VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2019 Bc. Peter Frnka

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

DIAGNOSTICKÝ NÁSTROJ PRO KOMUNIKAČNÍ PROTOKOL SD KARTY
DIAGNOSTIC TOOL FOR SD CARD PROTOCOL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Peter Frnka**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Petr Petyovský, Ph.D.**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření**
Ústav automatizace a měřicí techniky

Student: Bc. Peter Frnka ***ID:*** 164593
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Diagnostický nástroj pro komunikační protokol SD karty

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je navrhnout a realizovat diagnostický nástroj pro komunikační protokol SD karty.
1. Nastudujte standardy definující komunikační protokoly mezi SD kartou a koncovým zařízením.
2. Navrhněte koncepci nástroje umožňujícího diagnostiku a modifikaci komunikace mezi SD kartou a koncovým zařízením.
3. Navrhněte schéma zapojení a architekturu hardware i firmware celého diagnostického nástroje.
4. Sestavte hardware diagnostického nástroje.
5. Vytvořte firmware umožňující monitorování komunikace mezi SD kartou a koncovým zařízením.
6. Do zařízení navrženého a částečně realizovaného v rámci semestrálního projektu přidejte podporu pro real-time úpravu zpráv a vyhodnocování stavu komunikace.
7. Vytvořte PC aplikaci dovolující konfiguraci Vašeho výrobku.
8. Funkčnost vytvořeného systému ověřte na běžně dostupném komerčním zařízení, např. telefonu/tabletu. Vyhodnoťte a zdokumentujte výkon a omezení výrobku.
9. Zhodnoťte dosažené výsledky. Navrhněte další možná vylepšení SW i HW částí.

DOPORUČENÁ LITERATURA:

[1] VIRIUS, Miroslav. Jazyky C a C++: kompletní průvodce. 2., aktualiz. vyd. Praha: Grada, 2011.

ISBN 9788024739175.

[2] SD Card Association: SD Specifications Physical Layer Simplified Specification. [online]. Aug 29. 2018. Dostupné na WWW:
<https://www.sdcard.org/downloads/pls/click.php?f=Part1_Physical_Layer_Simplified_Specification_Ver6.00.pdf>.

Termín zadání: 4.2.2019 ***Termín odevzdání:*** 13.5.2019

Vedoucí práce: Ing. Petr Petyovský, Ph.D.
Konzultant: Jiří Macháček, Honeywell, spol. s r.o.

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

ABSTRAKT

Cieľom tejto práce je návrh a následná realizácia diagnostického nástroja pre komunikačný protokol SD pamäťových kariet. Diagnostický nástroj bude umožňovať monitorovanie a aktívny zásah do komunikácie s SD pamäťovou kartou. Realizácia bude založená na platforme s mikrokontrolérom Kinetis K66. Súčasťou diagnostického nástroja je nadradená aplikácia určená pre operačný systém Windows. V závere práce budú zhodnotené dosiahnuté výsledky a obmedzenia diagnostického nástroja.

KĽÚČOVÉ SLOVÁ

SD karta, diagnosticky nástroj, hardware, firmware, Kinetis K66

ABSTRACT

The objective of this diploma thesis is to design and implement a diagnostic tool for communication protocol of SD memory cards. The diagnostic tool will be able to monitor and interfere with communication of an SD memory card. The hardware implementation will be based on platform with Kinetis K66 microcontroller. A PC application designed for windows is also a part of diagnostic tool. There is a conclusion of the diagnostic tool limitation at the end of the thesis.

KEYWORDS

SD card, diagnostic tool, hardware, firmware, Kinetis K66

FRNKA, Peter. *Diagnostický nástroj pro komunikační protokol SD karty*. Brno, 2019, 72 s. Diplomová práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedúci práce: Ing. Petr Petyovský, Ph.D.

VYHLÁSENIE

Vyhlasujem, že som svoju diplomovú prácu na tému „Diagnostický nástroj pro komunikační protokol SD karty“ vypracoval samostatne pod vedením vedúceho diplomovej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej diplomovej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som poďakoval vedúcemu diplomovej práce pánovi Ing. Petrovi Petyovskému, Ph.D. za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci. Moje poďakovanie patrí aj konzultantovi Ing. Jířimu Macháčkovi za odbornú pomoc a konzultácie zadania práce.

Brno ....

podpis autora

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Diagnostika a testovanie produktov je nevyhnutnou súčasťou procesu vývoja v akomkoľvek odvetví techniky. Do procesu vývoja bezpochyby patrí aj monitorovanie rôznych komunikácií, komunikačných protokolov a zberníc medzi rôznymi zariadeniami. Koncové zariadenia využívajúce pamäťové karty nie sú výnimkou. Predložená práca sa zaoberá návrhom a realizáciou diagnostického nástroja pre komunikačný protokol pamäťovej karty.

Prvá kapitola práce sa bude venovať rozboru zadania, definícií požiadaviek kladených na diagnostický nástroj a popis úvodu do problematiky diagnostiky komunikácií pamäťových kariet.

Nasledujúcim bodom práce, ktorý bude potrebný k návrhu a realizácií diagnostického nástroja, je pochopenie a popis funkčnosti komunikačného protokolu pamäťových kariet.

Ďalej sa práca bude sústrediť na celkový návrh diagnostického nástroja. Súčasťou návrhu bude predstavenie možných konceptov diagnostického nástroja, z ktorých na jeden sa bude práca ďalej sústrediť. Dôležitým bodom návrhovej časti práce, založenom na výbere konceptu a požiadavkách kladených na nástroj, bude návrh jeho hardwaru a softwaru.

Hardware diagnostického nástroja, ako sa neskôr ukáže v rozbore zadania, bude založený na hardwarovej platforme. Súčasťou návrhu hardwaru pre nástroj bude popis výberu vhodnej platformy, ďalších hardwarových častí a schéma zapojenia.

Ďalej sa práca bude zaoberať návrhom firmwaru, ktorý bude schopný diagnostikovať a aktívne zasahovať do komunikácie medzi hostiteľským zariadením a pamäťovou kartou. Návrh softwaru taktiež pozostáva z návrhu nadradenej aplikácie, ktorá bude určená pre PC.

Následne bude prezentovaná realizácia diagnostického nástroja na základe návrhu z predošlých častí práce. V Závere budú zhodnotené dosiahnuté výsledky a možné obmedzenia aktuálnej realizácie nástroja.

1 Rozbor zadania

Úlohou tejto kapitoly je popis problematiky, na ktorú je táto práca zameraná, stanovenie jej hlavných cieľov a rozbor jej zadania. Toto zadanie vzniklo na základe požiadaviek pri testovaní rôznych zariadení, ktoré podporujú ukladanie dát na pamäťovú kartu. Primárne sa jedná o jednoduché vstavané zariadenia. Hlavným cieľom tejto práce je navrhnúť a realizovať prototyp zariadenia, ktoré bude schopné diagnostikovať komunikáciu medzi SD (Secure Digital) kartou a hostiteľským zariadením.

Predtým ako bude text práce pokračovať, je nutné definovať pojem diagnostika komunikácie. Pod týmto pojmom sa v tejto práci rozumie monitorovanie (sledovanie) prenášaných správ medzi hostiteľským zariadením pamäťovou kartou.

Typickým spôsobom monitorovania komunikácie s SD pamäťovými kartami je vyvedenie jej komunikačných signálov a využitie určitého meracieho prístroja, zvyčajne logického analyzátoru alebo osciloskopu. Tento spôsob monitorovania správ umožňuje správy monitorovať, no nezahŕňa žiadny aktívny zásah do komunikácie. V neposlednom rade je využitie spomenutých meracích prístrojov iba na tento účel finančne náročné a neefektívne.

Z tohto dôvodu vznikla myšlienka na realizáciu prototypu vlastného nástroja, ktorý bude schopný komunikáciu diagnostikovať, ale aj aktívne zasahovať do prenášaných správ medzi hostiteľským zariadením a pamäťovou kartou. Zásah do komunikácie je potrebný počas testovania hostiteľských zariadení pri ich vývoji alebo analýze problému v prípade chyby.

Je nutné poznamenať, že jedným z cieľov tejto práce je skúmanie možností návrhu a realizácie prvého prototypu diagnostického nástroja a nie jeho finálna komerčná realizácia.

1.1 Definícia použitých pojmov

V práci budú používané pojmy, ktorých definícia je nasledovná:

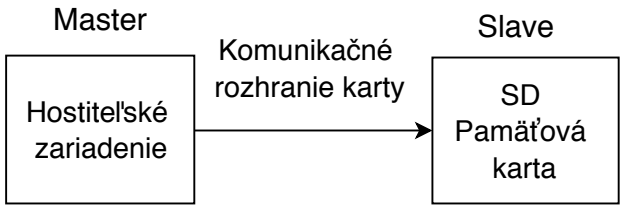
- **Hostiteľské zariadenie:** ako už z úvodu kapitoly vyplýva, jedná sa o zariadenie využívajúce pamäťovú kartu
- **Komunikačné rozhranie pamäťovej karty:** je rozhranie pomocou ktorého komunikuje hostiteľské zariadenie s pamäťovou kartou. Je možné ho rozdeliť na nasledujúce časti:
  - **Fyzická vrstva** definuje fyzické prepojenie (komunikačné signály) medzi hostiteľským zariadením a pamäťovou kartou
  - **SD komunikačný protokol** je súbor pravidiel a štandardov, pomocou ktorých prebiehajú všetky operácie medzi hostiteľským zariadením a pamäťovou kartou. SD komunikačný protokol musí byť implementovaný na

oboch stranách.

- **Diagnostické zariadenie:** je určené na samotnú diagnostiku a spracovávanie prenášaných správ od hostiteľského zariadenia pamäťovej karte
- **Užívateľské rozhranie:** vytvára rozhranie medzi diagnostickým zariadením a užívateľom
- **Diagnostický nástroj:** sa skladá z diagnostického zariadenia a užívateľského rozhrania

1.2 Základná koncepcia SD pamäťových kariet

Na obrázku 1.1 je zobrazená bloková schéma popisujúca štandardnú koncepciu pamäťovej karty a hostiteľského zariadenia. Komunikácia medzi hostiteľským zariadením a pamäťovou kartou je realizovaná pomocou komunikačného rozhrania.



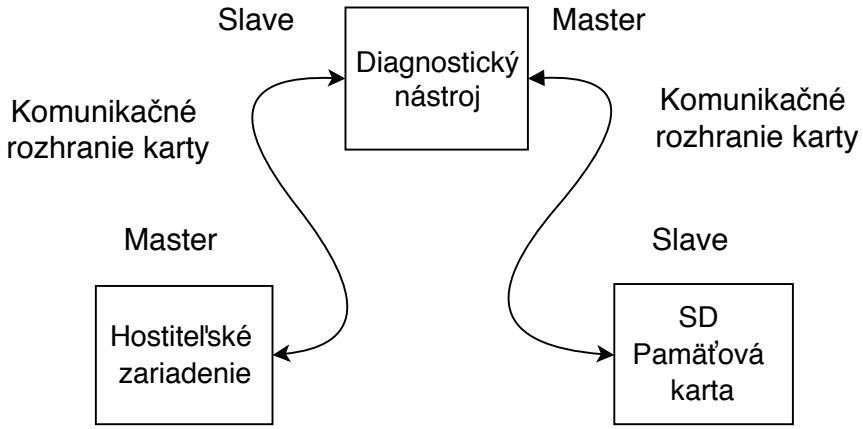
Obr. 1.1: Základná koncepcia pamäťovej karty

Základná koncepcia SD pamäťových kariet je založená na Master/Slave modely. Hostiteľské zariadenie (Master) je na nadradenej vrstve. Riadi komunikáciu a má plnú kontrolu nad pamäťovou kartou (Slave) alebo viacerými kartami. Každý prenos je iniciovaný zo strany hostiteľského zariadenia cez komunikačné rozhranie pamäťovej karty. Základnú koncepciu bude nutné modifikovať tak, aby diagnostický nástroj získal prístup k prenášaným správam.

1.3 Koncepcia SD pamäťových kariet s diagnostikou

Cieľom diagnostického nástroja bude diagnostikovať, respektíve monitorovať komunikáciu medzi hostiteľským zariadením a pamäťovou kartou a aktívne do nej zasahovať podľa požiadaviek užívateľa. Pretože diagnostický nástroj bude do komunikácie zasahovať, bude vyžadovať plnú kontrolu nad prenášanými správami. Bude vložený

medzi hostiteľské zariadenie a pamäťovú kartu. Užívateľ, ktorého cieľom bude diagnostikovať komunikáciu, vloží do hostiteľského zariadenia diagnostický nástroj, čím vytvorí základnú koncepciu zobrazenú na obrázku 1.2.



Obr. 1.2: Základná koncepcia pamäťovej karty s diagnostikou

Diagnostický nástroj bude v úlohe Slave pre hostiteľské zariadenie a zároveň v úlohe Master pre pamäťovú kartu.

1.4 Požiadavky zadávateľa na diagnostický nástroj

Počas diskusií so zadávateľom práce vznikali požiadavky, ktoré by mali byť v rámci realizácie diagnostického nástroja splnené. Úlohou tejto kapitoly je definovať požiadavky zadávateľa na hardware a software diagnostického nástroja.

1.4.1 Požiadavky na hardware

Jednou z požiadaviek zadávateľa na prototyp diagnostického nástroja je, aby bol realizovaný na voľne dostupnej hardwarovej platforme bez vývoja vlastného hardwaru. Táto požiadavka vyplýva z faktu, že ide o návrh a realizáciu prvého prototypu zariadenia a vôbec o možnosti realizácie diagnostického nástroja na volne dostupnej hardwarovej platforme. Vývoj špecifického hardwaru bude až ďalšou fázou tohto projektu v prípade, že sa zadávateľ rozhodne vo vývoji diagnostického nástroja pokračovať.

1.4.2 Funkcionálne požiadavky

V rámci rozboru zadania práce bolo potrebné definovať funkcionálne požiadavky, ktoré budú kladené na diagnostický nástroj. Prvou požiadavkou na diagnostický nástroj ako celok je prezentácia dát z diagnostiky pomocou určitého užívateľského rozhrania. Užívateľ by mal byť schopný diagnostický nástroj pomocou nadradeného zariadenia konfigurovať. Pod konfiguráciou sa rozumie definícia aktívneho zásahu diagnostického nástroja do komunikácie.

Z predošlej požiadavky vyplýva ďalšia požiadavka zahŕňajúca definíciu možných zásahov do komunikácie. Na základe diskusie so zadávateľom práce je možné predpokladať nasledujúce zásahy diagnostickým nástrojom do komunikácie medzi hostiteľským zariadením a pamäťovou kartou:

- **Odmietnutie príkazu:** užívateľ bude schopný definovať príkazy, ktoré diagnostický nástroj odmietne a neodošle pamäťovej karte. Primárne sa jedná o príkazy zápisu a čítania dátových blokov pamäťovej karty.
- **Odpoveď hostiteľskému zariadeniu:** užívateľ bude mať možnosť definovať odpovede, ktorými bude diagnostický nástroj hostiteľskému zariadeniu odpovedať - diagnostický nástroj bude odosielať hostiteľskému zariadeniu užívateľom požadované falošné odpovede pamäťovej karty.

2 SD pamäťové karty

Snahu ukladať dáta na rôzne dátové úložiská je možné sledovať už od počiatku informačných a počítačových technológií. Z hľadiska histórie prešiel vývoj dátových úložísk od jednoduchého magnetického princípu cez rôzne polovodičové úložiská až po ukladanie dát na servery a serverové riešenia. Do časti polovodičových úložísk sa nepochybne radia aj SD pamäťové karty, ktoré dnes často tvoria súčasť prenosných zariadení ako sú mobilné telefóny, tablety a podobne. V neposlednom rade sa pamäťové karty používajú aj v rôznych vstavaných zariadeniach, vývojových platformách a jedno-doskových počítačoch. V poslednej dobe sú pamäťové karty nahrádzané polovodičovými pamäťami zabudovanými priamo v zariadení, ale aj napriek tejto snahe tvoria pamäťové karty veľké percento dátových úložísk.

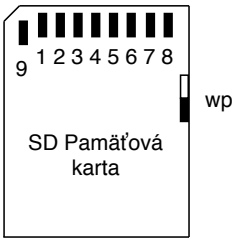
Hlavným cieľom tejto kapitoly je popísať základnú koncepciu pamäťových kariet, fyzickú vrstvu a SD komunikačný protokol použitý na komunikáciu s hostiteľským zariadením. Štandardy definujúce pamäťové karty zo spomenutých hľadísk sú dané a podliehajú pod organizáciu s názvom SDA (SD Card Association), do ktorej patria výrobcovia Panasonic, SanDisk a Toshiba. Špecifikácie sú dostupné na stránke `www.sdcard.org`, z ktorej sú aj voľne prevzaté a použité ako hlavný zdroj informácií v tejto práci [1].

2.1 Fyzické rozhranie SD pamäťových kariet

Fyzické rozhranie pamäťových kariet je popísané pomocou pamäťovej karty na obrázku 2.1 a jej pinov. Význam pinov je zobrazený a popísaný v tabuľke 2.1, ktorá je rozdelená podľa komunikačného módu pamäťovej karty.

Výber komunikačného módu ovplyvňuje fyzické rozhranie pamäťovej karty a šírku komunikačnej zbernice. Špecifikácia SD pamäťových kariet definuje SD a SPI (Serial Peripheral Interface) komunikačné módy. SD mód môže využívať až štyri dátové linky, po ktorých sú prenášané dáta. Linka CMD je určená výhradne na prenášanie príkazov. SD mód môže využívať na prenos dát všetky dátové linky alebo len jednu.

Komunikácia v SPI móde využíva SPI protokol. Dáta sú prenášané po dvoch dátových linkách, pričom jedna slúži na prijímanie a druhá na odosielanie dát. Komunikácia v obidvoch módoch je synchronizovaná pomocou CLK hodinového signálu.



Obr. 2.1: SD pamäťová karta

	SD Mód			SPI Mód		
Pin	**Názov**	**Typ**	**Funkcia**	**Názov**	**Typ**	**Funkcia**
1	DAT3		Dátová linka 3	CS		Chip select
2	CMD		Príkaz	DI		Vstupné dáta
3	VSS1		GND	VSS		GND
4	VDD		+3.3 V	VDD		+3.3 V
5	CLK		Hodinový signál	SCLK		Hodinový signál
6	VSS2		GND	VSS2		GND
7	DAT0		Dátová linka 0	DO		Výstupné dáta
8	DAT1		Dátová linka 1	-	-	Nepoužitý pin
9	DAT2		Dátová linka 2	-	-	Nepoužitý pin

Tab. 2.1: Fyzické rozhranie pamäťovej karty a jej piny

2.2 SD komunikačný protokol pamäťových kariet

Hlavným cieľom tejto kapitoly je popis základnej funkcionality SD komunikačného protokolu, pomocou ktorého je realizovaná komunikácia medzi hostiteľským zariadením a pamäťovou kartou. Detailný popis funkčnosti komunikácie a komunikačného protokolu zahŕňa množstvo častí, ktorých popis by bol nad rámec tejto práce a nie je potrebný, pretože je dostupný v pôvodnom dokumente vydaným organizáciou SDA [1]. Táto kapitola je zameraná na stručný úvod do SD komunikačného protokolu a popisuje len tie časti, na ktoré je potrebné sa pri návrhu a následnej realizácií prototypu sústrediť. Z tohto dôvodu je nutné popísať SD komunikačný protokol v rámci teoretického rozboru z hľadiska funkcionality a možných správ, ktoré budú počas komunikácie analyzované.

Komunikačný protokol je možné vo všeobecnosti definovať ako súbor pravidiel, pomocou ktorých sa riadia obidve strany komunikácie, v tomto prípade hostiteľské zariadenie a pamäťová karta. SD komunikačný protokol bol ako celok navrhnutý a

licencovaný spomenutou organizáciou SDA.

Základom SD komunikačného protokolu je súbor presne definovaných príkazov, odpovedí a dátových blokov.

- **Príkaz:** Každá operácia je inicializovaná príkazom, ktorý je odoslaný hostiteľským zariadením pamäťovej karte. Vo všeobecnosti existujú príkazy určené všetkým pamäťovým kartám pripojeným k hostiteľskému zariadeniu (broadcast) a príkazy, ktoré sú adresované len jednej konkrétnej karte (adresovateľné).
- **Odpoveď:** Pamäťová karta posiela hostiteľskému zariadeniu odpoveď, v ktorej je zahrnutý výsledok predošlého prijatého príkazu.
- **Dátový rámce:** Dátové rámce obsahujú dáta smerujúce od hostiteľského zariadenia do pamäťovej karty alebo opačne. Dátové rámce sú posielané obojsmerne, pretože karta podporuje operácie zápisu a operácie čítania dát.

