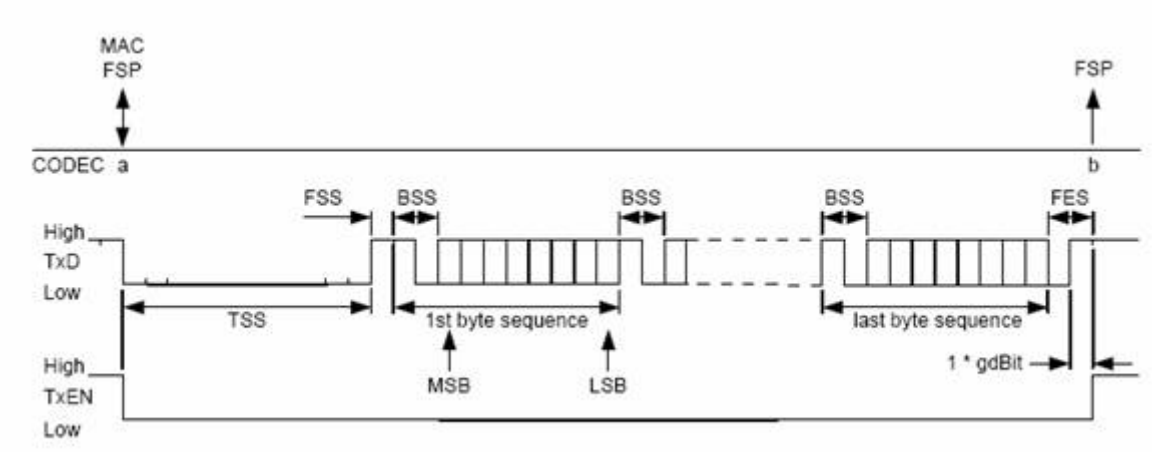
Druhou částí přenášeného datového rámce jsou samotná data. Tato datová část má proměnnou délku 2 až 254 byte v závislosti na údaji Payload lenght, tj 7-mi bitech z hlavičky.

Třetí a poslední částí datového rámce je CRC přenášených dat. Toto CRC má délku 24bitů.



Obrázek 8 - Datový rámec FlexRay

Mezi přenášenými informacemi v rámci jednoho rámce se přenáší tzv. řídící bity, respektive sekvence řídících bitů. Prvním sekvencí bitů je TSS, která je dlouhá 5 až 15 bitů, které jsou všechny v logické úrovni 0 a slouží k začátku komunikace, respektive správnému nastavení zařízení na obou koncích sběrnice při přechodu z klidového stavu. Ihned za sekvencí TSS je druhá kratší, tentokrát obsahující pouze jeden bit opačné úrovně s názvem FSS. FSS bit začíná datový rámec a druhou funkcí je kompenzace kvantizační chyby následující sekvence BSS po TSS. Třetí sekvencí je tzv BSS sekvence, která se skládá ze dvou bitů. Prvním bitem je bit logické hodnoty 1 následovaný logickou 0. Sekvence BSS se v komunikaci vkládá vždy po osmi přenesených bitech bez rozdílu, jestli vycházejí do nějakého většího logického celku, např. Frame ID nebo Header CRC. BSS sekvence má důležitou funkci v synchronizaci vysílajícího a přijímajícího hosta na obou stranách sběrnice. Poslední sekvencí je sekvence označená FES a obsahuje opět dva bity, které jsou oproti BSS negované. FES se vkládá vždy na konec celého rámce, který tímto ukončí a uvolní tak sběrnici pro další komunikaci.



Obrázek 9 - Řídící bity na sběrnici FlexRay

CRC

Velkým rozdílem mezi rámci pro CAN a Flexray sběrnici je, že u FlexRay sběrnice datový rámec neobsahuje pouze jeden kontrolní součet, ale dva. První kontrolní součet se počítá z bitů Sync frame indicator, Startup frame indicator, ID rámce a velikosti přenášených dat. K výpočtu CRC se využívá polynom CRC-11 skládající se z viz (4). U kontrolního součtu v hlavičce datového rámce se používá k výpočtu ještě tzv. inicializační vektor, který má hodnotu 0x1A. Tímto inicializačním vektorem se před výpočtem nastaví posuvný registr sloužící k výpočtu CRC.

$$x^{11} + x^9 + x^8 + x^7 + x^2 + 1 \tag{4}$$

Druhé CRC se počítá nejen ze samotných přenášených dat v druhé části datového rámce, ale je v něm zahrnuta celá hlavička datového rámce včetně CRC obsaženého v hlavičce. Zde se používá polynom CRC-24 viz (5). Opět jako u kontrolního součtu v hlavičce rámce se používá inicializační vektor. Je zde rozdíl kromě jeho délky také v tom, že pro každý z kanálu A nebo B se používá inicializační vektor s jinou hodnotou. U kanálu A je to hodnota 0xFEDCBA a u kanálu B je to hodnota 0xABCDEF. To slouží k tomu, aby byly vždy spojeny kanály A<->A a B<->B. V případě jejich prohození, A<->B nebo B<->A, by komunikace mezi zařízeními nefungovala.

$$x^{24} + x^{22} + x^{20} + x^{19} + x^{18} + x^{16} + x^{14} + x^{13} + x^{11} + \\ + x^{10} + x^8 + x^7 + x^6 + x^3 + x + 1 \tag{5}$$

2.4 Podporované přístroje

Tato kapitola popisuje použitá zařízení a jejich možnosti s ohledem na zadání práce. Jedná se především o níže uvedené řady osciloskopů různých výrobců. Osciloskopy se používají ve výuce a jsou k dispozici na naší fakultě.

2.4.1 Agilent DSO9254A

Osciloskop Agilent řady 9000, konkrétně typ DSO9254A s maximální možnou velikostí paměti, byl zakoupen z projektu ESF CZ.1.07/2.2.00/28.0050 Modernizace didaktických metod a inovace výuky technických předmětů.

0Osciloskop Agilent DSO9254A [7] byl vyvinut pro nejnáročnější použití. Proto také disponuje čtyřmi kanály s šířkou pásma 2,5 GHz, vzorkovací frekvencí 20 GSa/s při použití dvou a 10 GSa/s při použití všech čtyř kanálů. Osazen je osmibitovým A/D převodníkem. Zároveň má velkou vzorkovací paměť 1Gpts pro každý kanál, díky čemuž je možné vzorkovat i časově velice dlouhé signály, respektive při našem použití dlouhou dobu komunikace. Nevýhodou u takto pokročilého a drahého osciloskopu je nutnost dokoupit další software, který by umožňoval dekódování komunikačních kanálů. Podrobněji jsou vlastnosti popsané v bakalářské práci [8].

Z tohoto osciloskopu je možné data vyčítat dvěma základními možnostmi. První možností je vyčítat data po ethernetové síti nebo USB, pomocí vzdáleného ovládání osciloskopu. Druhou možností je data vyexportovat do datového souboru a na výběr je ze tří formátů. Prvním je do binárního souboru s příponou bin, druhý je standardní CSV/TSV, kdy oddělovačem je čárka v případě CSV a tabelátor v případě TSV. Posledním formátem je HDF5. Poslední možností je generovat přímo obrázky.

Pro tuto práci bylo zvoleno zpracování datových souborů, nicméně je samozřejmě možné vyčítat data z osciloskopů i dálkově. Nejzajímavějším formátem uložených dat je HDF5 soubor, který najde uplatnění v složitějších systémech a ve kterém jsou data uvnitř souboru strukturována obdobně jako soubory na pevném disku počítače. Přesněji je v daném souboru uspořádáno několik objektů, díky nimž je možné tento formát použít pro různé

typy uložených dat od naměřených hodnot, přes obrázky až po rozsáhlé projekty. Uvnitř souboru jsou definovány objekty *Groups*, *Dataset*, *Dataspace* a *Datatype*.

Binární soubor uložených dat s příponou bin je spolu s CSV/TSV nejuniverzálnějším možným způsobem uložení dat a každý se primárně hodí k jinému zpracování. Binární soubor je vhodný především pro strojové zpracování, přičemž CSV/TSV je vhodné nejen pro strojové zpracování, ale i pro ruční zpracování.

Struktura binárního souboru je následující. Na začátku celého dokumentu se nachází krátká hlavička celého binárního dokumentu, ve kterém je uložena informace o tom, že pochází z osciloskopu Agilent, počtu naměřených kanálů, velikosti souboru v bytech a také verze souboru. Za touto hlavičkou již následují naměřená data z jednotlivých kanálů, kdy jsou vždy uložena data z jednoho měřeného kanálu následovaná dalšími kanály. Datům z každého měřeného kanálu předchází ještě dvě hlavičky, které jsou uložené hned za sebou, a tak se dají považovat za jednu větší hlavičku. V této hlavičce jsou uložené informace o datu a času měření, modelu osciloskopu, sériovém číslu, nastavení osciloskopu, počtu vzorků, datovém typu, ve kterém jsou data uložena apod. [9]. Ilustrační tabulka jak vypadá binární datový soubor, viz Tabulka 5 - Uspořádání dat v binárním souboru.

Tabulka 5 - Uspořádání dat v binárním souboru

File Header (Hlavička souboru) 8 bytů
Waveform Header 1 (Hlavička 1. signálu) 136 bytů
Waveform Data Header 1 (Datová hlavička 1. signálu) 12 bytů
Data N_1 bytů
Waveform Header 2 (Hlavička 2. signálu) obsahuje informace o 136 bytů
Waveform Data Header 2 (Datová hlavička 2. signálu) 12 bytů
Data N_2 bytů
Waveform Header n (Hlavička n-tého. signálu) 136 bytů
Waveform Data Header n (Datová hlavička n-tého. signálu) 12 bytů
Data N_n bytů

U CSV/TSV souboru je struktura dat jednoznačná a daná konvencí vytváření tohoto typu souborů. V tomto typu soboru jsou data uložena vždy v textové podobě oddělena čárkou, středníkem, nebo tabelátorem. Struktura dat je také jednoznačně daná, kdy na počátku souboru je v několika řádcích uvedena hlavička, ve které jsou obsažené informace o počtu měřených kanálů, počtu naměřených vzorků, nastavení osciloskopu, datu a času apod.

2.4.2 GW Instek GDS 2072A

Osciloskop GW Instek GDS 2072A [10] má oproti osciloskopu Agilent DSO9254A nenáročné parametry, neboť fakticky spadá do nižší přístrojové kategorie, která je nicméně pro většinu laboratorních cvičení a měření v předmětech katedry dostačující. Disponuje pouze dvěma kanály s šířkou pásma 70MHz, vzorkovací frekvencí 2GSa/s a osmibitovým A/D převodníkem. Velikým omezení pro měření delších signálů, nebo komunikace je jeho relativně malá paměť, do které se vejde maximálně 2 miliony vzorků pro jeden kanál a po 1 milionu vzorků pro dva kanály.

Oproti modelu od firmy Agilent disponuje šestnácti kanálovým logickým analyzátorem, díky němuž je po doinstalování příslušných softwarových komponent, analyzovat základní komunikační kanály, mezi kterými nechybí ani sběrnice CAN. Sběrnice FlexRay již není u tohoto typu osciloskopu podporována.

2.4.3 Lecroy Wavesurfer 400

Poslední zvoleným osciloskopem je Lecroy Wavesurfer 434 z řady Wavesurfer 400. Tento osciloskop byl vybrán z důvodu, že v době pořízení patřil k nejlepším vyráběným osciloskopům. Dnes je do této práce vybrán nejen pro své parametry, ale také pro porovnání s dnešními top osciloskopy.

