VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

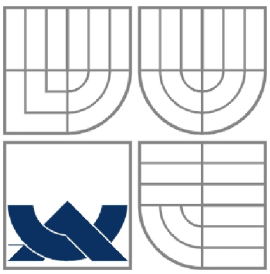
PŘEHRÁVAČ VIDEA VYUŽÍVAJÍCÍ FPGA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

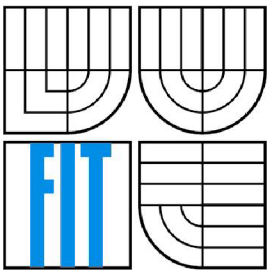
AUTOR PRÁCE
AUTHOR

Bc. Stanislav Sigmund

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

PŘEHRÁVAČ VIDEA VYUŽÍVAJÍCÍ FPGA

VIDEO PLAYER BASED ON FPGA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. Stanislav Sigmund

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Vašíček Zdeněk

BRNO 2009

Abstrakt

Tato práce se zabývá možnostmi a realizací dekomprese a přehrávání video-sekvencí na platformách, obsahující FPGA. Pro implementaci přehrávače je použita platforma FITKit, která má k dispozici VGA-konektor a dostatečně velkou paměť RAM. Jako paměťové médium je použit pevný disk se souborovým systémem FAT32.

Klíčová slova

FITKit, FPGA, video komprese, obrazová komprese, RLE, pohybový vektor

Abstract

This thesis deals with possible and realized decompression and playing of video on platforms, using FPGA unit. For implementation of this player is used platform FITKit, which has integrated VGA connector and large enough RAM memory. It uses a hard drive as memory medium with FAT32 file system.

Keywords

FITKit, FPGA, video compression, image compression, RLE, motion vector

Citace

Stanislav Sigmund: Přehrávač videa využívající FPGA, diplomová práce, Brno, FIT VUT v Brně, 2009

Přehrávač videa využívající FPGA

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením ing. Zdeňka Vašíčka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Stanislav Sigmund
1. 7. 2009

Poděkování

Tímto děkuji za pomoci s platformou FITKit a radami s diplomovou prací vedoucímu projektu, ing. Zdeňku Vašíčkovi.

Obsah

Úvod

Cílem diplomové práce je zjistit a realizovat možnosti komprese a přehrávání video-sekvencí na FPGA. Součástí je zvolení vhodné metody komprese s přihlédnutím na dostupné hardwarové prostředky. Jako úložiště dat se předpokládá pevný disk, či paměťová karta, řadič těchto médií je použit (a dále rozšířen) z mé bakalářské práce „Rozhraní IDE pro platformu FITkit“[6] s několika úpravami. Použita však mohou být i jiná média (nabízí se například jednotka CD/DVD-ROM).

Platforma FITKit, která je vyvíjená na Ústavě počítačových systémů (UPSY), je určena především pro výuku formou praxe. K tomu je vybavena mikroprocesorem, obvodem FPGA a několika konektory, z nichž nejpodstatnější pro tuto práci je konektor VGA a již implementovaný řadič ve VHDL. Obrazový výstup má k dispozici 3 bity na barvu, tedy celkem barevnou hloubku 9 bitů. Tomuto faktu je přizpůsobena i diplomová práce.

Práce je členěna následovně. První kapitola obsahuje stručný popis použitých komponent pro tuto DP (platforma FITKit s FPGA). Druhá kapitola popisuje metody obrazové komprese, třetí pak metody video komprese. Čtvrtá kapitola je věnována zvolené metodě komprese. Pátá kapitola se zabývá popisem rozhraní IDE, šestá pak souborovým systémem FAT32. Implementaci přehrávání na FPGA popisuje kapitola sedm, kapitola osm je zaměřena na popis zvoleného formátu video-souboru a vytvořenou aplikaci. Poslední devátá kapitola osvětluje dosažené výsledky a ovládání vytvořeného přehrávače.

1 Použité komponenty

1.1 Výuková platforma FITKit

Výuková platforma FITKit je určena pro získání praktických zkušeností s vestavěnými systémy, které jsou stále více používány v našem okolí (mobilní telefony, MP3 přehrávače). K tomuto účelu je vybavena obvodem FPGA (polem programovatelných hradel), na kterém je možno vytvářet různá jednodušší i složitější zapojení číslicových obvodů na jediném čipu. Toto zapojení lze programově velmi jednoduše změnit, umožňuje tak rychlé úpravy výsledného obvodu.

Tato práce vychází z platformy FITKit verze 1.2, v době tvorby však již existuje verze 2.0, pro kterou by se aplikace musela upravit, proto se s ní zatím nepočítá. Změny se týkají například zapojení samotného konektoru a přítomnosti audio kodeku, který s ním sdílí část vývodů.

Obvod FPGA je řízen 16-bitovým procesorem (též osazeným na platformě) a komunikující s ním přes sériové rozhraní SPI. Jelikož obvod FPGA obsahuje pouze volatilní paměť, je platforma osazena pamětí FLASH o kapacitě 256kB, v které je uloženo nastavení obvodu FPGA, umožňující tak případnou mobilitu platformy (nezávislost na počítači).

Procesor komunikuje s počítačem přes rozhraní RS232, simulovaného rozhraním USB, je tedy použitelný i na systémech bez rozhraní RS232 a poskytuje platformě napájení dostatečné pro většinu aplikací.

Platforma má k dispozici dva A/D a dva D/A převodníky na procesoru použitelné pro práci s analogovým signálem (např. Audio). Na obvodu FPGA pak 50-pinovou patici pro připojení dalších zařízení (je použit i pro připojení IDE konektoru), konektor pro výstup VGA signálu na monitor, výstup na rozhraní RS232 a dva porty PS/2. Platforma je osazena LCD displejem, maticovou klávesnicí a dynamickou pamětí 8MB. Vše je připojeno taktéž k obvodu FPGA.

Uvedené součásti platformy poskytují zázemí i pro náročnější aplikace. Více informací můžete nalézt na stránkách platformy [5].

1.2 Obvod FPGA

Použitý obvod FPGA [7] má k dispozici celkem 192 konfigurovatelných logických bloků, 4 násobičky 18x18 bitů a 4 paměti typu BRAM o velikosti 18 kbitů každá. S platformou komunikuje pomocí rozhraní SPI, ke kterému je implementován standardní řadič. K obvodu je připojena 8MB dynamická paměť, maticová klávesnice, LCD display, 2 PS/2 porty, RS232 port, VGA port a 50-ti pinový konektor, který má však část pinů sdílenou s výše uvedenými konektory. Při použití VGA portu pak tedy zůstává 27 pinů volných, což je ale dostatečný počet pro použití zařízení s IDE rozhraním po úpravě kabelu.

2 Metody obrazové komprese

Metody obrazové komprese je třeba vysvětlit před samotnými metodami video komprese, neboť ty jsou založeny z části právě na obrazových kompresích. Velké množství dat totiž nemohou být zkomprimované video-kompresními metodami a měly by velmi silný záporný vliv na výsledný kompresní poměr.

2.1 Metoda RLE

Bezztrátová kompresní metoda RLE (Run Length Encoding) využívá podobnosti sousedních bodů v obrazech. U mnoha obrázků (jako jsou například různé grafy a kreslené obrázky) jsou totiž velké oblasti tvořeny body stejné barvy. Komprimátor prochází body obrázku v zadaném pořadí (nejčastěji zleva doprava a shora dolů) a hledá skupiny za sebou jdoucích bodů stejné barvy. Tyto skupiny pak nahradí definovaným symbolem, barvou této skupiny bodů a jejich počtem.

Tento formát je použit u několika známých obrazových formátů, jako je PCX nebo BMP (ale u něj se téměř nevyužívá). Je velice jednoduchý a rychlý, ale jeho účinnost je velice závislá na podobnosti bodů. Nejhorší výsledky má u fotografií, kde se sousední body téměř vždy liší alespoň o malou hodnotu a často dokonce mohou být výsledná data větší. Její obdoba je použita i ve faxových přístrojích.

Účinnost této metody může být výrazně zvýšena zavedením ztrátovosti do komprese, ale za cenu snížení kvality obrázku. Toto je použitelné především u obrázků s vyšším rozlišením, kde se snížení detailů tak neprojeví.

2.2 Slovníkové metody

Slovníkové metody hledají výskyt skupin stejných bodů v datech a nahrazují je kratšími symboly. Tyto metody jsou použitelné v mnoha oblastech, u obrázků však dosahují různého stupně komprese. Přestože jsou účinnější, než metoda RLE, nedosahují na fotografiích moc dobrých výsledků, neboť jsou také bezztrátové.

Tato metoda je použita u grafického formátu GIF, při nižším počtu barev umožňuje dostatečnou kompresi, je proto používán u menších obrázků s nižším počtem barev.

2.3 Diskrétní kosinová transformace

Tato metoda převádí obraz do frekvenčního spektra a využívá toho, že ve většině obrázků se vyskytují jen ty nejnižší frekvence. Do výstupních dat pak může zapsat jen část frekvencí, které mohou být ještě vhodně upravené a často zakódované dalšími metodami (většinou se skládají z malých čísel), jako jsou například statistické metody. Kompresi lze ještě zvýšit převedením do jiného barevného prostoru a různým stupněm ztrátovosti komprese jednotlivých barev (nejvíce informace je obsaženo v jasu).

Tato metoda získala velikou oblibu pro vysoký stupeň komprese a možnosti měnit kvalitu dat, je však výpočetně náročnější kvůli většímu počtu násobení (64 operací násobení a součtu na bod). Je použita v mnoha formátech, z nichž nejznámější je JPEG.

3 Metody video komprese

Při kompresi videa se využívá skutečnosti, že většina za sebou jdoucích snímků je si velice podobných. Snaží se tedy najít podobné věci a teprve na zbytek dat použít jiný typ komprese. Na této skutečnosti jsou založeny téměř všechny video formáty.