Ako bolo spomenuté v kapitole 2.1, hostiteľské zariadenie môže s pamäťovou kartou komunikovať v SD alebo SPI komunikačnom móde. SD komunikačný mód je mód, pomocou ktorého pamäťová karta komunikuje primárne. To znamená, že po privedení napätia je karta v tomto móde. SPI komunikačný mód je mód, do ktorého je pamäťová karta uvedená pomocou špecifického postupu vykonaného počas jej inicializácie. Obidva komunikačné módy sú podmnožinou SD komunikačného protokolu, z čoho vyplýva, že hlavná funkcionalita komunikácie je rovnaká. Zmenou komunikačného módu sa mení fyzická vrstva. Rozdiel medzi SD a SPI komunikačným módom na úrovni fyzickej vrstvy je v komunikačnom rozhraní, pomocou ktorého sú prijímané alebo odosielané dáta. Zmena komunikačného módu ma za následok zmenu šírky komunikačnej zbernice, pretože SD mód využíva štyri dátové linky.

2.2.1 Príkazy hostiteľského zariadenia

Pamäťová karta je realizovaná ako stavový automat, ktorý čaká na spustenie určitej operácie zo strany hostiteľského zariadenia. Z tohto dôvodu je každá operácia iniciovaná špecifickým príkazom alebo sekvenciou príkazov, ktoré definujú o akú operáciu pôjde. Formát príkazov je rovnaký pre obidva komunikačné módy, preto je v rámci tejto podkapitoly popísaný detailnejšie.

Príkazy sa rozdeľujú do štyroch nasledujúcich skupín:

- **Broadcast príkaz bez odpovede:** je príkaz, ktorý je určený pre všetky pamäťové karty na komunikačnej zbernici bez akejkoľvek odpovede
- **Broadcast príkaz s odpoveďou:** je rovnako určený pre všetky pamäťové karty s tým rozdielom, že hostiteľské zariadenie očakáva odpoveď
- **Adresovaný príkaz:** je príkaz, ktorý je adresovaný konkrétnej pamäťovej

karte na komunikačnej zbernici. Počas inicializácie je možné pamäťovú kartu uviesť do adresovateľného režimu a získať adresu, pomocou ktorej ju hostiteľské zariadenie na komunikačnej zbernici identifikuje

- **Adresovaný príkaz s prenosom dát:** je určený pre príkazy, ktorých súčasťou sú aj dodatočné dáta pre pamäťovú kartu

SD komunikačný protokol definuje presný formát príkazu s dĺžkou 48 bitov. Formát príkazu je popísaný v nasledujúcej tabuľke, pričom symbol x znamená, že hodnota je závislá na konkrétnom príkaze.

Pozícia bitu	47	46	[45:40]	[39:8]	[7:1]	0
Počet bitov	1	1	6	32	7	1
Hodnota	0	1	x	x	x	1
Popis	Start bit	Smer prenosu	Index príkazu	Argument	CRC7	Stop bit

Tab. 2.2: Formát príkazu

Príkaz začína spúšťacím bitom (Start bit) signalizujúcim začiatok príkazového rámca tak, ako je popísaný v tabuľke 2.2. Spúšťací bit má vždy hodnotu 0. Následne je odoslaný bit, ktorý hovorí o smere prenosu. V prípade, že pamäťová karta odosiela dáta hostiteľskému zariadeniu, má tento bit hodnotu 0. Ak je však správa odosielaná z hostiteľského zariadenia pamäťovej karte, smer prenosu je nastavený na 1. Nakoľko sú príkazy posielané hostiteľským zariadením, je tento bit vždy nastavený na hodnotu 1.

Ďalšou časťou príkazu je takzvaný index. Veľkosť indexu je 6 bitov a jeho hodnota je odvodená od čísla v názve, ktoré nasleduje za skratkou CMD. To znamená, že ak hostiteľské zariadenie bude posielať napríklad CMD6, hodnota indexu bude 6. Niektoré z príkazov sú odosielané s dodatočnými dátami, ktoré sú nazvané ako argument. Veľkosť argumentu je stanovená na 32 bitov.

Prenos príkazov je chránený proti chybám, ktoré môžu nastať počas prenosu po zbernici, pomocou CRC (Cyclic redundnancy check). Ide o spôsob nazývaný ako kontrolný súčet, kde odosielateľ (hostiteľské zariadenie) dátového rámca vypočíta na základe odosielaných dát kontrolné číslo a vloží ho na koniec rámca. Prijímateľ (pamäťová karta) prevezme správu a na základe prijatých dát vypočíta rovnakým algoritmom číslo, ktoré porovná s prijatým kontrolným číslom. Ak sa prijaté a vypočítané číslo rovnajú, prijímateľ vyhodnotí prijaté dáta ako platné a vykoná potrebnú akciu (príkaz). Veľkosť kontrolného čísla je v tomto prípade stanovená na 7 bitov a preto je toto pole označené ako CRC7. Hodnota CRC je vypočítaná na základe

nasledujúceho polynómu:

$$G(x) = G(x) = x^7 + x^3 + 1 \tag{2.1}$$

Koniec príkazu je indikovaný ukončovacím bitom (Stop bit), ktorého hodnota je vždy 1.

2.2.2 Odpovede pamäťovej karty

SD komunikačný protokol definuje 5 rôznych odpovedí s rôznou dĺžkou a označením R1 až R7, s výnimkou odpovede R4 a R5. Tie sú určené pre pamäťové karty typu SDIO (Secure Digital Input Output) ktorými sa táto práca nezaoberá. Na rozdiel od príkazov, formát a význam odpovedí sa v závislosti na komunikačnom móde podstatne líši. Z tohto dôvodu je nutné popísať odpovede pre každý mód samostatne. Detailný popis odpovedí vrátane ich formátu je možné nájsť v špecifikácií SD komunikačného protokolu [1].

Formát odpovede v SD móde

Formát odpovede v SD komunikačnom móde, ktorý platí pre všetky definované odpovede (okrem R3), je popísaný v tabuľke 2.3. Pozícia bitov je popísaná všeobecne pomocou skratiek MSB (Most significant bit) a LSB (Least significant bit). MSB znamená najvýznamnejší bit odpovede a LSB zodpovedá najmenej významnému bitu, typicky bitu na pozícií 0. Veľkosť odpovedí sa líši v rozmedzí 46 až 136 bitov. Formát odpovedí použitých v SD komunikačnom móde je zobrazený v tabuľke 2.3, pričom znak x znamená, že sa informácia líši v závislosti na konkrétnej odpovedi.

Pozícia bitu	MSB	MSB-1	[MSB-2:MSB-7]	x	[LSB+7:LSB+1]	LSB
Počet bitov	1	1	6	x	7	1
Hodnota	0	0	x	x	x	1
Popis	Start bit	Smer prenosu	Index príkazu	Data odpovede	CRC7	Stop bit

Tab. 2.3: Formát odpovede v SD móde

Ako je vidno z tabuliek 2.2 a 2.3, popis častí odpovede a príkazu sú identické. Dáta odpovede pamäťovej karty sa menia v závislosti na príkaze obdržanom od hostiteľského zariadenia.

Formát odpovede v SPI móde

SPI komunikačný mód je jednoduchší ako SD mód, čo sa prejavuje aj vo formáte odpovedí odosielanými pamäťovou kartou. Odpovede majú v SPI móde veľkosť 8 až 40 bitov, čo je podstatne menej v porovnaní s SD módom. Odpoveď R1 je základom každej odpovede, ktorá sa vloží na pozíciu najvýznamnejšieho Bytu v odpoveďovom rámci. Veľkosť odpovede R1 je stanovená na 8 bitov a jej formát je uvedený v tabuľke 2.4.

Pozícia bitu	Popis
7	vždy 0
6	Chyba argumentu
5	Chyba adresy
4	Chyba mazacej sekvencie
3	Chyba CRC
2	Neplatný príkaz
1	Reset mazania
0	Idle stav

Tab. 2.4: Formát odpovede R1 v SPI móde

Bit číslo 7 má vždy hodnotu 0. Popis jednotlivých bitov je platný, ak je hodnota bitu nastavená na 1. Význam bitov v odpovedi R1 je nasledovný:

- **Chyba argumentu:** argument v prijatom príkaze nie je správny a bol mimo platný rozsah
- **Chyba adresy:** adresa, na ktorú majú byť zapísané dáta, je nesprávna
- **Chyba mazacej sekvencie:** počas mazania sekvencie nastala chyba
- **Chyba CRC:** CRC hodnota nie je správna
- **Neplatný príkaz:** prijatý príkaz bol neplatný
- **Reset mazania:** počas prijímania príkazov a sekvencie na mazanie bol prijatý nesprávny príkaz
- **Idle stav:** pamäťová karta je v stave Idle

Odpoveď v SPI móde je zložená z odpovede R1, ku ktorej sú pridané dáta špecifické pre odosielanú odpoveď. Ak pamäťová karta odpovedá pomocou R1, do odpoveďového rámca sa už nepridávajú žiadne dáta a odpoveď je odoslaná tak, ako to je uvedené v tabuľke 2.5.

Pozícia bitu	[MSB:MSB-7]	[MSB-8:LSB]]
Počet bitov	8	x
Hodnota	R1	x
Popis	Odpoveď R1 (2.4)	Dáta odpovede

Tab. 2.5: Formát odpovede v SPI móde

2.2.3 Dátové bloky

Prenos samotných dát medzi hostiteľským zariadením a pamäťovou kartou je založený na dátových blokoch, ktoré tvoria základnú prenosovú jednotku komunikácie v obidvoch komunikačných módoch. Hostiteľské zariadenie môže dĺžku dátového bloku nastaviť pomocou príkazu CMD16, pričom maximálna a zároveň predvolená hodnota je 512 bitov.

Dáta môžu byť prenášané dvomi spôsobmi. Hostiteľské zariadenie môže iniciovať operáciu čítania alebo zápisu dát s tým, že pamäťová karta odpovie len jedným dátovým blokom (Single block operation). Druhým spôsobom je odpoveď pamäťovej karty niekoľkými dátovými blokmi (Multiple block operation). Hostiteľské zariadenia využívajú zvyčajne druhý spôsob, pretože prenášané dáta sú obvykle väčšie a prenos po jednom bloku by bol neefektívny.

Každý odoslaný alebo prijatý dátový blok je ochránený proti chybe počas prenosu pomocou CRC o veľkosti 16 bitov. CRC hodnota je vložená na posledných 16 bitov dátového bloku a je vypočítaná pomocou nasledujúceho polynómu:

$$G(x) = x^{16} + x^{12} + x^5 + 1 \tag{2.2}$$

2.2.4 Riadiace príznaky

Riadiace príznaky sú špecifické dátové rámce, pomocou ktorých je riadený prenos dátových blokov v SPI komunikačnom móde a spolu s dátovými blokmi tvoria celý dátový rámec. SD komunikačný protokol definuje nasledujúce typy riadiacich príznakov:

- **Potvrdenie dátového bloku:** každý dátový blok je potvrdený pamäťovou kartou pomocou potvrdzovacieho príznaku, ktorý zahŕňa na bitoch 1 až 3 informáciu o akceptácií dátového bloku, odmietnutí kvôli nesprávnemu CRC alebo odmietnutí kvôli chybe počas zápisu.

7					0
x	x	x	0	Status	1

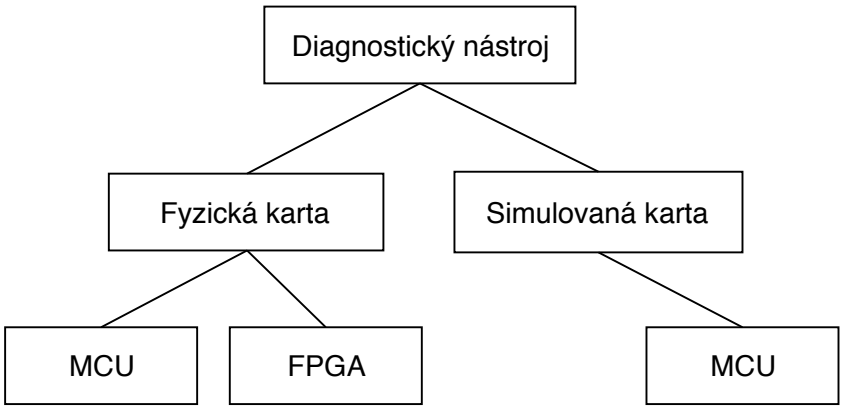
Tab. 2.6: Formát potvrdenia dátového bloku

- **Začiatok a koniec dátového bloku:** prenos dát je spojený s príznakmi, ktoré definujú začiatok a koniec dátového bloku. Príznak o začiatku je odoslaný ako prvý byte dátového bloku, po ktorom nasledujú prenášané dáta. Príznak o konci dátového bloku je odoslaný ako posledný byte, v prípade, že hostiteľské zariadenie žiada ukončenie komunikácie počas prenosu dátového rámca.
- **Chybné dáta:** ak nastala počas čítania dát chyba, pamäťová karta odošle namiesto platných dát chybový príznak.

3 Návrh konceptu diagnostického nástroja

V prvých fázach tejto práce vznikalo niekoľko možných konceptov diagnostického nástroja, ktoré sa po diskusiách so zadávateľom postupne formovali k reálnejšiemu návrhu. Súčasťou tejto kapitoly bude popis všetkých vzniknutých konceptov, pričom niektoré z nich budú vo výsledku nevyhovujúce, poprípade ponechané pre ďalší vývoj diagnostického nástroja.

Vzniknuté koncepty diagnostického nástroja sú zobrazené na obrázku 3.1, ktorý obsahuje pojmy MCU (Microcontroller unit) a FPGA (Field Programmable Gate Array). Význam týchto pojmov bude vysvetlený v samostatnej kapitole 3.1. Koncepty vychádzajú z predpokladu, že vložením diagnostického nástroja medzi hostiteľské zariadenie a pamäťovú kartu bude potrebné spĺňať náročné časové požiadavky dané hostiteľským zariadením.



Obr. 3.1: Koncepty diagnostického nástroja

Návrh diagnostického nástroja sa delí na základe pamäťovej karty na dva možné koncepty. Prvým konceptom je použitie fyzickej pamäťovej karty. Tento koncept plne reflektuje naznačenú blokovú schému na obrázku 1.2. Druhým konceptom je diagnostický nástroj, ktorý by využíval simulovanú pamäťovú kartu.

Koncepty diagnostického nástroja popísané v tejto kapitole používajú pojmy hardwarová platforma, nadradené zariadenie a rozhrania diagnostického nástroja, ktoré tvoria diagnostický nástroj ako celok. Pre dostatočné pochopenie popisu konceptov je potrebné definovať pojmy nasledovne:

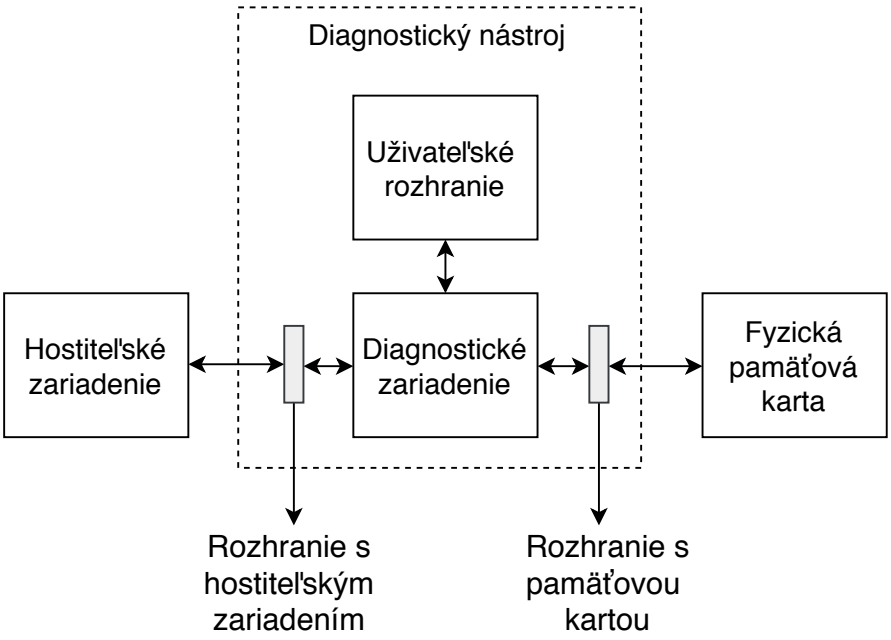
- **Hardwarová platforma:** tvorí blok diagnostického zariadenia, na ktorom bude bežať firmware. Realizácia založená na hardwarovej platforme vychádza z požiadaviek zadávateľa popísaných v kapitole 1.4. Pod pojmom hardwarová platforma sa rozumie hotové hardwarové riešenie, ktoré je pripravené na vývoj

prototypov zariadení. Požiadavky kladené na hardwarovú platformu a jej výber sú závislé na konkrétnom koncepte a budú diskutované neskôr

- **Rozhranie s hostiteľským zariadením:** tvorí rozhranie medzi hostiteľským a diagnostickým zariadením. Užívateľ vloží diagnostický nástroj do hostiteľského zariadenia na miesto pamäťovej karty. Z tohto dôvodu musí diagnostický nástroj disponovať rozhraním, ktoré bude na úrovni fyzickej vrstvy ekvivalentné pamäťovej karte
- **Rozhranie s pamäťovou kartou:** tvorí rozhranie medzi diagnostickým nástrojom a fyzickou pamäťovou kartou na úrovni fyzickej vrstvy

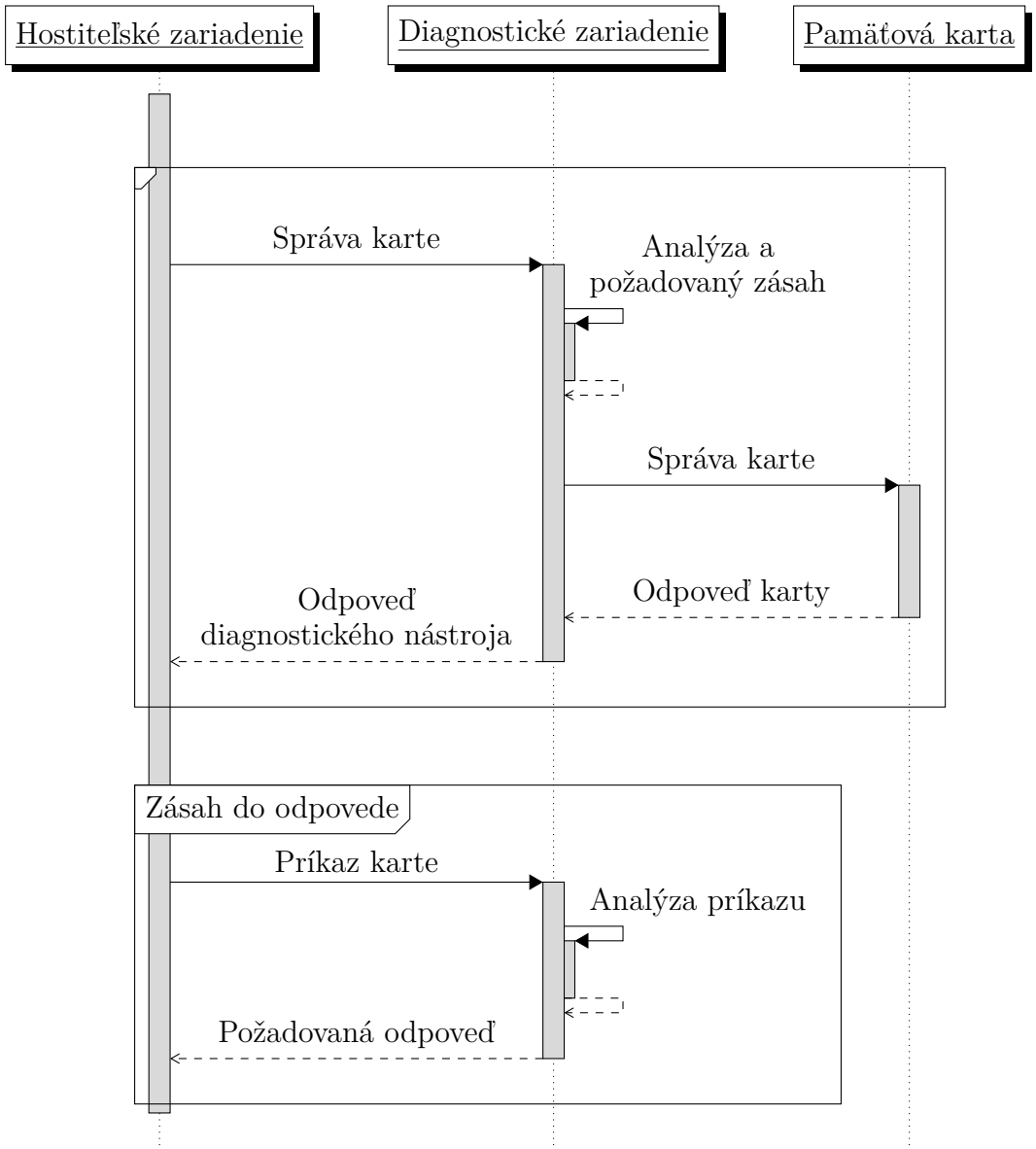
3.1 Diagnostický nástroj s fyzickou pamäťovou kartou

Koncept s fyzickou pamäťovou kartou vychádza z pôvodnej myšlienky diagnostického nástroja, ktorá bola naznačená v rozbore zadania na obrázku 1.2. Diagnostický nástroj s fyzickou pamäťovou kartou by bol vložený medzi hostiteľské zariadenie a pamäťovú kartu, čím by získal plnú kontrolu nad prenášanými správami a to z hľadiska diagnostiky a aktívneho zásahu zároveň. Bloková schéma popisujúca koncept diagnostického nástroja s fyzickou pamäťovou kartou je zobrazená na obrázku 3.2.