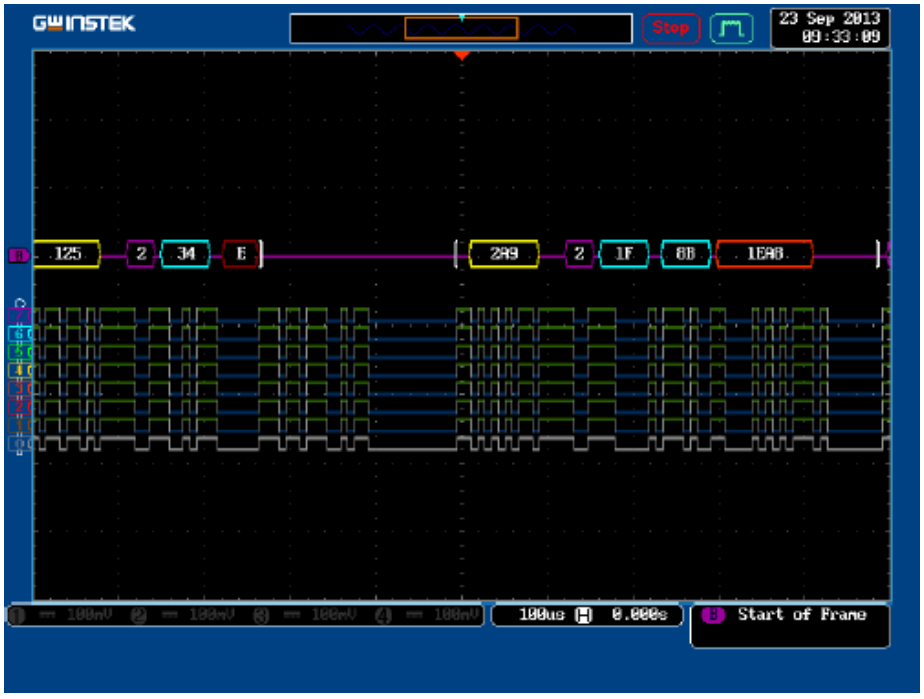
Lecroy Wavesurfer 434 je původně osciloskop ze stejné kategorie jako Agilent DSO9254A, avšak parametry dnes již nejlepším osciloskopům nekonkuruje z důvodu svého stáří a relativně rychlého vývoje v oblasti měřící techniky. Zvolený osciloskop disponuje čtyřmi nezávislými měřícími kanály, s maximální šířkou pásma 350MHz a vzorkovací frekvencí 1GSa/s. V případě periodických signálů se správným nastavením triggeru je možné využít tzv. RIS mode, kdy vzorkovací frekvence dosahuje hodnoty až 50GSa/s. Stejně jako předešlé modely osciloskopů od firem Agilent a GW Instek je osazen osmibitovým A/D převodníkem, který je pro běžné měření zcela dostačující. Nevýhodou tohoto osciloskopu pro měření komunikace na jakékoliv sběrnici je jeho relativně malá paměť, do které je možné navzorkovat maximálně 1ms signálu při 1GSa/s, což odpovídá 1Mpts pro jeden kanál.

Naměřená data lze uložit do několika formátů souboru. Důležitými formáty, do kterých lze data uložit jsou v daném osciloskopu nazvány Excel (csv), Matlab (dat), ASCII (txt) a Mathcad (prn). Ve všech těchto formátech jsou data uložena v textové podobě. Jedinou odlišností mezi těmito soubory je formát uspořádání těchto dat ve výsledném souboru. Ve formátu dat není přítomna hlavička naměřených dat a soubor tak obsahuje pouze naměřená data, kdy na každém řádku je vždy čas v sekundách a mezerou oddělené měřené vzorky ve voltech. V souborech csv a txt není žádný rozdíl kromě přípony souboru. V obou případech je přítomna hlavička souboru, ve které jsou informace o osciloskopu, typu měřených dat, množství vzorků, datum, čas a popis jak jsou naměřená data uložená, tj. že první je čas v sekundách a čárkou oddělené jsou naměřené vzorky v daný čas ve voltech. Posledním zmiňovaným formátem je prn, který se od txt a csv liší pouze ve hlavičce, kdy nejsou přítomné popisky jednotlivých informací, ale pouze jejich hodnoty a neobsahuje informaci o datu a času měření. Samotná naměřená data jsou uložena stejným způsobem jako u datového souboru s příponou dat., tj. čas v sekundách a naměřená data ve voltech oddělená mezerou.

2.4.4 Možnosti dekódování dat

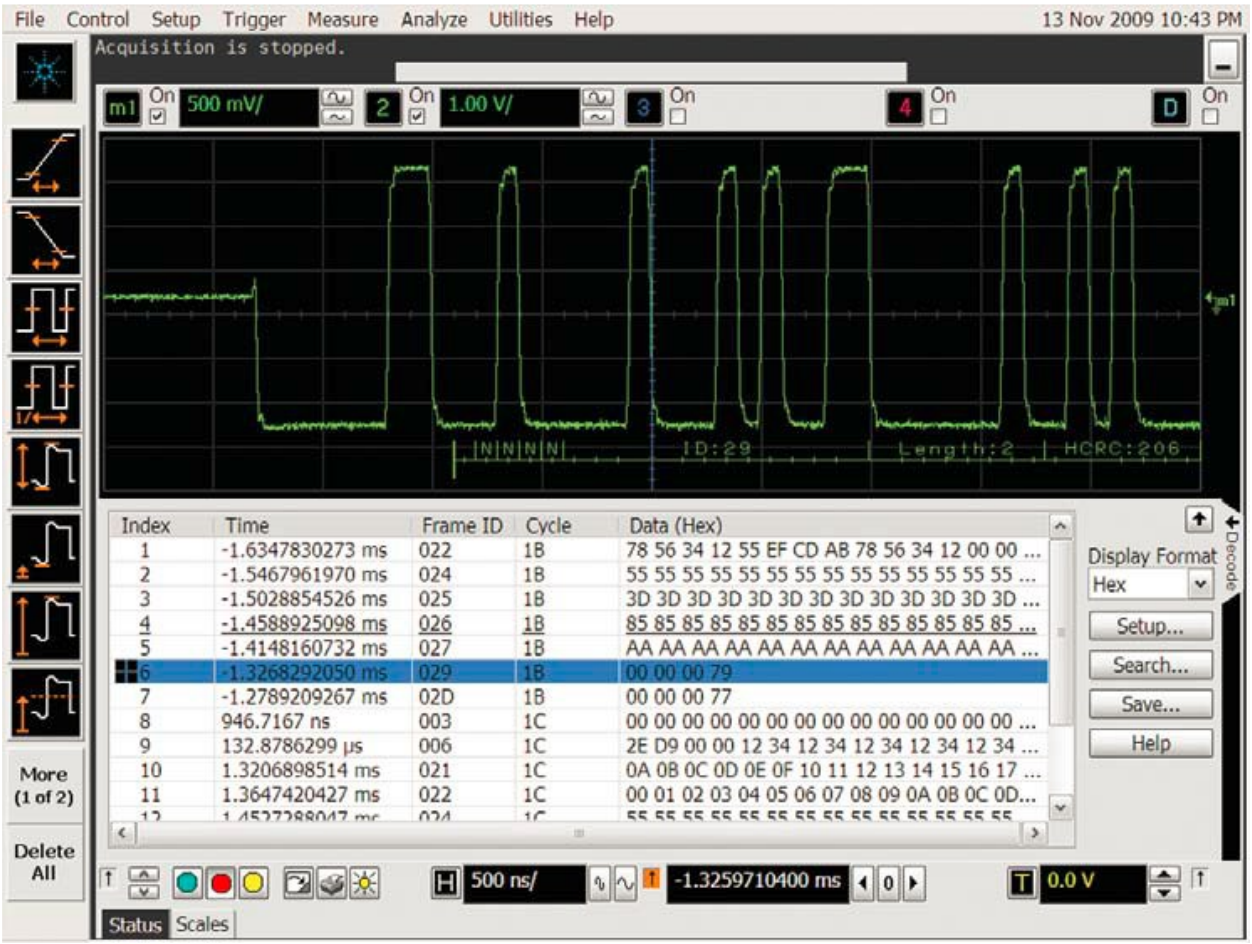
Z dostupných vybraných osciloskopů je možné přidat podporu dekódování dat na sběrnici CAN pouze u osciloskopů GW Instek GDS 2072A a Agilent DSO9254A. Pro sběrnici FlexRay je situace ještě horší, kdy lze podporu dekódování přidat pouze do osciloskopu Agilent DSO9254A pomocí softwarového rozšíření N8803A.

U obou výrobců osciloskopů je pouze základní detekce chyb a absence měření jakýchkoli parametrů navíc. Firma GW Instek [11] zapíše pouze, že na sběrnici je detekována chyba, ale nabízí například *trigger* na každou část paketu, nebo na chybějící ACK, případně chybějící *bit stuffing*.



Obrázek 10 - Dekódování CAN GW Instek [11]

Firma Agilent na chybu pouze upozorní ve svém výpisu, např. na neočekávaný konec paketu. Stejně jako GW Instek nabízí Agilent *trigger* na každou část paketu. Nevýhodou může být manuální nastavování parametrů pro dekódování u obou typů sběrnic například přenosová rychlost nebo ke kterému kanálu je připojen který typ vodiče. Výhodou ovšem je že zbytek nastavení osciloskopu jako například vzorkovací frekvence vertikální rozlišení apod. si osciloskop nastaví sám.



Obrázek 11 - Dekódování FlexRay Agilent DSO9254A [12]

2.5 Vývojový kit Freescale S12XF

Vývojový kit Freescale S12XF, přesněji tedy EVB9S12XF512E je osazen mikroprocesorem S12XF512MLM. Tento Vývojový kit byl vybrán jako ideální pro tuto diplomovou práci, jelikož podporuje jak sběrnici FlexRay tak i sběrnici CAN.

Použitý mikroprocesor je z řady Freescale S12X, která je vyvinuta z původní řady MC9S12. Tato nová řada má 16bitové jádro, nižší spotřebu a vyšší výkon. Konktrétně díky vysokému výkonu a podpoře sběrnic FlexRay a CAN je předurčen pro rychlou a spolehlivou komunikace, která je využitelná především v kritických aplikacích. Mezi vhodné kritické aplikace například v automobilovém průmyslu patří airbagy, brzdové ústrojí vozidla, bezpečností asistenti a další, kde je vhodnou náhradou za stávající řešení v podobě sběrnice CAN. Výkon tohoto mikroprocesoru je ještě vyšší díky integrování XGATE modulu. XGATE je periferní koprocesor umožňující autonomní přenos dat mezi jádrem a periferiemi, Dále umožňuje předzpracovávat přenášená data a provádět složité komunikační protokoly. Díky tomuto vylepšení je tak snížena náročnost na jádro mikroprocesoru a také na jeho systém přerušení a díky tomu je výkonnější až 5krát.

Tento mikroprocesor ke komunikaci na sběrnici FlexRay potřebuje ještě PHY, proto je na použitém vývojovém kitu pro každý kanál ještě integrovaný obvod TJA1080.

Základní specifikace

- 16bit architektura
- 512kB Flash
- Frekvence 50MHz
- Napájecí napětí 3,3V až 5,5V
- 16-ti kanálový 12-bit ADC
- 6-ti kanálový 15-bit ADC
- Pouzdro QFP 112

3 Vyvinutá aplikace

3.1 Výběr programovacího jazyka a vývojového prostředí

Nyní jsem měl výběr programovacího jazyka a vývojového prostředí usnadněný, jelikož jsem již dříve programoval v programovacích jazycích založených na jazku C. Z tohoto důvodu jsem si pro vývoj výsledné aplikace na dekódování dat vybral programovací jazyk C, který je dle mého názoru na zpracování velkého množství dat nejlepší. Jelikož se jedná o nejstarší programovací jazyk z této rodiny programovacích jazyků, je také nejjednodušší a v tom spočívá jeho největší výhoda, protože díky tomu je přeložený program velice rychlý v porovnání s ostatními programovacími jazyky. Na podpůrný program jsem si vybral programovací jazyk C#, ve kterém se snadno vytvoří GUI aplikace a také není u tohoto podpůrného programu kladen největší požadavek na rychlost.