Obraz je rozdělen na stejně veliké bloky, které mívají rozměry nejčastěji mocniny 2, kvůli snazší implementaci. Komprimátor pak hledá v okolí tohoto bloku v jiném snímku blok, který je mu dostatečně podobný. Toto je velice výpočetně náročná operace, ale existuje mnoho metod, jak ji zrychlit (často však na úkor nižší komprese, neboť se v nich často neprohledávají všechny možné pozice). Pokud je podobný blok nalezen, je místo něj zapsán do výstupních dat jen pohybový vektor, udávající relativní polohu zdrojového bloku. Není-li však nalezen, je nahrazen daty novými, často zkomprimovanými metodou DCT.

Komprese se dá ještě zvýšit, neboť většina pohybových vektorů míří stejným směrem a výsledná data se ještě komprimují další metodou (statistickou).

Ovšem vlivem použití ztrátové komprese a případnou přítomností chyb v datech se časem obraz degraduje. Proto se po určitém počtu snímků vkládá snímek bez pohybových vektorů, který je například ve formátu JPEG. Tyto snímky se označují I (indexační) a pokud je potřeba začít přehrávat video od jiného místa, začíná přehrávání právě od těchto snímků.

Snímky, které obsahují pohybové vektory jsou značeny jako P (predikční) a mohou být založeny na snímcích typu P a I. Ovšem jsou situace, kdy je vhodné hledat bloky i v následujících snímcích (například se na následujícím snímku odkryje věc, která je na aktuálním snímku částečně zakrytá). Tyto snímky jsou označeny jako B (obousměrné), jsou založeny na snímcích I a P a umožňují ještě výraznější kompresi. Ale pořadí příchozích snímků musí být jiné, než v jakém jsou zobrazovány, a tak musí přehrávač dopředu vypočítávat několik snímků. Většina přehrávačů však jich může zpracovávat více zároveň, neboť se tak zvýší jejich výkon.

4 Zvolená kompresní metoda

4.1 Video komprese

Kódování pohybových vektorů je osvědčená, často používaná metoda, dávající velice dobré výsledky u téměř všech videí, a je poměrně snadno implementovatelná v obvodu FPGA – jedná se o prosté překopírování bloku dat.

Video bude rozděleno do bloků o rozměrech 8x8 bodů, což je vhodná velikost při práci s použitou 8-bitovou šířkou dat a případných navrhovaných rozšíření přehrávače. Budou použity snímky typu I (indexovací snímek) a P (predikční snímek), použití snímků typu B (obousměrná predikce) se zvažuje (v závislosti na dostupných zdrojích a rychlosti).

Indexovací snímky budou též rozděleny do bloků a bude použita stejná metoda komprese bloků, jako u snímků typu B. O jiném typu komprese se zatím neuvažuje, protože komprese celého snímku upravenou zvolenou kompresní metodou vykazuje jen mírné snížení velikosti za cenu zvýšení komplexnosti dekompresní jednotky. Ve zvoleném formátu souboru však bude počítáno s možností zvolit si i jiný typ komprese.

4.1.1 Definice snímků

Každý snímek bude uveden svoji hlavičkou, udávající typ, typ komprese a celkovou velikost dat (snímek může být následován zvukovými daty, které se mohou ukládat do jiné vyrovnávací paměti). Poté budou za sebou následovat data pro jednotlivé bloky ve snímacím pořadí (zleva doprava, odshora dolů).

Formát dat je přizpůsoben práci s 8-bitovou šířkou dat, bitově orientovaná práce by totiž nemusela splňovat požadavky na rychlost na dostupné zdroje, její však bude zvažováno až na základě konkrétních výsledků. První byte každého bloku udává, zda se jedná o pohybový vektor či o plně definovaný blok – udává jej nejvyšší bit. U snímků typu I je ignorován a bude k dispozici pro případné udání typu komprese. Nižších 7 bitů je pak již přímo součástí pohybového vektoru nebo zvolené komprese.

4.1.2 Pohybové vektory

Pohybový vektor je v obou osách udán dvěma 7-bitovými čísly v doplňkovém kódu, což by mělo umožnit dostatečnou volnost kompresoru v hledání podobných bloků. Osmý bit druhého bytu může být použit u případných snímků typu B k výběru zdrojového snímku. Výsledná velikost je tedy 2 byty na jeden blok. O dalším kódování těchto vektorů se neuvažuje.

4.1.3 Plně definované bloky

Jako kompresní metoda obrazových dat byla zvolena metoda, založená na metodě RLE. Tato metoda je rychlá a snadno implementovatelná. Zatímco u normálních videí by vykazovala velice slabé výsledky vzhledem k rozdílnosti sousedních bodů (které se liší i o malé hodnoty, zvláště pak při kvalitnější kompresi), je při maximální dosažitelné barevné hloubce na platformě FITKit (3 bity na barvu) podobnost sousedů velice pravděpodobná a dosažitelný stupeň komprese již podstatně výraznější. Jeho účinnost je pak ještě možno navýšit vhodnou filtrací.

Formát dat kódovaného bloku je přizpůsoben maximální velikosti 64 bodů (bloky o velikosti 8x8 bodů). Použitý typ komprese vychází z komprese RLE, která je dostatečně jednoduchá na implementaci na obvodu FPGA. Skládá se z několika bloků, z nichž každý je označen úvodním bytem. Spodních 6 bitů udává velikost bloku (po přičtení 1 k jejich hodnotě), jeden kódovaný blok

může mít tak velikost až 64 bodů, stačí tedy pouze jeden kódovaný blok na vyplnění celého bloku. Sedmý bit pak udává typ kódovaného bloku, zda se jedná o nekomprimovaná data nebo o body se stejnou hodnotou. Osmý bit je zatím použit jen u prvního bloku (udává, zda se jedná o pohybový vektor nebo o plně definovaný blok).

Za úvodním bytem následují hodnoty bodů ve formátu RGB3:3:2 (3 bity na červenou a zelenou, 2 bity na modrou). Je-li požadován formát 3:3:3 (3 bity na barvu), je pro každých 8 hodnot bodů vložen 1 byte s hodnotami nejnižšího bitu modré barvy před zadáním hodnoty bodu, který již tento bit definován nemá. Běh bodů se stejnou hodnotou tedy může snížit počet těchto vkládaných bytů na méně, než maximálních 8.

V nejhorším případě může mít blok velikost 73 bytů, v nejlepším pak 3 byty. Formát RGB byl zvolen pro svoji jednoduchost a přímou použitelnost na zvolené platformě. Výstupní kvalita barev je příliš nízká a jiné formáty barev by ji spíše snížily, a přitom nepřinesly výrazné snížení velikosti video-sekvence.

4.2 Filtrování a kvantování dat

Filtrování před samotnou kompresí má veliký vliv na výslednou kvalitu a velikost video-sekvence.

4.2.1 Kvantování dat

Data mohou být kvantována jednoduchým snížením počtu bitů na barvu bitovými posuny, ale jak ukazuje obrázek 4.3, má výsledek příliš ostré přechody a především tmavší scény nemusejí být vůbec viditelné. Zato použití RLE komprese vykazuje velice dobré výsledky, přibližné snížení velikosti dat je na ½ až ¼ původní velikosti.

Zlepšením viditelnosti tmavších scén lze dosáhnout zaokrouhlováním hodnot. Toho lze snadno docílit přičtením poloviny kvantovací hladiny před bitovým posunem. Při hloubce 3 bity na barvu to odpovídá zvýšení jasu o 1/16 rozsahu, scény jsou sice pak světlejší, ale jsou lépe vidět detaily a výsledná komprese není moc ovlivněná.

Nejlepší dosažené kvality lze docílit kvantováním s opravou chyb. Spočívá k přičítání nepoužité části předchozího bodu před kvantováním. Komprese je stále velice rychlá, výsledný obraz obsahuje více detailů. Tato metoda je ale vhodná spíše při video-sekvence s vyšším rozlišením, neboť při nižším rozlišení je toto umělé zvýšení kvality dobře viditelné a může působit rušivě. Také dochází k výraznému navýšení rozdílnosti sousedních bodů, což vede k výraznému snížení účinnosti zvolené RLE komprese. Tato RLE komprese by pak musela být upravena, ale dosavadní zvolená metoda vykazuje snížení velikosti jen na ½ a méně..

Také lze pro zvýšení kvality použít metodu ditheringu – aplikováním umělého šumu vhodnou změnou barvy některých bodů. Obrázek je pak více přirozený a chyby vznikající při kvantizaci jsou méně viditelné. Tato metoda však také snižuje účinnost metody RLE.

4.2.2 Filtrace

Vhodná filtrace před kvantování nebo po něm může výrazně zvýšit účinnost komprese na úkor mírného snížení kvality, nebo naopak zvýšit kvalitu obrazu.

Účinnost zvolené komprese lze zvýšit změnou hodnot sousedních bodů na stejnou hodnotu a tolerancí malých rozdílů při hledání podobných bloků. Komprese tak bude ztrátovější, a navíc mohou být patrné rozdíly při příchodu následujícího I snímku, ale vliv na kompresi je velice výrazný.

Kvalita pak může být zvýšena rozšířením kvantování, například kvantování s opravou chyb, ale bude přizpůsobená snaze o snížení následků na účinnost komprese. Případně může vhodný filtr volit typ kvantování v závislosti na tmavosti snímku a míry ztráty detailů kvantováním.

4.3 Odhad kompresního poměru

Veškeré hodnoty jsou uvedeny pro formát RGB 3:3:2. Kompresní poměr zvolené RLE komprese se pohybuje (na základě měření na několika snímcích) v průměru kolem 3. Pohybový vektor vykazuje kompresní poměr 36 (původní velikost bloku 72B je snížena na 2B). Výsledný kompresní poměr tedy velice závisí na počtu bloků, kódovaných pohybovými vektory.

