VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

KNIHOVNA PRO KOMUNIKACI MIKROKONTROLERU KINETIS K60 S SD KARTOU

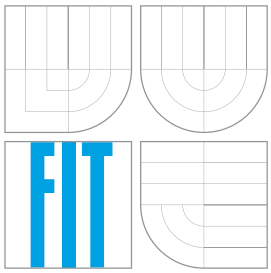
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

OLIVER NEMČEK

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

KNIHOVNA PRO KOMUNIKACI MIKROKONTROLERU KINETIS K60 S SD KARTOU

LIBRARY FOR COMMUNICATION BETWEEN KINETIS K60 MICROCONTROLLER AND SD CARD

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — OLIVER NEMČEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÁCLAV ŠIMEK
SUPERVISOR

BRNO 2015

Abstrakt

Cílem této bakalářské práce je navrhnout a sestavit knihovnu pro komunikaci s SD kartou. Platforma, pro kterou je knihovna určená, je vývojová deska osazená mikrokontrolorem Kinetis K60 od společnosti Freescale. Knihovna implementuje ovladač hostitelského řadiče sběrnice SD, který je zabudován v mikrokontroloru. Rozhraní knihovny musí umožňovat její jednoduchou použitelnost. Funkcionalita knihovny je demonstrovaná vzorovou aplikací pro komunikaci s SD kartou.

Abstract

The purpose of the Bachelor's thesis is to design and build library for communication with SD card. Target platform of library is a development board with Kinetis K60 microcontroller by Freescale Semiconductors. The library implements SD host controller driver, which is embedded in microcontroller. Interface of the library must be easy to use. Functionality is demonstrated by sample application for communication with SD card.

Klíčová slova

Knižnica, SD, SDHC, SD karta, Kinetis, K60, Freescale, Minerva, radič SDHC, mikrokontróler

Keywords

Library, SD, SDHC, SD card, Kinetis, K60, Freescale, Minerva, SD host controller, microcontroller

Citace

Oliver Nemček: Knihovna pro komunikaci
mikrokontroleru Kinetis K60 s SD kartou, bakalářská práce, Brno, FIT VUT v Brně, 2015

Knihovna pro komunikaci mikrokontroleru Kinetis K60 s SD kartou

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Václava Šimka

.....
Oliver Nemček
19. mája 2015

Poděkování

Chtěl bych podekovat pánu Ing. Šimkovi za užitečné rady a připomínky při řešení této bakalářské práce.

Obsah

Kapitola 1

Úvod

K popredným vlastnostiam Secure Digital (SD) karty patrí jej nízka cena, rýchlosť, spoľahlivosť a kompaktnosť. To z nej robí ideálny typ média na ukladanie dát v prenosných zariadeniach. Technológia SD kariet sa vyvíja od roku 1999 až do súčasnosti. Štandard SD má niekoľko verzií a v každej revízii sa pridávajú kartám nové vlastnosti a možnosti. Typicky sa tiež zvyšuje rýchlosť a kapacita kariet. Najčastejšie sa stretávame s kartami pre ukladanie dát, tzv. pamäťovými kartami. Okrem iného existujú aj iné typy špecializovaných kariet, ktoré využívajú SD protokol. K týmto druhom patria napríklad Wi-Fi alebo GPS karty.

Témou tejto bakalárskej práce je zostaviť sadu funkcií, ktoré umožnia obsluhovať pamäťovú kartu typu SD, prípadne Secure Digital High Capacity (SDHC). Naša cieľová platforma je vývojová doska Minerva disponujúca mikrokontrolerom od spoločnosti Freescale – Kinetis K60. Predpokladáme, že knižnica bude použitá v rôznych študentských projektoch, preto sa budeme snažiť prispôsobiť programové rozhranie tak, aby nevznikal problém pri jej využití.

V nasledujúcich kapitolách si predstavíme hardware, ktorý sa nachádza na vývojovej doske a podrobne sa budeme venovať technológii SD karty. Postupne uvedieme možnosti a funkcie modulu Secure Digital Host Controller (hostiteľský radič SD zbernice), ktorý je integrovaný priamo v mikročipe. Popíšeme komunikačný protokol pamäťovej karty a rozdiely medzi verziami SD a SDHC. V druhej časti práce popíšeme funkcie knižnice a demonštrujeme integráciu s knižnicou súborového systému FAT – FatFS[1]. V závere zhodnotíme dosiahnuté výsledky a podáme návrhy na vylepšenie knižnice.

Kapitola 2

SD karta

V tejto kapitole si predstavíme SD karty, ich konštrukciu a funkcionalitu. Informácie sú zhrnuté z viacerých zdrojov, predovšetkým z oficiálnej špecifikácie SD verzie 2.0 [3]. Táto špecifikácia je štandard, ktorým sú zaviazaní všetci výrobcovia kariet aj hostiteľských radičov. Verzia 2.0 proti verzii 1.0 a 1.10 prináša podporu High Capacity kariet, teda kariet s vysokou kapacitou a takisto nové vstupno-výstupné rozhranie SDIO[1]. Dodatočné poznatky sme čerpali z produktových manuálov výrobcov pamäťových kariet Sandisk [2] a Toshiba [4].

2.1 Krátka história

SD karta je evolučný nástupca karty MMC[2]. Štandard SD bol vytvorený a schválený v roku 1999 asociáciou SD. Jej zakladajúcimi členmi boli veľké spoločnosti SanDisk, Toshiba a Matsushita Electric(Panasonic). Skratka SD znamená *Secure Digital*, pretože oproti MMC kartám boli pridané obvody slúžiace na zabezpečenie dát pred neautorizovaným čítaním a zapisovaním. Z toho mal ťažiť hlavne hudobný priemysel, ktorý by využíval karty SD ako nosič nahrávok. Zatiaľ čo šifrovanie a ochrana dát sa v praxi používa výnimočne, samotná karta sa dočkala veľkej popularity. V priebehu času bolo vydaných niekoľko štandardov, posledný vo verzii 4.10 vydaný v roku 2013.

Odhaduje sa, že technológia SD bola implementovaná viac ako 400 výrobcami a má vyše 8000 produktových modelov[3]. Teoretická kapacita karty sa pohybuje v rozhsahu 1 MB až 2 TB. Spoločnosť SanDisk v roku 2014 uviedla na trh najväčšiu kartu, ktorá má neuveriteľných 512GB[10].

Od roku 2006 poskytuje SD Association zjednodušenú verziu štandardu na svojich webových stránkach. Vynechané v nej boli detaily týkajúce sa prevažne bezpečnostných funkcií karty. To umožnilo vytvoriť plnohodnotné ovládače SD kariet pre rôzne iné zariadenia. Dovtedy sa open-source ovládače tvorili iba pomocou reverzného inžinierstva proprietárnych ovládačov. Kompletný štandard SD sa poskytuje za $2000/rok. Bohužiaľ, na trhu sa objavujú aj repliky kariet renomovaných výrobcov, ktoré nemajú správne značenie. Nielen, že majú nižšiu rýchlosť prenosu ale aj menšiu kapacitu pamäte ako je napísané na výrobku.

[1] Wi-Fi, Bluetooth, GPS, ...
[2] MultiMediaCard
[3] Prevzaté z: https://www.sdcard.org/join/glossary/

2.2 Fyzické parametre

Z konštrukčného hľadiska sa SD karta skladá z nevolatilnej pamäti typu Flash a zabudovaného mikroradiča. Zaujímavým zistením, ktoré publikovala hackerská dvojica na Chaos Computer Congress[8], je fakt, že značná časť výrobcov používa namiesto špecializovaného radiča procesor ARM, prípadne upravený procesor Intel 8051. Skupina dokázala využitím reverzného inžinierstva spustiť na mikroprocesore vlastný podvrhnutý kód, čím by sa dal efektívne realizovať MITM[1] útok. Ďalšou možnosťou, ako sa uvádza na stránke skupiny, je využitie SD karty ako lacného procesora s integrovanou flash pamäťou pre hardwarových nadšencov.

Pre úplnosť informácií si uvedieme základné údaje o karte SD

- Kapacita pamäte až do 2 GB(SDSC) alebo 32 GB(SDHC)
- Napäťový rozsah: 2.7 – 3.6V
- Frekvenčný rozsah
  - až 25 MHz (Standard-Speed)
  - až 50 MHz (High-Speed)
- Rýchlosť prenosu dát:
  - až 12.5 MB/s (Standard-Speed)
  - až 25 MB/s (High-Speed)
- Mechanický prepínač na karte pre ochranu proti prepisu dát
- Zabudovaná podpora pre detekciu vloženia a odstránenia karty do slotu
- Šírka dátovej zbernice: 1 alebo 4 bity
- Komunikačné rozhranie: SD bus alebo SPI

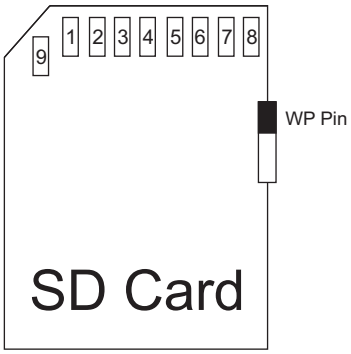
V priebehu času sa objavila potreba karty zmenšovať. Vznikli tak zmenšené typy SD kariet označované ako miniSD a microSD. Líšia sa predovšetkým v rozmeroch. Drobných zmien sa dočkalo aj fyzické rozhranie. MiniSD karta pridáva 2 nepripojené piny do strednej časti konektora, zatiaľ čo microSD karte chýba 1 pin, ktorý na SD karte plnil funkciu uzemnenia.(SD karta má 2 uzemnenia). Túto nekompatibilitu odstraňujú adaptéry, ktoré obsahujú len pasívne elektronické obvody. Po ostatných stránkach sú tieto zmenšené karty identické s klasickou SD kartou štandardnej veľkosti.

2.3 Rozhranie SD karty

Na puzdre karty sa nachádza 9 signalizačných vodičov. V závislosti na zvolenom type komunikácie sa môžu využívať v režime SPI alebo v režime SD Bus. Nasledujúci obrázok ilustruje tvar karty, komunikačné piny a ich číslovanie.

Všimnime si, že na bočnej strane karty sa nachádza mechanický prepínač ochrany pamäte pred zápisom. Takýmto zariadením nedisponovala žiadna iná karta v dobe vzniku SD

[1] Man In the Middle - typ útoku. Útočník je v strede medzi komunikujúcimi stranami, odpočúva a mení dátový tok medzi stranami



Obr. 2.1: Tvar SD karty a číslovanie konektorov

karty. Bohužiaľ, ako si uvedieme neskôr, tento prepínač je len mechanický. Zamedzenie zápisu na kartu je výhradne v kompetencii hostiteľského radiča alebo ovládača. Podrobnosti sa nachádzajú v sekcii 2.8.

Každá karta vie podľa potreby pracovať buď v režime SD Bus alebo SPI. Režim SPI je podporovaný z dôvodu kompatibility a takisto z dôvodu licenčných podmienok asociácie SD. Ďalšie obmedzenia režimu SPI zahŕňajú nižšiu prenosovú rýchlosť a redukovanú množinu príkazov protokolu SD. Licenciou zaťažený protokol SD Bus podporuje všetky príkazy a okrem 1-bitového režimu prenosu dokáže pracovať v 4-bitovom režime, viď funkcia konektorov v tabuľke 2.1, čím sa teoretická prenosová rýchlosť zvyšuje 4-krát. Maximálne prenosové rýchlosti sú uvedené v kapitole 2.2.

Pin	SD Názov	SD Bus funkcia	SPI Názov	SPI funkcia
1	DAT3/CD	Data linka 1/Detekcia karty	CS	Selekčný signál
2	CMD	Príkaz/odozva	DI	Data vstup
3	V_{SS1}	Napájanie – uzemnenie	VSS	Napájanie – uzemnenie
4	V_{DD}	Napájanie + 2.7–3.6V	VDD	Napájanie + 2.7–3.6V
5	CLK	Hodinový signál	SCLK	Hodinový signál
6	V_{SS2}	Napájanie–uzemnenie	VSS2	Napájanie–uzemnenie
7	DAT0	Data linka 0	DO	Data výstup
8	DAT1	Data linka 1	RSV	—
9	DAT2	Data linka 2	RSV	—

Tabuľka 2.1: Rozdielne zapojenie konektorov pri režime SD Bus a SPI podľa normy SD

V tejto bakalárskej práci sa budeme zaoberať výhradne rozhraním SD Bus. Cieľová platforma disponuje hostiteľským radičom SDHC, preto využijeme práve túto natívnu zbernicu. Druhým pozitívom je ušetrenie jedného SPI modulu pre iné účely.

2.4 Zbernica SD Bus

Komunikácia na zbernici SD Bus je založená na súbore bitov s presne definovanou štruktúrou, ktoré sú oddelené začiatočným a koncovým bitom. Komunikáciu vždy iniciuje hostiteľské zariadenie, nie karta. Špecifikácia odporúča, aby každá karta bola pripojená na vlastnú nezdieľanú zbernicu. Pokiaľ hostiteľský systém podporuje režim vysokej rýchlosti

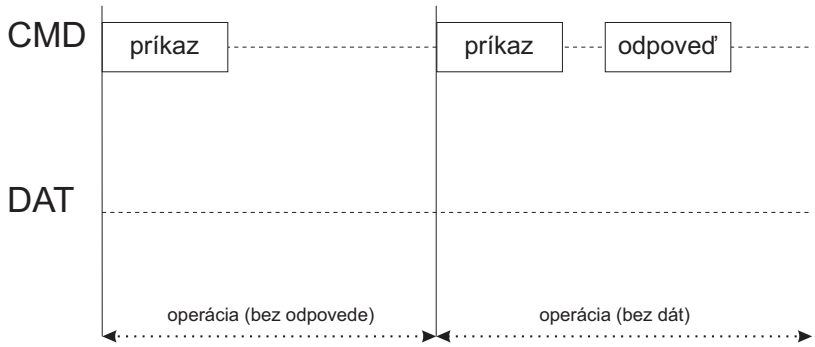
(High Speed), tak je to povinnosť.

Rozlišujeme tri typy správ, ktoré sa môžu objaviť na zbernici:

1. **Príkaz**: využíva iba *CMD* vodič, používa sa na posielanie príkazov smerom z hostiteľského systému do karty
2. **Odpoveď** na príkaz: využíva iba *CMD* vodič rovnako ako príkaz, avšak odpoveď smeruje vždy z karty smerom do hostiteľského zariadenia
3. **Dáta**: Využívajú sa len dátové vodiče označené *DATx*, komunikácia je obojsmerná ale v jednom čase dokáže prenášať dáta len jedným smerom.

Výhodou oddelenia vodiča pre riadenie a dáta je možnosť zasielať karte správy počas dátového prenosu. Praktické využitie ukončovania dátového prenosu pomocou príkazu *CMD12* si popíšeme neskôr.

Ochrana proti chybám prenosu je zabezpečená na riadiacom vodiči aj dátových vodičoch. Využíva algoritmus CRC-7 na riadiacom vodiči a CRC-16 na dátových vodičoch. Presné polynómy pre výpočet sú uvedené v špecifikácii[3]. Pri 4-bitovom prenose sa CRC-16 počíta nezávisle pre každý dátový vodič *DAT*.

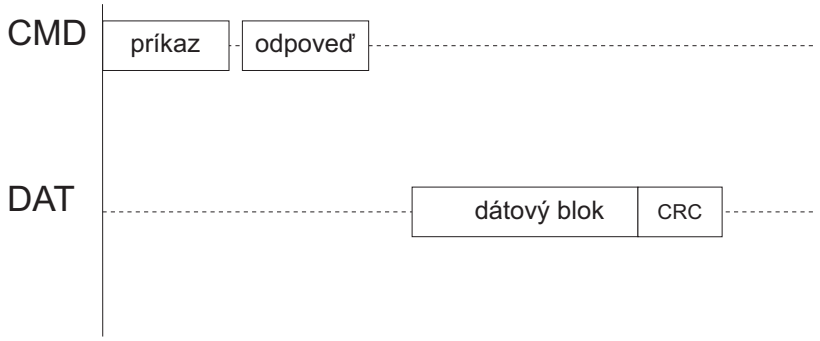


Obr. 2.2: Príklad komunikácie s kartou. Zaslanie príkazu bez odpovede a s odpoveďou.