Vývojové prostředí jsem měl vybrané také velice rychle, jelikož jsem si zvykl na vývojové prostředí Visual Studio 2010 Professional. Druhým důvodem je fakt, že jsem v něm mohl programovat všechny potřebné aplikace pro PC, a také, že jsem v něm programoval své předešlé práce. Vývojovým prostředím pro firmware jsem zvolil Freescale CodeWarrior, které bylo dodáno spolu s vývojovými kity.

3.2 GUI

Uživatelské rozhraní bylo zvoleno v podobě klasického příkazového řádku (konzole), protože výsledná aplikace je vyvinuta pro existující systém inteligentního měřícího pracoviště se systémem dálkové správy a zpracování dat. Dalším důvodem je její jednoduché ovládání, které nepotřebuje žádná složitá nastavení, jelikož je celý návrh aplikace cílen pokud možno na co nejautomatizovanější zpracování dat, a proto grafické uživatelské rozhraní není nutné.

Uživatelský vstup byl vyřešen s ohledem na nasazení aplikace do systému inteligentního měřícího pracoviště a proto je možno uživatelsky zasáhnout pouze pomocí parametrů při spuštění aplikace. Při tomto řešení aplikaci postačí pouze takto získané informace od uživatele a další uživatelský zásah tak není nutný po dobu zpracování dat. Jediným povinným vstupem je název vstupního souboru s naměřenými daty včetně kompletní cesty k danému souboru. Více o problematice zadávání vstupních informací v kapitole 0.

Rovněž je v aplikace navržena pro práci v tzv. ladícím módu, kdy je nutné ji znovu zkompilovat s příslušným nastavením. Nastavení ladícího módu je možné pomocí syntaxe *#define*, kdy je definuje pouze název *DEBUG*. Při takto nastavené a zkompilované aplikaci s ladícím módem se během chodu vypisují potřebné informace přímo do konzole, a ne jen a pouze výsledná data do souboru.

Stejným způsobem lze u aplikace měnit jazyk, ve kterém bude generovat výstupní soubory. Na výběr je z češtiny, kdy se definuje název *LANG_CZ* a angličtiny, kdy se generuje *LANG_EN*. Tento způsob výběru jazyka byl vybrán s ohledem na rychlost běhu aplikace, protože řešit při každém výpisu během programu podmínkou s proměnnou by znamenalo zpomalení aplikace, protože míst, kdy se vypisují data je relativně rodně.

3.3 Program

Z požadavků na výsledný program, které jsou uvedené v zadání, dále v úvodu, v kapitole 132.1, a také z předchozí zkušenosti jsem vymyslel, respektive upravil původní základní koncept celé aplikace. Ponechal jsem myšlenku na rozdělení celé problematiky do více dílčích částí, které na sebe v logické souvislosti navazují a tyto logické celky implementovat pomocí jednotlivých funkcí. Obdobně jako v mé bakalářské práci [8], nebo projektu [13]. Prvním cílem bylo tedy oddělit základní části na jednotlivé bloky, tedy uživatelský vstup, data z osciloskopu, zpracování a uložení dat.

Toto rozdělení bylo potřeba navrhnout jinak než v případě bakalářské práce či projektu, protože v obou případech se vždy jednalo pouze o jednu sběrnici a proto se v původních návrzích objevovaly nedokonalosti, které způsobily, že dané řešení nebylo zcela univerzální.

Proto v této práci je přístup a celé řešení navržené znovu a univerzálně, ačkoli hlavní myšlenka s drobnými úpravami je totožná. Lepší univerzálnost je ověřena, jelikož se zde kombinují dvě sběrnice a tři modelové řady osciloskopů.

V prvním velkém bloku uživatelského vstupu došlo k jeho celému přepracování, kdy se původní řešení ukázalo nevyhovující pro přehlednost z pohledu uživatele a zajištění naprosté volnosti uživatelsky měnit parametry, což by v původním řešení znamenalo velké úpravy a velice dlouhý program k jeho zajištění. Nové řešení je založeno na parametrizaci programu, tak jak je známé z příkazové řádky více v kapitole 3.3.2.

Druhý blok by se dal nazvat rozhraním mezi naměřenými daty z osciloskopu a používanou reprezentací těchto dat v popisované aplikaci. Díky této myšlence, a jejímu novému implementování, ve kterém se opravily nedostatky předchozích řešení, je nyní možné naprosto libovolně přidávat další modely a modelové řady osciloskopů bez dalšího zásahu do zbylé aplikace. K tomuto slouží jednotlivé funkce zajišťující načítání dat ze souborů pro příslušné řady osciloskopů popsané v kapitole 3.3.2.

Samotné zpracování načtených dat pro obě sběrnice zajišťuje řada funkcí popsaných v kapitole 3.3.2, kde jsou popsané i funkce pro uložení výsledků. Ukládání výsledků, ať se jedná o dekódovaná data, nebo o chybové vzorky se prování během dekódování tak, že se uloží vždy jeden paket po jeho dekódování, nebo jeden chybný vzorek po jeho detekování. Tímto přístupem se tak dosáhne úspory operační paměti, kdy se nemusejí držet všechny dekódované pakety, nebo detekované chybové vzorky, v paměti až do konce a poté je uložit. Zároveň s tím se zajistí, že uživatel v případě náhlé chyby, která by se teoreticky mohla objevit přímo v aplikaci, v naměřených datech, nebo na straně operačního systému tak nepřijde o již dekódovaná data a detekované chyby.

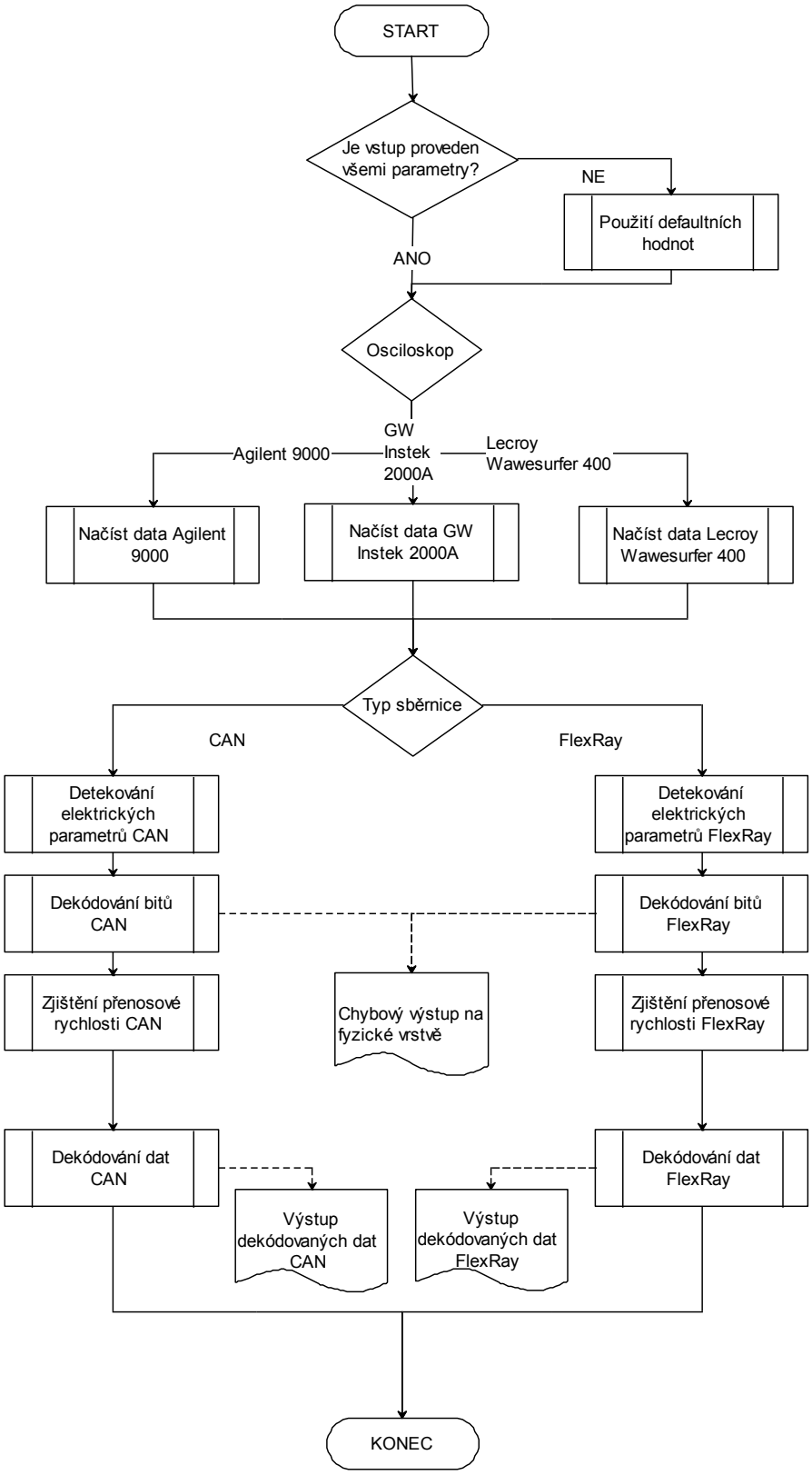
3.3.1 Struktura programu

Kapitola 3.3 popisuje základní princip výsledné aplikace, která je rozdělená na tři základní části. Názorně zobrazené uspořádání a funkce aplikace je na Obrázek 12 – Diagram programu.

Po spuštění aplikace se provede načtení všech uživatelských parametrů. Pokud není některý z parametrů zadán, je použita jeho výchozí hodnota. Toto platí pro všechny parametry, kromě vstupního souboru. To odpovídá první části dle kapitoly 3.3.

Následně je podle vstupního souboru vybrána správná funkce, respektive správný osciloskop a jeho funkce pro načtení dat. Načtou se data a podle uživatelského vstupu se provede výběr sady funkcí pro zpracování zvolené sběrnice. V případě obou možných sběrnic se postupuje podle stejného postupu.

Ve třetí části dle kapitoly 3.3 se nejprve detekují elektrické parametry sběrnice, například rozsahy napětí, ve kterých se pohybují jednotlivé vzorkované kanály, jakým kanálem osciloskopu byl vzorkován který vodič sběrnice, maximální hodnota šumu, nebo RMS hodnota šumu. Následuje krok, ve kterém se data převedou na jednotlivé stavy sběrnice, respektive na logické hodnoty jim odpovídající a zároveň s tím se testují napěťové rozsahy obou měřených kanálů i jejich rozdílu. V případě chyby se tyto stavy zapisují do výstupního souboru s chybovými stavy na fyzické vrstvě. V předposledním kroku se zjistí přenosová rychlost používaná na sběrnici z naměřených dat a zároveň se porovná se standardizovanými rychlostmi. Posledním krokem je samotné dekódování posílaných dat včetně chybových stavů s průběžným ukládáním do souboru.



Obrázek 12 – Diagram programu

3.3.2 Ovládání aplikace

Jak již bylo zmíněno v předchozích kapitolách, jediným možným uživatelským vstupem jsou parametry, se kterými se aplikace spouští. K přehlednému a komfortnímu zadávání a zpracování vstupních parametrů slouží funkce *getopt*, která je známá především z linuxu. K implementování této funkce je použitá stejnojmenná knihovna *getopt.h*.

Funkce *getopt* vyhledává ve vstupním poli funkce *main*, ve kterém se nachází všechny uživatelem zadané parametry, předem definované deskriptory podle jejich nastavení. V praxi tak vyhledává spojení pomlčky a písmena, za kterým buď musí, nebo nemusí být zadaná hodnota parametru. V případě že musí být zadaná hodnota parametru, funkce vrátí v jedné proměnné písmennou hodnotu parametru a pomocí ukazatele na pole znaků vrátí zadanou hodnotu. V případě, že za parametrem být hodnota nemusí, vrátí pouze znak parametru.

Pro zpřehlednění a zjednodušení pro uživatele byla zvolena upravená varianta funkce *getopt* pro dlouhé názvy, která se jmenuje *getopt_long*. Rozdíl mezi těmito funkcemi je, že *getopt_long* má o dva vstupní parametry více, ze kterých je pro dané použití důležitý jen jeden, ve kterém se předává struktura s informacemi o dlouhém názvu, typu parametru tj. jestli je hodnota povinná nebo volitelná. Dále ještě pointer na proměnnou, která se má nastavit a nakonec hodnota, kterou pro daný parametr bude funkce vracet. Při použití dlouhých názvů je nutné místo jedná pomlčky použít pomlčky dvě.