Obr. 3.2: Koncept diagnostického nástroja s fyzickou pamäťovou kartou

Sekvenčný diagram popisujúci prenášanie a prípadný zásah do správ je zobrazený na obrázku 3.3 na ktorom pojem správa karte môže byť vo všeobecnosti príkaz alebo dátový rámec. Diagnostické zariadenie by bez aktívneho zásahu preposielalo správy od hostiteľského zariadenia pamäťovej karte a preposielalo odpovede od karty hostiteľskému zariadeniu. V prípade zásahu by boli dáta správy zmenené. Ak by užívateľ vyžadoval falošné odpovede, diagnostické zariadenie by prerušilo komunikáciu s kartou a odosielalo požadované odpovede hostiteľskému zariadeniu.



Obr. 3.3: Sekvenčný diagram pre koncept s fyzickou pamäťovou kartou

Z diagramu je zrejmé, že firmware diagnostického zariadenia musí nad prenášanými správami vykonať analýzu, pomocou ktorej bude možné rozoznávať prenášané

správy a zasahovať do komunikácie.

3.1.1 Časová náročnosť konceptu

Dôležitou úvahou konceptu s fyzickou pamäťovou kartou je, že vložením diagnostického nástroja medzi hostiteľské zariadenie a pamäťovú kartu vzniká dôležitá požiadavka na časovú náročnosť riešenia.

Jedným zo spôsobov kontroly správnosti komunikácie na strane hostiteľského zariadenia je používanie časového limitu, do ktorého musí hostiteľské zariadenie obdržať odpoveď od pamäťovej karty. V prípade, že časový limit vyprší a hostiteľské zariadenie neobdrží odpoveď od pamäťovej karty, komunikácia sa vyhodnotí ako neplatná. Z tohto vyplýva, že diagnostické zariadenie musí vykonať operácie zobrazené na sekvenčnom diagrame na obrázku 3.3 v časovom limite stanovenom hostiteľským zariadením.

Na reprezentáciu časovej náročnosti konceptu je možné použiť napríklad definíciu chyby Command Timeout Error v SD komunikačnom móde, ktorá je popísaná v špecifikácii hostiteľského zariadenia a registru s názvom Error Interrupt Status Register. Chyba Command Timeout Error nastane v prípade, že hostiteľské zariadenie odoslalo príkaz a pamäťová karta neodpovedala do časového limitu stanoveného 64 periódami CLK signálu. Vzorec na výpočet časového limitu, po ktorom vyhodnotí hostiteľské zariadenie chybu Command Timeout Error, je nasledovný:

$$t = \frac{1}{f_{CLK}} 64 = \frac{64}{f_{CLK}} \qquad [\mathrm{s}] \tag{3.1}$$

V prípade SD komunikačného módu je uvažovaná maximálna frekvencia CLK signálu u SD pamäťových kariet 50 MHz v High Speed móde, čo je ekvivalent prenosovej rýchlosti 25 MB/s [2]. V tabuľke 3.1 sú vypočítané hodnoty časového limitu pre rôzne frekvencie signálu CLK, do ktorého by diagnostické zariadenie muselo vykonať všetky potrebné operácie.

$\mathbf{f_{CLK}}$ **[MHz]**	**t** [μ**s**]
50	1,28
25	2,56
10	6,4
5	12,8
1	64

Tab. 3.1: Príklad časových limitov pre chybu Command Timeout Error

Ak hostiteľské zariadenie využíva SPI komunikačný mód, definícia chyby Command Timeout Error neplatí, pretože SPI komunikačný mód využívajú jednoduché zariadenia, o ktorých špecifikácia hostiteľského zariadenia [2] nehovorí. Pre potreby tohto príkladu je možné uvažovať, že zariadenia využívajúce SPI komunikačný mód komunikujú na nižších rýchlostiach, sú vo vyhodnocovaní komunikačných chýb menej striktné a hodnota časových limitov sa pohybuje rádovo v stovkách ms.

Koncept s reálnou pamäťovou kartou má niekoľko výhod, ktoré vychádzajú z jeho hlavnej podstaty.

1. Vložením diagnostického nástroja medzi hostiteľské zariadenie a reálnu pamäťovú kartu by diagnostický nástroj získal plnú kontrolu nad prenášanými dátami.
2. Použitím fyzickej pamäťovej karty by boli prenášané dáta uložené na použitú pamäťovú kartu a diagnostické zariadenie by malo prístup k reálnym odpovediam od pamäťovej karty.
3. V prípade realizácie tohto konceptu by implementácia firmwaru diagnostického zariadenia nezahŕňala implementáciu celého SD komunikačného protokolu, ale len spomínanú analýzu nad prenášanými dátami.

Na základe výsledkov v tabuľke 3.1 je možné konštatovať, že časové nároky kladené na diagnostický nástroj s reálnou pamäťovou kartou sú vysoké a ich splnenie by bolo náročné, no stále realizovateľné a to pomocou dobre zvolenej hardwarovej platformy. Z tohto dôvodu sa prípadná realizácia hardwarovej platformy rozpadá na ďalšie riešenia. Prvým riešením je použitie MCU a v druhom prípade je možné na realizáciu diagnostického nástroja využiť FPGA.

3.1.2 Diagnostický nástroj založený na MCU

Prvou potencionálnou možnosťou realizácie konceptu diagnostického nástroja s fyzickou pamäťovou kartou je realizácia na hardwarovej platforme s jednoduchým MCU, napríklad na platforme Arduino. Hardwarové platformy založené na jednoduchom MCU používajú takt procesoru rádovo desiatky MHz a preto sú na realizáciu konceptu s reálnou pamäťovou kartou nepostačujúce z výkonového hľadiska. Nedostačujúci výkon MCU a celkovo vykonávania firmwaru diagnostického nástroja by nedokázal splniť požiadavky a časové limity popísané v kapitole 3.1.

Z tohto dôvodu je v prípade realizácie tohto konceptu nutné zvážiť výber hardwarovej platformy. Výber platformy by mal byť zameraný na platformy oveľa vyššej triedy, ako spomínaná platforma Arduino. Návrh vhodnej hardwarovej platformy je častokrát iteratívny proces, ktorého požiadavky sú splnené až po niekoľkých iteráciách a výkonové požiadavky MCU je možné na začiatku len odhadovať.

Ďalšou dôležitou úvahou v prípade realizácie konceptu na hardwarovej platforme založenej na MCU sú dostupné periférie. MCU na aktuálnom trhu disponujú perifériou schopnou komunikovať s pamäťovou kartou ako hostiteľské zariadenie, no nedisponujú perifériou schopnou prijímať dáta od hostiteľského zariadenia v SD komunikačnom móde. Jediným riešením by bolo využitie vstupno - výstupných pinov MCU na prijímanie dát v SD komunikačnom móde. Toto riešenie by však zvýšilo zložitosť firmwaru a tým aj časové nároky, ktoré je potrebné v rámci tohto konceptu držať na čo najnižšej úrovni. Ak by sa realizácia prototypu diagnostického nástroja sústredila iba na SPI komunikačný mód, diagnostický nástroj by využíval SPI perifériu, ktorá je dnes dostupná na väčšine MCU na trhu.

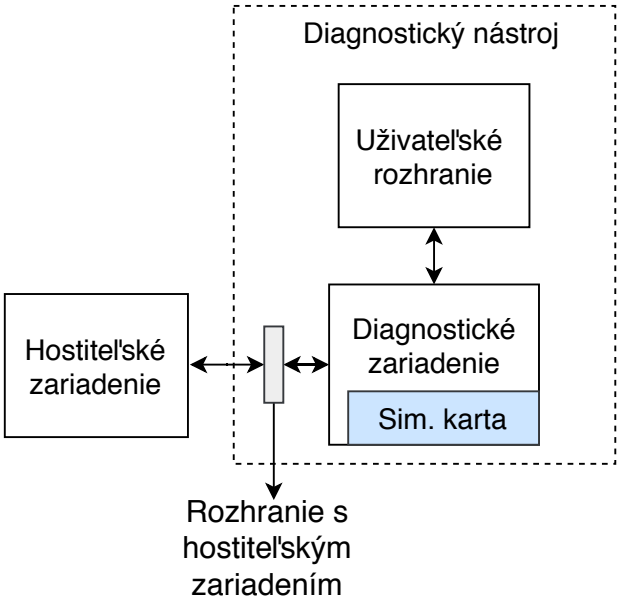
3.1.3 Diagnostický nástroj založený na FPGA

Diagnostický nástroj s fyzickou pamäťovou kartou by v druhom prípade mohol byť realizovaný na hardwarovej platforme založenej na FPGA. Sú to programovateľné logické hradlové polia, ktoré sú v porovnaní s MCU niekoľkonásobné rýchlejšie. Výhod v porovnaní s MCU je niekoľko, no tým najhlavnejším rozdielom je realizácia firmwaru na hardwarovej úrovní, to znamená na úrovni logických hradiel. FPGA je schopné pracovať s taktom rádovo stovky MHz, čím je všeobecne z hľadiska rýchlosti a diagnostiky komunikácií v porovnaní s riešením na MCU preferovanejšie.

V prípade výberu hardwarovej platformy založenej na FPGA by bolo možné realizovať chýbajúcu periférium z MCU, ktorá by bola schopná prijímať dáta od hostiteľského zariadenia a správať sa ako pamäťová karta. Je možné predpokladať, že použitím FPGA by problém s časovými požiadavkami nenastal.

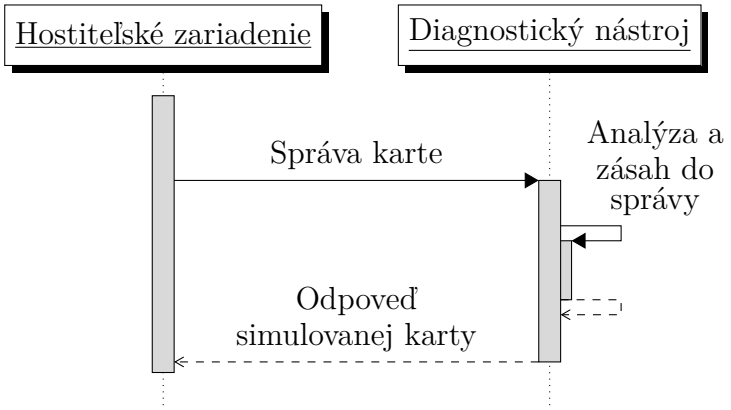
3.2 Diagnostický nástroj so simulovanou pamäťovou kartou

Druhou možnosťou realizácie diagnostického nástroja je koncept so simulovanou pamäťovou kartou, ktorý vychádza z predchádzajúceho hardwarového konceptu z kapitoly 3.1. Ako už z názvu konceptu vyplýva, fyzická pamäťová karta by bola nahradená simuláciou pamäťovej karty implementovanou vo firmwari diagnostického nástroja. Cieľom tohto konceptu je znížiť časovú náročnosť firmwaru vypustením komunikácie diagnostického nástroja s fyzickou pamäťovou kartou a tým získať čas na samotnú diagnostiku komunikácie zo strany hostiteľského zariadenia. V porovnaní s konceptom s fyzickou pamäťovou kartou je zrejmé, že hlavným rozdielom je vypustenie fyzickej pamäťovej karty zo základnej koncepcie zobrazenej na obrázku 3.4 a teda aj rozhrania s pamäťovou kartou.



Obr. 3.4: Koncept diagnostického nástroja so simulovanou pamäťovou kartou

Sekvenčný diagram popisujúci tento koncept je zobrazený na obrázku 3.5 a reflektuje základnú koncepciu so simulovanou pamäťovou kartou.



Obr. 3.5: Sekvenčný diagram pre koncept so simulovanou pamäťovou kartou

Simulácia pamäťovej karty by zahŕňala implementáciu celého SD komunikačného protokolu a správanie fyzickej pamäťovej karty. Firmware diagnostického nástroja je počas behu uložený v RAM (Random Access Memory) pamäti. Pamäte RAM sú požívané na úplne iný účel a ich kapacity sa u MCU pohybujú v niekoľkých KB. Z tohto dôvodu by simulovaná pamäťová karta mala podstatne nižšiu kapacitu a v prípade diagnostiky by nebolo možné pracovať s veľkým množstvom prenášaných dát. Koncept so simulovanou pamäťovou kartou by mal byť realizovateľný na hardwarovej platforme založenej na MCU.

Druhou možnosťou ukladania dát simulovanej pamäťovej karty je pamäť Flash. Veľkosti tejto pamäte sa u MCU pohybujú v jednotkách MB. Týmto spôsobom by sa kapacita simulovanej pamäťovej karty až tak nezväčšila a tým pádom sa jedná o rovnakú nevýhodu, ako pri ukladaní dát do pamäte RAM.

3.3 Výber konceptu diagnostického nástroja

V tejto kapitole boli prezentované dva základné koncepty diagnostického nástroja. Ďalej je potrebné vybrať jeden z nich, na ktorý sa bude návrh a následná realizácia sústrediť. Dôležitým faktorom vo výbere konceptu je zadávateľ práce, ktorému boli predložené popísané koncepty diagnostického nástroja. Po diskusiách so zadávateľom práce bol zvolený koncept diagnostického nástroja s reálnou pamäťovou kartou založený na MCU. Dôvody, ktoré viedli k tomuto rozhodnutiu, sú nasledovné:

1. Koncept s reálnou pamäťovou kartou je pre zadávateľa najbližší, pretože plne reflektuje pôvodnú myšlienku zadania a v porovnaní so simulovanou pamäťovou kartou umožňuje používanie reálnej pamäťovej karty v plnom rozsahu.
2. Ako bolo spomenuté v kapitole 3.2, simulácia pamäťovej karty by zahŕňala simuláciu správania sa fyzickej pamäťovej karty, čo je v porovnaní s konceptom s reálnou pamäťovou kartou z implementačného a časového hľadiska náročnejšie.
3. Obmedzenia kapacity simulovanej pamäťovej karty, ktoré vyplývajú z jej konceptu, zadávateľ neakceptuje, pretože vyžaduje kapacitu porovnateľnú s pamäťovými kartami na trhu a prístup k dátam, ktoré sú počas diagnostiky prenášané.
4. Zadávateľ je ochotný tolerovať obmedzenia vyplývajúce z konceptu s reálnou pamäťovou kartou založenom na MCU, pretože ide o prvý prototyp diagnostického nástroja a preferuje realizáciu na MCU.

Výberom hardwarového konceptu sa bude práca ďalej sústrediť na návrh a realizáciu diagnostického nástroja s fyzickou pamäťovou kartou založeného na MCU. Spolu so zvoleným hardwarovým konceptom bol vybratý aj SPI komunikačný mód, na ktorý bude diagnostický nástroj určený. V prípade, že by sa preukázalo, že je toto riešenie nepostačujúce, použije sa koncept založený na FPGA.

3.4 Koncepty užívateľského rozhrania

Užívateľské rozhranie, ako už zo samotného názvu vyplýva, by malo z praktického hľadiska čo najviac vyhovovať užívateľovi. Pomocou neho by mal byť užívateľ schopný konfigurovať diagnostické zariadenie a zobrazovať prenášané dáta medzi

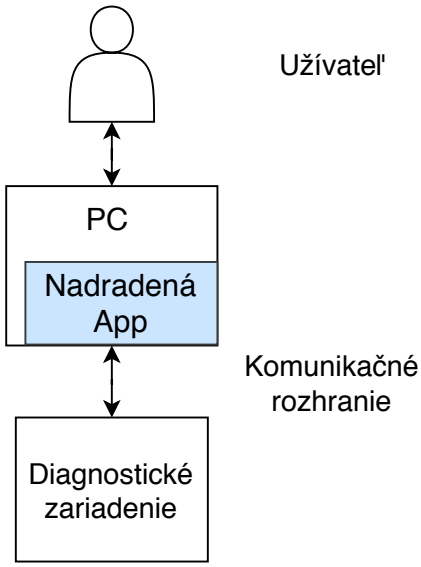
hostiteľským zariadením a pamäťovou kartou. Návrh užívateľského rozhrania vychádza z dvoch možností.

3.4.1 Užívateľské rozhranie s displejom

Prvou možnosťou je displej, ktorý by bol pripojený k diagnostickému zariadeniu. V tomto prípade by diagnostické zariadenie muselo obsahovať ďalšie časti firmwaru, ktorých úlohou by bolo riadenie displeja, zobrazovanie dát a celková interakcia s užívateľom. Displeje, ktoré je možné pripojiť k rôznym vstavaným zariadeniam sú zvyčajne použité na jednoduché aplikácie a neposkytujú dostatočnú veľkosť. Ak má užívateľské rozhranie zobrazovať prenášané správy, je možné predpokladať ich veľké množstvo. V kombinácií s ďalšou funkcionalitou užívateľského rozhrania je toto riešenie pre užívateľa nepraktické. Preto vznikol koncept s nadradenou aplikáciou.

3.4.2 Užívateľské rozhranie s nadradenou aplikáciou

Koncept s nadradenou aplikáciou je založený na komunikácií s diagnostickým zariadením a odosielaní dát na nadradenú vrstvu - nadradenú aplikáciu. Nadradená aplikácia by bežala na PC, pomocou ktorého by užívateľ konfiguroval a riadil diagnostiku a videl prenášané správy.



Obr. 3.6: Koncept užívateľského rozhrania s nadradenou aplikáciou

Nadradená aplikácia na PC bude umožňovať zobrazovanie veľkého počtu správ a ponúka viac možností rozšírenia ako displej. Výhodou je, že dáta z diagnostiky

budú dostupné priamo na PC a užívateľ bude schopný s dátami akokoľvek manipulovať. Ak by bol použitý koncept s displejom, bolo by nutné vyriešiť komplikovanejší prenos dát z diagnostiky do PC na ich ďalšie spracovanie.

Jedinou nevýhodou v porovnaní s predošlým konceptom je potreba PC počas používania nástroja. Užívateľ však bude mať počas práce s diagnostickým nástrojom dostupný PC. Užívateľské rozhranie bude z tohto dôvodu realizované na základe tohto konceptu, pričom nadradená aplikácia bude bežať na PC s operačným systémom Windows.