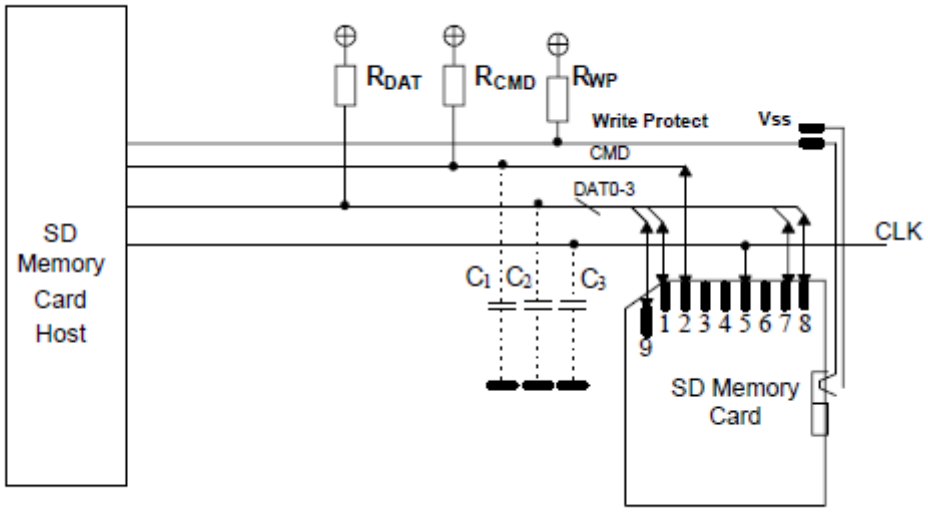
Pri komunikácii s kartou musí hostiteľské zariadenie ako prvé zaslať príkaz karte. Karta na tento príkaz odošle odpoveď. Jednoduchá vizualizácia komunikácie je zobrazená na obrázku 2.2. Signalizácia na vodiči *CMD* je sériová a 1-bitová. Pri popise príkazov uvedieme, ktoré príkazy využívajú len prenos na riadiacom vodiči a ktoré využívajú aj prenos na dátovom, prípadne dátových vodičoch. Pre úplnosť je na obrázku 2.3 demonštrovaný prenos dát na dátových vodičoch spolu s riadiacimi príkazmi ako nasledujú za sebou v priebehu času.

Produktový manuál[2] ukazuje schému zapojenia karty na zbernicu SD. Výňatok sa nachádza na obrázku 2.4 Pre pochopenie ďalšieho výkladu je dôležité uviesť konkrétne zapojenie zbernice, aby bolo možné poukázať na princípy detekcie prítomnosti karty.

Na obrázku 2.4 sú vyznačené odpory R_{DAT}, R_{CMD} a R_{WP}. Ich úlohou je znižovať spotrebu elektrického prúdu a takisto zamedziť kolísaniu napätia na zbernici pri pripojovaní alebo odpojovaní karty. Zbernica je navrhnutá tak, že v stave nečinnosti sa signály nachádzajú v stave logickej „1“. Signalizačné napätie na zbernici sa nachádza v rozmedzí 0V až 3.3V. Karta je najčastejšie napájaná napätím 3.3V. Mala by byť ale schopná pracovať aj na operačnom napätí 2.7V(záleží na napäťovom okne v registri *OCR*). V novších verziách



Obr. 2.3: Príklad komunikácie s kartou. Karta odpovedá na príkaz a posiela dáta.



Obr. 2.4: Zapojenie karty SD do zbernice SD Bus. Prebraté z [2].

štandardu sa pre dosiahnutie vyšších rýchlostí používa signalizačné napätie iba 1.8V, resp. 0.4V.

2.5 Štruktúra príkazu a odpovede

2.5.1 Príkaz

Príkazy protokolu SD majú predom stanovený pevný formát. Celková dĺžka príkazu je 48 bitov.

Tabuľka 2.2 vysvetľuje význam jednotlivých polí. Smer prenosu je nastavený vždy na hodnotu '1', čo znamená že príkaz vždy posiela hostiteľské zariadenie. Hodnoty označené ako 'x' môžu byť nastavené podľa potreby. Pre číslo príkazu je vyhradená 6-bitová hodnota. Parameter príkazu sa zapisuje do poľa argument a má šírku až 32-bitov. Kontrolný súčet CRC7 sa nachádza na konci štruktúry.

Špecifikácia uvádza nasledovné 4 typy príkazov:

- **BC**: Broadcast, všetky karty na zbernici príkaz príjmu a vykonajú

Pozícia	47	46	[45:40]	[39:8]	[7:1]	0
Šírka	1	1	6	32	7	1
Hodnota	0	1	x	x	x	1
Význam	štartovací bit	smer prenosu	index príkazu	argument	CRC7	ukončovací bit

Tabuľka 2.2: Formát príkazu SD protokolu

- **BCR**: Broadcast, všetky karty na zbernici príkaz príjmu a po vykonaní odošlú odpoveď
- **AC**: Adresovaný režim, príkaz prijme a po vykonaní odošle odpoveď adresovaná karta
- **ADTC**: Adresovaný režim, príkaz prijme a po vykonaní odošle odpoveď spolu s dátami adresovaná karta, tento typ prenosu sa používa na prenos dát

2.5.2 Odpoveď

Odpoveď nemá jednotný formát. Namiesto toho existuje až 6 typov odpovedí, ich veľkosť je buď 48 alebo 136 bitov. Dĺžka aj formát odpovede na jednotlivé príkazy sú dopredu známe, takže poznáme počet bitov, ktoré máme prijať, resp. ako interpretovať prijatú odpoveď.

Jednotlivé typy sa označujú skratkami *Rx*.

- **R1**: 48-bitov. Najčastejší typ odpovede. Správa obsahuje index príkazu, ku ktorému je viazaná, nasledovaná je 32-bitovým stavovým slovom, ktorým je možné kontrolovať výskyt chyby pri vykonávaní príkazu
- **R1b**: Je identický s odpoveďou R1, ale využíva signál(*DAT0*) indikujúci, že zariadenie je zaneprázdnené vykonávacím príkazu.
- **R2**: 136-bitov. Používa sa na prenos 128-bitový interných registrov karty(*CID*, *CSD*).
- **R3**: 48-bitov. Prenos registra *OCR*.
- **R6**: 48-bitov. Prenos registra *RCA* spolu s niekoľkými bitmi stavového registra.
- **R7**: 48-bitov. Odpoveď na príkaz *CMD8*. Pridané v špecifikácii verzie 2.00. Používa sa pri inicializácii SDHC karty.

Pre korektnú prácu s odpoveďou je kľúčové poznať presnú pozíciu bitov v prenesenej odpovedi. Formáty odpovedí sú dohľadateľné v špecifikácii a do tejto práce ich nebudeme uvádzať z priestorových dôvodov. Nemenej dôležitou informáciou je, že prenos odpovede sa uskutočňuje ako prenos n-bitového slova (n=48 alebo n=136), ktoré začína MSB(najdôležitejším) bitom a končí LSB(najmenej dôležitým) bitom. Všetky registre karty sa prenášajú touto konvenciou. Pri spracovaní odpovede na cieľovej platforme musíme vedieť, ako sa bity do pamäte ukladajú, aby mohli byť korektne interpretované. Význam bitov pri prenose dát si uvedieme neskôr.

2.6 Základné príkazy

Množina príkazov, ktoré podporujú SD karty je rozdielna. Každý príkaz patrí do jednej z 11 tried. Niektoré triedy sú rezervované pre budúce použitie. Podľa normy musia karty

implementovať príkazy niekoľkých tried. K základnému ovládaniu pamäťovej karty stačí len niekoľko príkazov, ktoré sa nachádzajú v tejto množine. Odpadá tak nutnosť testovať, ktoré množiny príkazov karta podporuje.

Popri základných príkazoch sú definované tzv. aplikačne špecifické príkazy označované *ACMDx*. Pred poslaním takéhoto príkazu musí byť karta inštruovaná, že nasledujúci príkaz má interpretovať ako aplikačne špecifický. To sa docieli zaslaním príkazu *CMD55*, ktorý prepína kartu do spomínaného stavu.

V tabuľke 2.3 uvádzame základnú množinu príkazov aj s popisom ich funkcie. Kompletný zoznam sa nachádza v špecifikácii SD[3].

Vzhľadom na fakt, že táto práca sa týka predovšetkým pamäťových SD kariet, najpodstatnejšie príkazy sú *CMD17* a *CMD24* realizujúce vstupno-výstupné operácie pre jediný dátový blok. Ekvivalentom pre skupinu dátových blokov sú príkazy *CMD18* a *CMD25*. Ich správne použitie bude uvedené v sekcii 2.11.

2.7 Detekcia prítomnosti karty

Aby bolo možné kartu vložiť do SD slotu hostiteľského zariadenia počas behu hostiteľského systému, tak ako to poznáme z technológie usb zariadení, musí existovať spôsob ako zdetekovať prítomnosť karty na zbernici SD. Rovnaký princíp sa uplatňuje aj pri odstránení karty zo slotu. Vychádzame zo zapojenia ukázaného na obrázku 2.4.

Existujú dve riešenia tohoto problému. Prvou možnosťou je využiť interný pull-up rezistor SD karty, ktorý je pripojený na *DAT3*. Pri priložení napätia na napájací konektor karty bude napätie nulové na všetkých pinoch okrem *DAT3*. Táto metóda vyžaduje, aby bol rezistor, ktorý je pripojený na vodič *DAT3* odstránený zo zapojenia zbernice. Jeho funkciu prevezme zabudovaný pull-up rezistor. V prípade potreby karta umožňuje pomocou špeciálneho príkazu tento rezistor ovládať.

Princíp druhého spôsobu je využitie prídavných signálov slotu, do ktorého sa SD karta zapája. Táto metóda funguje na základe mechanického zariadenia, ktoré musí SD slot obsahovať. Niektoré hostiteľské radiče majú možnosť pripojiť signál detekcie karty *CD*[1], čím sa efektívne dokáže zistiť prítomnosť karty v čítacom zariadení. Ak radič SD karty nedisponuje možnosťou pripojenia tohoto signálu, môže sa na detekciu použiť GPIO[2] rozhranie.

Výrobci a samotná pracovná skupina SD Association odporúčajú vo svojich materiáloch používanie druhého spôsobu a detekciu karty vykonávať pomocou GPIO rozhrania[7]. Dôvodom je spätná kompatibilita s MMC[3] kartami, ktoré nemajú interný pull-up rezistor a nedajú sa detekovať prvým spôsobom. Tento spôsob dokáže identifikovať aj prázdny SD adaptér v slote SD.

[1] CardDetect signál
[2] General Purpose Input Output
[3] MultiMediaCard - predchodca SD kariet

2.8 Ochrana karty proti zápisu a kopírovaniu

SD karta disponuje súborom technológii, ktoré zabezpečujú ochranu dát pred neautorizovaným čítaním aj zápisom. Najznámejší je mechanický posuvník na boku karty. Okrem neho sú v karte elektronické obvody implementujúce komplexné ochrany dát. V tejto sekcii si niektoré z nich bližšie popíšeme.

Funkciou posuvného prepínača 2.1, ktorý sa nachádza na boku SD karty, je signalizovať, že sa na danú kartu nesmie zapisovať žiaden obsah. Táto ochrana je založená na mechanickej detekcii prítomnosti zarážky v jej spodnej polohe. Zodpovednosť za detekciu preberá slot SD karty. Signál zo slotu musí byť dovedený do hostiteľského systému. Niektoré sloty nedisponujú týmto signálom. Analogicky, záleží na konkrétnom radiči, či podporuje tento signál. Samotné blokovanie operácie zápisu na základe prepínača je avšak v kompetencii ovládača modulu hostiteľského systému. Teda ak sa tvorca ovládačov rozhodne, že nechce dodržiavať špecifikáciu, môže na kartu zapísať dáta aj v prípade, že je posuvník nastavený v polohe ktorá to zakazuje.

Karta umožňuje pomocou špecializovaných príkazov prepínať medzi zabezpečeným a nezabezpečeným režimom. Táto ochrana je elektronická a stará sa o ňu radič karty. Pokiaľ je to vhodné, dá sa karta v zabezpečenom režime uzamknúť natrvalo.Potom nie je možné vrátiť kartu do pôvodného stavu. Ak sa karta prepne do zabezpečeného režimu, odmieta akékoľvek operácie zápisu ale povoľuje čítanie z karty.

V prípade, že je nutné ochrániť obsah proti čítaniu aj zápisu, dá sa nastaviť heslo tvorené až 16 bajtami. Zaheslovaná karta interaguje so systémom tak ako normálna karta ale odmieta všetky operácie čítania aj zápisu. Karta môže byť odheslovaná len použitím pôvodného hesla alebo celkovým zmazaním všetkých dát.

Karty(plne kompatibilné so špecifikáciou) disponujú technológiou, ktorá prináša podporu DRM[1]. Táto funkcia chráni obsah karty pred neautorizovanými operáciami, ochraňuje obsah, na ktorý sa vzťahuje autorské právo. Pôvodne sa malo jednať o ochranu hudobných diel pre hudobné vydavateľstvá. Táto technológia sa používa vo výnimočných prípadoch a nenašla svoje masové uplatnenie. Z tohoto dôvodu sa niektorí výrobcovia rozhodli vynechať technológiu DRM v svojich kartách.

Podľa dostupných informácií v súčasnosti neexistuje účinná technika na zlomenie hesla (alebo DRM) iná ako hrubou silou. Avšak trh je zaplavený lacnými kartami, hlavne z Číny[6], ktoré takéto ochrany neposkytujú. Niektoré karty dokonca umožňujú prepisovať interné registre(register CSD obsahuje informácie o kapacite), preto existujú 64 GB karty, ktoré sú fyzicky len 8 GB.

2.9 Interné registre

Karta obsahuje sadu registrov, ktoré udávajú jej stav a operačné vlastnosti. Popis registrov je nasledujúci.

- **CID** - Card identification. Obsahuje identifikačné údaje o karte ako sériové číslo, dátum výroby alebo kód výrobcu
- **RCA** - Relative card address. Obsahuje relatívne číslo karty, ktorým sa musí karta adresovať príkazom *CMD7*

[1] Digital Right Management

- **CSD** - Card Specific Data. Kapacita karty, frekvencia, spotreba karty, veľkosť dátového bloku.
- **SCR** - SD Configuration Register. Dopĺňa CSD o informácie o šírke zbernice a bezpečnostné parametre karty.
- **OCR** - Operation conditions register. Prevádzkové napätie.
- **SSR** - SD status register. Obsahuje chybové informácie o aktuálne vykonanom príkaze. Tento register tvorí obsah odpovede *R1*
- **CSR** - Card Status register. 512-bitový register. Obsahuje stavové informácie o proprietárnych vlastnostiach SD kariet.

Z hľadiska funkcie sú najdôležitejšie registre *RCA* a *CSD*. Prvý register obsahuje adresu, ktorú musíme uviezť v parametri príkazu *CMD7* pri adresácii karty. Karta sa prepne z režimu *Stand-by* do režimu *Transfer*. Pokiaľ sa adresa nezhoduje s hodnotou uloženou v registri *RCA* karta prechádza z ľubovoľného stavu do stavu *Stand-by*.

CSD slúži na výpočet fyzickej kapacity karty.

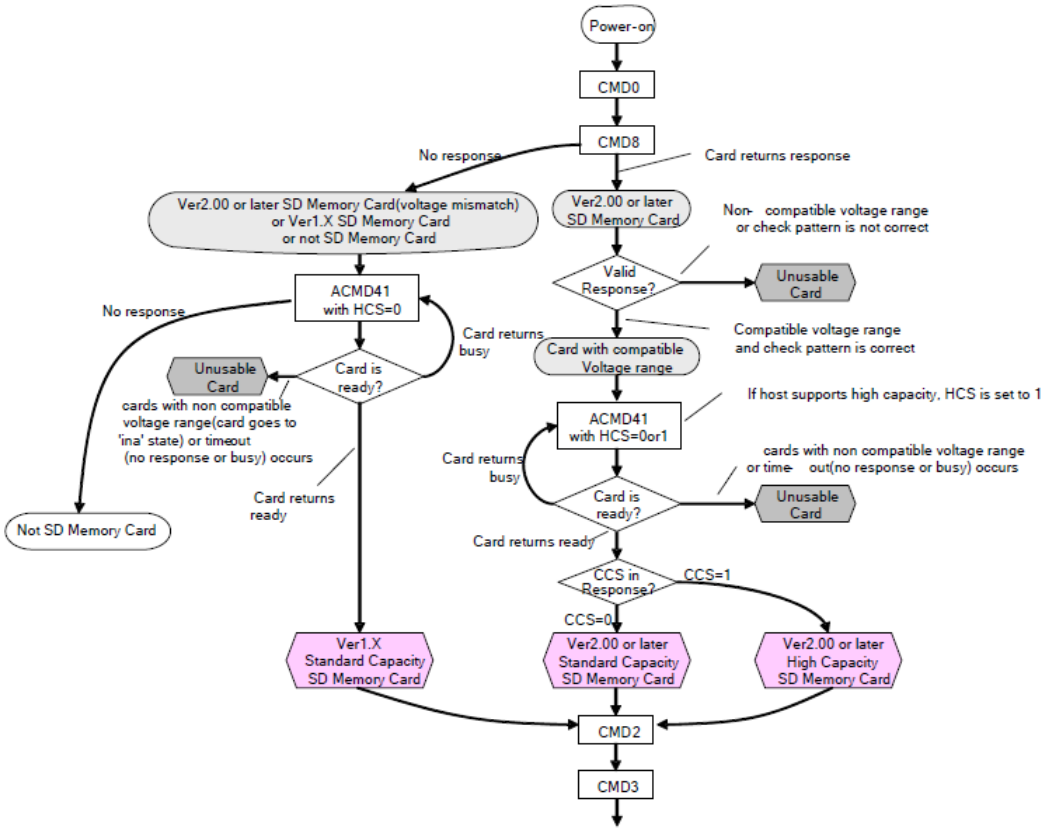
2.10 Konečný automat a inicializácia karty

Riadenie karty zabezpečuje stavový automat. Vstupná abeceda automatu sú príkazy, ktoré posielame karte. Špecifikácia jasne určuje korektné vstupy pre dané stavy. Aby sa karta mohla správne zinicializovať, musí dostať správnu sekvenciu príkazov. Do počiatočného stavu automatu sa dostávame zaslaním príkazu *CMD0*, vid obrázok 2.5. Zároveň týmto príkazom vyberáme režim komunikácie s kartou. Ak je *DAT3/CD* pri poslaní príkazu *CMD0* uzemnený (logická „0“), dôjde k prechodu karty do režimu SPI. Ak je na vodiči napätie(logická „1“), karta ostáva v natívnom režime SD Bus. Karta prechádza do stavu nečinnosti označovaného ako *Idle*.