Veškeré vstupní parametry týkající se napětí znamenají meze napětí, pokud daný signál meze překročí, bude to považované za chybový stav, který bude zaznamenán ve výstupním souboru s chybovými vzorky.

Popis všech možných parametrů, jejich dlouhých i krátkých názvů a výchozích hodnot je uveden v několika tabulkách. Obecné parametry jsou v Tabulka 6 - Obecné parametry aplikace, parametry týkající se pouze sběrnice CAN jsou v Tabulka 7 - Parametry aplikace - CAN a parametry týkající se pouze sběrnice FlexRay jsou v Tabulka 8 - Parametry aplikace - FlexRay.

Tabulka 6 - Obecné parametry aplikace

Popis parametru	Dlouhý název	Krátký název	Výchozí hodnota
Název vstupního souboru včetně cesty a přípony	--in	-i	-
Název druhého vstupního souboru včetně cesty a přípony, pouze Lecroy	--in2	-w	-
Název výstupního souboru včetně cesty a přípony	--out	-o	-
Název výstupního souboru s chybovými stavy včetně cesty a přípony	--outerr	-a	-
Název výstupního souboru s chybovými pakety včetně cesty a přípony	--outfail	-b	-
Typ dekódované sběrnice	--type --bus	-t	F

Tabulka 7 - Parametry aplikace - CAN

Popis parametru	Dlouhý název	Krátký název	Výchozí hodnota
Minimální napětí signálu CAN-H	--canhmin	-l	0,1V
Maximální napětí signálu CAN-H	--canhmax	-m	3,9V
Minimální napětí signálu CAN-L	--canlmin	-n	0,1V
Maximální napětí signálu CAN-L	--canlmax	-p	3,9V
Minimální napětí dominant	--dommin	-q	2V
Maximální napětí dominant	--dommax	-r	3,9V
Minimální napětí recessive	--recmin	-u	-0,1V
Maximální napětí recessive	--recmax	-v	0,5V

Tabulka 8 - Parametry aplikace - FlexRay

Popis parametru	Dlouhý název	Krátký název	Výchozí hodnota
Minimální napětí signálu BP	--bpmin	-c	-0,05V
Maximální napětí signálu BP	--bpmax	-d	3,6V
Minimální napětí signálu BM	--bmmin	-e	-0,05V
Maximální napětí signálu BM	--bmmax	-f	3,6V
Minimální napětí uBus - log.1	--1min	-g	0,6V
Maximální napětí uBus - log.1	--1max	-h	2V
Minimální napětí uBus - log.0	--0min	-j	-2V
Maximální napětí uBus - log.0	--0max	-k	-0,6V

3.3.3 Funkce

Main

Funkce *main* obstarává veškerý chod programu za pomocí jednotlivých funkcí, z nichž ty nejdůležitější jsou níže popsané. Ve funkci *main* se provede prvotní zpracování vstupních parametrů, na jejichž základě se provede správný výběr osciloskopu, kterým byla vstupní data vzorkovaná. Dále se provede inicializace všech potřebných globálních proměnných, otevření všech datových proudů ke vstupnímu i všem výstupním souborům. V neposlední řadě se začne provádět samotné zpracování dat, respektive volání jednotlivých funkcí. Při této činnosti se samozřejmě v souladu s požadavky na aplikaci využívají systémové prostředky optimálně primárně s ohledem na časovou náročnost a obsazení operační paměti.

Načtení dat ze souboru

Načítání dat ze souboru zajišťují celkově tři nezávislé funkce. K rozdělení tohoto problému, na první pohled triviálního na tři nezávislé funkce bylo přistoupeno s ohledem na jejich jednoduchost a zároveň kompatibilitu s ostatními funkcemi. Každá z těchto tří funkcí tak zastupuje jeden model, respektive modelovou řadu osciloskopů a také jeden typ vstupního souboru. Každá funkce tak načte a upraví vstupní data do jednotného formátu využitelného pro další zpracování. Jedná se tedy především o úpravu textových formátů naměřených dat jako je například *.txt, *.csv a jejich uložení do dvou polí datového typu float, ve kterých jsou uložené jednotlivé hodnoty napětí vzorkovaných v čase. Zároveň tyto funkce načtou dostupné informace o osciloskopu, datu a čase vzorkování, počtu vzorků načtených z naměřeného souboru. V neposlední řadě přečtou nebo dopočítají vzorkovací frekvenci.

```
int OpenAgilent9000(FILE *input, float **data1_ptr,
float **data2_ptr, int *samples, HeaderSignal *headerSignal)
```

Vstupem této funkce je otevřený datový proud na vstupní binární soubor s naměřenými daty z osciloskopu Agilent 9000, následují dva pointery na pointer, které ukazují na pole, do kterých se naměřená data, respektive hodnoty napětí jednotlivých vzorků uloží. Předposlední vstupní proměnnou je pointer na proměnnou *samples*, do které se uloží počet navzorkovaných respektive načtených hodnot. Posledním vstupem je pointer na strukturu hlavičky signálu, která obsahuje ostatní informace, které byli zmíněné v předchozím odstavci.

Ve funkci *OpenAgilent9000* se nejprve přečte hlavička celého vstupního souboru, viz Tabulka 5 - Uspořádání dat v binárním souboru, která musí obsahovat identifikátor výrobce osciloskopu. Zde se jedná o první dva znaky „AG“, následované verzí souboru, velikostí souboru a počtem měřených kanálů.

Po zpracování této první hlavičky souboru a vyhodnocení počtu kanálů se v případě splnění podmínky, že soubor obsahuje data pouze ze dvou kanálů osciloskopu, začne se s načítání dat. V opačném případě skončí funkce chybovou návratovou hodnotou. Další dvě hlavičky, které jsou přítomné u každé-ho měřeného kanálu, se přečtou vždy současně a zpracují se. Po přečtení hlavičky se alokuje příslušná část paměti pro načtení naměřených dat a ta se poté načtou a uloží se do pole. Více se data v této funkci již nezpracovávají, což má za následek univerzálnost, ačkoliv je tento způsob náročnější na operační paměť počítače.

```
int Read2ChannelGWInstekCSV(FILE *input, float **data_1ptr,
float **data_2ptr, int *samples, HeaderSignal *headerSignal)
```

Funkce *Read2ChannelGWInstekCSV* zpracovává vstupní soubory ve formátu CSV z osciloskopů od firmy GW Instek modelové řady 2000A. Tato funkce má zcela stejné vstupní proměnné, jako předešlá funkce *OpenAgilent9000*. Hlavním rozdíl, kromě typu zpracovávaného souboru a výrobce osciloskopu, je ve způsobu načítání dat. U funkce *OpenAgilent9000* se data načítala binárně, zde jsou data uložena v textové podobě. Načtená data se tedy ještě musí navíc převést z textové podoby na číslo v desetinném formátu.

Průběh samotného načítání je obdobný. První se přečte hlavička, ze které se vyberou důležité informace například o vzorkovací periodě, datu, času, počtu naměřených vzorků

apod. Až následně se začnou postupně načítat samotná naměřená data a převádět do dále zpracovatelné podoby.

```
int Read2ChannelLecroyTXT(FILE *input, FILE *input2,
float **data_1ptr, float **data_2ptr, int *samples,
HeaderSignal *headerSignal)
```

Funkce *Read2ChannelLecroyTXT* zpracovává vstupní soubory ve formátu TXT z osciloskopů od firmy Lecroy, konkrétně modelové řady Wavesurfer 400. Na první pohled je vidět, že parametry této funkce jsou odlišné od předešlých dvou funkcí *OpenAgilent9000* a *Read2ChannelGWInstekCSV*. To je způsobeno tím, jak je možné data z tohoto osciloskopu uložit. Tento osciloskop nedokáže uložit navzorkovaná data z více kanálů do jednoho souboru, ale každý kanál se ukládá zvlášť.

Rozdílem oproti předchozím funkcím je o jednu vstupní proměnnou více, která představuje otevřený datový proud na druhý soubor. Stejně jako v předešlém případě u funkce *Read2ChannelGWInstekCSV* se zde zpracovávají data v textové podobě a je tedy nutné je po načtení ze souboru převést na číslo v desetinném formátu.

Průběh načítání dat ze souboru je trochu odlišný od předchozích případů, ačkoliv hlavní postup je stejný. Nejprve se přečte hlavička z prvního souboru a vyčtou se data o modelu osciloskopu, počtu navzorkovaných dat, datu a času. Zde chybí informace o vzorkovací periodě, která se vypočítá z prvních dvou vzorků, respektive z jejich časových značek. Následně se postupně načítá hlavička druhého souboru a načtená data se porovnávají, jestli jsou shodná a soubory k sobě opravdu patří. Následně se přečtou první dva vzorky a spočte se vzorkovací perioda. Jelikož při převodu docházelo k velice malým zaokrouhlovacím chybám, tak se obě vzorkovací periody zaokrouhlí. To se může provést, protože vzorkovací frekvence bývá celé číslo, a tak vzorkovací perioda nebude mít nekonečný desetinný rozvoj. Obě tyto zaokrouhlené hodnoty se porovnají. Pokud všechna porovnání vyjdou pozitivně, z obou souborů se tedy načtou a převedou jednotlivé vzorky.

Změření elektrických parametrů

```
void CANElectricalParametersDetect(int samples, float **CAN_h,
float **CAN_l, HeaderCAN *headerCAN)

void FlexElectricalParametersDetect(int samples,
float** flexBP, float** flexBM, HeaderFlex* header)
```

Ke změření elektrických parametrů signálů obou sběrnic slouží funkce CANElectrical-ParametersDetect a FlexElectricalParametersDetect. U obou funkcí je vstupem hodnota udávající počet načtených vzorků ze souboru, pointery na pointery ukazující v paměti na pole s načtenými upravenými daty a posledním vstupem je pointer na strukturu obsahující informace o zjištěných parametrech dekódované sběrnice. Obě struktury *HeaderCAN* a *HeaderFlex* jsou obsahově totožné se strukturou *HeaderSignal*. Obě struktury se dále v programu využívají pouze pro zpřehlednění zdrojového kódu a tím i eliminaci chyb tím, že mají vnitřní proměnné pojmenované podle skutečných názvů souvisejících s danou sběrnicí.

U obou sběrnic se nejprve změří maximální a minimální hodnota napětí obou navzorkovaných kanálů. V případě sběrnice CAN podle rovnice (1) a v případě FlexRay podle rovnice (3). Zároveň se ověří, zda nejsou prohozené načtené kanály pro CAN-H a CAN-L, nebo BP a BM. U sběrnice CAN je jednoduší řešení, kdy se počítá rozdíl dle rovnice (1) a počítá se počet vzorků, vyhovujících a nevyhovujících podmínce U_{DIF} větší než nastavený práh pro stav *resessive*. Z těchto dvou počtů vzorků se určí, zda jsou kanály prohozené a případně se pointery na obě pole prohodí. V případě sběrnice FlexRay se volí jiný postup, kdy se počítá počet stavů sběrnice *Idle* do doby než se nalezne jiný stav sběrnice. Pokud je počet vzorků ve stavu *Idle* větší než empirická hodnota počtu vzorků, který by mohl značit pouze přechodový stav, provede se vyhodnocení prvního stavu sběrnice po stavu *Idle*. Pokud se jedná o stav odpovídající logické úrovni 1, značí to, že jsou měřené kanály prohozeny, a tak se musí pointery ukazující na načtená data ze souboru prohodit. V případě obou sběrnic se provede záznam o tom, který kanál osciloskopu odpovídá kterému vodiči sběrnice.