3.4.3 Komunikačné rozhranie nadradenej aplikácie

Výber komunikačného rozhrania medzi PC a diagnostickým zariadením je závislý na rýchlosti, ktorou budú dáta odosielané z diagnostického zariadenia do nadradenej aplikácie. Ak by bolo diagnostické zariadenie schopné všetky prenášané správy medzi hostiteľským zariadením a pamäťovou kartou odosielať okamžite na nadradenú vrstvu, rýchlosť prenosu medzi diagnostickým zariadením a nadradenou aplikáciou by bola ekvivalentná rýchlosti prenosu medzi hostiteľským zariadením a pamäťovou kartou. Ak by hostiteľské zariadenie komunikovalo s pamäťovou kartou v High Speed a SD komunikačnom móde, rýchlosť by bola približne 25 MB/s [2]. Komunikačné rozhrania PC pripadajúce v úvahu sú USB a Ethernet. Rýchlosť USB verzie 2.0 v High Speed móde je teoreticky 60 MB/s, čo je pre potreby diagnostického nástroja s dostatočnou rezervou postačujúce [3]. Použitím Ethernetu by bolo potrebné vyriešiť všetky sieťové požiadavky a vznikla by potreba lokálnej siete pre funkčnosť diagnostického nástroja. Z tohto dôvodu bolo ako komunikačné rozhranie medzi diagnostickým zariadením a PC vybraté USB.

4 Hardware diagnostického nástroja

Úlohou tejto kapitoly je popis hardwarového návrhu diagnostického nástroja, ktorý sa delí na dve časti. Prvou je hardware diagnostického zariadenia, ktorý bude popísaný v nasledujúcich kapitolách. Druhou časťou je hardware, na ktorom bude realizované užívateľské rozhranie a na ktorom bude bežať nadradená aplikácia.

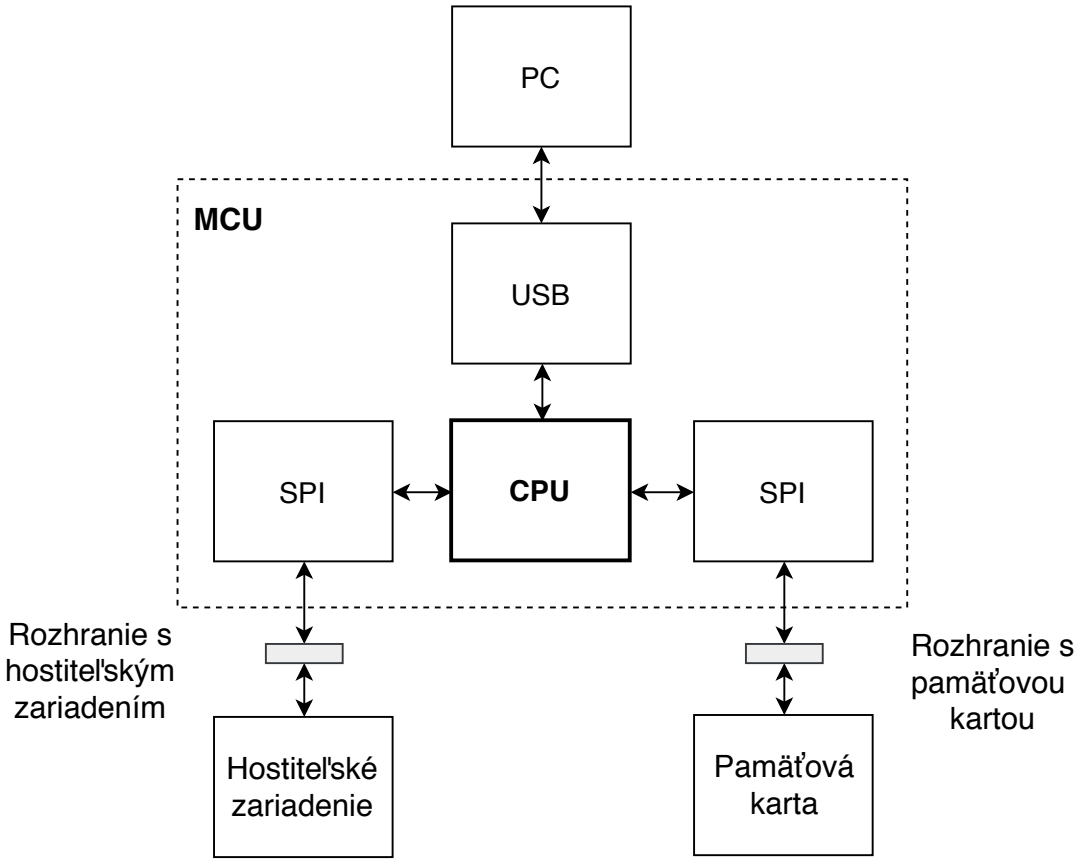
V návrhu užívateľského rozhrania v kapitole 3.4 bolo definované, že sa bude jednať o PC. Z tohto dôvodu nie je potrebné popisovať jeho hardware a kapitola sa bude sústrediť na hardware diagnostického zariadenia.

4.1 Návrh hardwaru diagnostického zariadenia

Hardware diagnostického zariadenia je jasne daný požiadavkou zadávateľa, ktorá vyžaduje jeho realizáciu na hardwarovej platforme. Hardwarová platforma bude založená na MCU, na ktorý sú výberom konceptu diagnostického nástroja kladené hardwarové požiadavky popísané v tejto kapitole.

Okrem hardwarovej platformy je nutné pri návrhu hardwaru diagnostického zariadenia uvažovať aj rozhrania medzi hostiteľským zariadením a diagnostickým nástrojom a diagnostickým nástrojom a pamäťovou kartou. Koncept diagnostického nástroja vyžaduje hardwarové časti zobrazené na blokovom diagrame na obrázku 4.1. Základom hardwaru diagnostického zariadenia je MCU, na ktorom bude bežať firmware a ktorý musí disponovať perifériami vyplývajúcimi zo zvoleného konceptu.

Prvou perifériou MCU vyplývajúcou z návrhu užívateľského rozhrania v kapitole 3.4, ktorá je zobrazená na obrázku 4.1, je USB. Úlohou USB bude komunikovať s nadradenou aplikáciou pomocou USB zbernice. Návrh užívateľského rozhrania ráta s USB High Speed verzie 2.0, ktoré podporujú dostupné MCU na aktuálnom trhu. Koncept s reálnou pamäťovou kartou založenom na MCU je určený len pre SPI komunikačný mód pamäťovej karty. Z tohto dôvodu je nutné, aby MCU disponoval minimálne dvomi SPI perifériami, ktorých úlohou bude komunikovať s hostiteľským zariadením a pamäťovou kartou na úrovni diagnostického zariadenia.



Obr. 4.1: Blokový diagram hardwaru diagnostického zariadenia

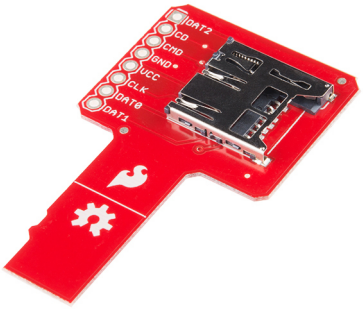
Keďže je ťažké predpokladať a odhadovať požadovaný výkon MCU v tejto fáze, je na začiatok nutné stanoviť typ a rodinu MCU, na ktorú sa bude návrh zameriavať. Výber hardwarovej platformy pre diagnostický nástroj sa bude sústrediť na MCU založenom na Cortex - M4 jadre s čo najvyšším taktom CPU, čo znamená od 100 MHz.

4.1.1 Rozhrania diagnostického zariadenia

Výberom hardwarového konceptu vznikajú rozhrania diagnostického zariadenia s hostiteľským zariadením a pamäťovou kartou. Diagnostické zariadenie bude vložené do hostiteľského zariadenia namiesto pamäťovej karty. Preto je potrebné navrhnúť hardwarové rozhranie s hostiteľským zariadením, ktoré bude z fyzického hľadiska identické s pamäťovou kartou a pomocou ktorého budú sprístupnené komunikačné signály hostiteľského zariadenia.

Jednou z možností pre rozhranie s hostiteľským zariadením je návrh malého plošného spoja, ktorý by presne kopíroval rozmery pamäťovej karty a fyzickú vrstvu

zobrazenú na obrázku 2.1. Z plošného spoja by boli vyvedené vodiče, ktoré by viedli na SPI piny MCU. Po preskúmaní dostupných riešení na trhu bolo rozhodnuté, že nie je potrebné navrhovať vlastný plošný spoj. Rozhranie s hostiteľským zariadením bude realizované pomocou takzvaného snifferu pamäťových kariet vyrábaného firmou SparkFun Electronics.



Obr. 4.2: Sniffer pamäťových kariet TOL-09419 od firmy SparkFun [4]

Primárnym účelom snifferu je sledovanie signálov medzi pamäťovou kartou a hostiteľským zariadením pomocou osciloskopu alebo logického analyzátoru. V rámci diagnostického nástroja budú signály zo snifferu privedené na SPI piny MCU, čím vznikne komunikačné rozhranie s hostiteľským zariadením.

Rozhranie medzi diagnostickým zariadením a pamäťovou kartou je možné realizovať správnym výberom hardwarovej platformy. Niektoré hardwarové platformy poskytujú rozhranie s pamäťovou kartou priamo na doske. Ak bude hardwarová platforma poskytovať rozhranie s pamäťovou kartou, diagnostické zariadenie už nebude potrebovať žiadny prídavný hardware okrem spomínaného snifferu.

4.1.2 Výber hardwarovej platformy

Požiadavky na hardwarovú platformu vyplývajúce zo zvoleného konceptu a návrhu hardwaru diagnostického zariadenia sú nasledovné:

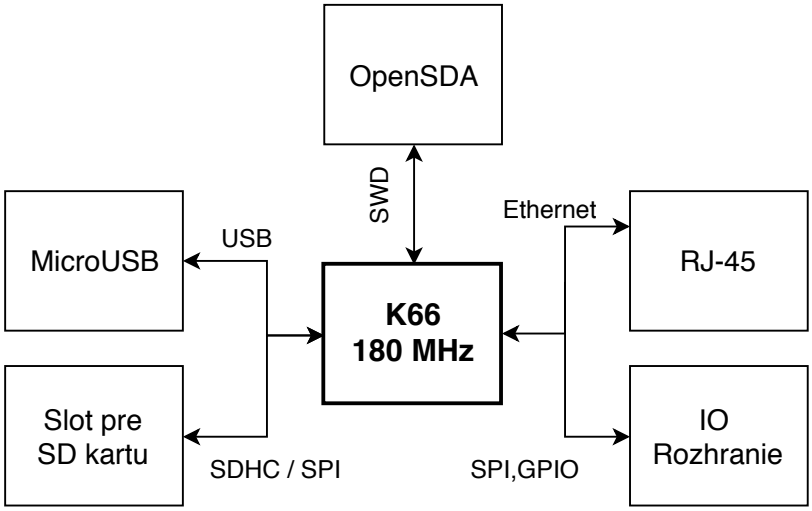
- MCU založený na jadre **Cortex-M4**
- **USB** periféria podporujúca High Speed verziu 2.0
- Dve **SPI** periférie
- Fyzické rozhranie s pamäťovou kartou
- Rozhranie na ladenie firmwaru

Rozhranie na ladenie firmwaru niektorí výrobcovia realizujú na hardwarovej platforme ako samostatnú hardwarovú časť, ktorej úlohou je ladenie firmwaru v MCU

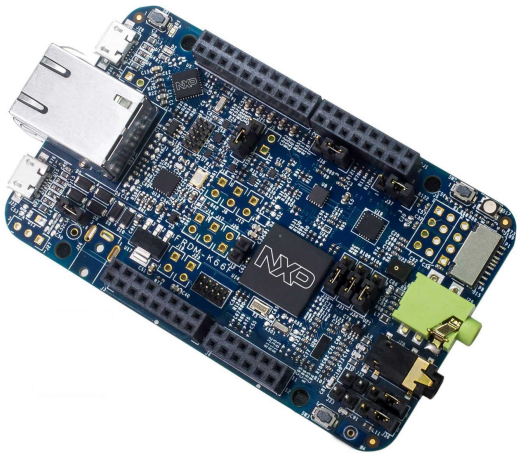
bez akéhokoľvek dodatočného nástroja. Toto riešenie je výhodné, pretože počas vývoja firmwaru a prototypu zariadenia je možné sledovať a ladiť jeho beh bez potreby iných ladiacich nástrojov ako je J-Link a podobne.

Preferovanými výrobcami hardwarových platforiem boli firmy Texas Instruments a NXP. Firma Texas Instruments poskytuje platformy, ktoré vyhovujú všetkým stanoveným požiadavkám až na fyzické rozhranie s pamäťovou kartou. Hardwarové platformy od firmy Texas Instruments, disponujúce fyzickým rozhraním s pamäťovou kartou neposkytujú dostatočný výkon a zvyčajne sú založené na mennej výkonných MCU. Z tohto dôvodu sa výber hardwarovej platformy sústredil na firmu NXP.

Po prieskume produktov firmy NXP bola vybratá platforma s označením FRDM-K66F, ktorá disponuje MCU Kinetis K66 založenom na jadre Cortex - M4 a taktom až 180 MHz. Ďalej vyhovuje požiadavkám na spomínané periférie. Ladenie a nahrávanie firmwaru do MCU je možné pomocou OpenSDA technológie, ktorá podporuje komunikáciu s vývojovým prostredím cez USB. Hardwarová platforma FRDM-K66F navyše disponuje aj rozhraním Ethernet, ktorú by bolo možné použiť v prípade, že by diagnostický nástroj v budúcnosti podporoval komunikáciu po lokálnej sieti [5]. Zjednodušený blokový diagram platformy FRDM-K66F založený na užívateľskej príručke [5] je zobrazený na obrázku 4.3.



Obr. 4.3: Blokový diagram platformy FRDM-K66F



Obr. 4.4: Platforma FRDM-K66F [8]

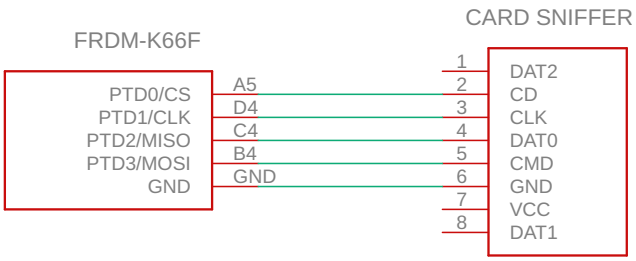
4.1.3 Prepojenie hardwarových častí diagnostického zariadenia

Hardwarové časti diagnostického zariadenia sa v podstate skladajú z rozhrania s hostiteľským zariadením, to znamená snifferu, a zo samotnej hardwarovej platformy. Ostatné hardwarové časti sú realizované na doske hardwarovej platformy od výrobcu, preto nie je potrebné pripájať ďalšie hardwarové moduly. Využitie pinov platformy FRDM-K66F na privedenie SPI signálov od hostiteľského zariadenia pomocou snifferu je zobrazené v tabuľke 4.1. Signály do hostiteľského zariadenia (piny snifferu) sú označené na základe signálov z SD módu. Tabuľka preto obsahuje stĺpec popisujúci SPI signál ku každému signálu do hostiteľského zariadenia sprístupneného pomocou snifferu.

Pin platformy	**Pin do host zariadenia (sniffer)**	**Popis**	**SPI označenie**
PTD0	CD	Chip Select	CS
PTD1	CLK	Hodinový signál	CLK
PTD2	DAT0	Výstupné dáta	MISO
PTD3	CMD	Vstupné dáta	MOSI
GND	GND	GND	GND

Tab. 4.1: Prepojenie pinov platformy a snifferu

Schéma zapojenia hardwarových častí je symbolicky zobrazená na obrázku 4.5. Hardwarová platforma disponuje niekoľkými ďalšími pinmi, ktoré nebudú využité a preto sa v schéme ani nevyskytujú.



Obr. 4.5: Schéma zapojenia platformy FRDM-K66F a snifferu

5 Software diagnostického nástroja

Software diagnostického nástroja je možné rozdeliť na dve časti. Prvou je firmware diagnostického zariadenia, ktorý bude bežať na hardwarovej platforme. Druhou časťou je nadradená aplikácia, ktorá bude vytvárať užívateľské rozhranie. V prvej časti je potrebné definovať softwarovú štruktúru diagnostického nástroja. Následne bude popísaný firmware diagnostického zariadenia a nadradená aplikácia bežiaca na PC.

5.1 Štruktúra softwaru diagnostického nástroja

Štruktúra softwaru diagnostického nástroja ako celku je zobrazená na obrázku 5.1.

Obr. 5.1: Štruktúra softwaru diagnostického nástroja

Fyzická vrstva definuje fyzické rozhranie použité na komunikáciu medzi hostiteľským zariadením a diagnostickým zariadením a medzi diagnostickým zariadením a pamäťovou kartou. Fyzická vrstva diagnostického zariadenia je závislá na komunikačnom móde, na ktorý je zariadenie určené. Návrh by mal rátať s prípadnou zmenou komunikačného módu. V prípade zmeny komunikačného módu, na ktorý je zariadenie určené, by bolo potrebné zmeniť nasledujúce časti:

- **Fyzickú vrstvu** tak, aby vyhovovala SD komunikačnému módu
- **Ovládače periférií** podľa použitej fyzickej vrstvy

Ostatné vrstvy softwaru by mali ostať rovnaké, pretože pracujú s dátami nezávislými na komunikačnom móde.

5.2 Firmware diagnostického zariadenia

Cieľ firmwaru diagnostického zariadenia bol naznačený v kapitole 1. Firmware bude vyvíjaný v programovacom jazyku C. Hardwarová platforma podporuje niekoľko vývojových prostredí, ktoré je možné pri programovaní a ladení použiť. Výrobca hardwarovej platformy ponúka vlastné vývojové prostredie založené na Eclipse prostredí. Na základe predošlých skúseností bolo však vybraté vývojové prostredie IAR Embedded Workbench [7], ktoré plne podporuje nahrávanie, vývoj a ladenie firmwaru na zvolenej hardwarovej platforme.

V rámci návrhu firmwaru je potrebné zvážiť použitie operačného systému reálneho času. V prípade použitia operačného systému reálneho času by boli úlohy diagnostickej aplikácie spúšťané operačným systémom. Návrh firmwaru predpokladá, že firmware nebude tak zložitý, aby potreboval operačný systém reálneho času a výhody plynúce z jeho použitia. Operačný systém by mohol vnášať do firmwaru ďalšiu réžiu, ktorá by mohla zvyšovať latenciu. Z tohto dôvodu budú úlohy diagnostickej aplikácií spúšťané z hlavného programu main v takzvanej super slučke. Úlohy firmwaru budú prerušované udalosťami od komunikačných periférií, ktoré vyžadujú okamžité spracovanie dát.

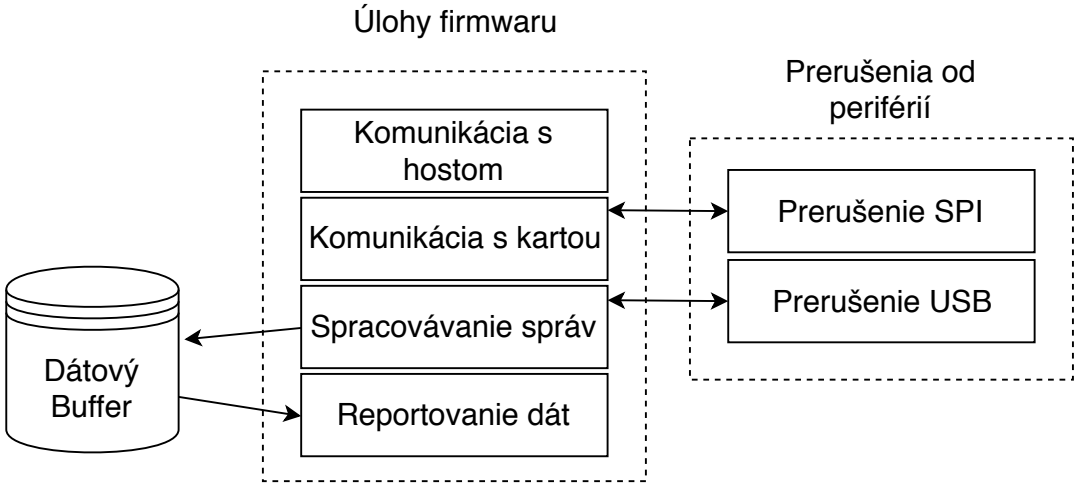
V neposlednom rade je nutné poznamenať, že ovládače periférií použité pri implementácií firmwaru sú prebraté od výrobcu MCU a sú dostupné z webovej aplikácie MCUXpresso SDK Builder [9].