Počet takto kódovaných bloků odpovídá typu video-sekvence a typu filtrování/kvantování a případném použití B snímků. Může se běžně pohybovat od 80% (minimální změny ve video-sekvenci, například dokumentární filmy) až do 30% (akční, rychle se měnící scény). To odpovídá výslednému kompresnímu poměru v rozsahu 4-11. S vhodnou filtrací lze docílit i u hůře komprimovatelných video-sekvencí průměrného kompresního poměru 10.

5 Rozhraní IDE

5.1 Konektor rozhraní IDE

K připojení standardního disku IDE se používá konektor, který obsahuje 40 pinů. Stejné rozhraní je možné beze změny využít i ke komunikaci s paměťovou kartou Compact Flash (CF). Význam a umístění jednotlivých pinů je uvedeno v tabulce 5.1. Symbol „ - “ u některých signálů značí jejich aktivitu v log. 0. Ke komunikaci se používá úrovní signálů TTL, které jsou základním nastavením obvodu FPGA na platformě FITKit. Následující popis vychází ze specifikace ATA-3, která by měla být v dnešní době podporována téměř všemi pevnými disky.

Signál	I/O	číslo pinu	Signál	I/O	číslo pinu
RESET-	I	1	Zem		2
DD7	I/O	3	DD8	I/O	4
DD6	I/O	5	DD9	I/O	6
DD5	I/O	7	DD10	I/O	8
DD4	I/O	9	DD11	I/O	10
DD3	I/O	11	DD12	I/O	12
DD2	I/O	13	DD13	I/O	14
DD1	I/O	15	DD14	I/O	16
DD0	I/O	17	DD15	I/O	18
Zem		19	klíč		20
DMARQ	O	21	Zem		22
DIOW-	I	23	Zem		24
DIOR-	I	25	Zem		26
IORDY	O	27	CSEL		28
DMACK-	I	29	Zem		30
INTRQ	O	31	Rezervováno		32
DA1	I	33	PDIAG-		34
DA0	I	35	DA2	I	36
CS0-	I	37	CS1-	I	38
DASP-		39	Zem		40

Tabulka 5.1: Jednotlivé piny konektoru (I=disk pouze čte, O=disk pouze zapisuje, I/O=obousměrný)

Funkce jednotlivých pinů:

- RESET-

Hardwarový reset (inicializace). Po inicializaci disky provedou diagnostiku

- DD0-DD15
  Obousměrné datové vodiče. Přenos dat probíhá po 16-ti bitech, přístup k registrům po 8-mi bitech (signály DD0-DD7)
- klíč
  Tento pin bývá odstraněn a měl by zabránit špatnému otočení kabelu
- DMARQ
  Disk signalizuje, že může přenést jedno slovo dat v DMA režimu
- DMACK-
  Tímto signálem signalizuje řadič disku platnost dat v DMA režimu
- DIOW-
  Tento signál signalizuje zápis dat na disk (použit ve všech režimech)
- DIOR-
  Tento signál signalizuje čtení dat z disku (použit ve všech režimech)
- IORDY
  Signalizace disku, že je připraven na další přenos dat (jen PIO režim). Je nutný jen v PIO módech 2 a výš, v nižších je pouze volitelný
- INTRQ
  Signalizuje dokončení operace, je vypínatelný
- DA0, DA1, DA2, CS0-, CS1-
  Tyto signály slouží k adresaci registru (viz Tabulka 2)
- CSEL
  Umožňuje konfiguraci disků podle umístění na kabelu
- DASP-,PDIAG
  Jsou použity disky k diagnostice a vzájemné identifikaci

5.2 Využití signálů

Disk může v dnešní době pracovat v několika režimech, lišících se přenosovou rychlostí a řízením přenosu. Nejjednodušším je přenos dat v PIO režimu, kdy jsou využity signály DD0-DD15, DA0-DA2, CS0-, CS1-, DIOW- a DIOR-. Komplikovanější avšak výkonnější DMA režim navíc vyžaduje obsluhu signálů DMARQ a DMACK-.

Je vhodné též využít signál RESET-, nebo jej trvale umístit do stavu log. 1. Signály INTRQ a IORDY mohou být použity volitelně. Ostatní signály by měly zůstat nezapojené.

5.3 Komunikace s pevným diskem

Komunikace probíhá prostřednictvím zápisu/čtení do/z řídících/datových registrů. Jednotlivé registry a jejich adresy jsou uvedeny v tabulce 5.2 (hodnoty signálů CS1- a CS0- jsou konečné, tedy po invertování).

Adresa					Význam při čtecí a zápisové operaci	
CS1-	CS0-	DA2	DA1	DA0	čtení	zápis
1	1	0	0	0	klidový stav + DMA přenos	
1	0	1	1	0	alternativní status + řízení zařízení	
0	1	0	0	0	přenos dat (PIO)	
0	1	0	0	1	chybový registr	funkce
0	1	0	1	0	počet sektorů	
0	1	0	1	1	LBA(7-0)	
0	1	1	0	0	LBA(15-8)	
0	1	1	0	1	LBA(23-16)	
0	1	1	1	0	LBA(27-24) + výběr disku	
0	1	1	1	1	status	příkaz

Tabulka 5.2: Adresy jednotlivých registrů

5.3.1 Registry rozhraní

- klidový stav + DMA přenos: Tato kombinace adresních bitů definuje klidový stav a je používána v případě DMA přenosu, kdy je nutné nastavit tuto adresu, což umožňuje přerušit probíhající DMA přenos
- alternativní status : vrací stejné výsledky, jako registr status, ale neruší stav chyby.
- řízení zařízení : bit 1 = vypnutí přerušení (0=přerušení zapnuto), bit 2 = softwarový reset
- chybový registr : pokud je nastaven bit 0 v registrech status/alternativní status, vrací seznam chyb:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
r	UNC	MC	IDNF	MCR	ABRT	TK0NF	AMNF

Tabulka 5.3: Přiřazení jednotlivých bitů chybového registru

Význam bitů chybového registru:

- AMNF: adresa nebyla nalezena i po nalezení správného ID sektoru
- TK0NF: stopa číslo 0 nenalezena
- ABRT: příkaz přerušen (neznámý příkaz, nebo nastala chyba)
- MCR: žádost o změnu média (pouze u vyměnitelných médií)
- IDNF: sektor nenalezen
- MC: médium změněno (pouze u vyměnitelných médií)
- UNC: neopravitelná data (při zápisu/čtení)
- r: rezervován

- status registr/alternativní status : vrací stav média, čtení tohoto registru ruší chybové stavy

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
BSY	DRDY	DF	DSC	DRQ	CORR	IDX	ERR

Tabulka 5.4: Přiřazení jednotlivých bitů stavového registru

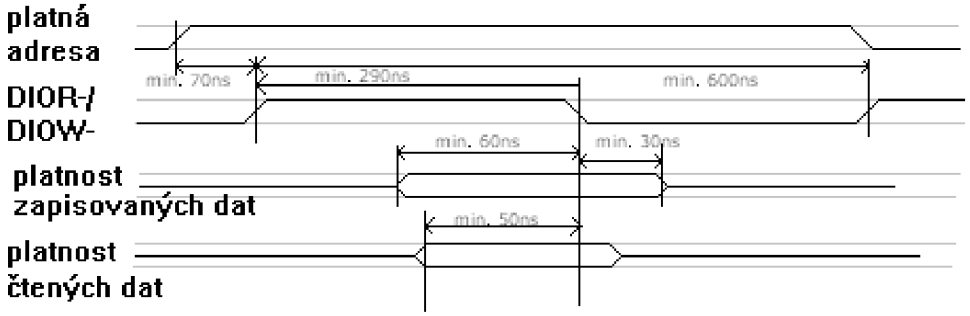
Význam bitů stavového registru:

- ERR: nalezena chyba, popis chyby je v chybovém registru
- IDX: záleží na výrobci
- CORR: nalezena opravitelná data, přenos dat je nepřerušen

    - DRQ: žádost o data - disk je připraven přenášet data
    - DSC: stopa nalezena
    - DF: chyba zařízení - porucha disku
    - DRDY: zařízení připraveno
    - BSY: zařízení zaneprázdněno - nepřijímá další příkazy
- funkce: slouží pro zapsání dodatečných parametrů pro některé příkazy
- počet sektorů: určuje počet přenášených sektorů naráz, 0 = 256
- LBA(23-0): adresa sektoru v režimu LBA28
- LBA(27-24) + výběr disku: nejvyšší 4 bity adresy, bity 7 a 5 mají být nastaveny na 1, bit 6 je nastaven na 1 pro LBA režim, bit 4 určuje vybraný disk (0=MASTER)
- příkaz: provedení příkazu disku, příkaz se spouští zápisem kódu na tento registr (všechny potřebné hodnoty už musejí být nastaveny)

5.3.2 Přenosový režim PIO

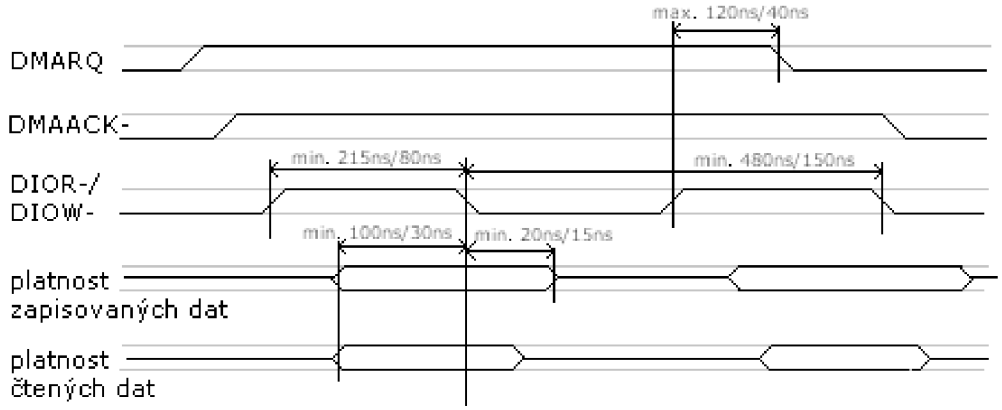
V režimu PIO je celý přenos v režii řadiče komunikujícího s diskem. Veškeré doby jsou uvedeny pro rychlost PIO 0. Přenos znaku (v případě registru) nebo slova (jsou-li přenášena data) musí být vždy zahájen nastavením správné adresy (CS1-, CS0-, DA2-DA0) -viz tabulka 5.2. Po ustalovací době adresních bitů (min. 70ns) může být aktivován signál DIOW- v případě jedná-li se o zápis nebo signál DIOR- v případě čtení. Vybraný signál musí být nastaven min. po dobu 290ns. Při zápisu dat musí být na datových vodičích data přítomna min. 60ns před deaktivací signálu DIOW- a min. 30ns po jeho deaktivaci. Čtená data jsou platná min. 50ns před deaktivací signálu DIOR-. Minimální délka cyklu je 600ns.