V poslední části se v těchto funkcích projdou data znovu a spočítají se hodnoty šumu [14] [15]. Jak maximální hodnota šumu, tak RMS hodnota [16] dle vzorce (6).

$$U_{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \quad (6)$$

Dekódování jednotlivých bitů

```
unsigned int DecodeBitsCAN (char **data_ptr, int *samples,
float *can_h, float *can_l, HeaderCAN headerCAN)
unsigned int DecodeBitsFlex (char **data_ptr, int *samples,
float *BP, float *BM, HeaderFlex headerFlex)
```

Zde se již jednotlivé funkce dělí dle potřeb pro obě sběrnice, avšak vstupy jsou téměř stejné, prvním vstupem je ukazatel na ukazatel pole, do kterého se uloží jednotlivé dekódované bity, následuje ukazatel na počet načtených vzorků. Další dva ukazatele jsou na pole načtených a upravených dat ze souboru. Poslední vstup je struktura, která obsahuje informace o parametrech dekódované sběrnice, respektive o zjištěných parametrech. Uvnitř těchto funkcí se provádí kromě dekódování navzorkovaných napětí na obou vodičích sběrnice také kontrola napěťových úrovní jednotlivých signálů, i jejich rozdílu dle uživatelem zadaných hodnot. V případě, že uživatel hodnoty nezadá, provede se kontrola podle výchozích hodnot aplikace, které jsou voleny nejen podle norem [17], [18], [19], [4], [20], ale také podle využití dané aplikace, která bude využívána především pro výukové účely, viz Tabulka 9 - Přehled výchozích hodnot napětí pro dekódování.

Tabulka 9 - Přehled výchozích hodnot napětí pro dekódování

	U_{MIN} [V]	U_{MAX} [V]
CAN – Dominant	0,10	3,90
uBus – log. 0	2,00	3,90
CAN - Recessive	-0,10	0,50
BP, BM	-0,05	3,60
uBus – log. 1	0,60	2,00
uBus – log. 0	-2,00	-0,60

Zjištění přenosové rychlosti

```
void CANBaudRate(int samples, char* data,
HeaderCAN *headerCAN);
void FlexBaudRate(int samples, char* data,
HeaderFlex* headerFlex)
```

Odhad přenosové rychlosti sběrnice CAN nebo FlexRay zajišťují funkce *CANBaudRate* a *FlexBaudRate*. Obě tyto funkce ze vstupních dat, informace o počtu načtených vzorků a již zjištěných parametrech sběrnice detekuje přenosovou rychlost. Ta se zjistí tak, že se v celém signálu hledá minimální počet vzorků, kdy je sběrnice ve stavu odpovídající logické úrovni 1. Ze znalosti vzorkovací frekvence a počtu vzorků nejkratšího stavu na sběrnici se dopočítá přenosová rychlost. Ta se následně porovnává se standardizovanými přenosovými rychlostmi, viz Tabulka 2 - Přehled přenosových rychlostí sběrnice CAN dle CiA 102 a Tabulka 4 - Přehled přenosových rychlostí sběrnice FlexRay a zjistí se standardizovaná přenosová rychlost sběrnice a jejich vzájemná odchylka, která se spočítá dle vzorce (7).

$$BaudRate_{err} = \frac{100}{BaudRate_{Approx}} \cdot BaudRate_{calc} - 100 \qquad (7)$$

Dekódování dat

```
void DecodeDataCAN (char *data, int samples,
HeaderCAN *headerCAN)
```

Dekódování samotných přenášených dat v případě sběrnice CAN obstarává funkce *DecodeDataCAN*, jejímiž vstupy je pointer na přenášená data již převedená do příslušných stavů sběrnice, dále proměnná obsahující informaci o počtu navzorkovaných hodnot, nyní odpovídající počtu jednotlivých stavů sběrnice. Posledním vstupem je struktura s parametry sběrnice, která obsahuje důležitý údaj o počtu vzorků na jeden přenášený bit.

Celé dekódování přenášeného paketu na sběrnici CAN je založeno na vyhledávání startovacího bitu přenosu. Při detekování tohoto bitu je nastaven obsah výsledné struktury obsahující veškerá dekódovaná data na výchozí hodnoty. Následně se provede kontrola, zda je při komunikaci správně prováděn bit stuffing pro maximálně dlouhý přenášený rámec. A zároveň jsou odstraněny všechny bity, které byly do komunikace vloženy právě kvůli

bit stuffingu a do pomocného pole jsou ze všech vzorků uloženy jen hodnoty jednotlivých bitů přenášených na sběrnici CAN. Tyto bity jsou vybírány dle normy [21], nebo specifikace [3]. Konkrétně jde o to, že se jako hodnota přenášeného bitu vezme vzorek, který byl navzorkován ve ¾ časového rámce pro daný bit. Pokud bude detekován problém s chybějícím bitem opačné hodnoty, než bylo posledních pět po sobě jdoucích bitů, je tato chyba zaznamenána a bude zobrazena ve výsledném souboru s dekódovanými daty. Tento typ chyby navíc způsobí, že se v daném rámci s velkou pravděpodobností objeví i další chyby, neboť nebude moci být v pořádku struktura rámce, vyjít kontrolní součet apod.

Dekódování dále pokračuje nalezením bitu číslo 13 a 14 z daného rámce. Tyto dva bity indikují, o jaký typ paketu se jedná, jestli o základní dle CAN 2.0A nebo rozšířený dle CAN 2.0B (oba bity by nabývali logické hodnoty 1). Po zjištění o jaký paket se jedná je dekódován celý paket. Během dekódování paketu se také kontrolují možné chyby v podobě kontroly CRC, které je součástí této funkce, jeho potvrzení, potvrzení paketu, zda byl rámec správně ukončen, potvrzování příjmu dat nebo správné zakončení přenášeného rámce. Po dekódování celého paketu se zavolá funkce *RepresentResultFlex*, která dekódovaná data uloží do výstupního souboru.

```
void DecodeDataFlex (char *data, int samples, HeaderFlex*
header);
```

Dekódování přenášených dat v případě sběrnice FlexRay obstarává funkce *DecodeDataFlex*, tato funkce má stejné vstupy s funkcí *DecodeDataCAN*.

Dekódování přenášeného paketu je založeno podobně jako u sběrnice CAN na hledání startovací podmínky, která má v případě sběrnice FlexRay podobu 5-ti až 16-ti po sobě jdoucích bitů v logické úrovni 0, za kterým musí následovat jeden bit FSS a podmínka BSS. V případě, že je nalezena posloupnost bitů TSS, je nastaven obsah výsledné struktury na výchozí hodnoty a začnou se do ní ukládat důležitá dekódovaná data. Jako první však čas začátku vysílání paketu počítaný od začátku měření. Po detekování náběžné hrany prvního bitu v TSS se všechny bity začnou vzorkovat dle normy [5] [20], pokud je to díky vzorkovací frekvenci osciloskopu možné dodržet. V opačném případě se volí rozhodující vzorek nejblíže normou danému. V normě je uvedena vzorkovací frekvence osmkrát vyšší, než je rychlost přenášených dat, aby bylo zajištěno 8 vzorků pro každý bit. Jako relevantní vzorek pro hodnotu daného bitu se vybere vždy 5-tý vzorek.

Dekódování postupně pokračuje po jednotlivých bitech hlavičky, přes ID přenášeného paketu, délku přenášených dat, CRC hlavičky a čítač komunikačních rámců. Z těchto dat hlavičky se vyberou bity Sync frame indicator, Startup frame indicator ke kterým se přidá celé ID rámce s délkou dat. Z těch se pomocí funkce *CalcFlexRayCRC* spočítá CRC hlavičky, které se porovná s dekódovaným CRC a také se uloží do výstupní struktury. Ve funkce se dále pokračuje dekódování přenášených dat a nakonec i celkovým CRC paketu. Toto dekódované CRC se pak porovná opět s výstupy funkce *CalcFlexRayCRC*, která se zavolá dvakrát, vždy s jinými parametry. Do funkce *CalcFlexRayCRC* se vloží pointer na dekódovaná data, zde již celý paket, s jejich velikostí, délka výsledného CRC, inicializační vektor a polynom reprezentující polynomiální funkci CRC. Rozdílným parametrem je v tomto případě inicializační vektor, který je jiný pro kanál A a kanál B. Tím je také možné detekovat, jaký kanál byl osciloskopem vzorkován.

Během samotného dekódování se také testují různé chybové stavy, které se zaznamenávají do stejné výsledné struktury. Těmito chybami je například chybné TSS, FSS, BSS, FES, chyba čítače komunikačních cyklů, která nesmí být menší než předchozí, kromě jedné výjimky, kterou je jeho přetečení. Zároveň se také kontrolují oba kontrolní součty.

3.4 Testovací program

3.4.1 Program pro S12XF

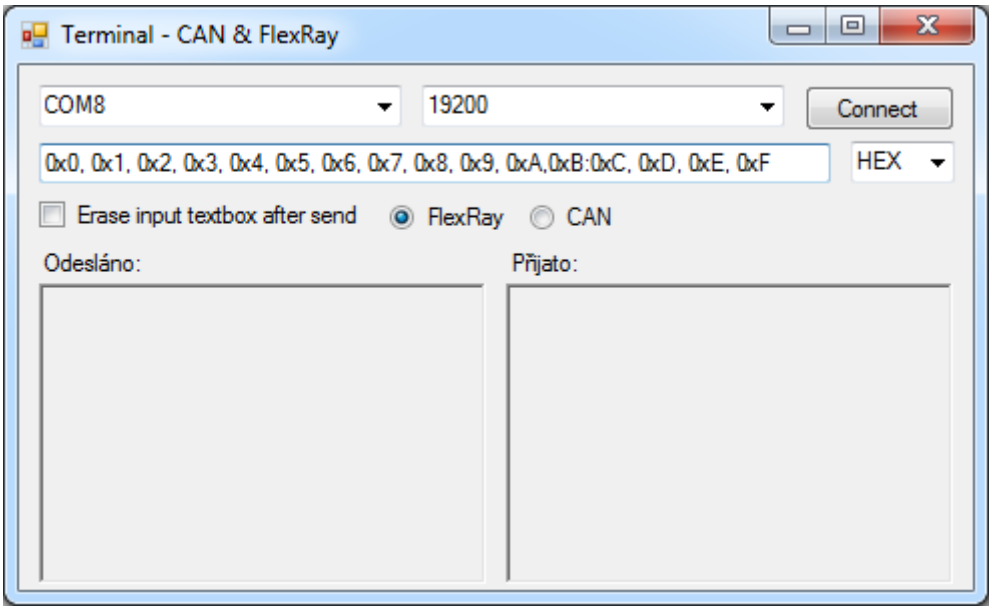
Pro mikroprocesor Freescale S12XF512MLM byl vytvořen testovací program pro sběrnice CAN a FlexRay. Tento program vznikl jako nadstavba vzorového programu dodávaného spolu s vývojovými kity EVB9S12XF512E. Program je založen na knihovnách pro sběrnici FlexRay dodávaných přímo od firmy Freescale s názvem *FR_Unified_driver*. Díky těmto knihovnám je zajištěna samotná komunikace na sběrnici FlexRay. Komunikaci na sběrnici CAN se obsluhuje pomocí funkcí z knihoven pro tento procesor a sběrnici CAN.

Bylo tedy nutné doprogramovat knihovnu pro komunikaci s PC, která překvapivě nebyla součástí SW balíčku k vývojovým kitům. Jako základní komunikační rozhraní byla zvolena sériová linka. Po naprogramování knihovny sériové linky je v obsluze přerušení naprogramovaná funkce, zajišťující interface mezi PC a sběrnicemi CAN a FlexRay. Tento interface přijímá data z PC po sériové lince, ukládá je do připraveného bufferu. Pokud jsou data ve správném formátu, jsou následně pomocí knihovních funkcí odeslány na příslušnou sběrnici.