Obrázok 2.5 popisuje priebeh inicializácie karty v režime SD Bus. Všimnime si, že tok programu je rozdelený na dve vetvy pomocou príkazu *CMD8*. Tento príkaz bol pridaný vo verzii SD v2.00 na odlíšenie kariet SD v1.xx a v2.00. Karta verzie 1.xx neodpovedá na tento príkaz, dôjde k uplynutiu časovej doby vyhradenej na prijatie odpovede. Radič hostiteľského zariadenia to detekuje a nastaví príznak *Timeout*. Parameter príkazu *CMD8* obsahuje 4-bitové pole, ktoré môže nadobúdať iba jedinú hodnotu, ktorá označuje operačné napätie karty 2.7–3.6V. Konkrétna bitová hodnota je „0001“. Parameter takisto obsahuje 8-bitové pole, ktoré plní funkciu testovacieho vzoru. V odpovedi na príkaz musí byť vzor nastavený na rovnakú hodnotu aká bola nastavená v parametri príkazu. Ak táto požiadavka nie je splnená, karta s najväčšou pravdepodobnosťou nie je schopná korektne pracovať. Je poškodená či nefunkčná alebo nie je schopná funkcie na zvolenom napätí.

Druhým príkazom potrebným pri inicializácii karty je aplikačne špecifický príkaz *ACMD41*, pred ktorým je potrebné poslať *CMD55*. Parametrom pri zasielaní *ACMD41* je register OCR, viď obrázok 2.6, v ktorom sa nachádza napäťové okno[1]. Jednotlivé bity označujú napäťové úrovne, ktoré je systém schopný karte dodávať. Bit 30 označovaný ako *HCS* indikuje či je systém schopný obsluhovať kartu typu SDHC. Odlišnosti medzi kartami typu SD a SDHC budú vysvetlené v ďalších kapitolách.

[1] V špecifikácii je označované ako „voltage window“



Obr. 2.5: Inicializácia karty v režime SD. Prebrané z [3].

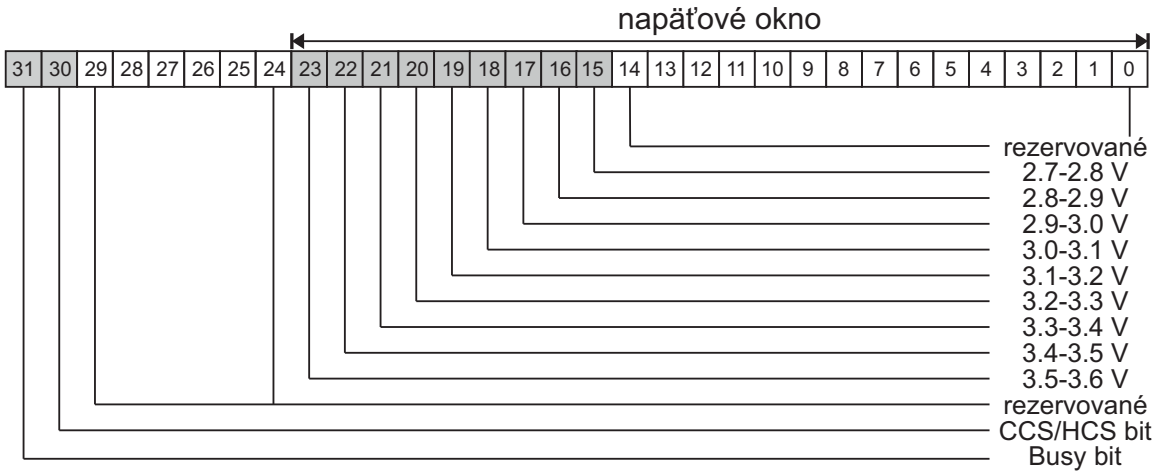
Karta potvrdí zvolený napäťový rozsah v odpovedi tým, že odošle napäťové okno s rovnako nastavenými hodnotami. V prípade, že v príkaze *ACMD41* uvedieme nulové okno, karta odpovie napäťovým oknom, na ktorom je schopná pracovať. Tento spôsob sa používa v systémoch, ktoré obsahujú regulátor napätia. Systém zvolí najnižšiu možnú úroveň z dôvodu šetrenia spotreby energie a vyššej komunikačnej rýchlosti.

Ak je bit *Busy* nastavený na hodnotu 0, znamená to, že karta nie je pripravená na používanie. Opätovným posielaním príkazu *ACMD41* môžeme zisťovať stav karty. Odporúčaná doba dotazovania sa na stav je až 1 sekunda. Ak karta ani po tejto dobe nie je pripravená(*Busy* bit je stále nastavený na 0), potom nie je možné kartu používať a to buď z dôvodu chybného nastavenia bitu *HCS*, nevyhovujúcemu napätovému oknu, alebo má karta internú závadu.

CCS bit(Card Capacity Status) môže byť testovaný iba v prípade, že sme karte odoslali príkaz s nastaveným bitom *HCS*. Bit indikuje či je karta typu SDHC alebo SDSC[1]. Ak karta vyhovuje štandardu SDHC potom je tento príznak nastavený a ovládač radiča musí s týmto faktom počítať. V prípade, že karta je typu SDHC ale v príkaze *ACMD41* nebol nastavený bit *HCS*, môže sa karta chovať ako SDSC (obmedzená kapacita do 1GB) alebo odmietne dokončiť vlastnú inicializáciu tým, že nenastaví bit *Busy* na hodnotu 1.

V tomto kroku sme zistili o karte najnutnejšie informácie. Poznáme verziu špecifikácie, ktorej karta vyhovuje a takisto či je typu SDSC alebo SDHC.

[1]SDSC - SD Standard Capacity



Obr. 2.6: Význam bitov registera OCR

Ďalší príkaz, *CMD2*, slúži na identifikáciu karty. Obsahom odpovede je 128-bitový register *CID*. Nachádza sa v ňom sériové číslo karty, kód výrobcu, produktové označenie, revízne číslo a dátum výroby.

Posledný povinný príkaz *CMD3* sa používa na vygenerovanie adresy, ktorá bude pridelená karte. Toto číslo označované *RCA* podľa registra, ktorý ho uchováva, je dôležité v prípade, že je na zbernicu pripojených viacero kariet. Aby bolo možné adresovať konkrétnu kartu, musí byť hodnota interného registra *RCA* a parameter príkazu *CMD7* v zhode. Iba adresovaná karta smie pristupovať k dátovej časti zbernice. Všetky ostatné karty, ktoré sú pripojené ku zbernici prejdú do stavu *Stand-by*.

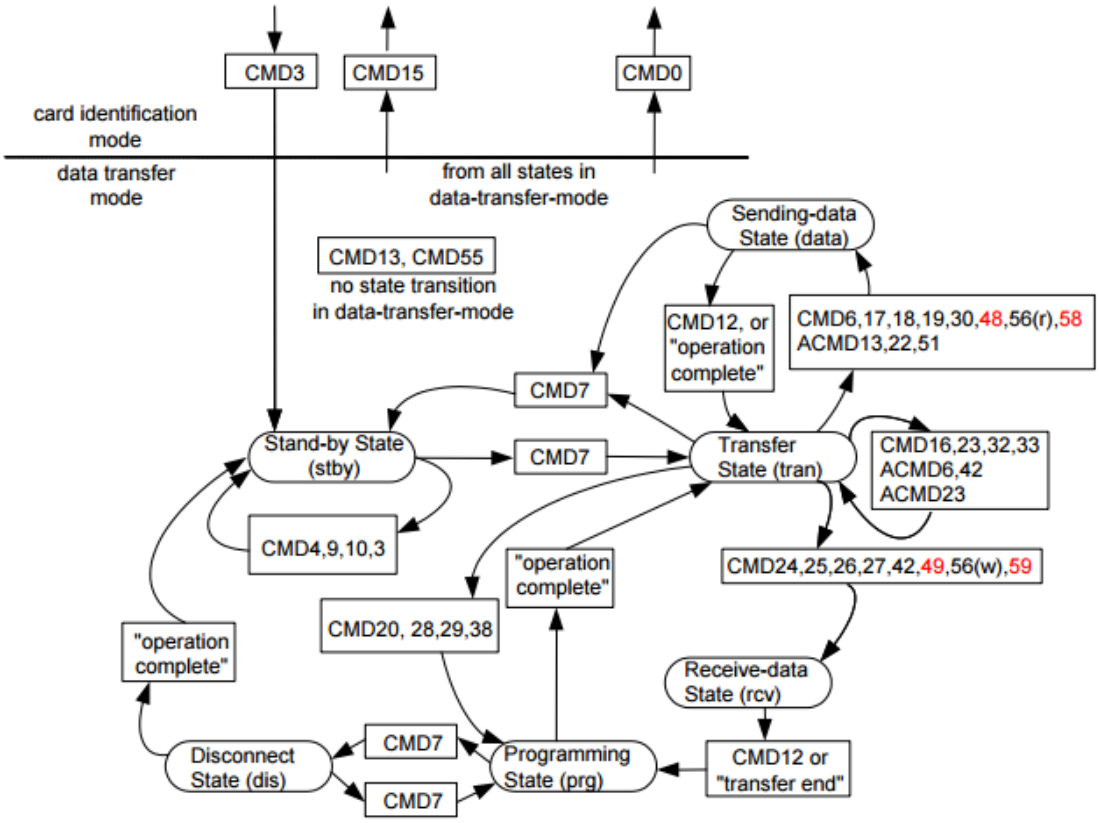
Dôležitým parametrom vo fáze inicializácie je frekvencia hodinového signálu. Tá nesmie prekročiť hodnotu 400kHz. Po úspešnom dokončení tejto fázy môže hostiteľské zariadenie nastaviť takt až 25MHz alebo 50MHz, podľa možností karty.

Na obrázku 2.7 je stavový automat tak ako je uvedený v špecifikácii. Po vykonaní príkazu *CMD3* karta prechádza do stavu *Stand-by*, v ktorom zostáva do doby, kým nie je adresovaná. Pre autora hostiteľského ovládača radiča je stavový automat na obrázku 2.7 najdôležitejší. Príkazy popisujú do akého stavu sa karta prepne po spracovaní príkazu. Najviac času karta strávi v stave *Stand-by State* alebo v jednom zo stavov prenosu dát: *Transfer State*, *Sending-data State* a *Receive-data State*.

2.11 Dátový prenos

Po úspešnej inicializácii karty a nastavení doplnkových parametrov prenosu ako sú šírka dátového prenosu a frekvencia, ktorou bude karta taktovaná, je karta pripravená na operácie čítania a zápisu. V nasledujúcom výklade predpokladáme, že karta pripojená na zbernicu je adresovaná, teda nachádza sa v stave *Transfer State*. Karta posiela dáta v bloku. To znamená, že pri jednej vstupno-výstupnej operácii sa prenesie skupina bajtov. SDSC karty disponujú funkciou nastaviteľného počtu bajtov v bloku. Karty typu SDHC od verzie 2.00 používajú fixný počet bajtov v jednom bloku. Ten je nastavený na 512 bajtov (táto veľkosť sa implicitne používa aj v kartách SDSC).

Všimnime si, tabuľka 2.3, že príkazy čítania aj zápisu majú dve varianty. Prvý typ číta/zapíše len jeden blok dát, zatiaľ čo druhý typ pracuje so sériou blokov. Parametrom



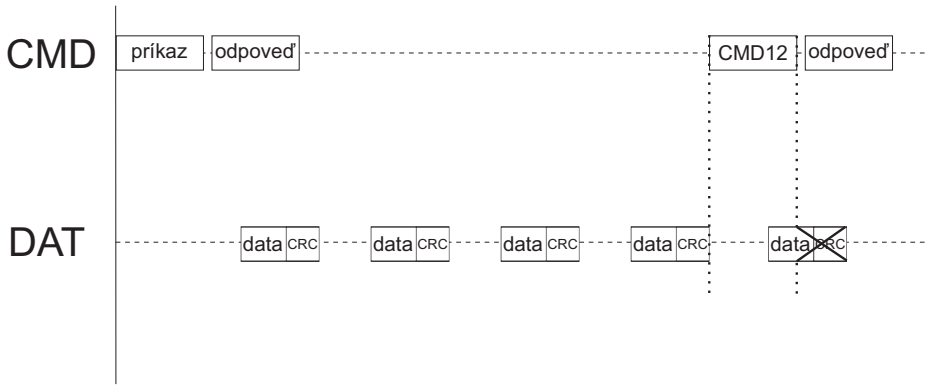
Obr. 2.7: Stavový automat operácií SD kart. Prevzaté zo špecifikácie[3].

príkazov je adresa, na ktorej sa vykoná požadovaná operácia.

Pri používaní operácií pre prenos jedného bloku je schéma nasledovná. Pred prenosom dát sa musí karta prepnúť do stavu *Transfer State*. Potom sa iniciuje prenos dát zvoleným smerom použitím príkazu čítania alebo zápisu pre jeden blok. Karta prechádza do stavu prijímania alebo posielania dát. Po dátovom spoji sa prenesie jeden dátový blok. V prípade úspešného prenosu sa karta prechodmi *operation complete* dostáva späť do stavu *Transfer State*, viď obrázok 2.7.

Pri vyvolaní operácie viacnásobného čítania2.9 alebo zápisu sa započne prenos dát od adresy, ktorá je uvedená v parametri príkazu *CMD18* pre čítanie, resp. *CMD25* pre zápis. Adresa sa s každým preneseným blokom inkrementuje. Dáta sa odosielajú, pokiaľ hostiteľské zariadenie nezastaví prenos odoslaním príkazu *CMD12*. Po prijatí príkazu sa karta prepína do stavu *Transfer state* a čaká na ďalší príkaz, viď schéma 2.7. Takto je možné urýchliť prenos veľkých objemov dát, pretože nedochádza k príliš veľkej réžii pri prenose. Pokiaľ by sa využíval pre prenos n blokov príkaz pre prenos jedného bloku, bolo by nutné pred každým transferom poslať karte príkaz, teda dokopy n príkazov. Pri tomto type prenosu stačí na prenos n blokov odoslať príkaz jeden.

Všimnime si, že pri odoslaní príkazu *CMD12* počas prenosu bloku dát sa tento prenos okamžite ukončuje. Dáta, ktoré sa nestihli odoslať, sú odstránené z prenosovej fronty. Karta sa okamžite prepína do stavu *Transfer state* a odosiela odpoveď na príkaz. Radič SD karty na hostiteľskom zariadení musí detekovať, že blok nebol prenesený celý a načítané dáta zahodiť.

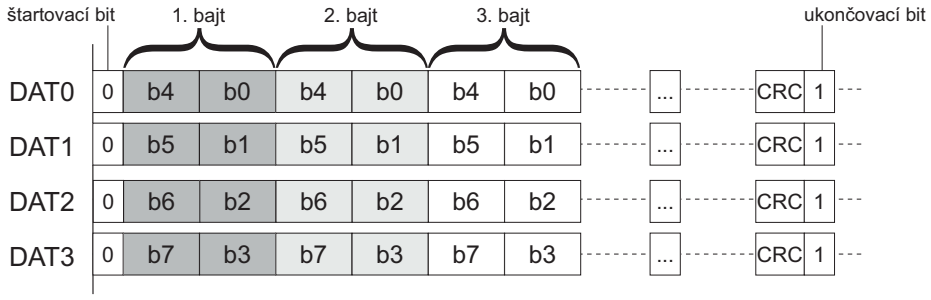


Obr. 2.8: Ilustrácia čítania 4 blokov dát v režime viacnásobného prenosu.

Operácia zápisu dát na kartu SD je v skutočnosti mierne zložitejšia. Po prijatí bloku dát karta začne programovať svoju dátovú pamäť, čo trvá určitú dobu. Počas tejto doby nie je možné poslať karte príkaz, ktorý by inicioval dátový prenos (čítanie, zápis). Jediná povolená komunikácia môže prebiehať výlučne po riadiacej linke *CMD*. Indikátorom, že karta sa nachádza v stave *Programming State* je, že karta drží dátový vodič *DAT0* v stave „log. 0“(nízke napätie). Akonáhle karta dokončí programovanie pamäte a prejde do stavu *Transfer State*, dátový vodič je nastavený na „log. 1“. Pre kontrolu stavu karty počas programovania sa používa príkaz *CMD13*, ktorý po riadiacej linke presunie obsah stavového registra *SSR*. Príznak `READY_FOR_DATA` signalizuje, že karta je alebo nie je pripravená na prenos dát.