Obrázek 5.1: PIO režim

5.3.3 Přenosový režim DMA

V režimu DMA je přenos řízen diskem. Během přenosu musí řadič nastavit neutrální adresu (signály CS0- a CS1- jsou na 1 po zinvetování). Veškeré doby jsou uvedeny pro rychlosti DMA0/DMA1. Přenos zahajuje disk nastavením signálu DMARQ, pokud řadič může data zpracovat či je dodat, tak nastaví signál DMAACK. Poté vždy řadič nastaví signál DIOR-/DIOW- podle směru dat (nastaveny musejí být min. po dobu 215ns/80ns), data musejí být nastavena diskem/řadičem (čtení/zápis na disk) min. 100ns/30ns před zrušením těchto signálů. Při zápisu musí řadič data na výstupu podržet min. 20ns/15ns po ukončení DIOW-. Disk může zrušit signál DMARQ nejpozději 120ns/40ns po nastavení signálů DIOR-/DIOW-. Minimální doba mezi náběžnými hranami signálů DIOW-/DIOR- je 480ns/150ns, což určuje max. přenosovou rychlost.



Obrázek 5.2: DMA režim

5.3.4 Přehled nejdůležitějších příkazů

Příkazy se provedou zapsáním potřebných hodnot do registrů a pak zapsáním čísla příkazu do registru příkaz. Následuje seznam příkazů a jejich číselných kódů:

Příkazy čtení/zápisu:

- PIO čtení 0x21
- PIO čtení s návratem 0x20
- PIO zápis 0x31
- PIO zápis s návratem 0x30
- DMA čtení 0xC9
- DMA čtení s návratem 0xC8
- DMA zápis 0xCB
- DMA zápis s návratem 0xCA

vstupy:

- adresa prvního sektoru a číslo disku v registrech LBA(23-0) (předpokládáme vždy režim LBA).
- počet sektorů (1-256) ke čtení/zápisu v registru počet sektorů

popis:

- zahájení čtení/zápisu v PIO/DMA režimu a příp. s návratem (opakování čtení špatně načtených registrů).
- data jsou přítomna při nastavení stavu DRQ ve status registru
- ve specifikaci ATA 3 musí být přítomny všechny typy těchto příkazů, ale používány by měly být spíše typy s návratem (na testovaném disku verze bez návratu nepracovala tak, jak měla)

Příkazy identifikace disku

- Identifikace PIO 0xEC
- Identifikace DMA 0xEE

popis:

- přenos 1 sektoru (512B dat) z disku, obsahující informace aktuální o disku. Verze DMA je ve specifikaci ATA3 pouze dobrovolná (je tedy nutné počítat i s PIO verzí)
- některé (nejpoužívanější) položky (pro úplný seznam viz specifikace ATA):

Offset	Velikost [B]	Popis
14H	20	Sériové číslo disku (opačné pořadí znaků a znaky vyplněny do konce mezerami)
36H	40	Model disku (jako u sériového čísla)
78H	4	počet adresovatelných sektorů (pouze LBA)
7EH	2	bity 15-8 - aktivní DMA režim, bity 7-0 - podporovaný DMA režim
A0H	2	bity 15-1 podporovaná specifikace ATA (bit 1=ATA1, ...)

Tabulka 5.5: Některé položky identifikace disku

Příkaz nastavení vlastností disku:

- Nastav vlastnosti 0xEF

vstupy:

- registr funkcí obsahuje číslo podpříkazu
- další registry podle jednotlivých podpříkazů

popis:

- nastaví některou z vlastností, může například nastavovat vyrovnávací paměť, spotřebu disku, či režim přenosu
- režim přenosu: podpříkaz 03H, hodnota se nastavuje do registru počtu sektorů. Typ režimu je vybrán horními 5 bity, spodní 3 bity určují hodnotu.

Horních 5 bitů	Dolní 3 bity	Popis
00000	000	Základní PIO režim
00000	001	Základní PIO režim (bez řízení s IORDY)
00001	nnn	PIO režim nnn
00100	nnn	DMA režim nnn

Tabulka 5.6: Nastavení přenosového režimu

Další příkazy (viz specifikace ATA):

- příkazy pro přenášení skupin sektorů (čtení/zápis, PIO/DMA)
- příkazy pro verifikaci zápisu/čtení
- příkazy pro řízení stavu disku (spotřeby)
- příkazy SMART
- příkazy zabezpečení dat

Specifikace ATA2 a 3 viz [2], [3].

5.4 Připojení IDE k FITKitu

FITkit je připojen k IDE přes standardní 40-pinový kabel, pro FITkit verze 1.x musí být konekor upraven obroušením jeho okrajů, protože by jinak nešel použít kvůli pinům konektoru. FITKit 2.0 má jiné umístění konektoru (nevyžaduje jeho úpravu). FITKit 2.0 ale nemá připraven žádný typ architektury.

Obrázek 5.3: Oříznutí kabelu

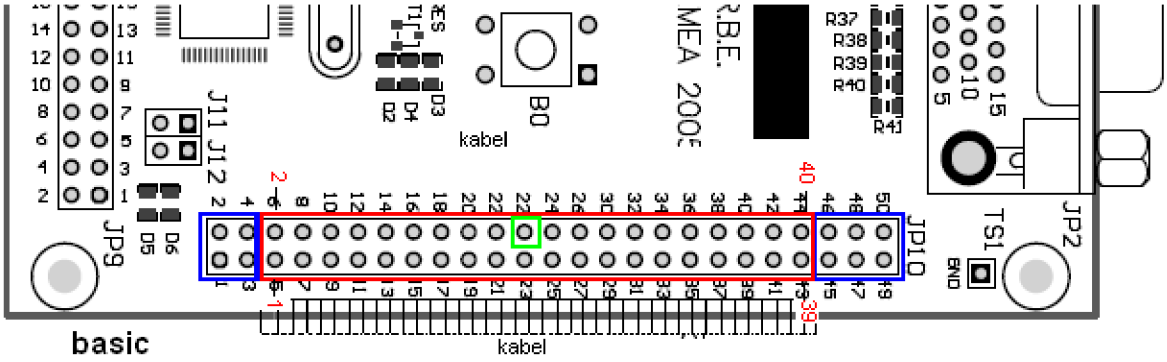
Obrázek 5.4: Komponenta IDE_port

- X: 46-bitový port, slouží pro přímé napojení na port X FPGA komponenty (konektor)
- IDE: 26-bitový port, napojuje se na port IDE řadiče, obsahuje všechny nutné signály

Komponenta má několik architektur, které jsou přizpůsobeny poloze konektoru:

5.4.1 Architektura basic

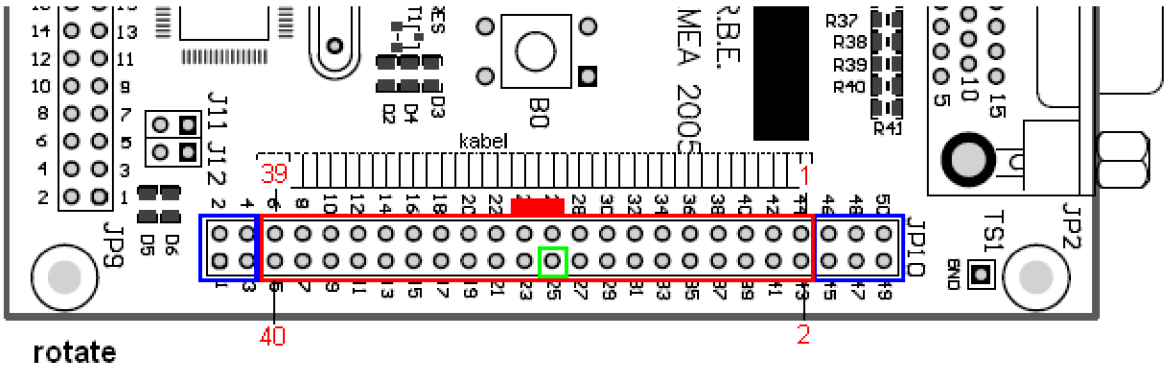
Je určena pro standardní polohu kabelu, viz obrázek 5.5.