3.4.2 Program pro PC

Pro uživatele byl naprogramovaný terminál uzpůsobený pro funkci uvnitř mikroprocesoru Freescale S12XF512MLM. Tento terminál byl naprogramovaný v programovacím jazyce C# a jeho funkcí kromě odesílání dat do mikroprocesoru přes sériovou linku je data od uživatele převést do jednotlivých bytů a naopak přijatá data převést do podoby, kterou uživatel požaduje. Nabízí tak možnost aby uživatel zadával a četl data binárně, hexadecimálně, nebo v desítkové soustavě. Z načtených dat je zjištěna jejich délka a z uživatelské volby je zjištěna sběrnice, na kterou mají být data odeslána. Terminál tedy jako první odešle ASCII hodnotu znaku F nebo znaku C, podle zvolené sběrnice, následně odešle byte obsahující počet přenášených bytů. Po této hlavičce následují za sebou převedená vstupní data od uživatele do mikroprocesoru, který data zpracuje a odešle na sběrnici. Náhled terminálu viz Tabulka 10 – Terminál pro CAN a FlexRay

Tabulka 10 – Terminál pro CAN a FlexRay



3.5 Výstup programu

Výstupem z této aplikace byly zvoleny soubory ve formátu HTML. Důvodů ke zvolení právě tohoto formátu bylo několik. Prvním důvodem byla podpora systému inteligentního pracoviště AP9 a univerzálnost tohoto datového formátu, díky které půjdou výstupní soubory číst, zobrazit a zpracovat na libovolném zařízení od počítače, tabletu, telefonu s OS či bez OS až po jednočipové mikroprocesory. Druhým důvodem je fakt, že daný výstupní formát je plně dostačující pro potřeby zobrazení výsledků a zároveň ho již mám vyzkoušený z předešlých prací [8] [13]. Posledním důvodem je široká dostupnost dokumentace ať již v knižní podobě [22], nebo na internetu.

Výstup z aplikace byl rozdělen do několika souborů, které tvoří ucelený souhrn výsledků. Celkem e jedná o tři přehledné soubory. V prvním souboru se nacházejí chybové stavy na fyzické vrstvě sběrnice. V druhém jsou uloženy dekódovaná data včetně všech detekovaných chybových stavů. Poslední soubor je spíše redundantní, protože obsahuje informace, které jsou již obsaženy v druhém souboru. Ve třetím souboru jsou pouze dekódovaná data, respektive pakety, ve kterých byl detekován chybový stav. Tento soubor bude zajímat například uživatele, kteří hledají problém s nespolehlivostí nebo obecně problémy komunikace a sběrnice.

3.5.1 Výstupní soubory

Fyzická vrstva

V tomto souboru jak již bylo naznačeno v kapitole 3.5 jsou uvedeny chybové stavy sběrnice na fyzické vrstvě. Zároveň s těmito chybovými stavy tento soubor obsahuje informace o fyzické vrstvě sběrnice. Mezi těmito informacemi nechybí ty osciloskopu, se kterým se data vzorkovala, v případě že je to možné tak i sériové číslo. Dále datum a čas měření dat včetně informace o vzorkovací frekvenci. Pod těmito informacemi jsou uvedené kanály osciloskopu, kterými se měřili jednotlivé vodiče sběrnice a následují již jen informace o fyzické vrstvě. Konkrétně jde o rozsahy napětí, ve kterých se nachází napětí na obou vodičích sběrnice, jejich absolutní hodnota šumu a RMS hodnota šumu. U chybových stavů na sběrnici je vždy zvýrazněna hodnota, která byla mimo uživatelsky definované rozsahy napětí. Ukázka výstupního souboru viz Obrázek 13 - Výstupní soubor s chybovými stavy fyzické vrstvy.

Chybové vzorky na sběrnici CAN

Osciloskop	GW Instek 2000A
Zdroj	CSV
Datum	12-Apr-15
Čas	13:32:19
Vzorkovací frekvence	1000.000 kHz
Signál CANH byl připojen ke kanálu	CH1
Signál CANL byl připojen ke kanálu	CH2
Rozsah napětí signálu CANH	0.176 - 0.408 V
Rozsah napětí signálu CANL	0.100 - 0.264 V
Maximální hodnota šumu CANH	0.075 V
Maximální hodnota šumu CANL	0.047 V
Šum RMS CANH	0.000 V
Šum RMS CANL	0.000 V

Chybové vzorky

Index	Vstupní vzorek	Čas	Napětí uBP	Napětí uBM	Napětí uBUS
0	1403	0.001403s	0.240V	0.172V	0.068V

Obrázek 13 - Výstupní soubor s chybovými stavy fyzické vrstvy

Dekódovaná data

V tomto souboru, ve kterém jsou uloženy informace nejen o samotných dekódovaných datech ve všech paketech, ale také informace o parametrech sběrnice. V tomto výstupním souboru jsou opět informace o osciloskopu, datu a času vzorkování dat, ale také informace o vzorkovací frekvenci osciloskopu. Pokračuje změřená rychlost sběrnice spolu se standardizovanou a jejich procentuálně vyjádřená odchylka. Pod tabulkou s informacemi o mařených datech a sběrnici se nachází v případě sběrnice FlexRay dvě legendy viz Obrázek 14 - Výstupní soubor s dekódovanými daty - FlexRay a v případě sběrnice CAN legenda pouze jedna.

Data přenášená na sběrnici FlexRay

Osciloskop	DSO9254A:MY53130116
Zdroj	BIN
Datum	22 FEB 2016
Čas	13:31:08
Vzorkovací frekvence	500000.000 kHz
Rychlost FlexRay - změřená	10416.667 kBaud
Rychlost FlexRay - standardizovaná	10000.000 kBaud
Odchylka rychlosti	4.2 %

Legenda - chybové stavy

Více detekovaných chyb	Chyba CRC - hlavička	Chybné CRC - paket	Chyba čítače komunikačních cyklů	Chyba startovací sekvence (TSS)	Chyba začátku rámce (FES)	Chyba oddělovače bytů (BSS)	Chyba ukončení rámce (FES)

Legenda - zkratky

RB	Rezervovaný bit
PPI	Payload Preamble Indicator
NFI	Null Frame Indicator
SyFI	Startup Frame Indicator
StFI	Startup Frame Indicator
CC	Číslo komunikačního cyklu

Dekódovaná data

Index	Čas	Vzorkovaný kanál FlexRay	RB	PPI	NFI	SyFI	StFI	ID rámce	Délka dat	CRC hlavičky - dekódované	CRC hlavičky - spočtené	CC	Data	CRC - dekódované	CRC - spočtené
0	0.000465s	A	0	0	1	1	1	0x01	0x20	0x00F2	0x00F2	0x39	0x00, 0x00	0x1D7415	0x1D7415
1	0.000615s	A	0	0	1	1	1	0x04	0x20	0x06D3	0x06D3	0x39	0x00, 0x00	0x48121A	0x48121A

Obrázek 14 - Výstupní soubor s dekódovanými daty - FlexRay

V obou případech sběrnic je přítomna legenda obsahující chybové stavy na sběrnici, kdy každému chybovému stavu odpovídá jedno políčko v tabulce legendy a jedna barva, kterou bude podbarven celý řádek s paketem v případě, že se v něm daná chyba objevila. Toto řešení bylo zvoleno kvůli jeho přehlednosti a lepší orientaci v případě že uživatel bude data procházet pouze zběžně. V případě, že bude detekováno více chyb, bude řádek s paketem podbarven tomu odpovídající červenou barvou a následně vložen další řádek tabulky se všemi vyznačenými chybami daného paketu. Názornost řešení viz Obrázek 15 - Výstupní soubor s dekódovanými daty - CAN

Data přenášená na sběrnici CAN

Osciloskop	GW Instek 2000A
Zdroj	CSV
Datum	12-Apr-15
Čas	13:32:19
Vzorkovací frekvence	1000.000 kHz
Rychlost CAN - změřená	41.667 kBaud
Rychlost CAN - standardizovaná	41.666 kBaud
Odchylka rychlosti	0.001603 %

Legenda

Více detekovaných chyb	Chyba bitstuffing	Chybné CRC	Nepotvrzené CRC (CRC delimiter)	Nepotvrzený packet (ACK)	Nepotvrzený ACK bit (ACK delimiter)	Chybný konec packetu

Dekódovaná data

Index	Čas	Typ paketu	Identifikátor A, B	RTR	IDE	Data	CRC - dekódované	CRC - spočtené	CRC delimiter	ACK	ACK delimiter	EOF
0	0.000105s	CAN 2.0A	0x070E	0	0	0x25, 0x80, 0x49, 0xB9, 0x55, 0x6D, 0x6E	0x4B5E	0x456A	1	0	1	0x77
		Detekované chyby:		X	X				X			
1	0.002727s	CAN 2.0A	0x0343	0	0	0x96, 0xB5, 0x7B	0x0754	0x0754	1	0	1	0x7F

Obrázek 15 - Výstupní soubor s dekódovanými daty - CAN

Dekódovaná chybová data

Poslední výstupní soubor obsahuje téměř stejné informace jako výstupní soubor s dekódovanými daty. Rozdíl mezi těmito soubory je na první pohled patrný v tom, že zde jsou obsaženy pouze pakety, ve kterých se objevil libovolný chybový stav. Náhled viz Obrázek 16 - Výstupní soubor s chybovými pakety - FlexRay.

Chybové pakety na sběrnici FlexRay

Osciloskop	DSO9254A:MY53130116
Zdroj	BIN
Datum	16 NOV 2015
Čas	11:19:23
Vzorkovací frekvence	100000.000 kHz
Rychlost FlexRay - změřená	11111.111 kBaud
Rychlost FlexRay - standardizovaná	10000.000 kBaud
Odchylka rychlosti	11.1 %

Legenda - chybové stavy

Více detekovaných chyb	Chyba CRC - hlavička	Chybné CRC - paket	Chyba čítače komunikačních cyklů	Chyba startovací sekvence (TSS)	Chyba začátku rámce (FES)	Chyba oddělovače bytů (BSS)	Chyba ukončení rámce (FES)

Legenda - zkratky

RB	Rezervovaný bit
PPI	Payload Preamble Indicator
NFI	Null Frame Indicator
SyFI	Startup Frame Indicator
StFI	Startup Frame Indicator
CC	Číslo komunikačního cyklu

Dekódovaná data

1870	3.117496s	-	0	0	1	1	1	0x02	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1872	3.117499s	-	0	1	0	2	2	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1874	3.117502s	-	0	0	0	0	0	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1877	3.117697s	-	0	0	0	0	0	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1891	3.152658s	-	0	0	0	0	0	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1897	3.162647s	-	0	0	0	1	1	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1899	3.162650s	-	0	0	0	0	0	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1903	3.167648s	-	0	0	0	0	0	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1905	3.167663s	-	0	0	0	0	0	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000
1907	3.167701s	-	0	0	0	0	0	0x00	0x00	0x0000	0x0000	0x00	0x00	0x000000	A: 0x000000 B: 0x000000

Obrázek 16 - Výstupní soubor s chybovými pakety - FlexRay

4 Dosažené výsledky

Před implementací výsledné aplikace do inteligentního měřícího pracoviště se systémem dálkové správy zpracování dat bylo nutné ještě provést testy rychlosti. K testům jsem zvolil několik různých naměřených souborů a aplikaci vyzkoušel na několika počítačích.

4.1 Test rychlosti aplikace

4.1.1 Konfigurace

Jak již bylo zmíněno, vybral jsem několik počítačů a různých souborů. Konkrétně se jedná o jeden binární 3,81 GB soubor, druhý binární 76,2 MB a třetí nejmenší 228 kB textový soubor. Volba takto velikých souborů byla jasná. Největší soubor byl vybraný jako zátěžový test celé aplikace a zbylé menší soubory, jako vzorky souborů, které by reálně během používání inteligentního měřícího pracoviště mohli vznikat.

Výběr testovacích počítačů viz Tabulka 11 - Konfigurace testovacích počítačů nebyl náhodný, šlo mi o to, zjistit, jak se bude výsledná aplikace chovat na různě výkonném HW. Pro lepší porovnání důležitosti rychlosti pevného disku je jeden počítač do testu zařazen dvakrát, jednou se data budou načítat z SSD disku a podruhé z plotnového disku.