Prenos dát je zabezpečený algoritmom CRC-16. Na konci každého bloku sa prenesie kontrolný súčet, ktorým je možné overiť správnosť prenesených dát. Pri zápise na kartu kontroluje CRC karta, pri čítaní z karty kontroluje CRC hostiteľský radič.

Ako bolo spomenuté, dátový prenos môže byť vedený po jednom dátovom spoji(1-bitový). Každá karta by však mala podporovať aj prenos po 4 dátových vodičoch(4-bitový). Paralelne sa tak dá posielať až 4-krát väčšie množstvo dát, čo efektívne zrýchľuje prenosovú rýchlosť 4-krát.



Obr. 2.9: Formát prenosu dát pri šírke zbernice 4-bity. Prebratá a upravená verzia zo špecifikácie [3]

Pri dátovom prenose karta odosiela najprv bajt s nižšou adresou, potom s adresou o 1 vyššou, atď. To odpovedá konvenciám „Little Endian“. Postupnosť bitov v bajte je nasledovná: prvý sa odošle najvýznamnejší bit(MSB) a posledný sa odosiela najmenej významný bit(LSB). Obrázok 2.8 zobrazuje radenie bitov na jednotlivých vodičoch v prípade, že sa využíva zložitejší 4-bitový režim. Za povšimnutie stojí fakt, že kontrolný súčet sa počíta pre

každú dátovú linku separátne.

2.12 SDSC vs. SDHC

S príchodom typu SDHC sa výrazne zvýšila kapacita kariet. Maximálna kapacita štandardných SDSC kariet je určená na 2 GB, zatiaľ čo vysoko kapacitné SDHC karty majú limit až 32 GB. Fyzické parametre oboch typov SDSC a SDHC nie sú rozdielne. Majú rovnakú veľkosť aj počet a funkciu konektorov rozhrania. Rozdiel spočíva v inom chovaní pri prístupe k dátam a vyššej pracovnej frekvencii karty SDHC. Ako sme si už spomínali, pri vstupno-výstupných operáciách sa ako parameter príkazu uvádza adresa(bloku) v pamäti. SDSC karty adresujú dáta na úrovni bajtov. Vzhľadom na fakt, že parameter príkazu je 32-bitová hodnota, maximálna teoretická veľkosť adresovanej pamäte je 4 GB. Nový štandard používa stále 32-bitovú šírku parametra príkazu z dôvodu zachovania kompatibility. Rozdielna je adresácia pamäte, preto treba pri obsluhe karty rozlišovať, či operujeme s kartou štandardnej alebo vysokej kapacity. SDHC karty adresujú pamäť po blokoch 512 bajtov.

Rozdielna adresácia pamäte		
Typ	Adresa	Popis
SDSC	0001h	Bajt na adrese 0001h
	0200h	Bajt na adrese 0200h
SDHC	0001h	Blok na adrese 0001h, rozsah bajtov(0x200h–0x3FFh)
	0200h	Blok na adrese 0001h, rozsah bajtov(0x40000h–0x401FFh)

Adresácia po blokoch veľkosti 512 bajtov, a šírka parametra 32-bitov umožňuje maximálnu teoretickú kapacitu rovnajúcu sa 2 TB = $512*2^{32}$ B.

Ďalšie rozdiely boli už spomenuté v iných kapitolách. SDSC karty podporujú prístup na adresy, ktoré nemusia byť zarovnané na 512 bajtov. Umožňujú taktiež zvoliť si veľkosť jedného prenášaného bloku dát. Tieto možnosti boli pri tvorení špecifikácie SDHC kariet odstránené.

S akým typom karty hostiteľský systém pracuje je možné rozlíšiť podľa odpovede na príkaz *CMD8*. Tento príkaz bol zavedený kvôli starším zariadeniam a zámerne prináša nekompatibilitu s SDSC kartami. Staršie zariadenia by totiž mohli pracovať s SDHC kartou tak ako s SDSC kartou a nebrali by v úvahu rozdiely medzi adresovacími režimami. Detailnejšie informácie sa nachádzajú v sekcii 2.10.

Index	Typ	Parameter	Odp.	Skratka	Popis
CMD0	bc	[31:0]='x'	–	GO_IDLE_STATE	Reset do počiatočného stavu
CMD2	bcr	[31:0]='x'	R2	ALL_SEND_CID	Karta odošle CID
CMD3	bcr	[31:0]='x'	R6	SEND_RELATIVE_ ADDR	Karta odošle RCA
CMD7	ac	[31:16]=RCA [15:0]='x'	R1b	GO_IDLE_STATE	Adresácia/ deadresácia karty
CMD8	bcr	[31:12]=rezervované [11:8]=pracovné napätie [7:0]=testovací vzor	R7	SELECT/ DESELECT_CARD	Príkaz operačných podmienok karty
CMD9	ac	[31:16]=RCA [15:0]='x'	R2	SEND_IF_COND	Karta odošle CSD.
CMD12	ac	[31:0]='x'	R1b	STOP_ TRANSMISSION	Vynútené ukončenie prenosu
CMD17	adtc	[31:0]=adresa dát	R1	READ_SINGLE_ BLOCK	Čítanie jedného bloku
CMD18	adtc	[31:0]=adtc	R1	READ_MULTIPLE_ BLOCK	Čítanie viacerých blokov
CMD24	adtc	[31:0]=adtc	R1	WRITE_SINGLE_ BLOCK	Zápis jedného blokov
CMD25	adtc	[31:0]=adtc	R1	WRITE_MULTIPLE_ BLOCK	Zápis viacerých blokov
CMD55	ac	[31:16]=RCA [15:0]='x'	R1	APP_CMD	Informuje kartu, že nasledujúci príkaz je ACMD.
ACMD6	ac	[31:2]='x' [1:0]=šírka zbernice	R1	SET_BUS_WIDTH	Definuje šírku dátovej časti zbernice
ACMD13	adtc	[31:0]='x'	R1	SD_STATUS	Odošle SD Status register
ACMD41	bcr	[31]=rezervované [30]=HCS bit [29:24]=rezervované [23:0]=napäťové okno	R3	SD_SEND_OP_COND	Zašle karte informácie o podpore SDHC kariet a hodnoty napätia, ktoré dokáže systém dodávať karte.

Tabuľka 2.3: Prehľad dôležitých príkazov ('x' znamená ľubovoľná hodnota)

Kapitola 3

Cieľová platforma

V tejto kapitole oboznámime čitateľa s platformou, pre ktorú vznikala knižnica pre prácu s SD kartami. Vymenujeme komponenty, ktoré sú k dispozícii, ich funkciu a popis použitia.

3.1 Vývojový kit Minerva

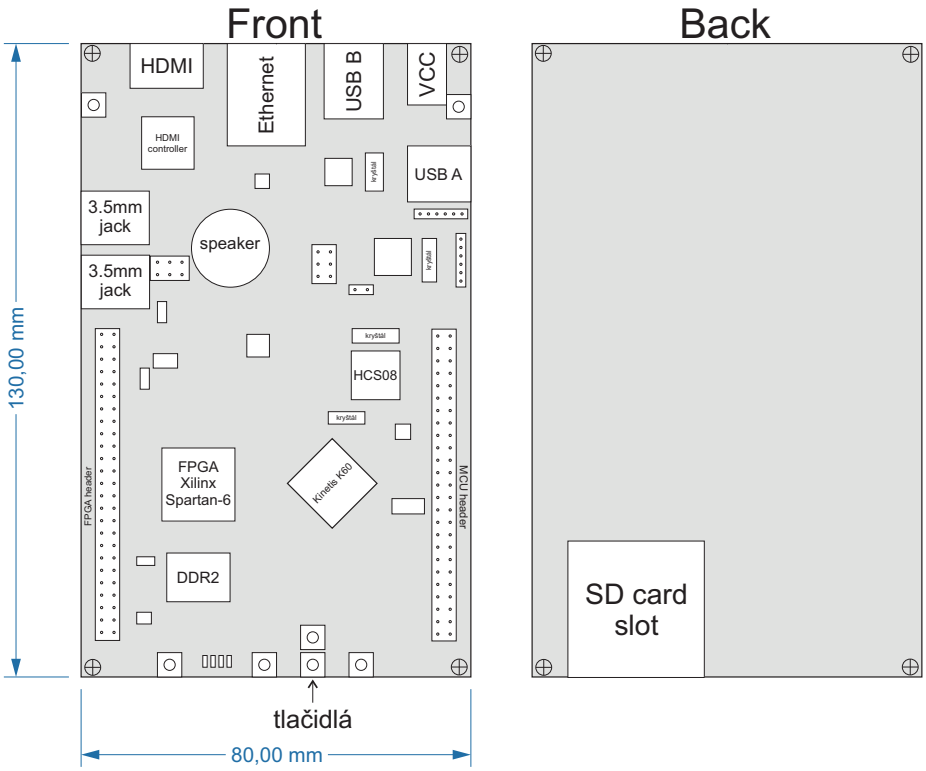
Naša cieľová platforma je vývojová doska v rozmeroch zhruba dvoch kreditných kariet vedľa seba. Je osadená mikrokontrolérom od spoločnosti Freescale – MK60DN512VMD10. Jedná sa o mikrokontrolér rady K60, ktorý je osadený s integrovanou 512 kB Flash pamäťou pre program. Pracovná frekvencia dosahuje až 100 MHz. Jadro mikrokontroléru tvorí moderný 32-bitový ARM Cortex-M4, vysokovýkonný jednojadrový procesor s nízkou spotrebou. Mikrokontrolér obsahuje množstvo modulov, ktoré rozširujú jeho použitie. RTC, Ethernet, CRC, ADC, DAC, USB, CAN, GPIO, UART a množstvo iných modulov je pripravených priamo od výrobcu. Pre potreby tejto bakalárskej práce je najdôležitejší modul SDHC(SD Host Controller).

Platforma vznikla ako náhrada za dosluhujúci prípravok FITkit, ktorý svojimi hardwarovými súčiastkami nepatrí medzi najaktuálnejšie vývojové dosky. Vývoj začal v roku 2012 a o rok na to bola k dispozícii funkčná vzorka. Spoluautorstvo na vývoji má aj vedúci tejto práce pán Ing. Václav Šimek[1].

Podobne ako FITkit, aj Minerva je osadená jednotkou FPGA od firmy Xilinx Spartan-6. Disponuje rýchlou DDR2 pamäťou o kapacite 512 kB pripojenou na FPGA čip. Okrem iného obsahuje HDMI port, ethernetový konektor, USB typu A a B, niekoľko tlačidiel, a rozhraním GPIO.

V súčasnej dobe sa mikropočítače tešia veľkej obľube. Príkladom môže byť projekt Raspberry Pi, ktorý na trh vstupoval s myšlienkou lacného a ekonomického mikropočítača pre nadšencov v cene nepresahujúcej 35 dolárov. Zaujímavým faktom je, že operačný systém sa do týchto typov zariadení zavádza na SD karte. Možno niekedy v budúcnosti bude naša knižnica plniť funkciu bootloadera pre operačný systém, ktorý sa bude nachádzať na SD karte.

[1] Informácie čerpané zo stránky: http://www.fit.vutbr.cz/research/prod/index.php.cs?id=290



Obr. 3.1: Podoba vývojového kitu Minerva.

3.2 Vývojové prostriedky

Spoločnosť Freescale dodáva k svojim mikrokontrolérom integrované vývojové prostredie CodeWarrior. Aktuálna verzia na webových stránkach spoločnosti je CodeWarrior 10.6. Toto prostredie umožňuje vyvíjať software napísaný v jazyku C alebo C++, spúšťať preložený kód na zariadení a debugovať ho.

V roku 2014 sa spoločnosť rozhodla uviesť na trh nový balík vývojového prostredia určeného pre jej procesory rady Kinetis. KDS alebo Kinetis Design Studio je prispôsobený pre miktrokontroléry Kinetis, ktoré obsahujú jadrá ARM Cortex. Hlavičkové súbory obsahujú množstvo nových funkcií a konštánt, ktoré uľahčujú prácu pri vývoji. CodeWarrior nevyhovuje štandardu CMSIS[1], ktorý je určený pre procesory s jadrami ARM Cortex a slúžia ako hardwareovo nezávislá vrstva pre prístup k registrom, resp. funkciám jadra Cortex.

Komunikácia vývojového prostredia je zabezpečená pomocou osvedčeného open-source rozhrania OSBDM, ktoré obsahuje procesor od spoločnosti Freescale HCS08. Procesor je zapojený na debugovacie porty mikrokontroléra a do počítača sa pripája pomocou zbernice USB.

Počas testovania oboch vývojových prostredí CodeWarrior aj KDS sme sa stretli s drobnou nekompatibilitou, hlavne pri prostredí KDS. Niekedy si odstránenie chyby vyžadovalo reštartovanie pracovnej stanice. Z toho dôvodu sme sa rozhodli knižnicu tvoriť v programe CodeWarrior, ktorý oproti KDS obsahuje aj funkciu *Register View*, ktorá zobrazuje obsah interných registrov procesora spolu s ich popisom na úrovni jednotlivých bitov. Činnosť programu pri vývoji sa tým pádom ľahšie kontroluje. V druhej fáze preportujeme knižnicu

[1] Cortex Microcontroller Software Interface Standard

pre používanie v prostredí KDS. Budeme sa snažiť, aby knižnica spĺňala požiadavky oboch prostredí.

Zaujímavým doplnkom pri vývoji je nástroj ProcessorExpert. Slúži pre rýchly a jednoduchý vývoj aplikácií. Okrem iného obsahuje už ovládač modulu SDHC. Po preskúmaní kódu sme zhodnotili, že je príliš zložitý a nejasný.

Na internete sa nachádzajú zaujímavé a jednoduché implementácie ovládačov SDHC pre mikrokontroléry Kinetis[1]. Ich spoločným rysom je fakt, že nepoužívajú pre prenos radič DMA ani viacnásobný prenos dátových blokov. Vynechanie týchto techník dramaticky znižuje rýchlosť prenosu dát z alebo do karty SD.

Dňa 5. 5. 2015 uvoľnila spoločnosť Freescale na svojich webových stránkach Kinetis Design Studio vo verzii 3.0.0. Táto verzia v sebe integruje nástroj na zobrazovanie registrov periférii, čím ponúka alternatívu k *Register View*. Knižnicu sa pokúsime preportovať aj pod toto vývojové prostredie.

[1] Príklad https://github.com/laswick/kinetis/blob/master/phase2_embedded_c/sdhc.c

Kapitola 4

Modul SDHC

Hostiteľský radič zbernice SD tvorí rozhranie medzi procesorom a kartou SD. Tvorca štandardu SD, SD Association, vydáva podrobnú špecifikáciu nie len pre výrobcov SD kariet ale aj pre konštruktérov hostiteľských modulov SD. Štandard vo verzii 2.00 je dostupný tu[5]. Tento dokument slúži iba ako návod pre tvorbu radiča, ako príručka pri návrhu. V žiadnom prípade to nie je záväzný dokument. Spoločnosť Freescale svoj radič v mikrokontroléri MK60DN512VMD10 označuje ako „*Freescale SDHC version 2.2*“. Tento radič je kompatibilný so špecifikáciou pre hostiteľské zariadenia vo verzii 2.00. Okrem podpory SD kariet dokáže obsluhovať aj evolučne staršie MMC karty. Pod SD protokolom pracuje s pamäťovými kartami, SDIO kartami a špeciálnym typom – combo kartami. Combo karty majú spojenú funkcionalitu pamäťovej karty a SDIO karty.