Obrázek 5.5: Poloha kabelu pro architekturu basic

5.4.2 Architektura rotate

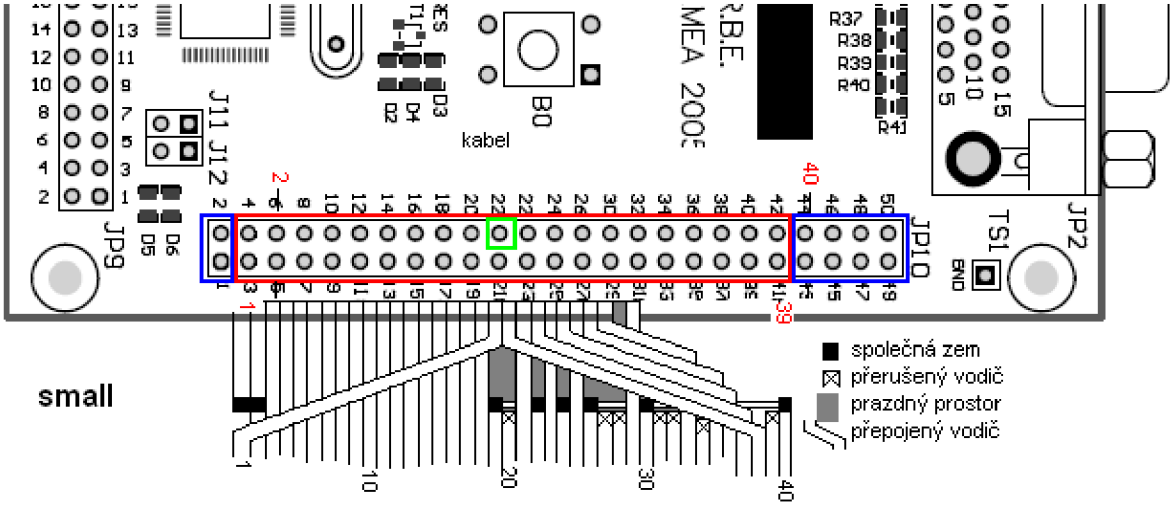
Je určena pro obrácenou polohu kabelu, vhodné pro umístění FITKitu na disk - zabraňuje přetočení kabelu.



Obrázek 5.6: Poloha kabelu pro architekturu rotate

5.4.3 Architektura small

Je určena pro minimální využití portu X, umožňuje použití standartních komponent, vhodné ve spojení s architekturou čipu IDE (port X je přítomen společně se standartními porty). Pro tuto variantu je však potřeba speciálně upravit kabel a konektor se umísťuje jinak, aby se využily piny se zemí.



Obrázek 5.7: Poloha kabelu pro architekturu small

5.4.4 Úprava kabelu pro architekturu small

Je potřeba přepojit několik vodičů - lze využít samotný konektor, neboť obsahuje samořezná upínací místa. Při návrhu se předpokládalo přehnutí vývodů 33-40 ke kabelu, což umožňuje snížení rozdílu v délce původních a přepojených vodičů. Nadstavovat je nutné pouze vývod č. 1. Tabulka 5.7 ukazuje seznam vývodů a jejich přepojení. Je vhodné (ale ne nutné) pospojovat vodiče, označené "Na společnou zem" a připojit je na vývody 1 a 2.

Vodič	Úprava	Připojit vodič	Vodič	Úprava
1	Přepojit na 19	Společná zem	29	Beze změny
2	Zem		30	Na společnou zem
3-18	Beze změny		31	Zrušit
19	Na společnou zem	1	32	Zrušit
20	Zrušit	38	33	Přepojit na 27
21	Beze změny		34	Zrušit
22	Na společnou zem	37	35	Přepojit na 26
23	Beze změny		36	Přepojit na 24
24	Na společnou zem	36	37	Přepojit na 22
25	Beze změny		38	Přepojit na 20
26	Na společnou zem	35	39	Zrušit
27	Zrušit	33	40	Na společnou zem
28	Zrušit			

Tabulka 5.7: Přepojení pinů na konektoru

6 Souborový systém FAT32

Souborový systém FAT je poměrně jednoduchý na implementaci a je podporován téměř ve všech operačních systémech. I když je v dnešních dobách nahrazován jinými systémy, které jsou spolehlivější a dosahují vyšší účinnosti, je tento systém stále přítomen například na přenosných paměťových kartách (především tedy systém FAT32). Existuje ve 3 variantách: FAT12, používaný na disketách a discích do 40MB, FAT16 a FAT32. Tento systém byl zvolen pro tuto práci právě díky jednoduché implementaci a velké podpoře mnoha operačních systémů, zvolena byla varianta FAT32 (jako jediná vhodná pro větší kapacity).

6.1 Master Boot Record MBR

Tabulka MBR se nachází v sektoru 0 a popisuje jednotlivé oddíly, nacházející se na disku. Na začátku sektoru se nachází 446B spustitelného kódu. Za touto oblastí se nachází čtyři 16-bytové popisy jednotlivých oddílů. Sektor je uzavřen znakem AA55H, který označuje MBR sektor. Jednotlivé popisy mají strukturu, uvedenou v tabulce 6.1, v tabulce 6.2 je pak seznam jednotlivých typů souborových systémů.

Offset	Velikost [B]	Popis
00H	1	Stav(0 = neaktivní, 80H = aktivní)
01H	1	Začátek – číslo hlavy
02H	2	Začátek – cylindr/sektor
04H	1	Typ oddílu (tabulka 6.2)
05H	1	Konec – číslo hlavy
06H	2	Konec – cylindr/sektor
08H	4	Logické číslo počátečního sektoru
0CH	4	Počet sektorů oddílu

Tabulka 6.1: 1 položka MBR

Typ	Popis
00H	Neznámý/žádný
01H	FAT12
04H	FAT16 (menší než 32MB)
05H	Rozšířený oddíl DOS
06H	FAT16 (větší než 32MB)
0BH	FAT32
0CH	FAT32 s LBA rozšířením (služby 13H BIOS)
0EH	FAT16 s LBA
0FH	Rozšířený oddíl DOS s LBA

Tabulka 6.2: Typy souborových systémů

Rozšířený oddíl DOS umožňuje větší počet oddílů na disku, než jsou základní čtyři. Ukazuje na strukturu, podobnou MBR, která obsahuje popis dalšího oddílu FAT a případně odkaz na další rozšířený oddíl. Adresa je ovšem vztažena k pozici aktuální tabulky, je tedy nutné je sečíst. Poslední rozšířený oddíl obsahuje již pouze poslední oddíl FAT.

6.2 FAT Boot Record

Podrobněji bude popisován systém FAT32, ostatní systémy pouze okrajově. První sektor oddílu FAT je Boot Record (viz tabulka 6.3). Obsahuje všechny potřebné údaje ke správnému nastavení oddílu. V tabulce jsou přiložena i běžně používaná jména položek.

Offset	Velikost [B]	Popis	Jméno
00H	3	Adresa spustitelné oblasti	JumpCode
03H	8	Jméno systému, který oddíl vytvářel	OEMName
0BH	2	Počet bytů na sektor (512,1024,2048,4096)	BytsPerSec
0DH	1	Sektorů na cluster	SecPerClus
0EH	2	Rezervovaných sektorů (běžně 32)	RsvdSecCnt
10H	1	Počet FAT tabulek (běžně 2)	NumFATs
11H	2	Počet položek v Root Directory (pro FAT32=0)	RootEntCnt
13H	2	Celkový počet sektorů.=0: údaj je ve 32b v.	TotSec16
15H	1	Typ disku, = F8H: pevný	Media
16H	2	Počet sektorů na FAT t.=0: údaj je ve 32b v.	FATSz16
18H	2	Sektorů na stopu	SecPerTrck
1AH	2	Počet hlaviček	NumHeads
1CH	4	Počet skrytých sektorů	HiddSec
20H	4	Celkový počet sektorů.=0: údaj je ve 16b v.	TotSec32
24H	4	Počet sektorů na FAT t.=0: údaj je ve 16b v.	FATSz32
28H	2	Aktivita FAT tabulek, bit 7 = 1: pouze jedna aktivní FAT tabulka (její číslo v bitech 0-3), = 0: mirroring	ExtFlags
2AH	2	Verze FAT, měla by být = 0	FSVer
2CH	4	Číslo clusteru kořenového adresáře	RootClus
30H	2	Číslo informačního sektoru (obvykle 1)	FSInfo
32H	2	Číslo záložního boot sektoru	BkBootSec
34H	12	Rezervováno	
40H	1	Písmeno disku, pevné disky začínají od 80H	DrvNum
41H	1	Rezervováno	
42H	1	Následující 3 položky jsou přítomné:	BootSig
43H	4	Číslo svazku, pro identifikaci	VolID
47H	11	Jméno svazku	VolLab
52H	8	Typ systému, obvykle "FAT32 "	FilSysType

Tabulka 6.3: Boot Record

6.3 Informační struktura

Informační struktura, pokud se nachází na oddílu (pouze pokud FSInfo>0), obsahuje některé informativní údaje, především údaj o volném počtu clusterů (=volné místo) a naposledy alokovanému clusteru (pro urychlení hledání volného clusteru). Tyto údaje jsou pouze informativní a nemusejí být správné.