Predtým, ako bude popísaný firmware diagnostického nástroja, je potrebné definovať jeho hlavné úlohy. Úlohy vyplývajúce zo stanoveného konceptu sú nasledovne:

- **Riadenie firmwaru:** firmware bude bežať v 2 stavoch - IDLE a RUN. Počas stavu RUN bude odosielať prenášané správy na nadradenú vrstvu a spracovávať ich
- **Komunikácia s hostiteľským zariadením:** prijímanie správ od hostiteľského zariadenia a odosielanie odpovedí od reálnej pamäťovej karty
- **Komunikácia s pamäťovou kartou:** odosielanie správ od hostiteľského zariadenia pamäťovej karte a prijímanie odpovedí od karty
- **Spracovávanie prenášaných správ:** diagnostický nástroj musí prenášané správy spracovať - analyzovať, pretože hostiteľské zariadenie môže posielať alebo prijímať 3 typy správ, tak ako to bolo popísané v kapitole 2. Do spracovania správ je nutné zahrnúť aj aktívny zásah podľa požiadaviek užívateľa
- **Komunikácia s nadradenou vrstvou:** odosielanie prenášaných správ na nadradenú vrstvu, to znamená nadradenú aplikáciu

Firmware bude realizovaný v super slučke a bude vykonávať úlohy v nekonečnom cykle. Z tohto návrhu vyplývajú dve úrovne - úlohy a prerušenia. Hlavné úlohy

firmwaru vrátane prerušení od komunikačných periférií a dátového bufferu sú zobrazené v štruktúre diagnostickej aplikácií na obrázku 5.2 a budú popísané v nasledujúcich podkapitolách.



Obr. 5.2: Štruktúra firmwaru diagnostického zariadenia

Na základe predošlého návrhu nadradeného zariadenia a koncepcie diagnostického nástroja je nutné, aby firmware odosielal prenášané správy medzi hostiteľským zariadením a pamäťovou kartou na nadradenú vrstvu, teda nadradenej aplikácií.

Jednou z možností je odosielať dáta počas spracovania správ. Spracovanie správ je z časového hľadiska náročná úloha, ktorú je potrebné realizovať s čo najvyššou rýchlosťou. Ďalej návrh firmwaru predpokladá, že okamžité odosielanie dát by bolo neefektívne a v časovom okne medzi jednotlivými správami by nebolo možné komunikovať s nadradeným zariadením.

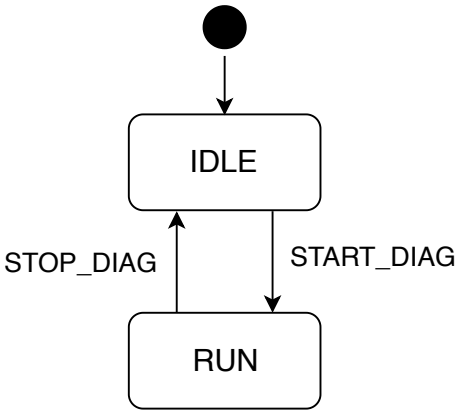
Preto bolo potrebné túto úlohu rozdeliť na dve časti - spracovanie správ a reportovanie dát. Dátový buffer tvorí rozhranie medzi týmito dvomi úlohami, do ktorého sú z jednej strany ukladané dáta z úlohy spracovania správ a z druhej strany sú dáta z diagnostiky odoslané na nadradenú vrstvu pomocou úlohy reportovania dát.

5.2.1 Riadenie firmwaru diagnostického zariadenia

Firmware môže bežať v dvoch diagnostických stavoch - IDLE a RUN. Predvoleným stavom diagnostiky je IDLE v ktorom sa správy od hostiteľského zariadenia neodosielajú pamäťovej karte a hostiteľské zariadenie vyhodnotí, že pamäťová karta nie je dostupná. V stave IDLE je možné komunikovať s diagnostickým zariadením z nadradenej vrstvy a konfigurovať diagnostiku.

Po prechode do stavu RUN je firmware pripravený na diagnostiku a na aktívne

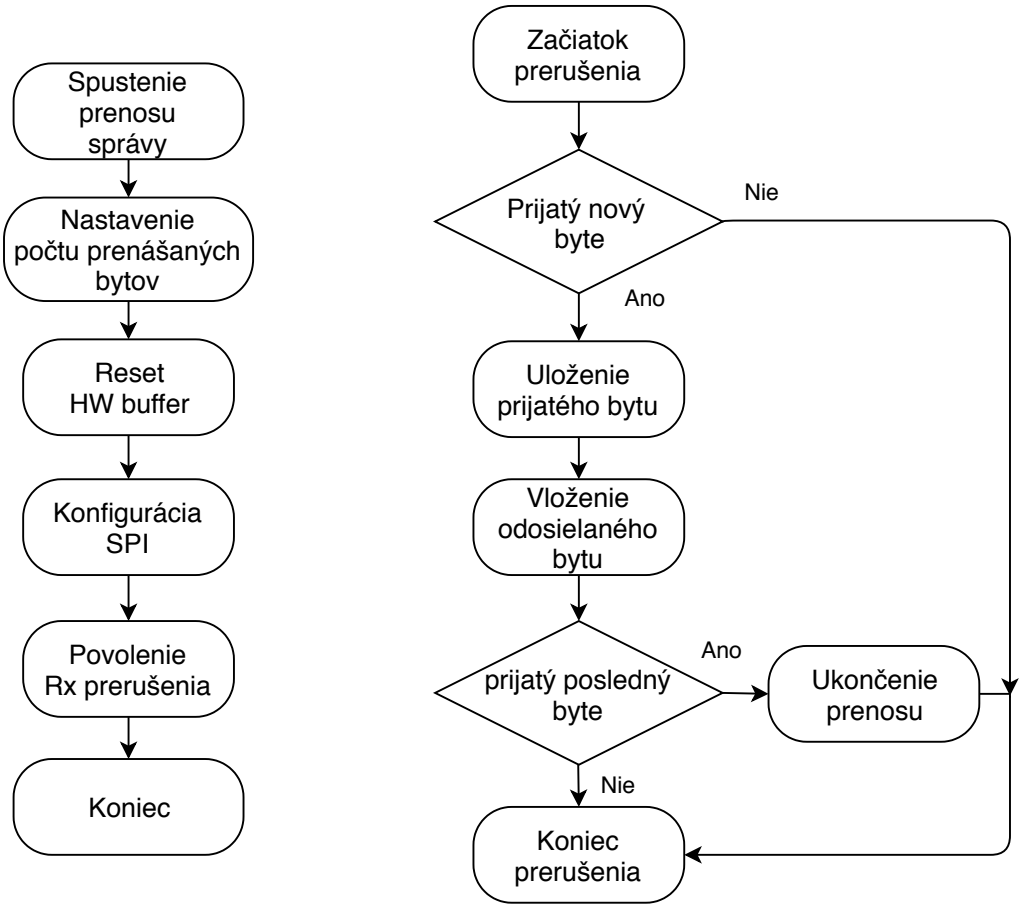
zásahy nakonfigurované užívateľom. V stave RUN nie je možné diagnostiku konfigurovať. Týmto spôsobom je zamedzené, aby bola aktuálna konfigurácia diagnostiky za behu zmenená. Prechody do jednotlivých stavov budú riadené z nadradenej aplikácie.



Obr. 5.3: Stavy firmwaru diagnostického zariadenia

5.2.2 Komunikácia s hostiteľským zariadením

Ako už bolo spomenuté v rozbore zadania na obrázku 1.2, diagnostické zariadenie je pri prijímaní správ od hostiteľského zariadenia v úlohe Slave. To znamená, že prichádzajúce dáta je potrebné spracovávať pomocou prerušenia. Prerušenie spolupracuje so samotnou SPI perifériou, ktorá prijaté dáta ukladá do bufferu na hardwarovej úrovni a nastavuje stavové príznaky o prijatí nových dát. MCU disponuje niekoľkými SPI perifériami. Na komunikáciu s hostiteľským zariadením bola zvolená periféria SPI0. Prenášanie správ od hostiteľského zariadenia ako Slave je naznačené na nasledujúcom diagrame:



Obr. 5.4: Prijímanie správy od hostiteľského zariadenia

Predtým ako sa začne prijímať správa do hostiteľského zariadenia, firmware musí nastaviť počet bytov, ktoré budu prijaté, a nakonfigurovať SPI periférium na príjem dát. Následne je potrebné resetovať dátový buffer periférie a povoliť prerušenie na prijímanie dát. Samotný prenos je riadený z hostiteľského zariadenia. Firmware obslúži prerušenie a keď je prenesený požadovaný počet bytov, prenos je ukončený.

Časť firmwaru starajúca sa o komunikáciu s hostiteľským zariadením musí byť schopná vykonávať tieto operácie:

- Prijímanie príkazu a odosielanie odpovede
- Prijímanie dátového rámca
- Odosielanie dátového rámca

V tejto práci bude popísaná a implementovaná funkcionalita prenášania dátových rámcov pre samostatné bloky (single block), ktoré tvoria základ pre komunikáciu s pamäťovou kartou. V prípade ďalšieho rozšírenia bude potrebné implementovať algoritmus na prenášanie viacerých blokov (multiple blocks).

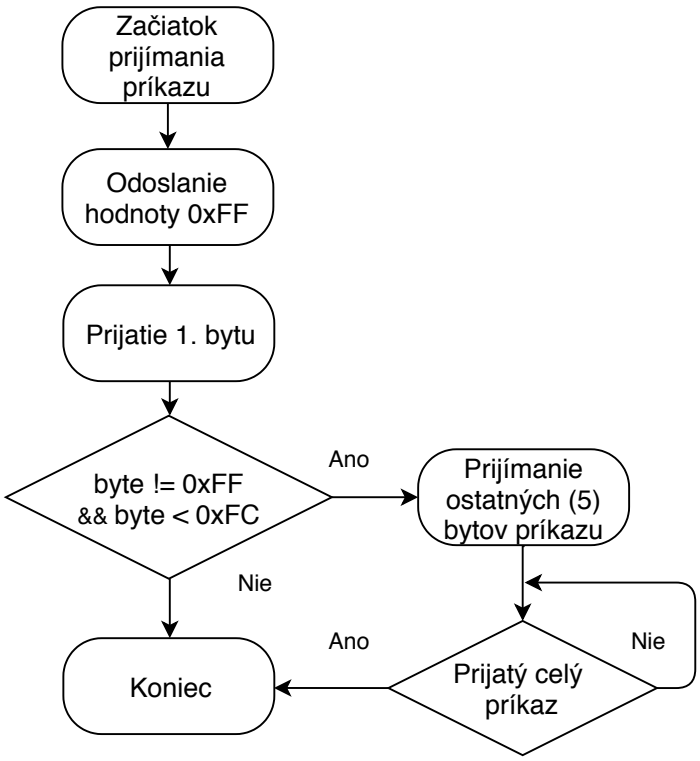
Prijímanie príkazu a odosielanie odpovede

Hostiteľské zariadenie odosiela pamäťovej karte na začiatku akejkoľvek správy nasledujúce dáta:

- 0xFF ako dotaz na pripravenosť karty. Odpoveď 0xFF znamená, že karta je pripravená. Naopak 0x00 znamená, že karta ešte nie je pripravená na prijímanie ďalších dát
- Hodnoty väčšie ako 0xFC vyhradené pre riadiace príznaky (kapitola 2.2.4)
- Prvý byte príkazu na základe popísaného formátu v tabuľke 2.2

Na základe tejto úvahy je prijímanie príkazu spustené odoslaním hodnoty 0xFF. Hostiteľské zariadenie môže začať prenášať príkaz alebo dátový rámec. Firmware rozozná príkaz len ak sa prvý byte nerovná riadiacemu príznaku, to znamená je menší ako hodnota 0xFC. Následne je prijatých zostávajúcich 5 bytov príkazu, čím je ukončený tento proces.

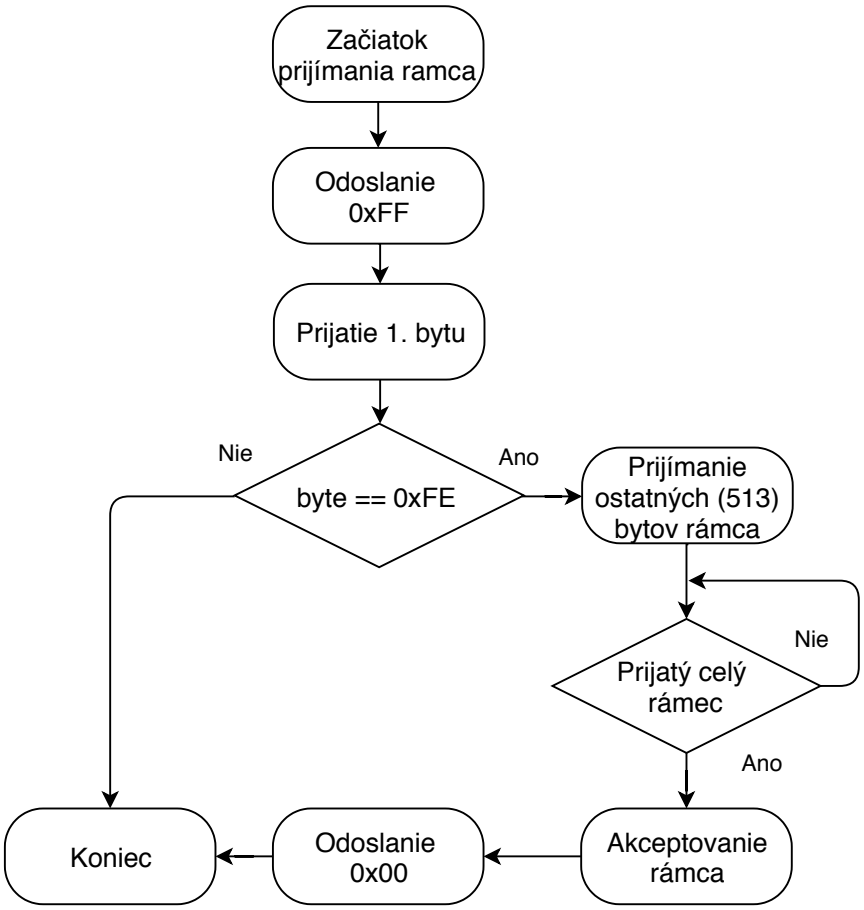
Pri samotnom odosielaní odpovede na príkaz nie je potrebná žiadna komplikovaná logika. Dáta odpovede sa len odošlú hostiteľskému zariadeniu.



Obr. 5.5: Prijímanie príkazu od hostiteľského zariadenia

Prijímanie dátového rámca

Prijímanie dátového rámca je založené na podobnom princípe, ako prijímanie príkazu. V tomto prípade ide o prijímanie rámca, ktorý hostiteľské zariadenie odosiela na príkaz CMD24 - Zápis jedného bloku. Tento proces kontroluje prijatie riadiaceho príznaku 0xFE, ktorý indikuje začiatok dátového rámca. Po prijatí celého rámca sa odošle hostiteľskému zariadeniu riadiaci príznak popisujúci akceptáciu rámca. Hodnota 0x00 indikuje, že pamäťová karta je zaneprázdnená a hodnota bude odosielaná až pokiaľ sa nezačne prijímať ďalší príkaz.



Obr. 5.6: Prijímanie dátového rámca od hostiteľského zariadenia

Odosielanie dátového rámca

Odosielanie dátových rámcov je jednoduché odoslanie dát hostiteľskému zariadeniu. Ak sa odosiela dátový rámec, na začiatku sa odošle riadiaci príznak 0xFE indikujúci začiatok rámca. Po ňom nasledujú samotné dáta prenášaného rámca.

5.2.3 Komunikácia s pamäťovou kartou

Diagnostické zariadenie je počas komunikácie s fyzickou pamäťovou kartou v úlohe Master. To znamená, že iniciuje a riadi celý prenos dát. Rovnako ako pri komunikácií s hostiteľským zariadením, prenos dát je potrebné riadiť a obsluhovať pomocou prerušenia od SPI periférie.

Algoritmus prijímania a odosielania dát pomocou prerušenia je podobný ako na obrázku 5.4 s tým rozdielom, že komunikácia je riadená z firmwaru a fyzická pamäťová karta je v úlohe Slave zariadenia. Hlavným rozdielom je konfigurácia samotnej SPI periférie, ktorá je v Master móde.

Na slot pamäťovej karty, ktorý je na hardwarovej platforme, sú vyvedené signály SPI1 zbernice. Z tohto dôvodu je nutné na komunikáciu s pamäťovou kartou využívať len túto periférium.

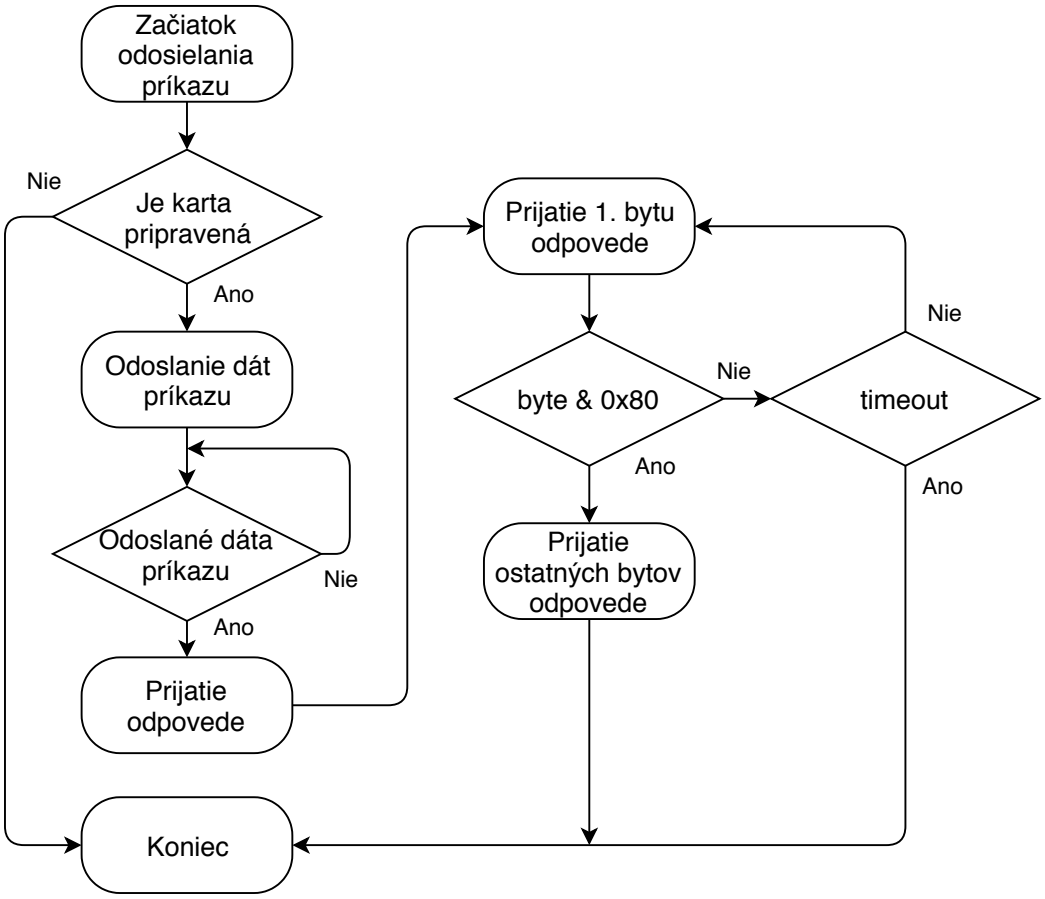
Komunikácia s pamäťovou kartou reflektuje operácie zo strany hostiteľského zariadenia ale v opačnom smere, ktoré sú nasledovné:

- Odosielanie príkazu a prijímanie odpovede
- Odosielanie dátového rámca
- Prijímanie dátového rámca

Pojem prijímanie dátového rámca je prípad, kedy firmware prijíma dátový rámec od pamäťovej karty - hostiteľské zariadenie číta dátový blok karty. Firmware odosiela dátový rámec pamäťovej karte v prípade, keď hostiteľské zariadenie zapisuje blok na pamäťovú kartu.

Odosielanie príkazu a prijímanie odpovede

V prvom kroku je potrebné zaručiť, že je pamäťová karta pripravená prijať ďalší príkaz. Ak áno, dáta sa odošlú a začne sa proces prijímania odpovede. Príjem odpovede je spustený, ak má prvý byte odpovede na pozícií 7. bitu hodnotu 0 (tabuľka 2.5) alebo ak nevyprší timeout, ktorý je daný 255 pokusmi o príjem odpovede od karty. Následne sú prijaté všetky dáta odpovede v prípade, že sa nejedná o odpoveď R1. Ak sa jedná o odpoveď R1, dáta odpovede sú rovne prvému prijatému bytu. Popísaný algoritmus je zobrazený na obrázku 5.7.