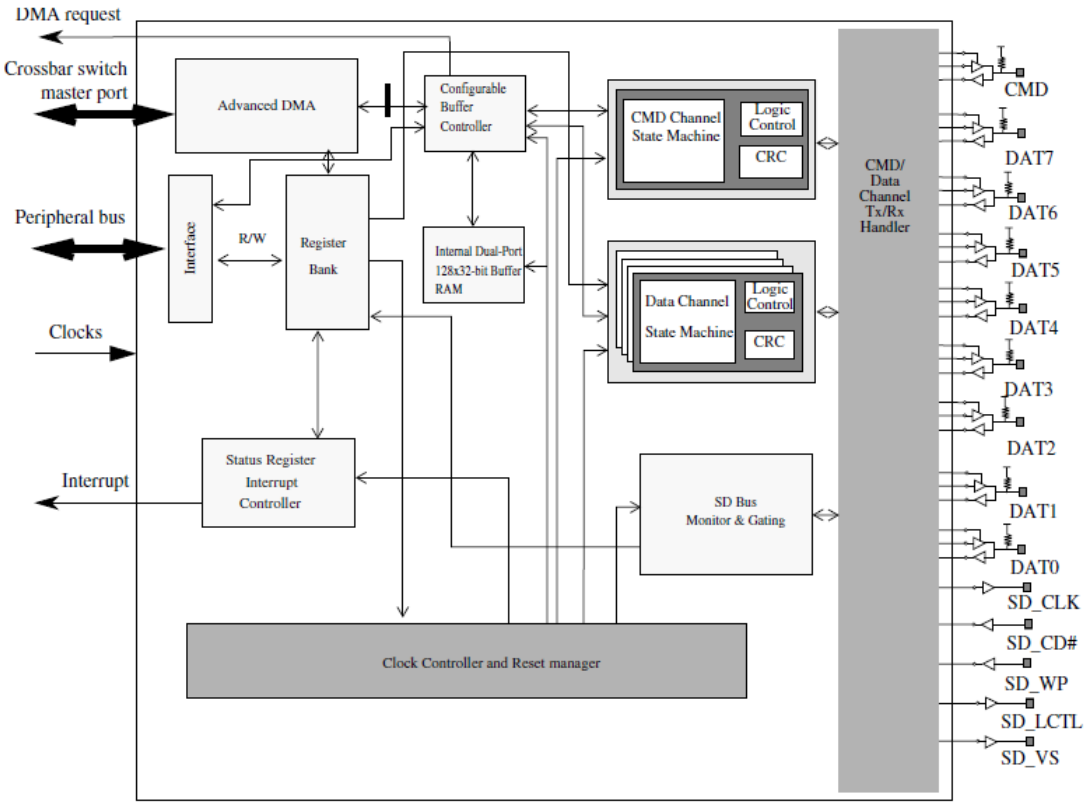
Najdôležitejšie vlastnosti radiča SD:

- Vyhovuje dokumentu SD Host Controller Standard Specification verzie 2.00 [5]
- Podpora MMC kariet štandardu 4.2/4.3
- Podpora SD kariet štandardu 2.00 a vysoko-kapacitných SDHC kariet
- Podpora SDIO kariet verzie 2.00
- Frekvencia hodinového signálu až 52 MHz
- Podpora pre 1-bitový, 4-bitový a 8-bitový režim(iba MMC)
- Automatické zasielanie príkazu *CMD12*
- Nastaviteľná veľkosť bloku dát v rozmedzí 1–4096 bajtov
- Podpora pre prenos dát pomocou radiča DMA

Maximálna teoretická rýchlosť pri 4-bitovom režime a SD karte činí až 200 Mbps $\approx$ 25 MB/s. Táto maximálna rýchlosť sa dá dosiahnuť pri frekvencií 50 MHz(High Speed) a pri prenose viacerých blokov naraz.

4.1 Fyzické rozhranie

Modul je pripojený na 10 vodičov, ktoré vedú k externým konektorom nachádzajúcim sa na púzdre mikrokontroléra. Časť z nich sa nachádza na porte E, zvyšok je zapojený na port

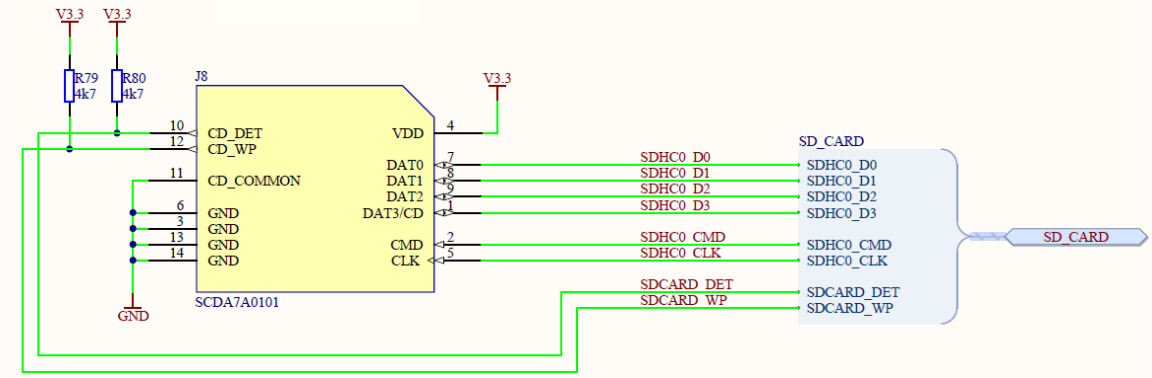


Obr. 4.1: Bloková schéma radiča SD kariet, ktorý je osadený na platforme Minerva. Prevzaté z [11]

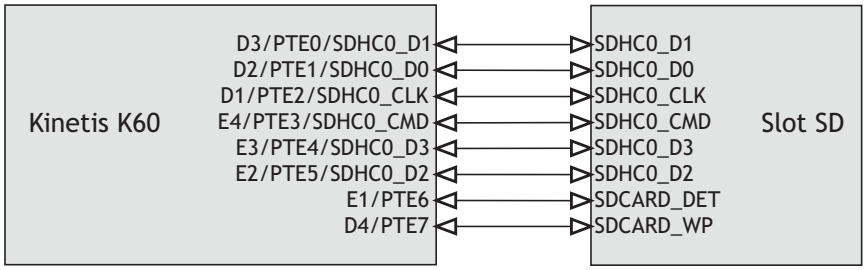
D. Tieto piny môžu byť pridelené rôznym periférиám, preto je nutné zvoliť ich funkciu pred tým ako budú používané. Obrázok 4.1 zobrazuje blokovú schému radiča. Konektory zbernice SD sa nachádzajú na pravej strane obrázka. Signály označené ako *SD_LCTL*, *SD_VS* a *SD_WP* sa na tomto type radiča nenachádzajú. Zvyšné vodiče sú vyvedené až k slotu pre SD kartu. Podľa schémy zapojenia vývojovej dosky Minerva vieme zistiť, ktoré signály sú pripojené, na ktoré konektory na SD slote[1]. Výňatok zo schémy je na obrázku 4.2. Všimnime si, že slot SD má zapojené iba 4 dátové vodiče označené ako *DATx*, preto sa nebude dať používať 8-bitový režim pre MMC karty. Signály na obrázku 4.2 označené ako *SDCARD_DET* a *SDCARD_WP* nie sú vyvedené do radiča SDHC, namiesto toho sú pripojené na GPIO port E. Ukážka zapojenia signálov na strane mikrokontroléra je na obrázku 4.3.

Vychádzame z odporúčaného zapojenia zbernice SD podľa obrázku 2.4. Všimnime si, že na vodičoch zbernice sa nenachádza tzv. pull-up rezistor(odpor, ktorý drží úroveň napätia v „log. 1“). Z tohoto dôvodu je absolútne nevyhnutné použiť interné odpory, ktoré obsahuje mikrokontrolér a to na všetky signály zbernice okrem *SDHC0_CLK*, pretože inak by hodinový signál nekmital.

[1]Kompletná schéma k prípravku minerva sa nachádza na priloženom CD



Obr. 4.2: Schéma zapojenia SD slotu na platforme Minerva. Zdroj: Schéma vývojovej dosky Minerva(príloha CD)



Obr. 4.3: Schéma zapojenia SD slotu na mikroprocesor.

4.2 Úroveň abstrakcie

Nespornou výhodou integrovaného radiča zbernice SD je abstrahovanie tvorcov ovládača SD od najnižšej fyzickej úrovne. Radič sám rozpoznáva napäťové hodnoty na zbernici, povoľuje alebo pozastavuje hodinový signál, vykonáva CRC kontrolu, detekuje udalosť vypršania času(Timeout), disponuje vyrovnávacou pamäťou s veľkosťou až 4096 bajtov. Obsahuje stavové automaty pre príkazovú linku a pre každú dátovú linku. Detekcia udalosti zapojenia karty do zbernice SD sa realizuje monitorovaním signálu *DAT3*, tak ako bolo vysvetlené v kapitole 2.7. Aby nedošlo k pretečeniu alebo podtečeniu vyrovnávacích pamätí pri operácii, ktorá zahŕňa prenos dát, radič automaticky kontroluje hodinový signál a v prípade potreby ho pozastavuje.

Dôležitou súčasťou je integrovaný DMA radič, ktorý dokáže významne urýchliť prenos dát z vyrovnávacej pamäte do operačnej pamäte. Existujú až 3 typy DMA režimov. Prvý, označovaný SDMA(Simple DMA) je najjednoduchší. Dokáže presúvať dáta z vyrovnávacej pamäti na iné miesto špecifikované adresou, prípadne opačne. Druhý a tretí režim ADMA a ADMA2(Advanced DMA) využíva dátovú štruktúru, v ktorej sa nachádzajú deskriptory, ktoré určujú koľko bajtov sa má presunúť na akú adresu a v akom smere. Tieto pokročilé režimy sa využívajú, keď je pamäť rozdelená na stránky(predpokladá sa použitie operačného systému).

Ovládanie všetkých funkcií radiča je možné jedine pomocou čítania a zápisu do sady registrov, ktoré sú pripojené na zbernicu mikrokontroléra, viď komponent *Register Bank* na obrázku 4.1. Okrem komunikácie pomocou protokolu SD sa radič stará aj o taktovanie karty. Obsahuje sadu deličiek, ktoré slúžia k úprave taktovacej frekvencie vstupného hodinového

signálu. Bohužiaľ, taktovacia frekvencia modulu musí byť známa. Inak si nemôžeme byť istí, aký bude výsledný takt.

4.3 GPIO

Na detekciu prítomnosti karty a na zisťovanie polohy, v ktorej sa nachádza posuvník proti zápisu na kartu sa používajú dva piny, ktoré sú pripojené na GPIO rozhranie. Podľa obrázka 4.2 vidíme, že na vodiče sú pripojené pull-up rezistory. Preto môžeme predpokladať, že keď bude karta vložená do slotu, bude signál *SDCARD_DET* v „logickej 0“. Analogicky ak sa karta v slote nenachádza, bude signál v „logickej 1“. Táto konvencia detekovania karty mechanickým zariadením vo vnútri slotu je odporúčaná a overená. Detekcia protekcie proti zápisu je, rovnako ako detekcia prítomnosti karty, mechanická a takisto využíva pull-up rezistor. Ako už bolo spomínané, je zodpovednosťou ovládača radiča nedovoliť zápis na kartu, pokiaľ je označená tým že má posuvník v spodnej polohe.

Jednoduchý a pritom efektívny postup ako reagovať na udalosť vloženia a odstránenia karty je zachytiť nástupnú alebo klesajúcu hranu signálu *SDCARD_DET*. Pri udalosti sa generuje prerušenie, ktoré obslúži jadro mikrokontroléra. To si žiada vytvoriť obslužnú rutinu a nastaviť správne udalosti, pri ktorých má dochádzať k prerušeniu. Jednoduchým riešením detekcie zapnutia ochrany proti zápisu je otestovanie logickej hodnoty, ktorá sa nachádza na vstupe pinu ktorý je pripojený na signál SDCARD_WP.

4.4 Popis registrov

Jediné rozhranie, ktoré nám poskytuje radič SD je sada 24 registrov, ktoré sú mapované do pamäti mikrokontroléra. Pre napísanie knižnice na ovládanie karty SD je nutné poznať ich popis a použitie. Nasledujúce obrázky sú z manuálu k mikrokontroléru Kinetis K60, zo sekcie SDHC.

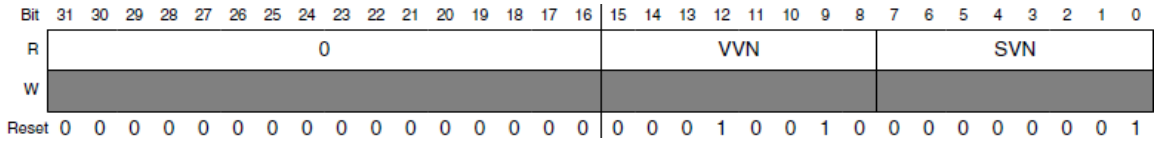
Niektoré registre slúžia na testovacie účely. Tieto registre nebudú v knižnici používané, takže ich neuvedieme v detailnom prehľade registrov. Ďalej registre, ktoré využívajú zložitejšiu verziu DMA prenosu (ADMA a ADMA2) nebudú používané. Z týchto dôvodov ich vynecháme taktiež.

- **SDHC_FEVT** - Zápisom jednotlivých bitov do tohoto registra môžeme simulovať udalosti, ktoré by bežne nastávali počas komunikácie s kartou. Tento register má testovací charakter.
- **SDHC_MMCBOOT** - Tento register kontroluje funkciu FastBoot, ktorú majú výhradne MMC karty.
- **SDHC_ADSADDR** - Slúži pri prenose ADMA ako ukazovateľ na zoznam deskriptorov.
- **SDHC_ADMAES** - Register zaznamenávajúci chybu pri prenose typu ADMA.

4.4.1 SDHC_HOSTVER

Host Controller Version - Verzia hostiteľského radiča

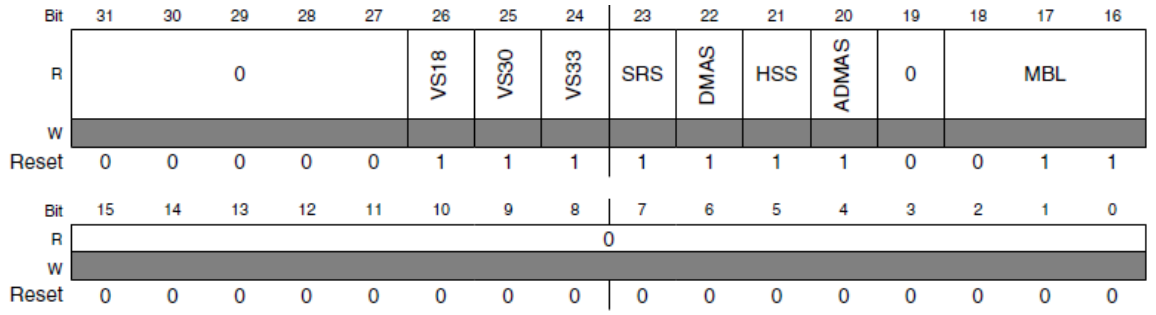
VVN	Označuje produktové meno „Freescale SDHC version 2.2“
SVN	Hodnota 01h označuje, že tento radič je kompatibilný so štandardom hostiteľského radiča verzie 2.00[5]



Obr. 4.4: Formát registra SDHC_HOSTVER, zdroj [11]

4.4.2 SDHC_HTCAPBLT

Obsahuje bitové príznaky označujúce vlastnosti, s ktorými disponuje radič

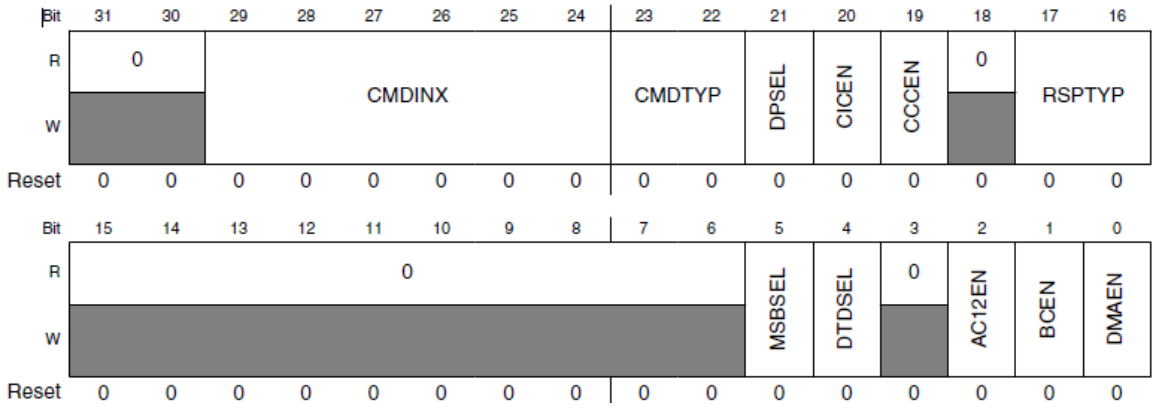


Obr. 4.5: Formát registra SDHC_HTCAPBLT, zdroj [11]

VS_{xy}	Podpora pre signalizačné napätie X.Y Volt [1.8V, 3.0V, 3.3V]
SRS	Podpora pre funkciu pozastavenia a obnovenia prenosu dát
DMAS	Radič má podporu DMA prenosu. Presnejšie tento bit označuje či má radič schopný používať interné DMA(SDMA)
HSS	Podpora rýchlostného režimu High-speed
ADMAS	Radič má podporu ADMA prenosu
MBL	Maximálna veľkosť bloku dát. Hodnota udáva, že maximálna veľkosť je nastavená na 4096 bajtov

4.4.3 SDHC_XFERTYP a SDHC_CMDARG

Register *XFERTYP* sa používa, keď chceme zaslať karte príkaz. Z hľadiska knižnice pre prácu s SD kartou ho považujeme za najdôležitejší register, ktorý sa nachádza v radiči SD. Spolu s registrom *CMDARG* tvoria dvojicu, z ktorej radič vygeneruje 48-bitový príkaz tak, aby spĺňal formát protokolu SD2.5.1.