Offset	Velikost [B]	Popis
1E8H	4	Počet volných clusterů, = -1: neznámý
1ECH	4	Číslo clusteru, který byl naposledy alokován (zrychlení vyhledávání)

Tabulka 6.4: Informační struktura

6.4 Základní informace

Nejprve je třeba zjistit počet položek, alokovaných zvlášť pro kořenový adresář(pouze FAT12 a FAT16):

RootDir = [(RootEntCnt * 32) + BytsPerSec - 1]/BytsPerSec

Pro systém FAT32 bývá RootEntCnt = 0, Tedy i RootDir = 0. Poté je nutné zjistit číslo sektoru clusteru č. 2 (první cluster v systému). Před daty jsou tabulky FAT, rezervované sektory a sektory kořenového adresáře (pouze FAT12 a FAT16):

ClusStart = (NumFATs * FATSize) + RsvdSecCnt + RootDir

Nesmíme zapomenout, že čísla sektorů jsou vztažena k začátku oddílu. Následuje výpočet počtu clusterů v oddíle:

Clusters = (Sectors - ClussStart) / SecPerClus

Typ FAT tabulky by měl být určen právě z počtu clusterů, nikoliv tedy ze jména svazku (FilSysType) nebo jiných typických znaků. Pokud je počet clusterů menší než 4085, jedná se o FAT12, je-li větší nebo roven 4085 a menší, než 65525, jedná se o FAT16 a je-li větší nebo roven 65525, jedná se o FAT32. Číslo sektoru, na kterém se nachází cluster, lze spočítat:

Sector = (Cluster-2)*SecPerClus + ClusStart

6.5 Adresářová struktura

Adresáře jsou podobné souborům, neobsahují však informaci o svoji délce a mají nastaven atribut DIRECTORY. Přístup k nim je však podobný. Adresář obsahuje jednotlivé položky, dlouhé 32B, které popisuje tabulka 9. Položky jsou řazeny za sebou a konec adresáře je označen zarážkou (je tvořena vynulovanou položkou) . Kořenový adresář se ve FAT32 nachází na clusteru č. RootClus.

Offset	Velikost [B]	Popis	Jméno
00H	8	Jméno položky, prázdná místa vyplněna znakem 32	Name
08H	3	Koncovka p., prázdná místa vyplněna znakem 32	Ext
0BH	1	Atributy položky	Attr
0CH	1	Rezervováno	NTRes
0DH	1	Čas vytvoření – setiny sekundy (pro 2 sekundy)	CrtTimeTenth
0EH	2	Čas vytvoření – granularita jsou 2 sekundy	CrtTime
10H	2	Datum vytvoření	CrtDate
12H	2	Datum posledního čtení/zápisu	LstAccDate
14H	2	Číslo prvního clusteru (horní polovina)	FstClusH
16H	2	Čas poslední změny (včetně vytvoření souboru)	WrtTime
18H	2	Datum poslední změny (včetně vytvoření souboru)	WrtDate
1AH	2	Číslo prvního clusteru (dolní polovina)	FstClusL
1CH	4	Délka souboru	FileSize

Tabulka 6.5: Položka adresáře

Typ položky je určen prvním znakem jména:

- E5H – volná položka
- 05H – normální položka, první znak = E5H
- 00H – prázdná položka, za ní se nenachází žádná jiná položka (zarážka)
- jiná hodnota – normální položka

CrtDate, LstAccDate, WrtDate			CrtTime, WrtTime		
Rok	Měsíc	Den	Hodiny	Minuty	Vteřiny
(Bity 15 – 9)	1980|Bity 8 – 5	Bity 4 – 0	Bity 15 – 11	Bity 10 – 5	(Bity 4 – 0) * 2

Tabulka 6.6: Formát data a času

Atributy položky (kromě dlouhého jména jsou všechny komutativní):

- 01H: Pouze pro čtení
- 02H: Skrytý soubor/adresář
- 04H: Systémový soubor/adresář
- 08H: Jméno svazku (smí být pouze v kořenovém adresáři)
- 10H: Adresář
- 20H: Archivní soubor/adresář (byl změněn)
- 0FH: Dlouhé jméno

6.6 Dlouhá jména

Pokud má nějaká položka dlouhé jméno, je toto uloženo v položkách jí předcházejících a s atributem "Dlouhé jméno" (0FH). Každá taková položka obsahuje 13 znaků v kódování UTF16 a popisuje ji tabulka 6.7.

Offset	Velikost [B]	Popis	Jméno
00H	1	Pořadové číslo záznamu	SequenceNumber
01H	10	5 znaků jména	Name1
0BH	1	Atribut = 0FH	Attr
0CH	1	Rezervováno	Type
0DH	1	Kontrolní součet	CheckSum
0EH	12	6 znaků jména	Name2
1AH	2	Musí být 0	FirstClusterLO
1CH	6	3 znaky jména	Name3

Tabulka 6.7: Položka dlouhého jména

Položky jsou řazeny v opačném pořadí (od konce jména po jeho začátek), pořadové číslo je od N po 1, položka s číslem N má k tomuto číslu přičteno ještě 40H.

6.7 Tabulka FAT

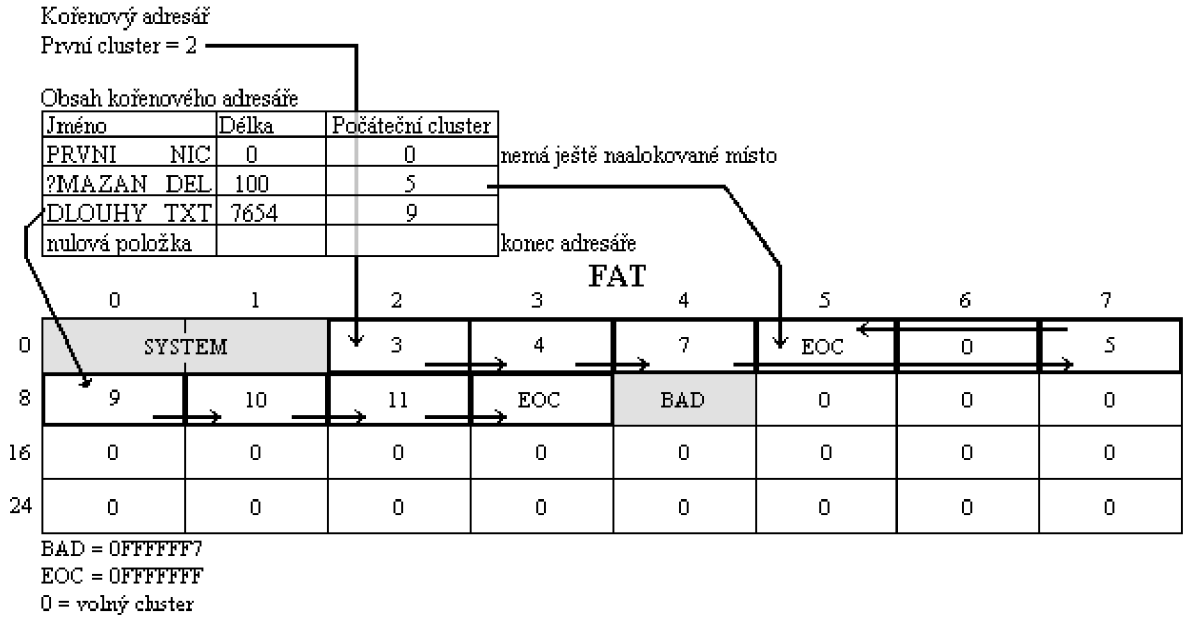
FAT tabulky obsahují údaje o použití clusterů oddílu (File Allocation Table), označují prázdné, vadné, nebo použité clustery a uchovávají číslo příštího clusteru, čímž vytvářejí jejich řetězce (a tedy i obsah souborů).

FAT tabulky se nacházejí za oblastí rezervovaných sektorů. První sektor tabulky (FATNum) se dá vypočítat pomocí vztahu:

FATSector = RsvdSecCnt + FATNum * FATSz

Pro FAT32 je tabulka tvořena polem položek o velikosti 4B, z nichž je použito 28bitů, nejvyšší 4bity jsou rezervovány a neměly by se měnit. Pro velikost sektoru 512B (nejpoužívanější) se tedy v jednom sektoru nachází 128 položek. Každá položka obsahuje číslo následujícího clusteru souboru/adresáře.

Hodnota větší nebo rovna 0FFFFFF8H znamená konec řetězce, řetězec dále již nepokračuje a aktuální cluster je poslední. Číslo prvního clusteru řetězu je uloženo v položce adresáře ve FstClusL a FstClusH. Soubor o nulové délce, který dosud nemá alokována data, má číslo 0. Hodnota 0FFFFFF7H znamená vadný cluster. Hodnota 0 označuje volný cluster. Clustery číslo 0 a 1 nesmějí být použity.



Obrázek 6.1: Ukázka tabulky FAT

Na obrázku je ukázka tabulky FAT (v tomto případě FAT32) s velikostí 2KB na cluster. Kořenový adresář má přidělených 5 clusterů (2,3,4,7,5) a obsahuje 3 položky (zbytek místa by měl být vynulován, nulová položka je poslední položkou v adresáři).

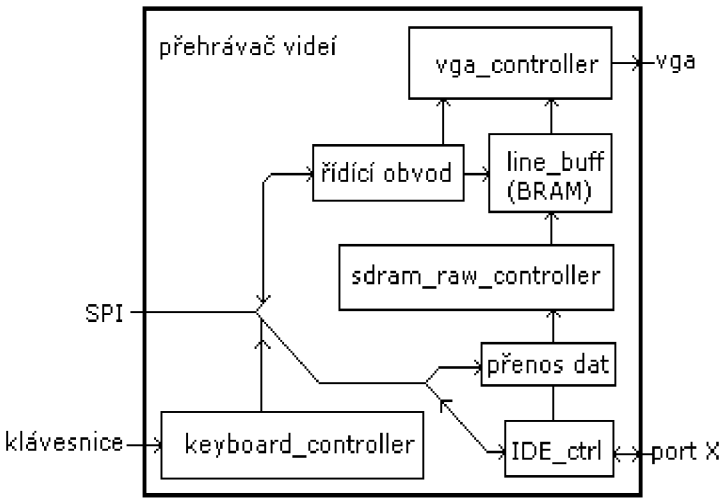
- První položka obsahuje soubor PRVNI.NIC (název by měl být psán velkými písmeny, FAT je necitlivá na velikost písmen). Ten však nemá ještě přiděleno místo, neboť jej dosud nepotřeboval, a tak má jako počáteční cluster hodnotu 0.
- Druhá položka byla smazána, avšak předtím měla přidělena cluster číslo 5 (a pravděpodobně i 6, což vysvětluje skok v řetězci adresáře).
- Třetí položka má přiděleny 4 clustery (8,9,10,11), aby mohla obsahovat všech 7654B dat.