Tabulka 11 - Konfigurace testovacích počítačů

	PC 1	PC 2	PC 3	PC 4
Procesor	Core i5-3230M	Core i7-4610M	Core i5-3210M	Core i5-2500
Operační paměť	8 GB 1600 MHz	8 GB 1600 MHz	4 GB 1066 MHz	8 GB 1333 MHz
Pevný disk	SP600N34 (SSD) HGST725050A7E630 (2,5“, 7200RPM)	Samsung SSD PM871 2.5 7m	TOSHIBA MQ01ABD075 ATA Device	ST3500413AS (3,5“, 7200RPM)
Operační systém	Windows 7 Professional x64	Windows 7 Professional x64	Windows 10 Pro x64	Windows 7 Professional x64

4.1.2 Podmínky testu

Na všech testovaných počítačích, na kterých byl 64-bitový operační systém byla spuštěna zkompilovaná 64-bitová verze výsledné aplikace s drobnou úpravou, kterou je vypisování časových údajů jednotlivých vybraných důležitých kroků aplikace. Na každém počítači se provedlo celkem 15 měření, vždy 5 pro každý typ souboru a jako výsledek jsem vybral to měření, které odpovídá mediánu celkového času běhu aplikace.

Rozdíl mezi celkovým časem potřebným pro vykonání programu ve výsledcích označený jako celkový čas a součtem jednotlivých měřených kroků je uveden pod položkou režie. Čas režie je tak dopočítávanou veličinou.

4.1.3 Vyhodnocení testu

Binární soubor 3,81 GB

U testu rychlosti aplikace při použití téměř 4 GB souboru nebylo možné naměřit jednotlivé časy u PC 3, protože se u tohoto počítače sešlo několik okolností dohromady. Jelikož má 4 GB operační paměti a zároveň téměř plný pevný disk, na který mohl swapovat, nebylo možné výslednou aplikaci na tomto PC dokončit celou operaci. Celkové výsledky viz Tabulka 12 - Výsledky testu rychlosti aplikace - 3,81 GB soubor.

Tabulka 12 - Výsledky testu rychlosti aplikace - 3,81 GB soubor

	PC 1 - SSD	PC 1 - HDD	PC 2	PC 3	PC 4
Inicializace programu	0,010 s	0,002 s	0,010 s	-	0,019 s
Načtení dat ze souboru	11,445 s	55,012 s	12,160 s	-	30,342 s
Detekování el. parametrů	8,720 s	8,650 s	4,210 s	-	4,539 s
Dekódování bitů	7,090 s	7,080 s	1,910 s	-	2,496 s
Detekování přenosové rychlosti	1,410 s	1,420 s	0,380 s	-	0,718 s
Dekódování dat	1,550 s	1,550 s	0,350 s	-	0,343 s
Režie	0,740 s	0,630 s	0,340 s	-	0,262 s
Celkový čas	30,965 s	74,344 s	19,360 s	-	37,719

Binární soubor 76,2 MB

Tabulka 13 - Výsledky testu rychlosti aplikace - 76,2 MB soubor

	PC 1 - SSD	PC 1 - HDD	PC 2	PC 3	PC 4
Inicializace programu	0,010 s	0,001 s	0,010 s	0,015 s	0,031 s
Načtení dat ze souboru	0,220 s	1,130 s	0,260 s	0,094 s	0,655 s
Detekování el. parametrů	0,200 s	0,160 s	0,080 s	0,266 s	0,094 s
Dekódování bitů	0,160 s	0,140 s	0,030 s	0,125 s	0,046 s
Detekování přenosové rychlosti	0,030 s	0,020 s	0,010 s	0,031 s	0,016 s
Dekódování dat	0,030 s	0,030 s	0,010 s	0,015 s	0,000 s
Režie	0,010 s	0,009 s	0,010 s	0,016 s	0,016 s
Celkový čas	0,660 s	1,490 s	0,410 s	0,562 s	0,858 s

Textový soubor 228 kB

Tabulka 14 - Výsledky testu rychlosti aplikace - 228 kB soubor

	PC 1 - SSD	PC 1 - HDD	PC 2	PC 3	PC 4
Inicializace programu	0,001 s	0,000 s	0,000 s	0,000 s	0,015 s
Načtení dat ze souboru	0,029 s	0,070 s	0,010 s	0,031 s	0,016 s
Detekování el. parametrů	0,000 s	0,000 s	0,000 s	0,000 s	0,000 s
Dekódování bitů	0,010 s	0,000 s	0,000 s	0,000 s	0,000 s
Detekování přenosové rychlosti	0,000 s	0,000 s	0,000 s	0,000 s	0,000 s
Dekódování dat	0,000 s	0,000 s	0,000 s	0,015 s	0,000 s
Režie	0,000 s	0,000 s	0,000 s	0,000 s	0,000 s
Celkový čas	0,004 s	0,070 s	0,010 s	0,046 s	0,031 s

4.1.4 Vyhodnocení naměřených dat

Z výsledků všech testů je patrné, které části aplikace jsou časově nejnáročnější. Celkově nejnáročnější je načíst data z uloženého souboru i přesto, kdy nejvíce záleží na typu pevného disku. Dalšími dvěma částmi, které vyžadují oproti zbytku aplikace větší časový prostor, jsou detekování elektrických parametrů a dekódování bitů. To je způsobeno tím, že například funkce pro detekování elektrických parametrů prochází všechna data několikrát a funkce pro dekódování bitů musí počítat rozdílové napětí, všechna napětí porovnávat s definovanými mezními úrovněmi. Zároveň v případě chybového vzorku volá další funkci, která tyto vzorky zapisuje do souboru, tudíž čas potřebný pro dekódování bitů je závislý také na počtu nalezených chyb. Rychlost těchto částí se dá také ovlivnit výkonem PC, především procesoru. Dosažené hodnoty jsou obecně velmi uspokojivé, u běžných měření a souborů běžné délky v desítkách až stovkách MB se dosahuje velmi přijatelných časů v řádu zlomků až jednotek sekund.

4.2 Doporučená nastavení osciloskopu

Pro správnou funkci aplikace je nutné, aby bylo dodrženo dodržení správné vzorkování. Při vzorkování obou sběrnic musí tedy být splněna podmínka na vzorkovací frekvenci, aby neodporovala Shannon-Kotělnikovovu teorému, který v našem případě definuje minimální vzorkovací frekvenci alespoň jako dvojnásobek přenosové frekvencedat na sběrnici.

Je tedy doporučeno a zařízení splňující normy obou sběrnic počítají s mnohem vyšší vzorkovací frekvencí. Je tedy vhodné i s ohledem na dekódování, které je schopna aplikace provádět i s nižší vzorkovací frekvencí oproti jednotlivým zařízením, počítat s vyšší vzorkovací frekvencí než je jen zmíněný dvojnásobek. Mohlo by totiž dojít v řetězci vzorkování, uložení dat, načtení a zpracování dat k tomu, že by při dekódování byl některý bit náhodně přeskočen, nebo naopak započítán dvakrát. Z toho také vychází pro obě sběrnice doporučená a teoreticky minimální vzorkovací frekvence, která se dá snadno nastavit na osciloskopu. Sběrnici CAN odpovídá Tabulka 15 - Doporučená nastavení vzorkovací frekvence - CAN a sběrnici FlexRay Tabulka 16 - Doporučená nastavení vzorkovací frekvence - FlexRay.

Tabulka 15 - Doporučená nastavení vzorkovací frekvence - CAN

Rychlost sběrnice	Minimální vzorkovací frekvence	Doporučená vzorkovací frekvence
1 Mbit/s	2 MSa/s	5 MSa/s
500 kbit/s	1 MSa/s	5 MSa/s
250 kbit/s	500 kSa/s	2 MSa/s
100 kbit/s	200 kSa/s	500 kSa /s
50 kbit/s	100 kSa /s	250 kSa /s
25 kbit/s	50 kSa /s	250 kSa /s
10 kbit/s	20 kSa /s	50 kSa /s

Tabulka 16 - Doporučená nastavení vzorkovací frekvence - FlexRay

Rychlost sběrnice	Minimální vzorkovací frekvence	Doporučená vzorkovací frekvence
10 Mbit/s	20 MSa/s	50 MSa/s
5 Mbit/s	10 MSa/s	50 MSa/s
2,5 Mbit/s	5 MSa/s	25 MSa/s

4.3 Integrace do inteligentního měřícího systému

Program vyvinutý v rámci této diplomové práce byl po ověření jeho funkčnosti a otestování jeho vlastností především s ohledem na rychlost zpracování dat byl úspěšně zakomponován do systému inteligentního měřícího pracoviště se systémem dálkové správy [23] [24], který vznikl modernizací výuky v učebně AP9 [25]. Je tedy možné na žádost obsluhy (studenta) daného pracoviště nebo vyučujícího, který připravil měřící úlohu na cvičení z předmětu [26], tento program zavolat pomocí PHP funkce přímo na serveru, který si nejprve vyžádá a stáhne naměřená data z konkrétního osciloskopu, který je jako ostatní přístroje, které tento systém podporuje připojen k počítači na daném pracovišti. PHP skript vyčká, až bude mít veškerá data z osciloskopu k dispozici a následně zavolá aplikaci pro analýzu sběrnic CAN a FlexRay, která na výstupu vrací HTML soubory. Skript v poslední fázi výsledky zobrazí zpět uživateli na jeho pracovišti. Tím byl bod 6 zadání splněn.

5 Závěr

Cílem této diplomové práce bylo seznámení se se standardy sběrnic CAN a FlexRay a parametry osciloskopů s velkou hloubkou záznamu naměřených dat s ohledem na jejich využití při analýze komunikace na zkoumaných sběrnicích. Dalším cílem byl samotný návrh a realizace programu, který provede analýzu naměřených dat z osciloskopu a vytvoří výpis komunikace do HTML souboru. V neposlední řadě výsledný program měl obsahovat analýzu a upozornění na základní chybové stavy.

Nejprve tedy bylo provedeno samotné nastudování standardu sběrnic CAN s FlexRay s ohledem na realizaci samotné diplomové práce. K tomuto účelu bylo nutné rovněž zakoupit některé normy CAN, a především všechny normy FlexRay, neboť informace dostupné na internetu nebo v publikacích byly velmi neúplné. Především v případě sběrnice FlexRay bylo zakoupení norem přímo nutností ke zdárnému dokončení této diplomové práce, protože se nenacházely ani v univerzitní a Krajské vědecké knihovně. Dále před samotnou realizací proběhlo nastudování parametrů osciloskopů, především však osciloskopu Agilent DSO9254A z uživatelské dokumentace k tomuto osciloskopu a využitím předešlých znalostí tohoto osciloskopu z předchozí práce. Následovalo nastudování příslušných částí programátorské dokumentace.

Samotný návrh a realizace výsledné aplikace se povedlo dokončit do stavu, kdy aplikace splňuje zadání diplomové práce včetně dalších požadavků definovaných v úvodu. Povedlo se tedy vytvořit aplikaci, která je univerzální pro další rozšíření, umožňuje zpracovat velké množství dat ve velice krátkém čase, kdy rychlost celého procesu zpracování je závislá především na rychlosti pevného disku počítače a rychlosti procesoru. Díky tomu bohužel nejde zpracování již více významně zrychlit, jelikož samotné zpracování dat je již řádově v setinách sekundy maximálně v jednotkách sekund u větších souborů případně pomalejších počítačů. Po načtení naměřených dat dojde k jejich zpracování v podobě detekování žádaných parametrů a až následně dojde k jejich optimalizaci v takové míře, že výsledné paměťové nároky na operační paměť jsou čtvrtinové oproti původní velikosti naměřených dat, aniž by došlo ke ztrátě informace nutné pro dekódování dat. Jelikož by z optimalizovaných dat nešli naměřit elektrické parametry sběrnice, je nutné načíst všechna data do paměti. Další optimalizace

by byla z důvodu univerzálnosti řešení, které počítá s budoucím rozšířením, nelze více optimalizovat paměťové nároky při zachování stávající funkčnosti a na nahrávání celého souboru do operační paměti. Dosažené časy zpracování dat vyvinutým programem jsou obecně velmi uspokojivé, u běžných měření a souborů obvyklé délky se dosahuje velmi přijatelných časů v řádu jednotek sekund.