Obr. 4.6: Formát registra SDHC_XFERTYP, zdroj [11]

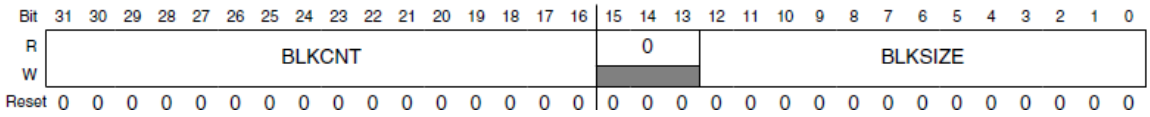
CMDIND	Index príkazu SD protokolu, viď tabuľka 2.3
CMDTYP	Typ príkazu. Štandardný príkaz, príkaz pozastavenia, príkaz opätovného spustenia a príkaz zrušenia aktuálneho prenosu(zrušenia transkakcie). Toto pole informuje radič o činnosti na zbernici.
DPSEL	Bit musí byť nastavený pokiaľ príkaz vyvoláva prenos dát po dátovej linke
CICEN	Ak je tento bit nastvený, tak radič porovná indexy príkazu a prijatej odpovede. Ak sa nezhodujú tak vygeneruje chybu.
CCCEN	Radič skontroluje CRC v odpovedi na príkaz.Ak je chybný kontrolný súčet tak vygeneruje chybu.
RSPTYP	Toto pole určuje aká bude odpoveď na príkaz. Odpoveď môže byť 136-bitová(*R2*), štandardná 48-bitová(*R1*), s busy signálom(*R1b*) alebo žiadna odpoveď nemá byť prijatá
MBSEL	Nastavuje sa len pri operáciách čítania a zápisu. Určuje, či má operácia prebehnúť len s jedným dátovým blokom alebo skupinou blokov
DTDSEL	Zvolenie smeru prenosu dát. Ak je 1 tak dáta smerujú z karty do radiča, a opačne.
AC12EN	Nastavením tohoto bitu na 1 sa pri viacnásobnom prenose dátových blokov automaticky pošle príkaz *CMD12*, ktorý ukončí dátový prenos. Príkaz sa pošle, keď počítadlo prenesených blokov dosiahne nastavenú hodnotu.
BCEN	Tento bit zapína počítadlo prenesených dátových blokov. Je relevantný iba pre viacnásobné prenosy. Ak je bit nastavený na hodnotu 0 a zároveň je inicializovaný viacnásobný prenos, potom sa vstupno-výstupná operácia neskončí a bude prebiehať do nekonečna, až kým nebude manuálne ukončená zaslaním príkazu *CMD12*
DMAEN	Zapína funkciu DMA. Ak je tento bit nastavený na 1 tak sa prenos dát realizuje hneď ako sa nastaví bit *DPSEL*

Zápisom do registra *XFERTYP* radič odošle príkaz, ktorý je doplnený 32-bitovou hodnotou *CMDARG*.

4.4.4 SDHC_BLKATTR

Tento register realizuje počítadlo dátových blokov. Okrem toho nastavuje, koľko bajtov obsahuje jeden takýto blok. Veľkosť bloku v tomto registri musí byť nastavená rovnako

ako v karte. Radič musí vedieť ktoré signály na zbernici má interpretovať ako dáta a ktoré ako CRC kontrolný súčet. Ak by počet blokov alebo veľkosť jedného bloku neboli nastavené správne tak sa prenos nepodarí a vygeneruje sa chyba. V horšom prípade môže beh programu uviaznuť na nedokončenom čítaní dát(v prípade že chceme čítať viac ako poslala karta).



Obr. 4.7: Formát registra SDHC_BLKATTR, zdroj [11]

Okrem toho je počítadlo blokov dôležité pri automatickom zasielaní príkazu *CMD12*(ak je nastavený bit *AC12EN* v *XFERTYP*). Pokiaľ sa iniciuje viacnásobné čítanie s automatickým zasielaním príkazu *CMD12*, bit označujúci zapnuté počítanie blokov by bol nastavený a zároveň v registri *SDHC_BLKATTR* by bol nastavený nulový počet blokov tak dôjde k okamžitému poslaniu ukončovacieho príkazu. Radič by nenačítal žiadne dáta. Preto je dôležité aby bol počet nastavený na aspoň jeden blok, aby mohol prenos začať.

BLKCNT	Udáva počet dátových blokov pre aktuálnu transakciu. Rozsah 0–65535 blokov.
BLKSIZE	Špecifikuje veľkosť jedného dátového bloku v bajtoch. Rozsah 0–4096 bajtov.

Čítanie bitového poľa *BLKCNT* počas dátovej transakcie nie je odporúčané, lebo nemusí reflektovať skutočnú hodnotu. Takisto pokiaľ je bit *XFERTYP[BCEN]* nastavený na režim jedného bloku tak hodnota *BLKCNT* bude vždy čítaná ako 1, nehľadiac na pôvodnú hodnotu. Počas transakcie, ktorá prenáša viac dátových blokov radič automaticky dekrementuje túto hodnotu. Čítať z bitového poľa sa smie len keď sa aktuálny dátový prenos pozastaví. Po dokončení prenosu sa premenná nastaví na poslednú zapísanú hodnotu.

4.4.5 SDHC_CMDRSPx

Radič automaticky počká na odpoveď a načíta ju do skupiny 4 registrov pomenovaných *SDHC_CMDRSP0 – SDHC_CMDRSP3*. Každý z nich je 32-bitový a určený iba na čítanie. Pokiaľ je odpoveď 48-bitová, uloží obsah odpovede(32-bitová hodnota) do registra *SDHC_CMDRSP0*. 136-bitová odpoveď(128 bitov dát, iba *R2*) sa uloží do všetkých štyroch registrov. Výnimku tvorí odpoveď na automaticky vygenerovaný príkaz *CMD12*. Aby sa zabránilo prepisu predošlej odpovede(očakáva sa, že predošlá odpoveď bola *R1*) uložia sa dáta do registra *SDHC_CMDRSP3*.

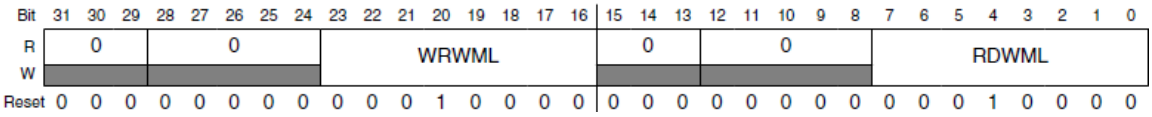
4.4.6 SDHC_DATPORT

Tento register tvorí jediné miesto k prístupu do vyrovnávacej pamäti radiča. Je to jeden 32-bitový register, ktorý mieri na začiatok fronty vyrovnávacej pamäte. Po každom prístupe do registra(čítanie aj zápis) sa nastaví na ďalšiu 32-bitovú hodnotu vo vyrovnávacej pamäti. Používa sa vtedy, keď sa presun dát realizuje pomocou inštrukcií procesora(nepoužíva sa DMA).

4.4.7 SDHC_WML

Tento register má dve funkcie. Podľa toho, či sa používa radič DMA alebo sa používa prenos, pri ktorom asistuje procesor.

Ak sa používa radič DMA tak hodnoty *WRWML* a *RDWML* určujú, koľko slov[1] sa musí načítať kým sa zaháji prenos DMA do hlavnej pamäte.



Obr. 4.8: Formát registra SDHC_WML, zdroj [11]

WRWML	Počet slov, ktoré chceme zapísať v dávke do bufferu, alebo prah, kedy sa má začať DMA operácia zápisu do bufferu
RDWML	Počet slov, ktoré chceme načítať v dávke do bufferu, alebo prah, kedy sa má začať DMA operácia čítania z bufferu

Ak sa používa procesorom asistovaný prenos, tak sa nastaví príznak, že sa smie čítať z registra *SDHC_DATPORT* práve vtedy, keď sa načíta z karty aspoň počet slov uložených v tomto registri. Ak je príznak „čítanie povolené“ nastavený, tak sa musí dávkovo, teda naraz, prečítať daný počet slov. Pri operácii zápisu sa bit „zápis povolený“ interpretuje ako stav, kedy sa vo vyrovnávacej pamäti nachádza miesto aspoň pre daný počet slov.

Pokusmi sme zistili, že jedno slovo označuje 32-bitovú hodnotu.

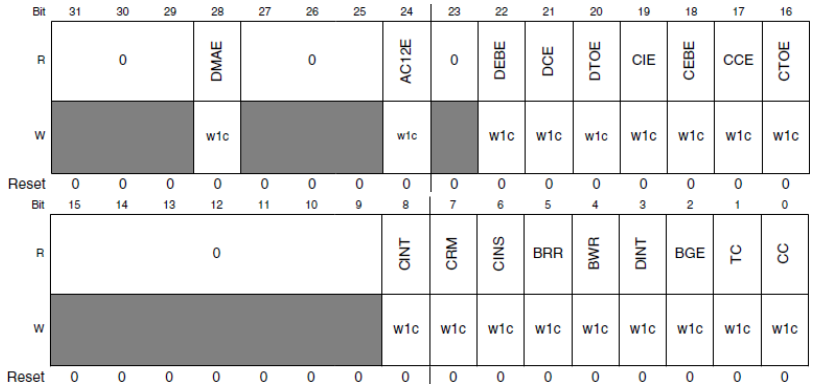
4.4.8 SDHC_IRQSTAT, SDHC_IRQSTATEN a SDHC_IRQSIGEN

Tieto registre ovládajú generovanie udalostí a povoľovanie prerušení. Register *SDHC_IRQSTAT* je stavový. Každý bit ma priradenú udalosť alebo stav. Ak je bit nastavený, tak k príslušnej udalosti došlo. Zapísaním bitu s hodnotou „1“ dochádza k vynulovaniu príznaku. Register *SDHC_IRQSTATEN* je povoľovací. Iba povolené udalosti budú v prípade výskytu premietnuté do stavového registra. Posledný register tejto skupiny je takisto povoľovací. V *SDHC_IRQSIGEN* sa nachádzajú povoľovacie bity pre generovanie prerušenia do jadra mikrokontroléra. Pokiaľ sa nastaví bit v stavovom registri a je povolené generovanie prerušenia pre danú udalosť, dochádza k prerušeniu behu jadra a vyvolaniu obslužnej rutiny.

Najdôležitejšie príznaky	
DTOE	Vypršal čas na dátovej linke
CCE	Chyba CRC príkazu
CTOE	Vypršal čas spracovania príkazu
BRR	Vyrovnávacia pamäť je pripravená na čítanie
BWR	Vyrovnávacia pamäť je pripravená na zápis
TC	Značí úspech, keď je operácia čítania alebo zápisu úspešne ukončená
CC	Nastaví sa, ak radič obdrží ukončovací bit odpovede na príkaz

Kompletný popis všetkých príznakov je popísaný v produktovom manuáli k mikrokontroléru.

[1] Manuál používa slovo „Word“, 1 Slovo = 4 bajty



Obr. 4.9: Formát registra SDHC_IRQSTAT, zdroj [11]

4.4.9 SDHC_PRSSTAT

Stav radiča SDHC je uchovávaný výhradne v tomto registri. Dokážeme pomocou neho zistiť, v akom stave sa nachádzajú vodiče zbernice, ktorú časť zbernice je možné používať(dátovú alebo riadiacu, prípadne obe), či je aktívny transfer dát, alebo či je povolený hodinový signál do karty SD.

4.4.10 SDHC_PROCTL

Riadi protokol SD a komunikáciu s kartou. V tomto registri sa nastavuje, ktorý z DMA typov prenosu sa má používať, aká je šírka dátovej zbernice alebo sa nastavuje možnosť monitorovania *DAT3* linky kvôli detekovaniu novej karty na zbernici.

4.4.11 SDHC_SYSCTL

Tento register obsahuje prevažne údaje o deličkách frekvencie hodinového signálu. Ďalej dokáže resetovať modul SDHC, poprípade jeho príkazovú alebo dátovú časť.

4.4.12 SDHC_DSADDR

32-bitová hodnota slúži ako ukazovateľ do pamäti na miesto, od ktorého sa začne prenos dát z buffera(prípadne do buffera) počas DMA prenosu typu SDMA. Adresa musí byť zarovnaná na 4 bajty, to znamená, že musí mať posledné 2 najmenej významné bity nastavené na hodnotu 0.

Kapitola 5

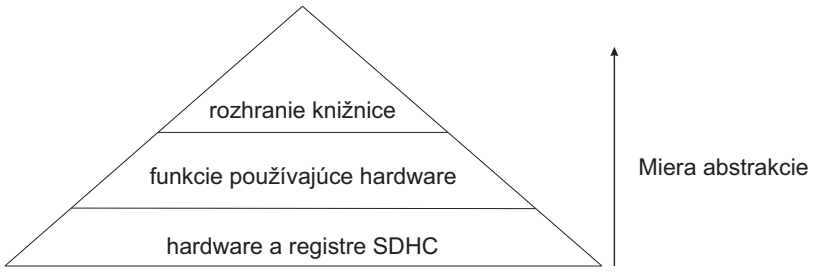
SDHC knižnica

5.1 Úvod

Cieľom tejto bakalárskej práce je navrhnúť a implementovať knižnicu funkcií na riadenie a obsluhovanie SD karty. Vzhľadom na to, že sa jedná o knižnicu pre zabudované zariadenie, budeme sa snažiť udržať minimálne pamäťové nároky. Pri tvorení sa postupovalo tak, aby vzniklo jednoduché a intuitívne rozhranie knižnice, bez zbytočných prvkov, ktoré by vytvárali neistotu v rukách používateľa. Inšpiráciou nám boli open-source riešenia, ktoré sa nachádzajú v jadre systému Linux, alebo ovládač SDHC_LLD, ktorý sa nachádza vo vývojovom prostriedku ProcesorExpert balíka CodeWarrior.

5.2 Architektúra

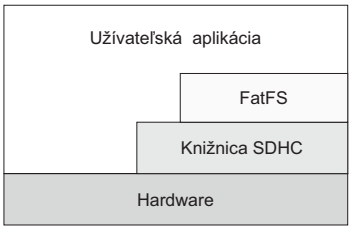
Knižnica je rozdelená do vrstiev. Vyššie vrstvy používajú nižšie vrstvy a dochádza tak k určitej miere abstrakcie, čo zaistí jednoduchšiu údržbu knižnice, pridávanie nových vlastností, prípadne jej adaptáciu na karty MMC alebo SDIO. Užívateľ by mal ku knižnici pristupovať výhradne skrz jej najvyššiu vrstvu. Predíde sa tým komplikáciám, ktoré môžu nastať. Najnižšiu vrstvu tvorí prístup k jednotlivým registrom modulu SDHC. Stredná vrstva implementuje funkcie s použitím týchto registrov. Najvyššia vrstva, ktorá slúži ako rozhranie, poskytuje vysokoúrovňový prístup k pamäťovej karte SD.



Obr. 5.1: Architektúra knižnice SD

Knižnica je napísaná v jazyku C a k svojmu behu nevyžaduje žiadne ďalšie neštandardné knižnice. Je rozdelená do dvoch súborov podľa konvencií jazyka C, `SDHC.c`(kód funkcií) a `SDHC.h`(hlavičky funkcií rozhrania, konštanty, dátové typy). Vývoj prebiehal v prostredí CodeWarrior. Po dokončení sa knižnica preportovala do prostredia KDS. Drobné rozdiely

medzi prostrediami sme vyriešili doplnením hlavičkového súboru `HAL.h`. Nachádzajú sa v ňom definície, ktoré sa medzi prostrediami líšili.



Obr. 5.2: Bloková schéma použitia knižnice

Obrázok 5.2 zobrazuje blokovú schému spolu s integráciou knižnice súborového systému FatFS a obecnou užívateľskou aplikáciou.

5.3 Vlastnosti

Knižnica dokáže pracovať s kartami SD aj SDHC. Podporuje karty verzie 1.00 – 2.00. Ostatné karty neboli pri vývoji k dispozícii ale mali by byť funkčné aj verzie vyššie ako 2.00. Knižnica má dopredu pripravenú rutinu pre obsluhu prerušenia na detekčnom porte. Pri vložení alebo odstránení karty sa zavolá užívateľom definovaná funkcia.

Pokiaľ karta podporuje režim High-Speed tak je do tohoto režimu prepnutá. Maximálna možná frekvencia pre taktovanie karty je z konštrukčného hľadiska obmedzená radičom. Špecifikácia uvádza maximum 50 MHz. Karta je taktovaná touto frekvenciou, pokiaľ podporuje režim *High-Speed*, inak sa používa *Standard-Speed* režim, ktorý podporujú všetky karty (25 MHz).