7 Implementace v FPGA

U implementace je nutné počítat s omezením dostupných zdrojů, především s omezeným počtem programovatelných logických obvodů, paměti BRAM a dostupnou dynamickou pamětí.

7.1 Zapojení přehrávače

Přehrávač se skládá z řadiče pevných disků s výstupním kanálem a grafického řadiče, blokové zapojení viz obrázek 7.1. Přenos dat je tvořen přímým kopírováním do paměti SDRAM, případná komprese by byla řešena v této části.



Obrázek 7.1: Blokové zapojení přehrávače

7.2 Grafický řadič

Grafický řadič je tvořen již existujícím řadičem VGA rozhraní (vga_controller), řádkovým vyrovnávačem, tvořeným pamětí BRAM a pamětí SDRAM, která je řízena též již hotovým řadičem (sdram_raw_controller). Řízení paměti SDRAM je navíc vybaveno řídícím obvodem priorit přístupu – k paměti SDRAM přistupuje také řadič IDE disků a vyšší prioritu má zobrazovací část.

Řádkový vyrovnávač má snižovat dobu přístupu k paměti SDRAM, jinak by paměť SDRAM byla po většinu času nedostupná. Má kapacitu na 2 řádky, což umožňuje zobrazovat 1 řádek a dopředu načítat další, získá se tím dostatečná časová rezerva pro případ, že by paměť SDRAM byla využívána jiným obvodem. V této části je též realizována možnost umělého snižování rozlišení pomocí kopírování řádků a sloupců, a to na ½, ¼ a 1/8 původního rozlišení.

O obnovování paměti SDRAM se stará také řídící obvod řádkového vyrovnávače, který po načtení posledního řádku obnoví polovinu paměti. Všechny řádky mají být obnoveny každých 64ms, zvolená obnovovací frekvence monitoru je 60 FPS, celá paměť je tedy obnovována každých 33ms, což je více než dostatečné.

Pro aplikaci bylo vybráno rozlišení 640x480 proto, že je podporováno prakticky všemi monitory a jinými zobrazovacími zařízeními, je dostatečné na zobrazování videí a menších obrázků a nemá velké nároky na paměť SDRAM. Přestože jak zobrazovací rozhraní, tak řádkový vyrovnávač umožňují zobrazování 9-ti bitových barev (3-3-3), je modrá barva omezena na 2 bity, neboť paměť

SDRAM je jen 8-mi bitová a rozšíření by spotřebovalo mnoho prostředků. Paměť je rozdělena na 16 oblastí, 4 oblasti na 1 banku (paměť SDRAM má 4 banky). Přepínání zobrazované banky se provádí během zpětného běhu paprsku. Paměť je řídkého typu, jednotlivé řádky tedy nejsou uloženy za sebou.

Řadič je řízen 4-mi 8b registry, které se nacházejí na adresách 40H-43H, jejich význam je vyznačen v tabulce 7.1.

adresa	Význam bitů
40H	bit 7 – zapnutí zobrazování (1=zobrazuje), bity 3-0 – výběr paměťové banky
41H	bity 6-0 – horizontální rozlišení/8 (80=640)
42H	bity 5-0 – vertikální rozlišení/8 (60=480)
43H	Bity 1 a 0 – změna měřítka (0=1, 1=1/2, 2=1/4, 3=1/8)

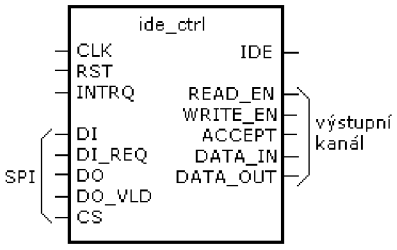
Tabulka 7.1: Řídící registry grafického řadiče

7.3 Řadič pevného disku

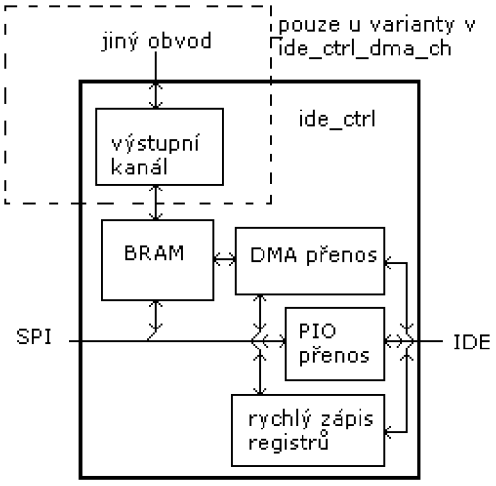
Řadič pevného disku, vycházející z již vytvořeného řadiče v mojí předchozí BP, již nevyhovuje po stránce rychlostní i po stránce možností. Využíval přenosu v PIO módu a data se přenášela přímo na MCU (mikrokontrolér). Použitý MCU má však omezené množství paměti, musel vždy přečíst celý sektor, a pak by musel data posílat zpět do FPGA. Rozhraní SPI je však poměrně pomalé (4Mbps). Byl proto rozšířen o paměť BRAM, komunikaci v režimu DMA a výstupní kanál pro přenášení dat do jiných obvodů (řízen řadičem). Existuje verze bez a s tímto kanálem.

7.3.1 Řadič IDE

Použitý řadič ide_ctrl se skládá z několika bloků a využívá komunikace přes rozhraní SPI. Existují 3 typy řadiče, 1 pouze pro PIO režim, 2 novější pro DMA režim, lišící se vstupně-výstupním kanálem pro přenos dat mezi dalšími komponentami na FPGA.



Obrázek 7.3: Řadič ide_ctrl



Obrázek 7.2: Blokové zapojení řadiče

signál	popis
IDE	26b port, určený pro připojení na komponentu ide_port
RST	reset řadiče
CLK	hodiny
DI, DI_REQ, DO, DO_VLD, CS	signály rozhraní SPI
INTRQ	plánován pro přerušení, není využit a u varianty s výstupním kanálem ani není přítomen
READ_EN	řadič má připravena data pro zařízení
WRITE_EN	řadič vyžaduje data ze zařízení
DATA_IN	vstupní data do řadiče (8b)
DATA_OUT	výstupní data ze řadiče (8b)
ACCEPT	potvrzení platnosti dat (v obou směrech) zařízením

Tabulka 7.2: Signály řadiče

7.3.2 DMA kanál

Paměť BRAM slouží nejen jako odkladiště dat, čímž umožňuje náhodný přístup k datům, ale také zároveň převádí 16-bitová data z disku na 8-bitovou šířku. Obvod DMA je nastavován pomocí 8b portu na adrese 05H, k paměti BRAM se přistupuje na registrech 0800H-0FFFH, jak ukazuje tabulka Tabulka 7.3. Obvod může používat DMA režimy 0 a 1, které lze volit. Paměť BRAM má kapacitu 2kB, umožňuje tedy uchovávat až 4 sektory a je použita i u případného výstupního kanálu.

DMA kanál může být spouštěn i po zadání příkazu čtení/zápisu, do jeho spuštění bude komunikace pozdržena (nejsou limity na dobu čekání).

adresa	Význam bitů
05H	Zápis: bit 7 - spouští DMA přenos, bit 6 – provádí RESET obvodu, bit 5 – rychlost DMA (0=DMA0, 1=DMA1), bit 4 – směr dat (0=čtení z disku, 1=zápis), bity 0-1 – výběr oblasti BRAM Čtení: bit 7 – řadič stále pracuje
0800H-0FFFH	Dolních 11 bitů adresy tvoří adresu dat v paměti BRAM, umožňuje inkrementální přístup k datům (přenášení bloku) – zápis i čtení

Tabulka 7.3: Registry DMA obvodu

7.3.3 PIO mód

Samotný řadič stále podporuje režim PIO, neboť přístup k registrům disku či některé operace (jako identifikace disku) jej vyžadují. Identifikace disku přes režim DMA je ve specifikaci ATA3 pouze volitelná, proto řadič podporuje ještě i tento typ přenosu. Port pro PIO režim je na adrese 0300H – 033EH, bity 5-1 určují číslo registru disku.

Neboť pro přečtení/zapsání sektoru je potřeba nastavit celkem 6 registrů (SCR – počet sektorů, 4 registry adresy a pak samotný příkaz) a na nastavení 1 registru je potřeba odeslat 5 bytů přes rozhraní SPI, je celkově potřeba odeslat 30 bytů, což je poměrně hodně a má za následek snížení rychlosti práce s diskem. Z toho důvodu byl přidán obvod pro rychlý přístup k registrům, který využívá toho, že výše zmíněné registry leží adresou hned vedle sebe. Registr na přístup k nim využívá

adresy 0000H-001FH, přičemž stačí zadat adresu registru SCR, používá se tedy adresa 0012H. Takto stačí přenést jen 9 bytů, což je výrazně méně. Tento registr je pro jednoduchost určen jen pro zápis.

adresa	Význam bitů
0300H – 033EH	adresní bity 5-1 určují adresu registru, bity 15-0 potom data
0000H-001FH	Adresní bity 4-0 určují adresu prvního registru, každých 8 bitů pak data (jen zápis)

Tabulka 7.4: Registry PIO režimu

7.3.4 Výstupní kanál

Pro přenos dat do dalších obvodů slouží výstupní kanál, který je tvořen signály, popsanými v tabulce Tabulka 7.5.

signál	směr	funkce
READ_EN	ven	Řadič má data pro zařízení
WRITE_EN	ven	Řadič vyžaduje data
ACCEPT	dovnitř	Potvrzení dat → přechod na další znak
DATA_IN	dovnitř	Vstupní data řadiče
DATA_OUT	ven	Výstupní data z řadiče

Tabulka 7.5: signály výstupního kanálu

Pokud chce obvod data číst z/zapisovat do vnitřní paměti BRAM (paměť sektorů), vystaví signál READ_EN/WRITE_EN a zapíše data na DATA_OUT/čte data z DATA_IN. Komunikující zařízení potvrzuje data signálem ACCEPT.