Obr. 5.7: Odosielanie a prijímanie odpovede od pamäťovej karty

Odosielanie a prijímanie dátového rámca

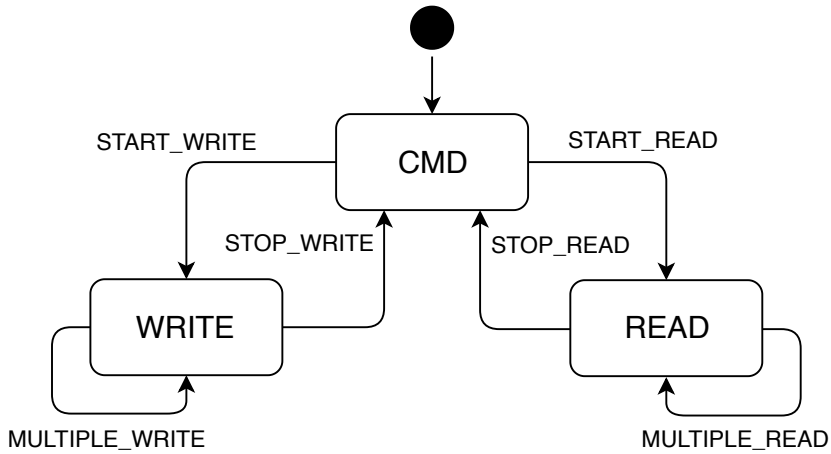
Algoritmus prijímania dátového rámca od pamäťovej karty je založený na podobnom princípe ako pri prijímaní rámca od hostiteľského zariadenia. Firmware začne komunikovať s kartou a kontroluje prvý prijatý byte. Príjem dát sa spustí, keď je prvý prijatý byte rovný riadiacemu príznaku s hodnotou 0xFE.

Pri odosielaní dátového rámca pamäťovej karte je pred samotné dáta vložený riadiaci príznak 0xFE, čím je pamäťovej karte indikovaný začiatok rámca. Následne sú odoslané všetky dáta prenášaného rámca.

5.2.4 Spracovanie prenášaných správ

Spracovanie prenášaných správ firmwarom je založené na stavovom diagrame na obrázku 5.8, ktorého stavy budú popísané v nasledujúcich podkapitolách. Význam jednotlivých stavov je nasledovný:

- **CMD:** firmware očakáva príkaz od hostiteľského zariadenia. Toto je predvolený stav stavového diagramu, pretože hostiteľské zariadenie iniciuje každú akciu príkazom
- **READ:** je stav, v ktorom hostiteľské zariadenie číta dáta z pamäťovej karty
- **WRITE:** je stav, v ktorom hostiteľské zariadenie zapisuje dáta na pamäťovú kartu



Obr. 5.8: Stavový diagram spracovania správ

Prechody do stavov READ a WRITE sú realizované na základe prijatých príkazov. Príkaz definuje, či bude hostiteľské zariadenie dáta zapisovať alebo čítať z pamäťovej karty. Príkazy určené na čítanie dát z pamäťovej karty vygenerujú podmienku START_READ, čím firmware prejde do stavu READ. Naopak príkazy určené na zápis dát vygenerujú podmienku START_WRITE. Príkazy určené na konfiguráciu karty, ktoré nevyžadujú prenos dátových rámcov, sú spracované v stave CMD.

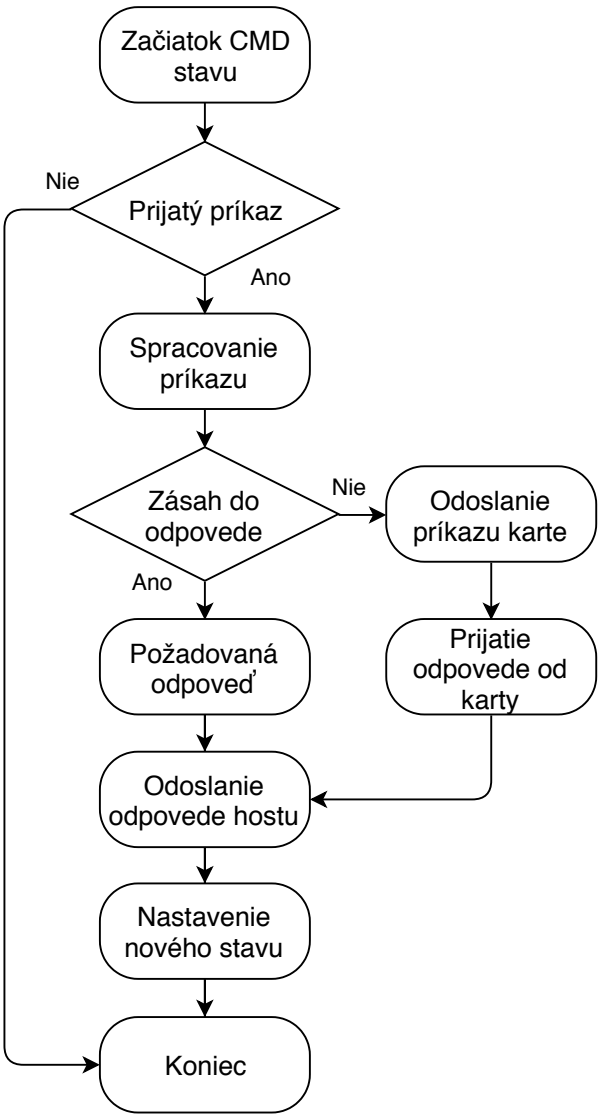
Stavy READ a WRITE sú nastavené pokiaľ nenastanú podmienky STOP_READ alebo STOP_WRITE. V prípade prenášania jedného dátového rámca je nastavený stav CMD hneď po skončení prenosu.

Úloha spracovávania správ je spúšťaná len ak je do diagnostického zariadenia vložená pamäťová karta a stav diagnostiky je nastavený na RUN.

Stav CMD

Prijímanie príkazu je realizované na základe algoritmu popísaného v predchádzajúcej kapitole 5.2.2. Ak je prijatý nový príkaz od hostiteľského zariadenia, začne sa spracovávať. Pod spracovaním príkazu sa rozumie zistenie čísla príkazu, získanie očakávaného typu odpovede a či je požadovaný zásah do odpovede. Ak je zásah do odpovede povolený, hostiteľskému zariadeniu sa odošle požadovaná odpoveď. Ak je

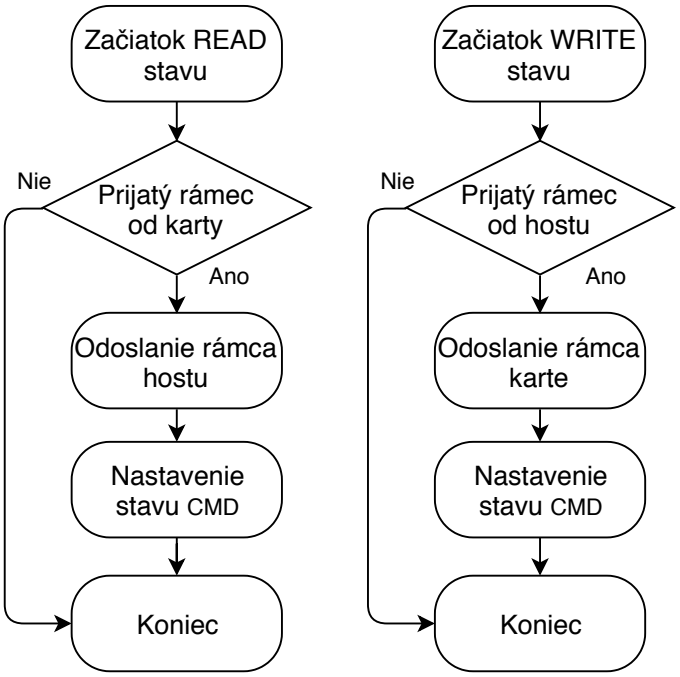
príkaz bez zásahu, odošle sa pamäťovej karte a od karty sa prijme odpoveď. Posledným krokom stavu CMD je prechod do nového stavu.



Obr. 5.9: Diagram algoritmu stavu CMD

Stav READ a WRITE

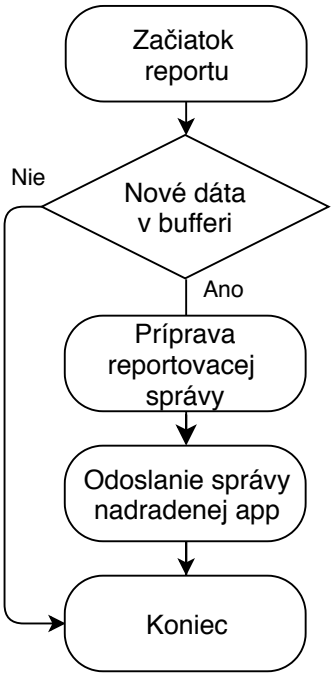
V stave READ a WRITE sú dáta preposielané pamäťovej karte alebo hostiteľskému zariadeniu. Obidva stavy sú vo svojej podstate totožné, rozdiel je iba v smere v ktorom sú dáta prenášané. Vývojový diagram popisujúci tieto stavy je zobrazený na nasledujúcom obrázku:



Obr. 5.10: Diagram algoritmu stavu READ a WRITE

5.2.5 Reportovanie dát

Cieľom tejto úlohy firmwaru je odosielanie prenášaných dát na nadradenú vrstvu, to znamená do nadradenej aplikácie. Ako už bolo spomenuté v úvode kapitoly, prenášané dáta sa počas spracovania ukladajú do dátového bufferu. Počas reportovania sú z bufferu vyzdvihnuté a odoslané cez USB.



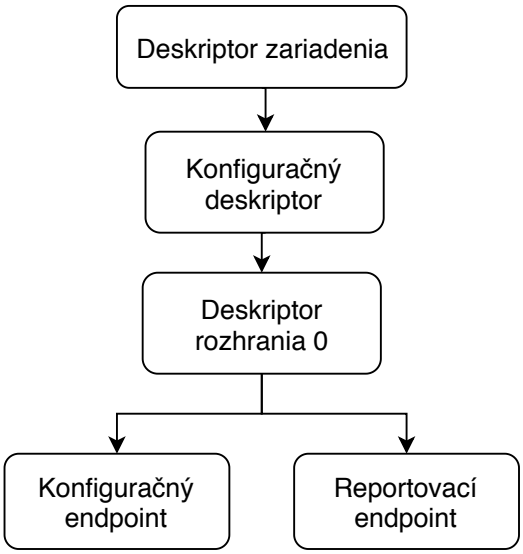
Obr. 5.11: Diagram algoritmu reportovania dát nadradenej vrstve

5.2.6 Komunikácia s nadradenou vrstvou

Firmware musí zahŕňať podporu USB komunikácie verzie 2.0 podľa požiadavky definovanej zvoleným konceptom. Na komunikáciu s nadradeným zariadením je využitá USB periféria a USB protokol. Výrobca MCU poskytuje implementáciu celého USB protokolu na zvolenom HW, ktorý sa všeobecne nazýva USB stack.

Komunikáciu s nadradenou vrstvou je možné rozdeliť do dvoch časti. Prvou je prijímanie konfiguračných príkazov. Pomocou nich je konfigurovaný zásah do komunikácie na základe požiadaviek užívateľa. Druhou časťou je odosielanie reportovacích správ, ktoré obsahujú dáta prenášaných správ medzi hostiteľským zariadením a pamäťovou kartou.

Toto vedie k návrhu dvoch USB koncových bodov (Endpoint) zariadenia - konfiguračný a reportovací. USB zariadenie je popísané pomocou deskriptorov, ktorých štruktúra je zobrazená na obrázku 5.12.



Obr. 5.12: Popis USB diagnostického zariadenia

V prípade ďalšieho rozšírenia firmwaru je možné pridávať koncové body alebo ďalšie USB rozhrania diagnostického zariadenia.

Konfiguračné príkazy

Konfiguračný príkaz je z nadradenej vrstvy odoslaný po bytoch. Jeho formát je zobrazený v tabuľke 5.1:

Pozícia bytu	0	1	2	3
Popis	Stav	ID Príkazu	Zásah povolený	Požadovaná odpoveď

Tab. 5.1: Formát konfiguračného príkazu

Význam jednotlivých častí konfiguračného príkazu je nasledovný:

- **Stav:** riadi stav diagnostiky. Ak je Stav nastavený na 0, firmware nastaví stav diagnostiky na IDLE. Ak je Stav nastavený na 1, diagnostika prejde do stavu RUN
- **ID príkazu:** je číslo príkazu, na ktorý bude firmware odpovedať odpoveďou podľa požiadavky užívateľa
- **Zásah povolený:** povoľuje zásah do daného príkazu

Reportovacie správy

Reportovacie správy sú odosielané firmwarom na nadradenú vrstvu počas úlohy reportovania dát. Sú rozdelené na dve kategórie - správy pre príkaz a pre dátový rámec.

Reportovacia správa pre príkaz obsahuje jeho dáta a odpoveď od pamäťovej karty. Reportovacia správa pre dátový rámec obsahuje jeho dáta, ktoré boli prenesené medzi hostiteľským zariadením a pamäťovou kartou. Formát diagnostickej správy pre príkaz je zobrazený v tabuľke 5.2.

Pozícia bytu	0	1:5	6	7:11
Počet bytov	1	6	5	1 - 5
Popis	Typ správy	Dáta príkazu	Typ odpovede	Dáta odpovede

Tab. 5.2: Formát diagnostickej správy pre príkaz

Význam jednotlivých častí reportovacej správy pre príkaz je nasledovný:

- **Typ správy:** definuje, či sa jedná o reportovaciu správu pre príkaz (hodnota 0) alebo pre dátový rámec (hodnota 1)
- **Dáta príkazu:** obsahuje dáta príkazu tak ako to definuje SD protokol
- **Typ odpovede:** je ekvivalentný jej veľkosti v závislosti na odpovedi R1 - R3 alebo R7
- **Dáta odpovede:** dáta odpovede na príkaz

Formát diagnostickej správy pre dátový rámec je zobrazený v tabuľke 5.3.

Pozícia bytu	0	1:64
Počet bytov	1	65
Popis	Typ správy	Dáta rámca

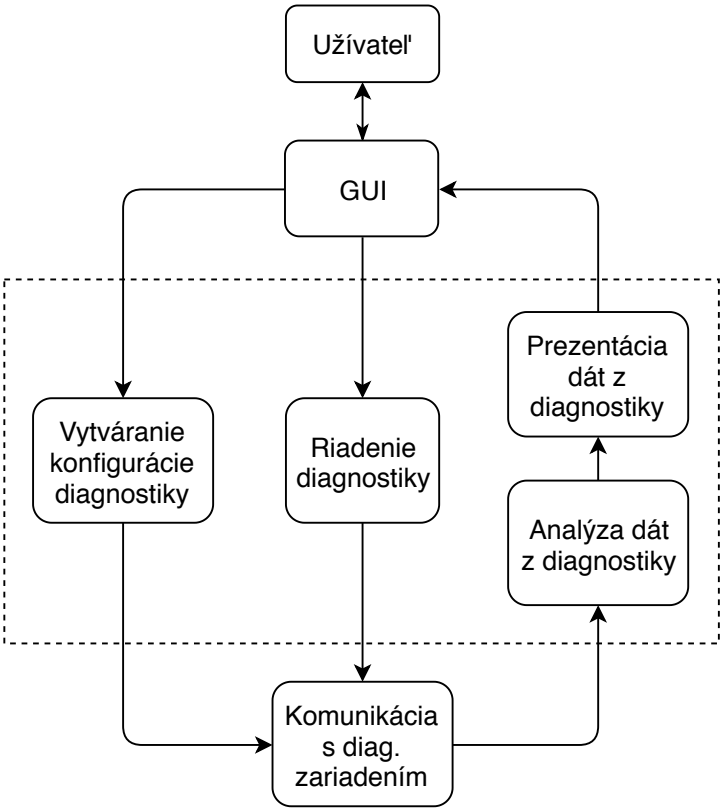
Tab. 5.3: Formát diagnostickej správy pre dátový rámec

Význam jednotlivých častí reportovacej správy pre dátový rámec je nasledovný:

- **Typ správy:** definuje či sa jedná o reportovaciu správu pre príkaz (hodnota 0) alebo pre dátový rámec (hodnota 1)
- **Dáta rámca:** obsahuje dáta dátového rámca

5.3 Nadradená aplikácia

Nadradená aplikácia, ktorá vytvára rozhranie s užívateľom, bude bežať na PC s operačným systémom Windows. Bude naprogramovaná v jazyku C# v .NET frameworku. Jeho výhodou je komplexnosť a rýchly vývoj užívateľských aplikácií. Blokový diagram popisujúci nadradenú aplikáciu a jej hlavnú funkcionalitu je zobrazený na obrázku 5.13.



Obr. 5.13: Blokový diagram nadradenej aplikácie

5.3.1 Grafické rozhranie s užívateľom

Framework .NET ponúka možnosť vytvárania takzvaných WinForms aplikácií. Ich vývoj je rýchly, prebieha pomocou grafického designeru a dostupných komponent ako sú tlačidla, formuláre, tabuľky a pod.

Ako bolo spomenuté predtým, hlavným cieľom nadradenej aplikácie bude prezentácia prenášaných správ medzi hostiteľským zariadením a pamäťovou kartou, to znamená reportovacích správ. Prvým konceptom zobrazovania reportovacích správ bolo ich vykresľovanie. Tento koncept je však zbytočne detailný, pretože diagnostický nástroj je určený na diagnostiku na protokolovej úrovni.

Z tohto dôvodu budú reportovacie správy zobrazované v tabuľke s nasledujúcimi atribútami:

- **ID:** je číslo, ktoré sa každou prijatou reportovacou správou inkrementuje
- **Typ:** typ reportovacej správy - pre príkaz alebo dátový rámec
- **Názov:** je názov reportovacej správy (v prípade príkazov to je napr. CMD8)
- **Raw Data:** sú samotné dáta reportovacej správy

5.3.2 Komunikácia s diagnostickým zariadením

Spôsob komunikácie medzi nadradenou aplikáciou a diagnostickým zariadením je daný výberom konceptu diagnostického nástroja. Na strane nadradenej aplikácie bude potrebné realizovať USB komunikačnú vrstvu pomocou USB ovládača. Najlepším spôsobom realizácie USB komunikačnej vrstvy je použitie hotového riešenia vo forme knižnice, ktorá bude podporovaná na operačnom systéme Windows a zvolenom programovacom jazyku.

USB komunikačná vrstva bude v rámci nadradenej aplikácie implementovaná pomocou libUSB knižnice, ktorá je dostupná na odkaze [6]. Knižnica podporuje niekoľko operačných systémov vrátane operačného systému Windows a Linux. Podpora viacerých operačných systémov je výhodou v prípade zmeny operačného systému, na ktorom bude nadradená aplikácia bežať.

Knižnica libUSB je určená pre programovací jazyk C. Z tohto dôvodu je potrebné použiť takzvanú wrapper knižnicu s názvom LibUsbDotNet určenú pre .NET framework dostupnú z [10].

5.3.3 Hlavná funkcionalita nadradenej aplikácie

Hlavná funkcionalita nadradenej aplikácie vyplýva z celkových požiadaviek na diagnostický nástroj. Užívateľ bude pomocou nadradenej aplikácie schopný odosielať diagnostické príkazy diagnostickému zariadeniu.

Hlavnou úlohou nadradenej aplikácie bude čítanie reportovacích správ a ich prezentácia užívateľovi. Predtým ako budú dáta z diagnostiky prezentované pomocou nadradenej aplikácie užívateľovi, bude potrebné dáta analyzovať na základe špecifikácie SD komunikačného protokolu.

Vytváranie konfigurácie a riadenie diagnostiky

Vytváranie a riadenie diagnostiky bude realizované pomocou konfiguračných príkazov popísaných v kapitole 5.2.6. Nadradená aplikácia vytvorí konfiguračný príkaz na základe požiadaviek užívateľa a odošle ho diagnostickému zariadeniu.