Vzhľadom na fakt, že iné zabezpečovacie mechanizmy, ako ochrana proti zápisu posuvníkom na boku karty sú použité raritne, pri vývoji knižnice sme s týmito technológiami nepočítali. Neznamená to ale, že táto vlastnosť by nemohla byť neskôr do zdrojového kódu pridaná

5.4 Dátové typy

Pre potreby knižnice boli vytvorené nové dátové typy. Ako návratový typ funkcií je najčastejšie používaný `SDHC_RETURN`. Jeho hodnoty odpovedajú stavom, ktoré môžu nastať pri používaní knižnice.

```
enum SDHC_RETURN_enum {
    SDHC_OK = 0, SDHC_ERROR,
    SDHC_CMD_TIMEOUT, SDHC_DATA_TIMEOUT,
    SDHC_NO_CARD, SDHC_PROTECTED
};
typedef enum SDHC_RETURN_enum SDHC_RETURN;
```

Hodnoty enumerácie SDHC_RETURN	
SDHC_OK	Nenastala chyba pri vykonávaní funkcie
SDHC_CMD_TIMEOUT	Pri vykonávaní funkcie došlo k vypršaniu času na riadiacej linke
SDHC_DATA_TIMEOUT	Došlo k vypršaniu času na dátovej linke
SDHC_NO_CARD	Karta sa nenachádza v slote SD
SDHC_PROTECTED	Karta je chránená proti zápisu
SDHC_ERROR	Obecná, bližšie nešpecifikovaná chyba

Ak sa funkcia skončí chybou SDHC_ERROR, tak je odporúčané aby sa SD karta znovu inicializovala, inak nie je garantovaná plná operabilita karty.

K svojmu behu potrebuje knižnica jednu globálnu premennú. Sú v nej uložené niektoré informácie o SD karte, ktorá je práve v slote SD. Deklarácia je zapísaná v jazyku C takto:

```
typedef struct {
    enum CardType Type;
    enum SdVersion Version;
    uint64_t capacity; // kapacita v B
    volatile int Ready;
    int RCA;
    SDHC_CALLBACK insert_detect;
    SDHC_CALLBACK remove_detect;
} SDHC_config;
SDHC_config SDHC;
```

Type	Typ karty [SD|SDHC|Neznáma karta|Žiadna karta]
Version	Verzia špecifikácie, ktorej karta vyhovuje [Neznáma verzia, Verzia 1.XX, Verzia 2.00+]
capacity	Kapacita karty vypočítaná z *CSD* registra uložená v 64-bitovej premennej
Ready	Príznak označujúci, že karta je zinicializovaná a pripravená na komunikáciu
RCA	Relatívna adresa karty
insert_detect	Ukazovateľ na funkciu, ktorá sa zavolá, keď je karta vložená do slotu SD
remove_detect	Ukazovateľ na funkciu, ktorá sa zavolá, keď je karta odstránená z slotu SD

Okrem toho sú v knižnici predpripravené štruktúrované dátové typy popisujúce *CSD*, *CID* aj *R1*.

5.5 Konfigurácia

Kód knižnice je napísaný na mieru mikrokontroléru Kinetis K60, avšak mal by byť schopný pracovať s celou rodinou mikrokontrolérov Kinetis, pokiaľ nedošlo k významnej zmene fungovania práce radiča SD. Ak si užívateľ nepraje aby dochádzalo k vyvolaniu prerušenia pri zasunutí, resp. odobraní karty zo slotu SD, musí v hlavičkovom súbore nastaviť konštantu preprocesora:

```
/* prerusenie pri detekcii kart
 *      0 - k preruseniu nedochadza
 *      1 - ak sa karta vlozi alebo odstrani zo slotu SD,
```

```
 *        dojde k vygenerovaniu prerusenia na porte E
 */
#define SDHC_INTERRUPT (1)
```

Podobne je to aj s využitím zabudovaného radiča DMA. Pre prenos sa používa najjednoduchší typ SDMA. Ak užívateľ nechce pri prenose využívať technológiu priameho prístupu do pamäti stačí aby nastavil príznak:

```
/* Pri prenose datovych blokov pouzivat DMA. Pre demonstracne ucely.
 *  Moznost porovnania rychlosti prenosu pri zapnutom a vypnutom DMA.
 * 1 - ANO
 * 0 - NIE
 */
#define SDHC_DMA_TRANSFER (1)
```

Pre testovacie účely je pripravená aj jednoduchá knižnica pre ovládanie LED diód, ktoré sú pripojené k mikrokontroléru. Užívateľ si môže overiť či detekcia a komunikácia s kartou funguje tým, že sleduje, ktoré diódy svietia. Ak si neželá používať túto doplnkovú vlastnosť, môže ju zakázať v hlavičkovom súbore.

```
/* pouzivat LED diody pre signalizovanie cinnosti
 * 1 - ANO
 * 0 - NIE
 */
#define SDHC_LED (1)
```

5.6 Konvencia komunikácie

Okrem počiatočnej inicializácie sa karta udržuje v stave *Stand-by State*(diagram na obrázku 2.7). Pred volaním funkcie z knižnice sa očakáva, že karta je v tomto stave. V tele funkcie sa s kartou pracuje nasledovným spôsobom.

1. Karta sa najprv zvolí príkazom *CMD7* a ako parameter sa uvedie relatívna adresa z globálnej premennej `SDcard.RCA`.
2. Následne sa karte pošle ďalší príkaz(skupina príkazov) podľa potreby.
3. Po dokončení práce sa odošle *CMD7* znovu. Tentokrát s parametrom *RCA* nastaveným na 0, čím karta znovu prechádza do stavu *Stand-by State*.

Nevýhodou tohoto prístupu je, že pre dátové prenosy, ktoré nasledujú obyčajne rýchlo za sebou musí karta prejsť selekciou a deselekciou, čo prináša zvýšenú mieru réžie. Nachádza sa tu priestor pre zlepšenie kvality a rýchlosti knižnice.

5.7 Rozhranie knižnice

Predpokladáme, že užívateľa nezaujímajú registre karty, stavový automat, manuálne posielanie príkazov karte a iné. Užívateľ chce mať jednoduchý prístup ku čítaniu a zápisu dát. Touto zásadou sme sa riadili pri tvorbe rozhrania. Inšpirovali sme sa podobnými knižnicami[1].

[1]Príklad: https://github.com/laswick/kinetis/blob/master/phase2_embedded_c/sdhc.c

`void SDHC_setup(SDHC_CALLBACK insert, SDHC_CALLBACK remove);`

Táto funkcia musí byť volaná pri inicializácií mikrokontroléru. Nastaví komunikačné porty a resetuje modul SDHC. Jej parametrami sú ukazovatele na funkcie, ktoré budú spätne volané(Callback), keď dôjde k detekcii vloženia resp. odstránenia karty zo slotu SD. Pokiaľ si užívateľ nepraje tieto funkcie používať, musí uviezť ako parameter hodnotu `NULL`. Funkcia musí byť volaná pred použitím ostatných funkcií.

`SDHC_RETURN SDHC_cardInit(int forceInit);`

Inicializuje kartu pre používanie. Návratový kód informuje o úspechu funkcie. Odporúča sa, aby sa v prípade neúspechu funkcia volala opakovane, buď po nejakú dobu alebo niekoľkokrát. Parameter `forceInit` sa používa ak užívateľ vynútiť inicializáciu už inicializovanej karty.

`SDHC_RETURN SDHC_sendCmd(uint32_t cmd, uint32_t arg, SDHC_CMDRSP* data);`

Pošle karte príkaz `cmd` s parametrom `arg`. Odpoveď na príkaz sa uloží do parametra `data`. V prípade, že sa `data` nastaví na hodnotu `NULL`, odpoveď sa neukladá.

`int SDHC_isCardPresent();`

Testovanie prítomnosti karty(aj adaptéru) v slote SD. Ak sa karta v slote nachádza tak je odpoveď nenulová. Inak je odpoveď rovná hodnote 0.

`int SDHC_isCardWriteProtected();`

Detekcia pozície prepínača ochrany proti zápisu. Ak je výsledkom nenulová hodnota, tak je prepínač v polohe *Karta uzamknutá*, inak je v polohe *Karta odomknutá*.

`void SDHC_setFrequency(unsigned mul, unsigned div);`

Táto funkcia slúži na ovládanie deličiek frekvencie, ktorou je taktovaná karta. Pre presné použitie a parametre je nutná znalosť hardware modulu SDHC a vstupná taktovacia frekvencia modulu SD.

`SDHC_RETURN SDHC_readStatusRegister(uint32_t *buffer);`

Načíta do premennej `buffer` obsah stavového registra SD karty. `buffer` musí byť veľký najmenej 64B a zarovnaný na 4 bajty. Preto sa ako premenná používa ukazovateľ na 32-bitový celočíselný typ `uint32_t`.

`SDHC_RETURN SDHC_getCID(SDHC_CID *buffer);`

Načíta do premennej `buffer` obsah identifikačného registra *CID*.

`SDHC_RETURN SDHC_getCSD(SDHC_CSD *buffer);`

Podobne ako predošlá funkcia, načíta do parametra `buffer` obsah registra karty *CSD*. Tieto dve funkcie demonštrujú konvenciu knižnice popísanú v sekcii5.6. Takisto ukazujú ako sa jednoduchým spôsobom môžu do knižnice pridávať nové funkcie.

```
SDHC_RETURN SDHC_diskRead(void *buff, int LBA, int count);
SDHC_RETURN SDHC_diskWrite(const void *buff, int LBA, int count);
```

Sú to najdôležitejšie funkcie knižnice. Slúžia pre blokové čítanie a zápis. Parameter `LBA` označuje blok v pamäti karty, s ktorým sa bude vykonávať operácia. Pre pripomenutie čitateľovi, jeden blok predstavuje 512 za sebou idúcich bajtov uložených na karte SD. Parameter `count` udáva, koľko blokov chceme načítať, alebo naopak zapísať. Pokiaľ je počet blokov väčší ako 1 tak sa pre transfer dát použije rýchlejší prenos viacerých blokov naraz v jednej transakcii. Parameter `buff` je ukazovateľ na miesto v pamäti, kde sa budú načítané dáta ukladať, alebo odkiaľ sa budú dáta načítavať a odosielať na kartu SD. Veľkosť poľa `buff` musí odpovedať hodnote (`512*count`) bajtov.

5.8 Obmedzenia

Počas používaní knižnice sa nesmú obsluhovať piny, ktoré sú pripojené na slot SD iným spôsobom, ako funkciami knižnice. Zahŕňa to aj signály GPIO zbernice *SDHC_DET* a *SDHC_WP*. Detailná schéma zapojenia je na obrázku 4.2 a 4.3.

Podobne, ak užívateľ zvolí, že prenos dát má byť vykonávaný s pomocou DMA, je nutné v užívateľskom programe vypnúť jednotku ochrany pamäti MPU[1] alebo sa v nej povolí prístup do operačnej pamäti pre modulu SDHC. Inak by sa prenos nemohol uskutočniť a skončil by vyvolaním výnimky. Pre jednoduchosť uvedieme len celkové vypnutie jednotky ochrany pamäte. Nasledovná ukážka je v jazyku C:

```
MPU_CESR &= ~MPU_CESR_VLD_MASK;
```

V niektorých častiach funkcií je implementované aktívne čakanie. Nie je to ideálne. Alternatívou by bolo použiť niektorý zo systémov časovačov. Aktívne čakanie bolo použité z dôvodu jeho jednoduchosti a je implementované funkciou `SDHC_DELAY(x)`, kde parameter `x` je počet opakovaní inštrukcie `NOP`.

Vstupno-výstupné funkcie pre prenos dát sú implementované v synchrónnej variante. Po zavolaní funkcie `SDHC_diskRead` alebo `SDHC_diskWrite` sa riadenie navráti volajúcemu vtedy, keď sa dátový prenos úspešne dokončí, alebo skončí s chybou. Pre použitie v operačných systémoch by bolo vhodnejšie dopísať funkcie s asynchrónnym chovaním spolu s využitím generovania prerušenia priamo v module SDHC. To však nepovažujeme za cieľ tejto bakalárskej práce, pretože by to vyžadovalo napísať operačný systém.

Z dôvodu kompatibility sme otestovali knižnicu s novou kartou Kingston 8GB SDHC microSD UHS-1. Karta vyhovuje štandardu SD verzie 3.01. Ultra High Speed(UHS) karty vyžadujú odlišnú procedúru prepnutia na najvyššiu dostupnú rýchlosť. Z tohoto dôvodu bola karta verzie 2.xx rýchlejšia ako karta verzie 3.xx. Novšie karty môžu byť taktované až do 208 MHz. Táto frekvencia sa v našej cieľovej platforme nedá dosiahnuť.

V konečnom dôsledku môžeme povedať, že knižnica je kompatibilná so všetkými verziami štandardu SD. Pre optimálne výsledky odporúčame použiť ľubovoľnú SDHC kartu verzie 2.00.

5.9 Použitie knižnice v reálnej aplikácii

Ako rozšírenie tejto bakalárskej práce sme sa rozhodli demonštrovať funkčnosť knižnice na skutočnom príklade. Pamäťové karty najčastejšie neobsahujú len strohé dáta ale adresárové

[1] Memory Protection Unit

štruktúry, súbory, . . . , to znamená, že musíme využiť služby knižnice pre súborový systém. Jednou knižnicou, ktorá je určená priamo pre zariadenia s obmedzenou pamäťou a zabudované systémy je knižnica s názvom FatFS[1]. Ako samotný názov napovedá, podporované súborové systémy sú FAT12, FAT16 a FAT32. Knižnica má veľmi podobné rozhranie ako štandardné funkcie z knižnice `stdio.h` v jazyku C. Na webovej stránke autora sa nachádza pomerne detailná dokumentácia a príklady použitia. Táto knižnica v staršej verzii sa nachádza aj v nástroji ProcessorExpert vo vývojových prostrediach od spoločnosti Freescale. Zdrojové kódy FatFS sú aj naďalej vyvíjané a spravované. Aktuálna verzia k dátumu písania tejto práce je R0.11 zo dňa 9. februára 2015. Zdrojové kódy k projektu s knižnicou sú k dispozícii na priloženom CD. Po spustení si môže ktokoľvek otestovať rýchlosť svojej SD karty na platforme Minerva. Zároveň tento projekt slúži ako demonštrácia knižnice a jej funkcií.

Pre testovacie účely sme importovali knižnicu aj do projektu od spoločnosti Kinetis[9]. Táto aplikácia implementuje čítačku pamäťových kariet, ktorá sa do počítača pripája pomocou USB portu. Nahradili sme pôvodné funkcie pre komunikáciu s kartou SD. Overili sme, že knižnica je funkčná a môže byť integrovaná aj do komplikovanejšieho projektu. Rýchlostné porovnanie bude budú prezentované v nasledujúcej kapitole.

5.10 Testovanie a zhodnotenie výsledkov

Zdrojové kódy boli priebežne testované a zlepšované. Testovanie prebiehalo formou preloženia kódu, potom sa binárny súbor nahral na prípravok a spustil pod ladiacim nástrojom vývojového prostredia. Počas behu programu sme vyskúšali kartu vložiť a odstrániť zo slotu SD. Kontrolovala sa detekcia a inicializácia karty. Pokusné čítanie a zápis na kartu sa stali bežnou rutinnou kontrolou činnosti knižnice. Pre potreby testovania sme využili 4 LED diódy, ktoré sú pripojené na k portu B. Každá dióda indikovala inú činnosť. Reálne karty SD, ktoré sme pri vývoji používali sú nasledovné:

- Sandisk 1GB SD
- Apacer 1GB SD
- Emtec 8GB SDHC, class 4
- Kingston 2GB microSD
- Kingston 8GB, microSD, class 4

Všetky karty vyhovovali štandardu SDv2.00, bohužiaľ karta staršej verzie 1.XX sa na začiatku vývoja pokazila.

Intenzívne testovanie priniesla podpora prenosu dát s využítím radiča DMA, ktorý je zabudovaný priamo v module SDHC. Zrýchlenie prenosu dát bolo badateľné s každou vývojovou iteráciou knižnice.

Pre meranie doby, po ktorú je dátový prenos aktívny sme použili svetelnú signalizáciu pomocou LED diódy, čas sme merali stopkami. Tento spôsob sa nedá považovať za exaktný ale je dostatočný v prvotných fázach. Neskôr sme pre meranie rýchlosti využívali interný register ARM procesora `SYSTICK`, ktorý sa dekrementuje rovnakou frekvenciou, na akej pracuje jadro procesora. Po dosiahnutí hodnoty 0 sa vyvolá prerušenie. Pokiaľ poznáme dobu, po ktorej sa bude prerušenie opakovať, dokážeme merať čas v relatívne veľkej presnosti[1].

[1] Presnosť je ovplyvnená stabilitou taktovacej frekvencie

Register `SYSTICK` sa potom nastaví na predvolenú hodnotu a dekrementuje sa opäť. V našom nastavení je jadro taktované na pracovnú frekvenciu 100 MHz a predvolená hodnota `SYSTICK` je nastavená na 100000. K vyvolaniu prerušenia dochádza každú 1 milisekundu. V tele obsluhy prerušenia sa inkrementuje globálna premenná používaná na meranie času v milisekundách. Rýchlosť sa testovala na vzorke dát o veľkosti 10 MB. Rýchlosť súborového systému FatFS bola testovaná na súbore rovnakej veľkosti. Overili sme predpoklad, že na rýchlosť vo veľkej miere vplýva režim prenosu. Prenos viacerých blokov naraz je omnoho rýchlejší ako prenos jedného bloku. Výsledky sú uvedené v tabuľke 5.1.