Výstup celé aplikace je nakonec nejenom v jednom detailním výpisu ve formátu HTML, ale jsou přidány ještě další dva detailní výpisy v HTML souboru, čímž se zvýšila přehlednost a možnost vybrat si pouze ta výsledná data, která uživatele skutečně zajímají.

V porovnání s komerčně dostupnými řešeními od výrobců osciloskopů má tato aplikace mnoho předností, například si lze definovat vlastní napěťová pásma, ve kterých bude komunikace považována za správnou, což se hodí u specifických zapojení, kde není možné dodržet přesné specifikace. Aplikace umožňuje detekovat velikost šumu, což komerční řešení pro osciloskopy neumí. Také navíc umí detekovat, jakým kanálem osciloskopu byl vzorkovaný který signál sběrnice, nebo spočítat odchylku přenosové rychlosti od standardizované. Oproti komerčním řešením dále nabízí měření šumu signálu a zároveň je schopna signál včetně tohoto šumu zpracovat. V neposlední řadě oproti některým komerčním řešením umožňuje kontrolu CRC a formální správnost přenášených paketů.

Tato diplomová práce slouží nejen jako výukový nástroj, který je zakomponován do systému inteligentního měřícího pracoviště, ale je také vhodnou pomůckou při vývoji zařízení se sběrnicemi CAN a FlexRay.

Uvažovanou možností jak dále daný software rozvíjet je například bezdrátové měření s nutností nejprve rekonstruovat bezdrátově naměřený signál do původní podoby, dekódování simultánně s měřením osciloskopem, doplnění programu o generování detailních grafů signálu se zakreslením paketů, na které jsou již ve zdrojovém kódu připravené první funkce, případně přidání dalších sběrnic, případně osciloskopů.

Seznam použité literatury

[1]. **CAN in Automation.** CAN Physical Layer. *CAN in Automation (CiA): Controller Area Network (CAN).* [Online] [Citace: 10. 9 2015.] https://support.dce.felk.cvut.cz/pub/hanzalek/_private/ref/canphy.pdf.

[2]. **CAN in Automation.** CiA Draft Standard 102 Version 2.0. *CAN in Automation (CiA): Controller Area Network (CAN).* [Online] 20. 4 1994. [Citace: 10. 9 2015.] http://affon.narod.ru/CAN/DS102.pdf.

[3]. **Philips Semiconductor.** SJA1000 Stand-alone CAN controller. *NXP Semiconductor.* [Online] 17. 8 1999. [Citace: 17. 9 2015.] http://www.nxp.com/documents/data_sheet/SJA1000.pdf.

[4]. **ISO, Norma.** Road vehicles -- FlexRay communications system -- Part 4: Electrical physical layer specification. Switzerland : ISO, 2013. ISO17458-4:2013(E).

[5]. **ISO, Norma.** Road vehicles -- FlexRay communications system -- Part 2: Data link layer specification. Switzerland : ISO, 2013. ISO17458-2:2013(E).

[6]. **ISO, Norma.** Road vehicles -- FlexRay communications system -- Part 3: Data link layer conformance test specification. Switzerland : ISO, 2013. ISO17458-3:2013(E).

[7]. **Agilent Technologies, Inc.** Infiniium 9000 Series Oscilloscopes - Data Sheet. *Web Agilent Technologies.* [Online] 6. 11 2013. [Citace: 7. 10 2014.] http://cp.literature.agilent.com/litweb/pdf/5990-3746EN.pdf.

[8]. **Vošta, Petr.** *Analýza komunikace na I2C sběrnici.* Liberec, 2014.

[9]. **Agilent Technologies, Inc.** Programmer's Reference for Infiniium 9000 Series Oscilloscopes. *Web Agilent Technologies.* [Online] 23. 1 2014. [Citace: 7. 10 2015.] http://www.home.agilent.com/upload/cmc_upload/All/9000_series_prog_ref.pdf?&cc=CZ&lc=eng.

[10]. **Good Will Instrument Co., Ltd.** GDS-2000A Specification. *GW Instek Simply Reliable.* [Online] 27. 2 2013. [Citace: 8. 10 2015.] http://www.gwinstek.com/download/DownloadFile/download%23%40%2301_Oscilloscope%23%40%2301_GDS%23%40%23GDS-2000A_Specification_EN.pdf.

[11]. **Good Will Instrument Co., Ltd.** GDS-2000A CAN/LIN bus application note. *GW Instek Simply Reliable.* [Online] 1. 4 2015. [Citace: 8. 10 2015.] http://www.gwinstek.com/download/DownloadFile/download%23%40%2301_Oscilloscope%23%40%2301_GDS%23%40%23GDS-2000A%23%40%23CANLIN_bus_AppNote_EN.pdf.

[12]. **Keysight Technologies.** N8803A CAN, LIN, and FlexRay Protocol Triggering and Decode for Infiniium 90000 and 90000 X Series scopes | Keysight (Agilent). *Keysight (Agilent).* [Online] 10. 5 2015. [Citace: 7. 10 2016.] http://www.keysight.com/en/pd-1654555-pn-N8803A/can-lin-and-flexray-protocol-triggering-and-decode-for-infiniium-90000-and-90000-x-series-scopes?cc=CZ&lc=eng.

[13]. **Vošta, Petr.** *Systém pro analýzu sběrnie CAN.* Liberec : TU Liberec, 2015. Semestrální projekt.

[14]. **Slavík, Lubomír.** Měřící technika [přednášky]. Liberec : TU Liberec, 2013.

[15]. **Jaksch, Ivan.** Číslicové měřící systémy [Přednášky]. Liberec : TU Liberec, 2014.

[16]. **Březina, Jan.** Pravděpodobnost a statistika [Přednášky]. Liberec : TU Liberec, 2015.

[17]. **ISO, Norma.** Road vehicles -- Controller area network (CAN) -- Part 2: High-speed medium access unit. Switzerland : ISO, 2003. ISO11898-2:2003(E).

[18]. **ISO, Norma.** Road vehicles -- Controller area network (CAN) -- Part 5: High-speed medium access unit with low-power mode. Switzerland : ISO, 2007. ISO11898-5:2007(E).

[19]. **ISO, Norma.** Road vehicles -- Controller area network (CAN) -- Part 6: High-speed medium access unit with selective wake-up functionality. Switzerland : ISO, 2013. ISO11898-6:2013(E).

[20]. **ISO, Norma.** Road vehicles -- FlexRay communications system -- Part 5: Electrical physical layer conformance test specification. Switzerland : ISO, 2013. ISO17458-5:2013(E).

[21]. **ISO, Norma.** Road vehicles -- Controller area network (CAN) -- Part 1: Data link layer and physical signalling. Switzerland : ISO, 2003. ISO11898-1:2003(E).

[22]. **Domes, Martin.** *Tvorba internetových stránek pomocí HTML, CSS a JavaScriptu.* Kralice na Hané : Computer Media s.r.o., 2005. ISBN: 80-86686-39-6.

[23]. **Sieber, Lukáš.** *Inteligentní měřicí pracoviště se systémem dálkové správy a zpracování dat.* Liberec : TU Liberec, 2015. Diplomová práce.

[24]. **Sieber, Lukáš a Pfeifer, Petr.** *Inteligentní měřicí pracoviště se systémem dálkové správy a zpracování dat.* Liberec : Studentská konference FM 2015, 2015.

[25]. **Pfeifer, Petr.** *Inteligentní měřicí pracoviště - Popis řešení modernizace AP9.* Liberec : ESF, Technická univerzita v Libereci, FM ITE, 2012.

[26]. **Pfeifer, Petr.** *Manuál sady přípravků do cvičení PMN (Pokročilé metody návrhu).* Liberec : ESF, Technická univerzita v Libereci, FM ITE, 2014.

[27]. **Plíva, Zdeněk, a další, a další.** Metodika zpracování bakalářských a diplomových prací. *Fakulta mechatroniky, informatiky a mezioborových studií, TUL.* [Online] 3 2014. [Citace: 13. 4 2014.] http://www.fm.tul.cz/cs/jak_psat_prace. ISBN 978-80-7494-049-1.

[28]. **Prata, Stephen.** *C++ primer plus 5th.* Indianapolis : Sams, 2005. ISBN 0-672-32697-3.

[29]. **Virius, Miroslav.** *Pasti a propasti jazyka C++. 2. aktualizované a rozšířené vydání.* Brno : CP Books, 2005. ISBN 80-251-0509-1.

[30]. **Infineon Technologies.** Final Data TLE 6262 G. *Web DatasheetCatalog.com.* [Online] 20. 3 2002. [Citace: 11. 4 2016.] http://pdf.datasheetcatalog.com/datasheet/infineon/1-TLE6262_24.pdf.

[31]. **ISO, Norma.** Road vehicles -- Controller area network (CAN) -- Part 3: Low-speed, fault-tolerant, medium-dependent interface. Switzerland : ISO, 2006. ISO11898-3:2006(E).

[32]. **ISO, Norma.** Road vehicles -- Controller area network (CAN) -- Part 4: Time-triggered communication. Switzerland : ISO, 2004. ISO11898-4:2004(E).

[33]. **ISO, Norma.** Road vehicles -- FlexRay communications system -- Part 1: General information and use case definition. Switzerland : ISO, 2013. ISO17458-1:2013(E).

[34]. **Arndt, Michael.** Implementation of a FlexRay Communication Interface for Linux. [Online] 14. 4 2009. [Citace: 25. 4 2016.] http://tmi.yokogawa.com/files/uploaded/bu7013_61e_020_1.pdf.

[35]. **Freescale Semiconductor.** MC9S12XF512 Reference Manual. [Online] 10. 11 2010. [Citace: 16. 1 2016.] http://cache.nxp.com/files/microcontrollers/doc/ref_manual/MC9S12XF512RMV1.pdf?pspll=1.

[36]. —. EVB9S12XF512E User Manual. [Online] 6 2007. [Citace: 16. 1 2016.] http://cache.nxp.com/files/soft_dev_tools/doc/user_guide/EVB9S12XF512E_UM.pdf?fpsp=1&WT_TYPE=Users%20Guides&WT_VENDOR=FREESCALE&WT_FILE_FORMAT=pdf&WT_ASSET=Documentation&fileExt=.pdf.

[37]. **Agilent Technologies, Inc.** Binary Oscilloscope File to MATLAB Translator Program | Agilent. *Web Agilent Technologies.* [Online] 10. 5 2007. [Citace: 7. 10 2015.] http://www.home.agilent.com/agilent/editorial.jspx?cc=CZ&lc=eng&ckey=1185953&nid=-11143.0.00&id=1185953.

[38]. **Karlsruhe Institute of Technology.** FlexRay Communications System - Protocol Specification. [Online] 10 2010. [Citace: 19. 1 2016.] https://svn.ipd.kit.edu/nlrp/public/FlexRay/FlexRay%E2%84%A2%20Protocol%20Specification%20Version%203.0.1.pdf.

[39]. **Koopman, Philip.** The FlexRay Protocol. [Online] 30. 11 2015. [Citace: 19. 1 2016.] https://www.ece.cmu.edu/~ece649/lectures/23_flexray.pdf.

Obsah přiloženého CD

Text diplomové práce

- Diplomova_prace_2016_Petr_Vosta.pdf
- Diplomova_prace_2016_Petr_Vosta.doc
- Diplomova_prace_2016_Petr_Vosta.docx

Zdrojové kódy

- DP_Vosta-C
- Terminal
- Vzorový firmware pro vývojové kity

Spustitelný program pro platformu x86

- DP_Vosta-C_CZE_x86
- DP_Vosta-C_ENG_x86
- Terminal_x86

Spustitelný program pro platformu x64

- DP_Vosta-C_CZE_x64
- DP_Vosta-C_ENG_x64
- Terminal_x64