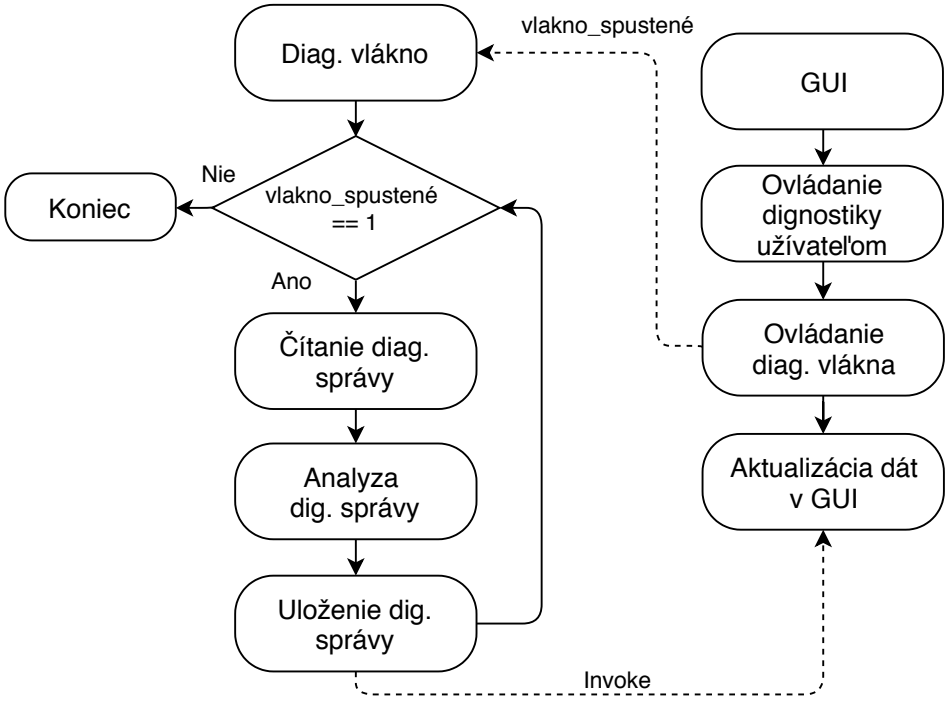
Užívateľ definuje príkaz hostiteľského zariadenia, na ktorý bude firmware odpovedať požadovanou odpoveďou. Následne nadradená aplikácia vytvorí konfiguračný príkaz a odošle ho diagnostickému zariadeniu.

Čítanie reportovacích správ

Reportovacie správy budú čítané z reportovacieho koncového bodu diagnostického zariadenia. Čítanie reportovacích správ bude bežať v samostatnom vlákne. Vlákno bude spustené, keď užívateľ spustí diagnostiku. Po prijatí reportovacej správy ju bude potrebné analyzovať a prezentovať užívateľovi pomocou GUI.

Vlákno bude ovládané pomocou signálu `vlákno_spustené` zobrazenom na obrázku 5.14. Ak užívateľ spustí diagnostiku, vlákno sa vytvorí a bude bežať pokiaľ signál `vlákno_spustené` nebude mať hodnotu 0 - užívateľ zastaví diagnostiku.

Po prijatí a analýze reportovacej správy je potrebné aktualizovať GUI v ktorom sú správy prezentované užívateľovi. Dáta zobrazované v GUI sú získavané v inom vlákne, preto je nutné využiť metódu Invoke. Tá zabezpečí, aby boli vlákna vykonané v správnom poradí. Po aktualizácií dát v GUI bude užívateľovi zobrazená prijatá reportovacia správa.



Obr. 5.14: Čítanie reportovacích správ

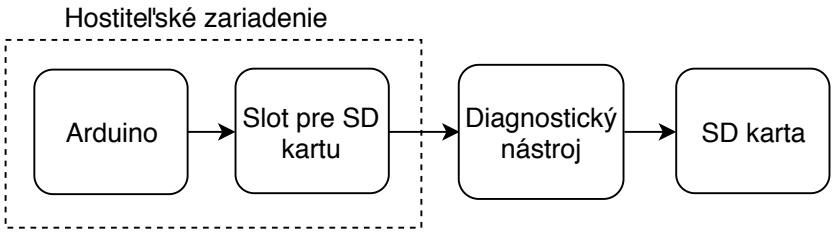
6 Realizácia diagnostického nástroja

Cieľom tejto kapitoly je popis realizácie diagnostického zariadenia na základe predošlého návrhu. Realizáciu, tak ako diagnostický nástroj ako celok, je možné rozdeliť do dvoch celkov. Prvým je realizácia firmwaru diagnostického zariadenia a druhým je realizácia nadradenej aplikácie. Prvým krokom tejto kapitoly je popis testovacieho prostredia, pomocou ktorého prebiehal vývoj firmwaru. Následne sa kapitola bude venovať realizácií obidvoch softwarových častí.

Posledným bodom tejto kapitoly je symbolické zobrazenie hardwarovej realizácie diagnostického zariadenia.

6.1 Testovacie prostredie počas vývoja

Počas vývoja diagnostického nástroja bolo potrebné realizovať hostiteľské zariadenie na testovanie funkčnosti nástroja po jednotlivých častiach a nakoniec jeho celkovej funkčnosti. Ako najvhodnejším hostiteľským zariadením komunikujúcim v SPI móde a nad ktorým je možné mať plnú kontrolu bolo zvolené Arduino s dodatočným slotom pre SD pamäťové karty. Usporiadanie testovacieho prostredia je zobrazené na obrázku 6.1.



Obr. 6.1: Usporiadanie testovacieho prostredia

Na komunikáciu s pamäťovou kartou bola využitá knižnica SD.h, ktorá je súčasťou inštalácie vývojového prostredia pre Arduino. Pomocou nej bolo možné vykonávať jednotlivé kroky komunikácie ako napríklad inicializácia karty, čítanie registrov karty, prenos dátového rámca a pod.

6.2 Implementácia firmwaru diagnostického zariadenia

Cieľom tejto kapitoly je popis jednotlivých skupín modulov firmwaru, ich implementácie a vzájomnej závislosti. Funkcie, ktoré budú použité v diagramoch, majú

skrátený názov a ich reálny názov v zdrojovom kóde je tvorený názvom modulu v ktorom sú implementované a názvom z diagramu.

Súčasťou aktuálnej implementácie firmwaru je dátový buffer pre príkazy, z ktorého sú potom reportovacie správy vytvorené a odoslané. V budúcnosti bude potrebné implementovať ďalší dátový buffer pre dátové rámce, ktoré budú odosielané na nadradenú vrstvu. Z tohto dôvodu aktuálna implementácia firmwaru odosiela na nadradenú vrstvu len reportovacie správy pre prenášané príkazy.

Popis firmwaru je rozdelený do 2 hlavných častí - spracovanie správ a komunikácia s nadradenou aplikáciou.

6.2.1 Spracovanie správ

Skupina modulov spracovania správ implementuje nasledujúcu funkcionalitu:

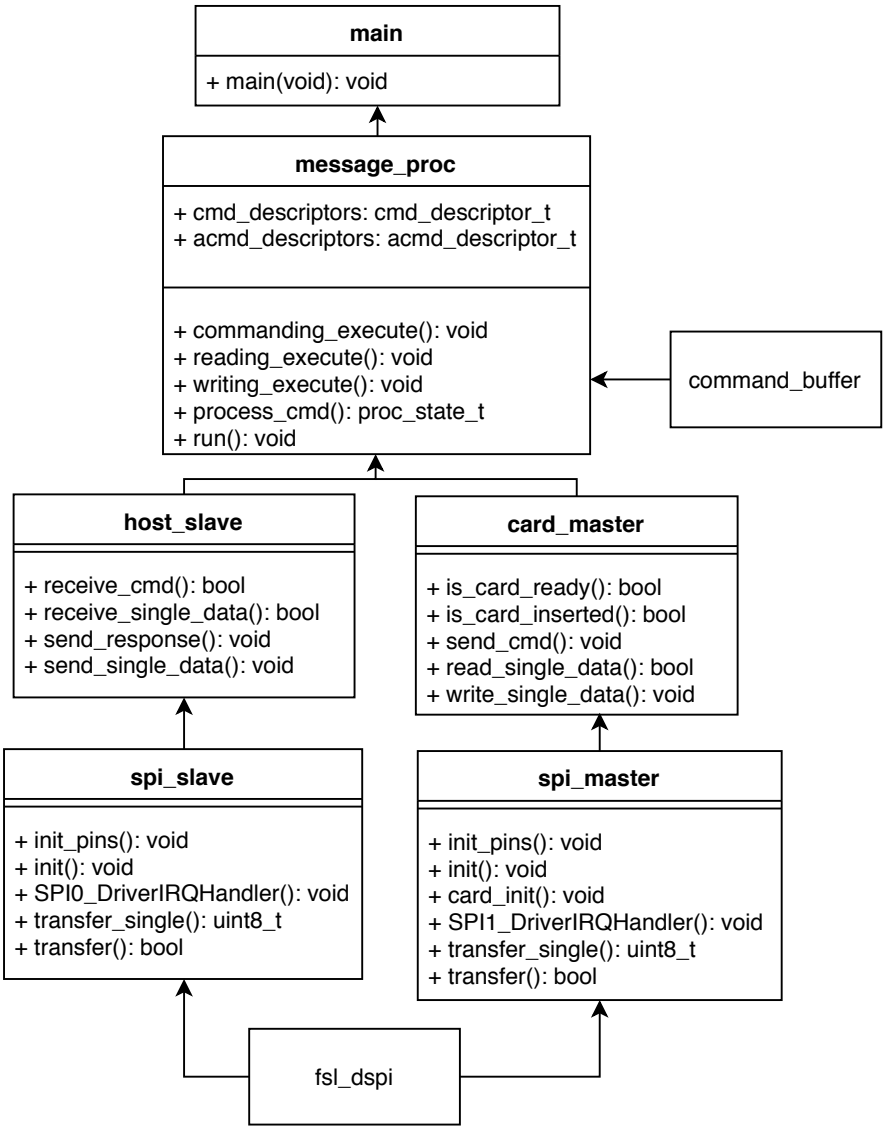
- **host_slave:** komunikácia s hostiteľským zariadením
- **card_master:** komunikácia s pamäťovou kartou
- **message_proc:** samotné spracovávanie správ

Diagram popisujúci štruktúru jednotlivých modulov je zobrazený na obrázku 6.2. Na najvyššej vrstve je modul message_proc, ktorý je volaný z hlavnej funkcie main.

V tomto module sú implementované stavy popísané v kapitole 5.2.4 pomocou takzvaných callback funkcii (`commanding_execute()`, `reading_execute()`, `writing_execute()`) volaných z funkcie `run()` na základe aktuálneho stavu. Dôležitými atribútmi tohto modulu sú premenné `cmd_descriptors` a `acmd_descriptors`. Jedná sa o pole typu `cmd_descriptor_t` pomocou ktorého je popísaný každý príkaz. Po prijatí príkazu od hostiteľského zariadenia je možné indexovať toto pole podľa ID (čísla) príkazu a zistiť potrebné informácie o prijatom príkaze ako napríklad typ odpovede, či je povolený aktívny zásah a požadovaná odpoveď.

Spracovávané správy sú ukladané do bufferu s názvom command_buffer. Z neho sú následne vytvárané reportovacie správy a odoslané na nadradenú vrstvu.

Komunikácia s hostiteľským zariadením a fyzickou pamäťovou kartou je implementovaná v moduloch host_slave a card_master. Tieto moduly obsahujú implementáciu funkcionality popísanej v predchádzajúcich kapitolách. Rozhranie s perifériami MCU tvoria moduly spi_slave a spi_master. Implementujú funkcie na prenos dát pomocou SPI periférie a volajú potrebné funkcie z výrobcom dodaného ovládača s názvom fsl_dspi.



Obr. 6.2: Diagram hierarchie zdrojových súborov pre spracovanie správ

6.2.2 Komunikácia s nadradenou vrstvou

Do implementácie komunikácie s nadradenou vrstvou patrí reportovanie správ a ovládanie diagnostiky. Tak ako to bolo spomenuté už predtým, na komunikáciu sa využíva USB stack.

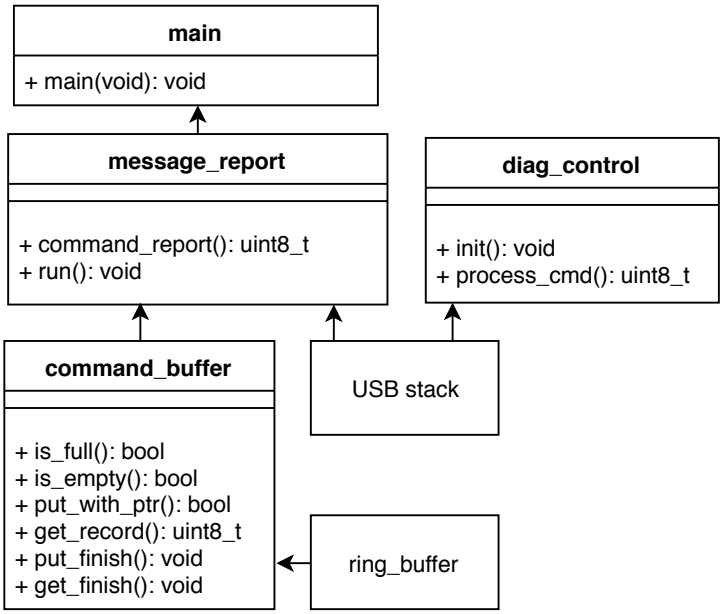
Modul reportovania správ je volaný pomocou funkcie `run()` z hlavnej funkcie `main()`. V module message_report je implementovaná funkcia `command_report()`. Funkcia na základe dát z bufferu command_buffer vytvorí reportovaciu správu pre príkaz a odošle ju na nadradenú vrstvu.

Modul ovládania diagnostiky diag_control nie je volaný z funkcie `main()`. Funkcia `process_cmd()`, ktorá spracuje prijatý konfiguračný príkaz z nadradenej vrstvy,

je volaná pomocou callback funkcie z USB stacku. Funkcia je volaná po každom prijatí nových dát z konfiguračného koncového bodu.

Modul command_buffer je implementovaný ako statický kruhový (Ring/Circular) buffer. Na vloženie dát do bufferu je využitá funkcia `put_with_ptr`, ktorá nastaví ukazateľ predaný ako argument, na volnú pozíciu v bufferi. Vkladanie dát týmto spôsobom je výhodne, pretože dáta sú vložené priamo do bufferu pomocou ukazateľa. Po ukončení priameho vkladania dát do bufferu je nutné zavolať funkciu `put_finish()`, ktorá pripraví buffer na vloženie nových dát.

Pri odosielaní reportovacích správ sú dáta správy získané pomocou funkcie s názvom `get_record()`. Funkcia vytvorí reportovaciu správu na základe dát v bufferi a vráti jej veľkosť.



Obr. 6.3: Diagram zdrojových súborov pre komunikáciu s nadradenou vrstvou

6.3 Nadradená aplikácia

Realizácia grafického rozhrania nadradenej aplikácie je zobrazená na obrázku 6.4. Grafické rozhranie je možné rozdeliť do 3 častí:

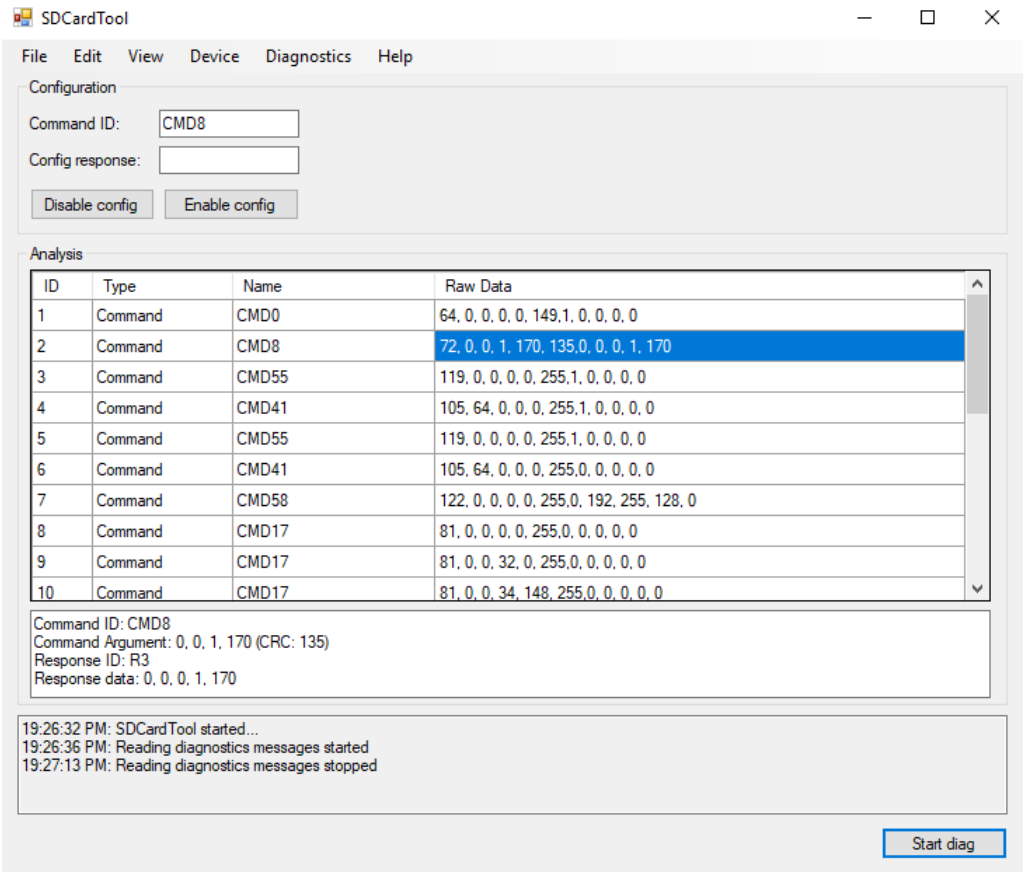
- Konfigurácia zásahu
- Analýza správ
- Ovládanie diagnostiky

Konfiguráciu zásahu do príkazu je možné vykonať pomocou dvoch textových polí, ktorých vstupom je ID (číslo príkazu) a požadovaná odpoveď typu R1 od užívateľa.

Po zadaní týchto vstupov môže užívateľ zásah povoliť alebo zakázať. Po povolení alebo zakázaní konfigurácie nadradená aplikácia odošle konfiguračný príkaz diagnostickému zariadeniu.

Ďalšou časťou je analýza správ, ktorá je realizovaná ako tabuľka s atribútami popísanými v kapitole 5.3.1. Po spustení diagnostiky sa do tabuľky dynamicky pridávajú diagnostické správy prijaté od diagnostického zariadenia. Po kliknutí na riadok v stĺpci Raw Data je pod tabuľkou zobrazená analýza diagnostickej správy.

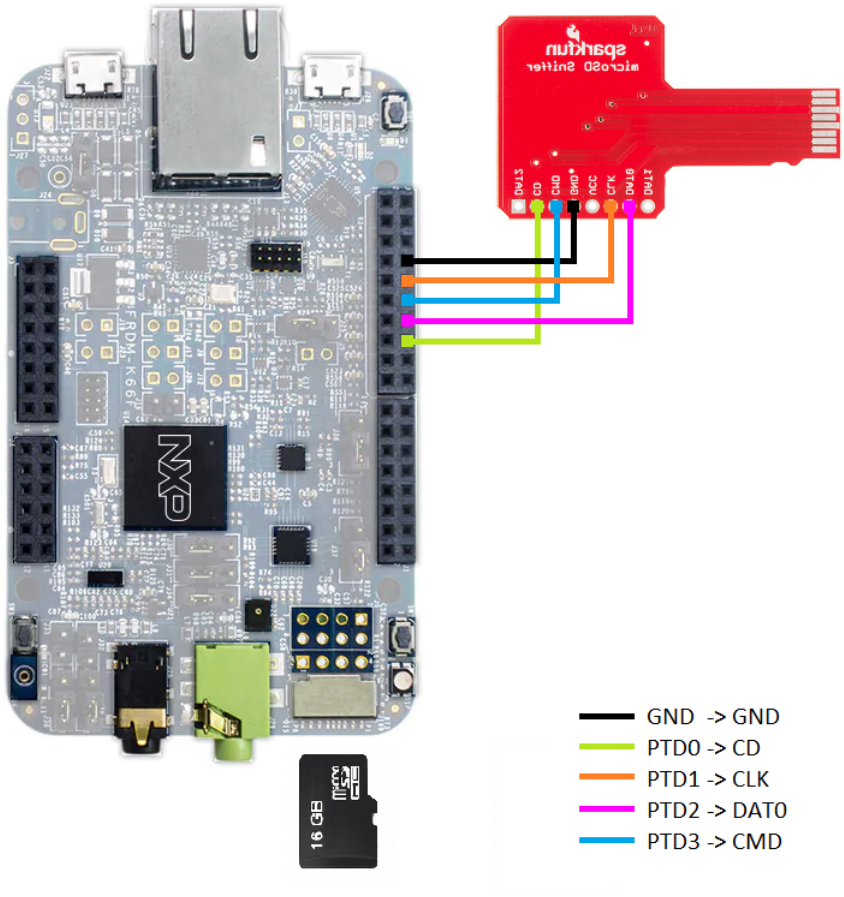
Pomocou tlačidla v pravom dolnom rohu je možné diagnostiku ovládať - spustiť alebo zastaviť. Po spustení diagnostiky sa spustí algoritmus popísaný v kapitole 5.3.3 na obrázku 5.14.