Kingston 8GB SDHC, class 4					
Parametre		Bez DMA		DMA	
Počet blokov	Frekvencia jadra [MHz]	Čítanie [KB/s]	Zápis [KB/s]	Čítanie [KB/s]	Zápis [KB/s]
16	100	8153.78	3931.67	11287.15	4997.98
8	100	6290.20	2934.72	8962.19	3447.00
4	100	4318.68	638.09	6061.13	641.12
2	100	2655.29	261.48	3680.51	273.25
1	100	1546.34	105.50	2089.63	106.77
16	20	4014.46	2542.62	4094.40	2680.41
8	20	3437.95	2113.21	3556.91	2119.62
4	20	2670.17	562.30	2815.73	568.77
2	20	1846.41	251.78	1987.45	254.25
1	20	1148.12	103.09	1266.86	104.13

Tabuľka 5.1: Prehľad dosiahnutých rýchlostí v závislosti na parametroch. Veľkosť súboru je 10 MB.

Z nameraných výsledkov môžeme usúdiť, že používanie technológie DMA, ktorá je zabudovaná v radiči SD, neznižuje rýchlosť komunikácie a teda nie je dôvod ju nepoužívať. Pri vyšších taktovacích frekvenciách je nárast v rýchlosti pri používaní DMA cca 35%. Pri 20 MHz je rozdiel minimálny.

Okrem našej vlastnej aplikácie pre porovnávanie rýchlosti komunikácie s SD kartou sme vyskúšali knižnicu integrovať do existujúceho riešenia. Úspešne sme nahradili podsystém SD v projekte od spoločnosti Freescale[9]. Ako už bolo spomenuté, táto aplikácia realizuje čítačku SD kariet, ktorá sa do počítača pripája cez zbernicu USB. S pôvodným rozhraním sa rýchlosť pohybovala a v priemere na úrovni 420 KB/s. S našou knižnicou sa rýchlosť zvýšila na 520 KB/s. Zlepšenie pripisujeme hlavne použitiu radiča DMA. Miera zrýchlenia oproti pôvodnému stavu sa pohybovala na úrovni cca 30%, čo korešponduje so zlepšením v prípade zapnutia DMA. Bohužiaľ, nepodarilo sa nám zistiť taktovaciu frekvenciu jadra a periférií. Všimli sme si, že počas čítania alebo zápisu sa manipulovalo s dátami o veľkosti jedného bloku (512 B). Z tohoto dôvodu sa režim viacnásobného prenosu blokov nevyužil, čo malo za následok nižšiu rýchlosť vstupno-výstupných operácií. V tejto aplikácii sme si overili, že knižnica môže byť integrovaná aj do náročnejšieho projektu.

5.11 Metriky

Počet riadkov v zdrojových súboroch a veľkosť kódu po skompilovaní s prepínačom `-O3`:

Názov súboru	Počet riadkov	Veľkosť kódu v bajtoch
`SDHC.c`	1235	14 996
`SDHC.h`	310	
`LED_hal.c`	50	256
`LED_hal.h`	29	
`HAL.h`	64	—
Spolu	1688	15 252

Kapitola 6

Záver

Cieľom tejto bakalárskej práce bolo vytvoriť jednoduchú knižnicu pre komunikáciu s SD kartami. Vzhľadom na fakt, že sme nemali predošlé skúsenosti s perifériou tohoto typu ani s vývojom aplikácie pre mikrokontroléry, začali sme štúdiom technológie SD kariet a pokračovali zoznámením sa s cieľovou platformou. Mikrokontrolér K60, ktorý je osadený na vývojovej doske, v sebe integruje hostiteľský radič zbernince SD. Kartu tak netreba prepínať do pomalého a obmedzeného režimu SPI.

Všetky dostupné produktové manuály, špecifikácie a poznatky sme čerpali z internetových zdrojov, kvôli absencii tlačenej literatúry. Špecifikácia SD a referenčný manuál od spoločnosti Freescale sú voľne dostupné na webových stránkach. Pre lepšie pochopenie SD potokolu sme využili aj produktové manuály výrobcov pamäťových kariet.

Z hľadiska riadenia nie je SD karta jednoduché zariadenie. Disponuje množstvom príkazov, registrov a zabezpečovacími technológiami. Komunikačný protokol sa používa aj pre karty typu SDIO, ktoré umožňujú zapojiť do slotu SD zariadenie pre príjem GPS alebo Bluetooth signálu(príklady). SD karta je zaujímavou variantou stavového automatu, ktorý ako vstupnú abecedu využíva prijaté príkazy z hostiteľského radiča. Doteraz bolo vydaných niekoľko verzií štandardu SD. Bohužiaľ, nie všetky zachovávajú spätnú kompatibilitu.

Počas vývoja knižnice sme postupovali systematicky smerom zdola nahor. Postupne sme vytvorili funkcie pre inicializáciu karty, prepínanie karty do vysokorýchlostného režimu (High-Speed) alebo nastavenia taktovacej frekvencie karty. Snažili sme sa využiť prostriedky, ktoré nám ponúka procesor s architektúrou ARM Cortex-M4. Cieľová platforma má oproti osobným počítačom veľmi limitovanú pamäťovú kapacitu a výpočtový výkon. Z týchto dôvodov sme sa snažili, aby knižnica bola efektívna a zároveň pamäťovo nenáročná.

Myslíme si, že ciele tejto práce sme úspešne splnili. Požiadavka jednoduchej použiteľnosti spočíva v minimalistickom rozhraní, ktoré knižnica užívateľovi ponúka. Nevyužíva všetky technológie, ktoré ponúka SD karta, ale je efektívna a pamäťovo nenáročná. Veľkosť knižnice v preloženej forme je asi 15 KB. Kapacita FLASH pamäte mikrokontoléru je 512 KB. Knižnica zaberá asi 3% systémových prostriedkov cieľovej platformy. Vytvorili sme funkčnú knižnicu, ktorá je pamäťovo nenáročná a efektívna.

Komplikácie, ktoré sa pri vývoji objavili, sme úspešne odstránili. Pre testovanie sme vytvorili projekty v prostrediach CodeWarrior 10.6, KDS 2.0.0 a KDS 3.0.0. Spolu s implementáciou knižnice sme vytvorili demonštračnú aplikáciu pre meranie rýchlosti vstupno-výstupných operácií s kartou. Maximálna rýchlosť komunikácie, ktorú sme namerali, bola 12,85 MB/s (Kingston 2GB microSD, operácia čítania). Overili sme predpoklad, že prenos dát viacerými blokmi naraz, tzv. *MultiBlock Transfer*, je omnoho rýchlejší ako prenos rovnakého množstva dát s prenosom po jednom bloku. Dosiahnuté výsledky prezentujeme

v tabuľke 5.1. Prekvapila nás neočakávane nízka rýchlosť dátovej komunikácie pri zápise jediného bloku dát na kartu.

Na priloženom CD dodávame zdrojové súbory knižnice, aplikáciu pre meranie rýchlosti a užitočné materiály, z ktorých sme počas vývoja čerpali. Nad rámec zadania dodávame knižnicu spolu so zdrojovými kódmi knižnice FatFS[1] pre prácu so súborovým systémom FAT.

V práci podávame návrhy na zlepšenie niektorých častí knižnice. Týka sa to prevažne monitorovania stavu, v akom sa karta nachádza. V praxi by táto zmena mohla pozitívne ovplyvniť komunikačnú rýchlosť s kartou. Jednoduché implementácie knižnice, ktoré sú dostupné na internete, neťažia z prítomnosti DMA radiča v module SDHC. V našej knižnici sme využitím tejto technológie dosiahli zlepšenie až 30% v ideálnych podmienkach.

Veríme, že v budúcnosti naša knižnica nájde uplatnenie v aplikáciách, ktoré budú využívať SD kartu.

Literatúra

[1] FatFs - Generic FAT File System Module [online]. ChaN. [cit. 2015-04-12], Dostupné z: http://elm-chan.org/fsw/ff/00index_e.html.

[2] SanDisk Corporation. SanDisk SD Card Product Manual Version 2.2 [online]. 2004-11-01 [cit. 2015-04-12], Dostupné z: http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Components/General/SDSpec.pdf.

[3] SD Association. SD Specifications Part 1 Physical Layer Simplified Specification [online]. 2006-09-25 [cit. 2015-04-12], Dostupné z: https://www.sdcard.org/downloads/pls/simplified_specs/archive/part1_200.pdf.

[4] Toshiba Corporation. TOSHIBA SD Card Specification [online]. 2006-12-29 [cit. 2015-04-12], Dostupné z: http://www.mikrocontroller.net/attachment/21920/SDHC_SDM04G7B7_08G7B7.pdf.

[5] SD Association: SD Host Controller Simplified Specification Version 2.00 [online]. 2007-02-08 [cit. 2015-04-19], Dostupné z: https://www.sdcard.org/developers/overview/host_controller/simple_spec/Simplified_SD_Host_Controller_Spec.pdf.

[6] On MicroSD Problems. Bunniestudios: bunnie's blog [online]. 2010-02-16 [cit. 2015-04-14], Dostupné z: http://www.bunniestudios.com/blog/?p=918.

[7] NXP Semiconductors: AN10911 [online]. 2013-04-04 [cit. 2015-04-14], Dostupné z: http://www.nxp.com/documents/applicationq_note/AN10911.pdf.

[8] On Hacking MicroSD Cards. Bunniestudios: bunnie's blog [online]. 2013-12-29 [cit. 2015-04-13], Dostupné z: http://www.bunniestudios.com/blog/?p=3554.

[9] Freescale Semiconductor, Inc. MEDICALUSB: USB Stack. [cit. 2015-04-30], Dostupné z: http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MEDICALUSB.

[10] Corporation, S.: SanDisk Premieres World's Highest Capacity SD Card for High Performance Video and Photo Capture. 2014-09-11 [cit. 2015-04-30], Dostupné z: http://www.sandisk.com/about-sandisk/press-room/press-releases/2014/sandisk-premieres-worlds-highest-capacity-sd-card-for-high-performance-video-and-photo-capture/.

[11] Freescale Semiconductor, I.: K60 Sub-Family Reference Manual. 2012-06-02[cit. 2015-05-12], Dostupné z: http://cache.freescale.com/files/32bit/doc/ref_manual/K60P144M100SF2V2RM.pdf.

Dodatok A

Obsah CD

Adresár	Popis
`\doc`	Zdrojové texty bakalárskej práce
`\manual`	Referenčné príručky a špecifikácie, z ktorých boli čerpané informácie
`\source`	Zdrojové text knižnice pre prácu s kartou SD
`\app\KDSv2`	Demoaplikácia pre prostredie KDSv2.0.0
`\app\KDSv3`	Demoaplikácia pre prostredie KDSv3.0.0

Dodatok B

Manuál

Manuál pre knižnicu pre prácu s kartou SD.

Pre používanie tejto knižnice s vývojovou doskou Minerva v prostredí Kinetis Design Studio je nutné:

1. Importovať zdrojové súbory knižnice do cieľového projektu.
2. V prípade, že užívateľ chce, aby knižnica reagovala na udalosť vloženia alebo odstránenia karty do slotu musí nastaviť vo vektore prerušení obslužnú rutinu pre portE a nastaviť makro `SDHC_INTERRUPT` v hlavičkovom súbore. Vektor prerušení sa nachádza v inicializačných súboroch prostredia, v prípade použitia kompomentu ProcessorExpert je vektor v súbore `Vector.c`. Pozícia prerušenia portu E je `107`.
3. V obslužnej rutine sa musí volať funkcia knižnice `SDHC_CardDetectRoutine()`. Číslo pinu, na ktorý je zapojený signál detekcie prítomnosti karty v slote je `6`. Inicializácia pinu a nastavenie prerušenia sa nachádza vo funkcii `SDHC_setup()`. Príklad obslužnej rutiny pre port E zobrazuje tento kód:

```
extern void PORTE_IRQHandler()
{
    if (PORTE_PCR(6) & PORT_PCR_ISF_MASK)
    {
        /* prekmity */
        SDHC_DELAY(0x3FFFF);
        SDHC_CardDetectRoutine();
    }
}
```

4. Pred použitím funkcií knižnice sa musí modul inicializovať a pripraviť na použitie. Dosiahneme toho funkciou `SDHC_setup(...)`. `CustomInsertTask` a `CustomRemoveTask` sú callback funkcie (ukazovatele na funkcie).

```
void SDHC_setup(SDHC_CALLBACK insert, SDHC_CALLBACK remove);
```

   Callback funkcia `insert` je volaná pri detekcii vloženia karty do slotu. Najprv sa knižnica pokúsi kartu inicializovať. `remove` funkcia je zavolaná, keď dôjde k odstráneniu karty zo slotu SD.

   Demonštračný kód pre `insert` callback:

```
void CustomInsertTask()
{
    if (SDcard.ready == 1)
        printf("Karta bola vlozena a inicializovana\n");
    else
        printf("Karta bola vlozena ale pri inicializacii doslo k chybe\n");
}
```

Ak užívateľ nechce aby knižnica volala callback, tak musí namiesto ukazovateľa na funkciu uviezť `NULL`. V tomto príklade sa zavolá callback funkcia pri vložení karty do slotu, ale nezavolá pri odstránení karty zo slotu:

```
SDHC_setup(insert, NULL);
```

5. Testovanie prítomnosti karty v slote sa vykonáva pomocou funkcie `SDHC_isCardPresent()`. Funkcia vráti kladnú hodnotu, pokiaľ sa karta(alebo adaptér) nachádza v slote.

6. Modul využíva globálnu premennú `SDcard`. Táto premenná by mala byť iba na čítanie. Užívateľ do tejto premennej nesmie zapisovať, inak knižnica nemusí pracovať korektne. Z tejto dátovej štruktúry sa podľa potreby užívateľ môže dozvedieť typ, verziu, kapacitu a príznak inicializácie karty.

7. To, či je karta inicializovaná, sa dá testovať pomocou príznaku `SDcard.Ready`. Pokiaľ karta nie je inicializovaná, tak knižnica nedovolí volať niektoré funkcie(ich návratová hodnota bude indikovať chybu).

8. Ak je karta v slote už pred tým, ako sa spustí aplikácia na mikrokontroléri tak nedôjde k vyvolaniu prerušenia a teda karta nie je automaticky inicializovaná. Užívateľ musí v svojom kóde kartu manuálne inicializovať pomocou `SDHC_CardDetectRoutine()`.

9. V prípade, že užívateľ chce inicializovať kartu mimo obsluhy prerušenia(predošlý krok), musí volať funkciu

```
SDHC_RETURN SDHC_cardInit(int forceInit);
```

Pokiaľ návratový kód funkcie je chybový, môže užívateľ funkciu volať opakovane niekoľkokrát, alebo ju volať opakovane po určitú dobu. Špecifikácia udáva časové rozpätie až 1 sekundu.

10. Pre optimálne výsledky odporúčame čo najvyšší takt procesora. Modul SDHC pre svoju funkciu využíva práve taktovaciu frekvenciu procesora. Najvyššia frekvencia, ktorou môže modul taktovať kartu je $\frac{\text{Frekvencia jadra}}{2} Hz$. Maximálna taktovacia frekvencia jadra je 100 MHz.

11. Knižnica automaticky detekuje typ karty SD a SDHC. Ak je karta typu SD tak adresa bloku vo funkciách `SDHC_diskRead()` a `SDHC_diskWrite()` je interne vynásobená veľkosťou jedného bloku, pretože SD karta využíva iné adresovanie. Vzhľadom na to, že knižnica automaticky upravuje adresy, užívateľ nemusí pre kartu SD a SDHC vytvárať špecializované kódy.

12. Dokumentácia funkcií knižnice sa nachádza v zdrojových súboroch.

Vzorový kód:

```
#include <...>
#include SDHC.h

extern void PORTE_IRQHandler() // portE obsluzna rutina
{
    if (PORTE_PCR(6) & PORT_PCR_ISF_MASK)
    // port 6 je detekcia pritomnosti karty
    {
        /* prekmity */
        SDHC_DELAY(0x3FFFF);
        SDHC_CardDetectRoutine();
    }
}

void CustomInsertTask()
{
    if (SDcard.ready == 1)
        printf("Karta bola vlozena a inicializovana\n");
    else
        printf("Karta bola vlozena ale pri inicializacii doslo k chybe\n");
}

void CustomRemoveTask()
{
    printf("Karta bola odstranena zo slotu\n");
}

int main(void)
{
        /* Inicializacia mikrokontroleru */
    SDHC_setup(CustomInsertTask, CustomRemoveTask);

    /* Detekcia pritomnosti karty v slote -- manualna inicializacia ak nedojde
     * k automatickej inicializacii
     */
    if (SDHC_isCardPresent()) {
        SDHC_CardDetectRoutine();
    }

    for (;;){
        if ( SDcard.Ready ){
                /* Miesto pre uzivatelsky kod ... */
                /* Priklad */

                // musi byt zarovnane na 4-Bajty, kvoli DMA radicu
                char buffer[512] __attribute__ ((aligned (4))) ;
                // nacitame 1 blok na logickej adrese 0
                SDHC_RETURN result = SDHC_diskRead(buffer, 0, 1);
                if (result != SDHC_OK){
                    /* OSETRENIE CHYBY */
                };
                /* ... */
        }

    }
}
```