Obr. 6.4: Grafické rozhranie nadradenej aplikácie

6.4 Hardware diagnostického zariadenia

Hardware diagnostického zariadenia sa skladá z hardwarovej platformy a snifferu. Realizácia je založená na návrhu hardwaru popísanom v kapitole 4.1. Realizácia diagnostického zariadenia je zobrazená na obrázku 6.5.



Obr. 6.5: Realizácia hardwaru diagnostického zariadenia [4] [8]

7 Zhodnotenie riešenia a výsledkov

Realizáciou návrhu diagnostického nástroja založenom na MCU bol vytvorený prvý funkčný koncept nástroja pracujúcom v SPI komunikačnom móde. Jeho funkcionalita bola ladená a overená pomocou testovacieho prostredia popísaného v kapitole 6.1. V tejto kapitole bude popísaný priebeh realizácie diagnostického nástroja, obmedzenia a návrhy na jeho zlepšenie.

7.1 Priebeh realizácie diagnostického nástroja

Počiatočným bodom tejto práce bol nápad na návrh konceptu a realizáciu diagnostického nástroja na volne dostupnej hardwarovej platforme bez návrhu vlastného hardwaru. V prvej fáze bolo potrebné naštudovať a pochopiť SD komunikačný protokol, jeho hlavné princípy a prípadné odlišnosti. Diagnostické zariadenie je určitým prostredníkom medzi hostiteľským zariadením a pamäťovou kartou. Preto bolo potrebné preskúmať štandardy definujúce druhú stranu komunikácie - hostiteľské zariadenie (dostupne z [2]).

Po dostatočnom naštudovaní komunikačného protokolu boli navrhnuté koncepty diagnostického nástroja, ktoré boli diskutované so zadávateľom práce. Koncepty sú založené na požiadavkách na diagnostický nástroj, primárne požiadavke na realizáciu na hardwarovej platforme.

Počas návrhu a implementácie firmwaru vznikalo niekoľko koncepcií. Prvá bola založená na podstate full duplex SPI komunikácie. Firmware prijal byte od hostiteľského zariadenia, odoslal ho karte a následne prijal byte, ktorý karta odoslala. Toto riešenie nie je možné, pretože v prípade ďalšej logiky (analýzy) nad prenášanými bytami celej správy by firmware nestíhal vykonať všetky potrebné operácie medzi jednotlivými bytami správy. Z tohto dôvodu bolo nutné, aby firmware pristupoval k prenášaným správam ako k celku.

Následne bolo potrebné vyriešiť spôsob prijímania prenášaných správ. Prvou možnosťou bolo prijímanie pomocou DMA periférie MCU. Cieľom bolo urýchliť beh firmwaru a prijímanie dát správy na hardwarovej úrovni. Hostiteľské zariadenie však odosiela najprv nejakú špecifickú hodnotu prvého bytu, za ktorou nasledujú dáta prenášanej správy. Prvý byte je už časťou prenášanej správy, to znamená, že firmware by musel spúšťať prenos pomocou DMA medzi prvým a druhým bytom správy. Z tohto dôvodu bol nakoniec princíp prijímania správ implementovaný pomocou prerušenia, čím má firmware kontrolu nad každým prijatým bytom a môže do komunikácie v prípade potreby zasahovať.

V neposlednom rade bolo potrebné počas vývoja firmwaru prehrať bootloader

hardwarovej platformy, ktorý obsahoval chybu. Pri pripojení platformy k PC s operačným systémom Windows 10 prestal bootloader komunikovať a jediným riešením bolo nahranie nového firmwaru do MCU, ktorý je určený na ladenie hlavného MCU.

Vývoj prototypu nadradenej aplikácie následne prebiehal bez väčších problémov. Primárne sa jednalo o vytváranie grafického rozhrania a prácu s knižnicou LibUsbDotNet.

7.2 Obmedzenia diagnostického nástroja

Prvé obmedzenie vyplýva z vybraného konceptu diagnostického nástroja popísaného v kapitole 3.1. Nástroj je určený len pre SPI komunikačný mód pamäťovej karty. Toto obmedzenie je dané výberom konceptu založenom na MCU a jeho hardwarovým vybavením. Toto obmedzenie je zadávateľom rešpektované, pretože ide o prvý prototyp nástroja a nie o jeho finálnu realizáciu.

Firmware diagnostického zariadenia je v prípade prenosu dát na/z pamäťovej karty schopný reagovať na prenos samostatných dátových rámcov (single read/write). Tento algoritmus bol popísaný v kapitole 5.2.2. V prípade ďalšieho rozšírenia bude potrebné implementovať algoritmus prijímania dát pri prenose viacerých dátových rámcov (multiple read/write). Toto obmedzenie však nezasahuje do prezentácie funkcionality diagnostického nástroja. Hostiteľské zariadenie popísané v kapitole 6.1 využíva na prenos dát samostatné dátové rámce, čím je nástroj s aktuálnou implementáciou schopný diagnostikovať túto komunikáciu.

Ďalším obmedzením diagnostického nástroja je rýchlosť komunikácie medzi hostiteľským zariadením a pamäťovou kartou, na ktorú je nástroj schopný reagovať. Maximálna rýchlosť komunikácie, ktorá bola počas testovania dosiahnutá bola 250 kHz. Toto obmedzenie vyplýva z návrhu firmwaru a algoritmu spracovania prenášaných správ. Ak firmware prijíma príkaz, celkovo čaká na 6 dátových bytov príkazu. V prípade dátového rámca firmware čaká na prijatie 514 dátových bytov a následne na odoslanie prijatého rámca pamäťovej karte. Týmto spôsobom sú možnosti diagnostického nástroja v rýchlosti diagnostikovanej komunikácií obmedzené na spomínanú hodnotu. Riešenie tohto obmedzenia bude navrhnuté v nasledujúcej kapitole.

Posledným obmedzením nástroja je, že nadradená aplikácia zobrazuje len diagnostické správy pre príkazy. Kvôli časovej tiesni nebol implementovaný buffer pre dátové rámce a ich následné reportovanie firmwarom na nadradenú vrstvu.

7.3 Možné zlepšenia diagnostického nástroja

Návrhy na zlepšenie diagnostického nástroja je nutné rozdeliť na dve možnosti, ktorými je možné pokračovať vo vývoji. Obidve možnosti sú založené na konceptoch diagnostického nástroja diskutovaných v kapitole 3. Prvou možnosťou je zachovanie konceptu diagnostického nástroja tak, ako to bolo popísané v tejto práci. Druhou možnosťou je zmena celého konceptu nástroja a tým spustenie novej iterácie vývoja.

7.3.1 Zachovanie konceptu diagnostického zariadenia

Táto možnosť by sa sústredila na zlepšenie a rozšírenie firmwaru diagnostického zariadenia a následné doplnenie funkcionality nadradenej aplikácie. Hlavnou nevýhodou aktuálnej implementácie firmwaru je nízka rýchlosť komunikácie, na ktorú je nástroj určený. Tento problém vyplýva z prvotného návrhu modulu spracovania správ a komunikácie s hostiteľským zariadením.

Pri prijímaní akejkoľvek správy od hostiteľského zariadenia sú využité blokovacie funkcie, ktoré čakajú na prijatie celej správy. V prípade príkazu nedochádza k veľkému obmedzeniu rýchlosti kvôli celkovému počtu bytov. Naopak, hlavným problémom sú dátové rámce. Riešením by bolo preposielať prijaté dáta správy priamo pamäťovej karte už počas prijímania od hostiteľského zariadenia a naopak. V prípade aktívneho zásahu by firmware do dát zasiahol a pokračoval v komunikácií.

Priestor na zlepšenie aktuálnej realizácie diagnostického nástroja vzniká aj v pridaní ďalších aktívnych zásahov. Nástroj umožňuje pokryť veľké množstvo prípadov zásahu, pretože každá akcia sa začína príkazom od hostiteľského zariadenia. Ak užívateľ nakonfiguruje diagnostiku na konkrétny príkaz, bude schopný simulovať chybovú odpoveď pamäťovej karty, teda určitý chybový stav. Prípady, ktoré niesu pokryté pomocou aktívneho zásahu do odpovedí príkazu v aktuálnej realizácií nástroja sú nasledovné:

- **Zásah do dátových rámcov:** diagnostický nástroj by bol schopný modifikovať aj dáta v prenášaných dátových rámcoch. Problémom, ktorý by bolo potrebné vyriešiť, je konzistencia dát na pamäťovej karte v prípade zásahu do rámca.
- **Zmena rýchlosti komunikácie:** tento zásah by bolo možné realizovať na strane diagnostického zariadenia - pamäťovej karty, pretože diagnostické zariadenie je v úlohe Master. Mohlo by meniť rýchlosti komunikácie s fyzickou pamäťovou kartou podľa požiadaviek užívateľa.
- **Továrenské registre karty:** v prípade zásahu do komunikácie pri zisťovaní továrenských registrov karty by bolo možné simulovať inú veľkosť, iného kódu výrobcu a podobne.

Ďalším možným zlepšením, ktoré kvôli nedostatku času nebolo realizované, je pridanie ďalšieho USB koncového bodu. Tento koncový bod by bol použitý na zisťovanie stavu diagnostického zariadenia z nadradenej aplikácie. Ako príklad informácií o stave diagnostického zariadenia je prítomnosť fyzickej pamäťovej karty, plný buffer na prenášané správy a podobne.

7.3.2 Zmena konceptu diagnostického nástroja

V prípade zmeny konceptu diagnostického nástroja by vývoj viedol na koncept založený na FPGA. Touto zmenou by bolo potrebné zmeniť celý návrh firmwaru, pretože by sa jednalo o úplne inú úlohu založenú na podpore iného hardwaru. Zmenou a realizáciou tohto konceptu by bolo možné využívať diagnostický nástroj pre SD komunikačný mód pamäťovej karty. V porovnaní s aktuálnou realizáciou by bola cena tohto konceptu niekoľko násobne vyššia kvôli potrebnému hardwaru.

7.3.3 Zlepšenia nadradenej aplikácie

Nadradená aplikácia bola implementovaná, tak isto ako aj firmware, ako prototyp na reprezentáciu základnej funkčnosti diagnostického nástroja. Jej hlavné menu je pripravené na implementáciu nových funkcií. Priestor na jej zlepšenie je v prvom rade v jej funkcionalite.

Ako prvú možnosť zlepšenia by bol export diagnostických správ do definovaných formátov, ako napríklad do CSV. Výstup z exportu by mohol byť použitý na ďalšie spracovanie dát.

Užívateľ zadáva v konfigurácií odpoveď na príkaz pomocou číselnej hodnoty odpovede R1. Tento spôsob by mohol byť vylepšený pomocou konfiguračného okna, kde by bol presný popis jednotlivých bitov odpovede. Užívateľ by následne definoval, aký zásah (chybový stav) vyžaduje.

8 Záver

V prvej kapitole sa práca venuje rozboru zadania, ktoré definuje základné pojmy a popisuje koncepciu diagnostiky komunikačného protokolu SD pamäťových kariet. Súčasťou rozboru je definícia požiadaviek kladených na diagnostický nástroj, ktoré vznikali počas úvodných diskusií so zadávateľom práce.

Kapitola 2 je zameraná na teoretickú časť práce. Jej cieľom bolo popísať štandardy definujúce komunikačný protokol SD pamäťovej karty. Kapitola popisuje základnú funkčnosť komunikačného protokolu a správy, ktoré sú prenášané medzi hostiteľským zariadením a pamäťovou kartou.

Ďalej sa práca delí na návrh a následnú realizáciu diagnostického nástroja, ktorý je rozdelený na diagnostické zariadenie a užívateľské rozhranie.

V kapitole 3 boli predstavené koncepty diagnostického nástroja umožňujúceho diagnostiku a prípadný zásah do komunikácie s pamäťovou kartou. Popísané koncepty sú v základe rozdelené na koncept s fyzickou alebo simulovanou kartou. Výstupom tejto kapitoly je výber konceptu diagnostického nástroja s fyzickou pamäťovou kartou založenom na MCU, ktorý bol diskutovaný so zadávateľom práce. Súčasťou tejto kapitoly je popis možností užívateľského rozhrania, ktorým sa stal PC s nadradenou aplikáciou. Výberom tohto konceptu vzniklo obmedzenie nástroja na SPI komunikačný mód pamäťovej karty.

Kapitola 4 sa venuje návrhu hardwaru pre diagnostický nástroj, ktorý primárne pozostáva z hardwaru diagnostického zariadenia, pretože užívateľské rozhranie bolo realizované ako aplikácia bežiaca na PC. Návrh hardwaru je založený na požiadavke realizácie na hardwarovej platforme a na zvolenom koncepte z kapitoly 3. Ako hardwarová platforma na realizáciu diagnostického zariadenia bola zvolená platforma od firmy NXP s označením FRDM-K66F. Súčasťou tejto kapitoly je aj výber rozhrania s hostiteľským zariadením - Sniffer TOL-09419 od firmy SparkFun.

Návrh softwaru diagnostického nástroja je v kapitole 5 rozdelený na návrh firmwaru a nadradenej aplikácie. V kapitole 5.2 bol navrhnutý a popísaný firmware, ktorý je schopný diagnostikovať komunikáciu a aktívne do nej zasahovať podľa požiadaviek užívateľa.

Kapitola 5.3 sa venuje popisu prototypu nadradenej aplikácie, ktorá vytvára rozhranie s užívateľom. Aplikácia bola naprogramovaná v jazyku C# vo frameworku .NET a na komunikáciu s firmwarom bola využitá knižnica LibUsbDotNet. Aplikácia je určená pre PC s operačným systémom Windows.

Kapitola 6 v úvode popisuje testovacie hostiteľské zariadenie, ktoré bolo využité na overovanie funkcionality diagnostického nástroja. Týmto zariadením sa stalo Arduino so slotom na SD pamäťové karty. Následne kapitola popisuje realizáciu na základe softwarového návrhu v predošlých kapitolách.

Výsledkom tejto diplomovej práce je funkčný prototyp diagnostického nástroja, ktorý je schopný zasahovať do komunikácie na základe požiadaviek užívateľa z nadradenej aplikácie. Užívateľ je schopný pomocou nadradenej aplikácie konfigurovať diagnostiku - definovať presnú hodnotu odpovede R1 diagnostického zariadenia na požadovaný príkaz. Pretože každá akcia od hostiteľského zariadenia začína príkazom, užívateľ je schopný simulovať chybové stavy pamäťovej karty ako napríklad: chybná inicializácia karty, odmietnutie zápisu/čítania bloku, neplatný príkaz a podobne.

Firmware diagnostického zariadenia má nedostatky popísané v kapitole 7.2. V rámci tejto kapitoly boli navrhnuté opatrenia, ktoré by nedostatky mohli odstrániť. Výsledné parametre, ktoré má aktuálna realizácia diagnostického nástroja sú nasledovne:

- Diagnostika SPI komunikačného módu
- Hodnota CLK signálu SPI komunikácie maximálne 250 kHz
- Diagnostika single read/write dátových rámcov
- Nadradená aplikácia zobrazuje diagnostické správy pre príkaz

Dôležitým výstupom tejto práce je aj fakt, že realizácia diagnostického nástroja, ktorý by podporoval obidva komunikačné módy a ich plnú rýchlosť komunikácie, je bez hardwarovej podpory takmer nemožná (50 MHz pre SD mód a približne 25 MHz pre SPI mód). Prvým dôvodom je špecifická fyzická vrstva pre SD mód, ktorá využíva 4 dátové signály, 1 dátový signál pre príkazy a 1 pre CLK signál. Druhým dôvodom je, že pri plných rýchlostiach komunikácie musí byť spôsob diagnostiky alebo aktívneho zásahu niekoľkonásobne rýchlejší, ako samotná komunikácia - rádovo desiatky ns. V pr9pade realiz8cie diagnostického nástroja pre SD komunikačný mód v plných rýchlostiach(High Speed mode) bude pravdepodobne nutné zmeniť koncept nástroja a zvoliť FPGA platformu.

Budúcnosť tejto práce je závislá na rozhodnutí zadávateľa. Nasledujúcim krokom bude prezentácia dosiahnutých výsledkov a poznatkov, ktoré budú, v prípade pokračovania, využité v budúcnosti.

Literatúra

[1] SD Card Association: *SD Specifications Part 1 Physical Layer Simplified Specification* [online]. Version 6.00 [cit. 15. 11. 2018]. Dostupné z URL: <https://www.sdcard.org/downloads/pls/>.

[2] SD Card Association: *SD Specifications Part A2 SD Host Controller Simplified Specification* [online]. Version 4.20 [cit. 24. 11. 2018]. Dostupné z URL: <https://www.sdcard.org/downloads/pls/>.

[3] SD Card Association: *Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips* [online]. Revision 2.0 [cit. 5. 12. 2018]. Dostupné z URL: <https://www.usb.org/document-library/usb-20-specification>.

[4] SparkFun Electronics: *SparkFun microSD Sniffer* [online]. TOL-09419 [cit. 18. 12. 2018]. Dostupné z URL: <https://www.sparkfun.com/products/9419>.

[5] SparkFun Electronics: *Freedom FRDM-K66F Development Platform User's Guide* [online]. Rev 0 [cit. 23. 12. 2018]. Dostupné z URL: <https://www.nxp.com/docs/en/user-guide/FRDMK66FUG.pdf>.

[6] libusb: *A cross-platform user library to access USB devices* [online]. Dostupné z URL: <https://libusb.info>.

[7] IAR Systems AB: *IAR Embedded Workbench IDE User Guide* [online]. Rev 0 [cit. 29. 12. 2018]. Dostupné z URL: <http://supp.iar.com/FilesPublic/UPDINFO/004916/arm/doc/EWARM_UserGuide.ENU.pdf>.

[8] NXP Semiconductors: *FRDM-K66F: Freedom Development Platform for Kinetis* [online]. Rev 0 [cit. 4. 1. 2019]. Dostupné z URL: <https://www.nxp.com/support/developer-resources/evaluation-and-development-boards/freedom-development-boards/mcu-boards/freedom-development-platform-for-kinetis-k66-k65-and-k26-mcus:FRDM-K66F>.

[9] NXP Semiconductors: *MCUXpresso SDK Builder* [online]. Rev 0 [cit. 18. 1. 2019]. Dostupné z URL: <https://mcuxpresso.nxp.com/en/welcome>.

[10] LibUsbDotNet: *LibUsbDotNet* [online]. Rev 2.2.8 [cit. 5. 4. 2019]. Dostupné z URL:
<https://mcuxpresso.nxp.com/en/welcome>.

Zoznam symbolov, veličín a skratiek

SD	Secure Digital
CLK	Clock
SPI	Serial Peripheral Interface
SDA	SD Card Association
CRC	Cyclic redundnancy check
RCA	Relative card address
CS	Chip select
MSB	Most significant bit
LSB	Least significant bit
MCU	Microcontroller unit
FPGA	Field Programmable Gate Array
MHz	Megahertz
kHz	Kilohertz
RAM	Random Access Memory
MB	Megabyte
KB	Kilobyte
USB	Universal Serial Bus
CPU	Central Processing Unit
PC	Personal Computer
μ**s**	mikrosekunda
ms	milisekunda
ns	Nanosekunda
GPIO	General-purpose input/output
SDIO	Secure Digital Input Output
HW	Hardware
GUI	Graphical User Interface
DMA	Direct Memory Access
CSV	Comma Separated Values

Zoznam elektronických príloh

A Text diplomovej práce
B Firmware diagnostického zariadenia
C Nadradená aplikácia
D Prezentačné video