Tento obvod pro jednoduchost blokuje čtení/zápis dat MCU, ale to by nemělo způsobovat problémy, neboť pokud není přenos zbytečně zdržován, jsou data přenesena dříve, než může MCU začít se čtením/zápisem (Testováno při vnitřní frekvenci 50MHz, přenosu 512B bloků a čekáním na uvolnění paměti SDRAM grafickým řadičem).

Obvod je řízen registrem na adrese 10H nebo 11H, který je 24-bitový a je popsán v tabulce Tabulka 7.6.

Port 10H-11H	Význam bitů
Zápis:	bit 23 – spouští obvod, bit 22 určuje směr dat (0=z obvodu), bity 21-11 jsou cílová adresa, bity 10-0 pak počáteční adresa
Čtení:	bit 23 – obvod běží, bity 21-11 jsou cílová adresa, bity 10-0 pak aktuální adresa

Tabulka 7.6: Registr výstupního kanálu

7.4 Dekompresní jednotka

Dekompresní jednotka zatím není realizována, neboť všechny uvedené části spotřebovaly příliš mnoho prostředků (86% konfigurovatelných logických bloků), byla zvažována komprese RLE, na kterou by měl být dostatek prostředků, ale ta sama o sobě nepřináší velké zlepšení. Je tedy potřeba zoptimalizovat všechny ostatní obvody a snížit jejich spotřebu prostředků, aby jich bylo dostatek na použití pohybových vektorů. To by mělo být usnadněno použitím 2 pamětí BRAM, které zůstaly k dispozici.

Samotná jednotka je nahrazena přímým spojením výstupního kanálu s pamětí SDRAM, na určení cílové adresy je použit 24-bitový registr na adrese 80H, bity 22-0 určují cílovou adresu.

7.5 Přehled použitých portů

Přehled všech použitých portů a jejich adresy, + v délce dat značí blokový přenos, pro délku dat 3 je vhodné použít upravené makro FPGA_SPI_RW_A8_D3.

adresa	režim	použití	délka adresy (B)	délka dat(B)
00H	W	hromadný přístup k registrům disku	2	1+
03H	W/R	zápis/čtení PIO režim	2	2
04H	W	řízení hardwarového resetu	1	1
05H	W/R	řízení/stav DMA kanálu	1	1
08H-0FH	W/R	čtení/zápis dat z sektorové paměti	2	1+
10H-11H	W/R	řízení/stav výstupního kanálu	1	3
40H	W	spouštění grafického rozhraní/výběr bank	1	1
41H	W	šířka obrazu	1	1
42H	W	výška obrazu	1	1
43H	W	snížení rozlišení	1	1
80H-81H	W	adresa v paměti pro výstupní kanál	1	3

Tabulka 7.7: celkový přehled portů

7.6 Programové vybavení MCU

Programové vybavení je tvořeno jednotkami ide.h a fat.h, které zajišťují komunikaci s pevným diskem a přístup na souborový systém FAT32. Popis funkcí a ukázky kódu se nacházejí v příloze č. 2.

Microcontrolér má na starosti přípravu pevného disku, nastavovaní DMA a výstupního kanálu, práci s oddíly na disku a souborovým systémem FAT. Ve vytvořené aplikaci také zpracovává informace o video-souboru, řídí přesun dat z pevného disku a nastavuje zobrazovací zařízení.

8 Vytvořená aplikace

8.1 Použitý video-formát

Základní koncovka pro zvolený formát je RAF – Raw Animation Format – demonstruje jeho jednoduchost, neboť složitější by pro danou úlohu nebyl vhodný.

8.1.1 Soubor typu RAF

Hlavička souboru je 28 bytů veliká a její tvar popisuje tabulka 8.1.

poloha	velikost	jméno	popis
0	4	type	Identifikace souboru, obsahuje znaky „RAF “
4	4	version	Verze souboru, zatím 00010000H (1.0)
8	2	headerSize	Velikost hlavičky = 28
10	2	width	Šířka videa
12	2	height	Výška videa
14	2	speed	Rychlost videa, zatím jen 0 (bez omezení)
16	4	frames	Počet snímků animace
20	2	comp	Typ komprese (0=snímky bez komprese)
22	2	index	Přítomnost indexu snímků, zatím 0 (bez indexu)
24	2	soundsize	Velikost bloku zvuku po každém snímku (zatím 0)
26	2	soundcomp	Typ komprese zvuku

Tabulka 8.1: Hlavička video-formátu

Po hlavičce by případně následoval index a popis zvuku, pokud by byly přítomny. Dále pak následují již jednotlivé snímky, uloženy přímo za sebou. Pokud by byla použita komprese, začínaly by identifikátorem použitého typu snímku a komprese pro aktuální snímek (předpokládá se velikost 1-2B). Data jsou zatím uložena ve formátu 3-3-2.

8.1.2 Konverze souboru

Soubor lze vytvářet pomocí vytvořeného programu avi2raf, který používá standardní API funkce a jednotku „Video for Windows“ na zpracování AVI souborů. Píše se ve tvaru:

avi2raf.exe zdrojový_avi_soubor cílový_raf_soubor

Videa lze také vytvářet ze série BMP snímků vytvořeným programem bmp2raf, který se píše ve tvaru:

bmp2raf.exe cílový_raf_soubor seznam_bmp_souborů

Je nutné, aby BMP obrázky měly barevnou hloubku 256 barev a byly stejně veliké.

8.2 Vytvořený přehrávač

8.2.1 Ovládání

Vytvořený program se ovládá z příkazové řádky, ale umožňuje omezeně použít též přítomnou klávesnici na platformě.

Nejdříve je nutné inicializovat disk a FAT, to se provádí příkazem init (též klávesa „A“). Poté je nutné zapnout zobrazování příkazem mon (klávesa „B“). Poté je možné používat další příkazy z následujícího seznamu:

příkaz	funkce
init	Inicializuje disk a FAT
mon	Zapíná monitor
dir jméno	Vypisuje soubory, odpovídající jménu, při neudání jména hledá vše
dirp jméno	Podobné dir, jen vypisuje jména bez vlastností
cd jméno	Přejde na adresář, používá i . a ..
cd \	Přejde do kořenového adresáře
cat jméno	Vypíše obsah souboru
play jméno	Přehraje soubor
size číslo	Nastaví velikost obrazovky (0 - 3)
Clear	Vyčistí obrazovku

Klávesy „1“ a „2“ pouštějí videa se jmény „t.raf“ a „t2.raf“, video se dá zastavit klávesou „D“.

8.2.2 Experimentální vyhodnocení

Rychlost přehrávání je výrazně nižší, než se předpokládala, neboť dochází ke zpoždění pomalou komunikací přes SPI. Případná komprese by rychlost mohla zvýšit, ale ta zatím není možná. Plynulého přehrávání při 25FPS lze dosáhnout u videí s rozměry 160x120, větší videa již dosahují nižších rychlostí. Dalšího zrychlení by se mohlo dosáhnout optimalizací jednotek ide.h a fat.h, případně jejich úpravou na akceleraci sekvenčního přístupu.

Jednou z možností komprese je použití komponenty picoCPU [8] (miniaturní procesor na FPGA), který by získal řízení většiny řadičů (včetně řadiče disku), což by vedlo ke snížení prostorových nároků ostatních komponent a také i snadněji programovatelnou kompresi.

9 Závěr

Cílem této diplomové práce je prověřit možnost přehrávání video-sekvencí na obvodu FPGA a realizovat ji. Tento cíl je splněn a je možno použít již hotovou platformu FITKit, která je snadno studentům dostupná a je vybavena běžně používaným typem FPGA obvodu. Na trhu jsou k dostání i výkonnější typy těchto obvodů, je tedy možné použít i mnohem náročnější (a účinnější) metodu komprese.

Přestože výkon vytvořené aplikace nedosahuje výkonu dnešních přehrávačů, dokazuje aplikace použitelnost obvodů FPGA i na náročnější úkoly a v rámci DP byl vytvořen řadič disků, využívající přenosu DMA a umožňující přenos dat přímo do dalších obvodů, který je zastřešený knihovními funkcemi. Popisy řadiče a aplikace se také nacházejí na stránkách FITKitu [5].

Literatura

[1] David Salomon, D.: *Data Compression*. Springer, 2004

[2] *AT Attachment Interface with Extensions (ATA-2)*. Revize 4c [online], poslední aktualizace 18. 3. 1996. Dostupné z WWW: <http://www.t13.org/Documents/UploadedDocuments/project/d0948r4c-ATA-2.pdf>

[3] *AT Attachment Interface with Extensions (ATA-3)*. Revize 7b [online], poslední aktualizace 27. 1. 1997. Dostupné z WWW: <http://www.t13.org/Documents/UploadedDocuments/project/d2008r7b-ATA-3.pdf>

[3] Microsoft Corporation. *Microsoft Extensible Firmware Initiative FAT32 File System Specification* Verze 1.03 [online]. c 2000, poslední aktualizace 6. 12. 2000. Dostupné z WWW: <http://www.microsoft.com/whdc/system/platform/firmware/fatgen.mspx>

[4] Dobiash, Jack. *FAT32 Structure Information* [online], poslední aktualizace 4. 12. 2005. Dostupné z WWW: <http://home.teleport.com/~brainy/fat32.htm>

[5] Platforma FITKit [online], Dostupné z WWW: <http://merlin.fit.vutbr.cz/FITkit/>

[6] Sigmund, Stanislav. Rozhraní IDE pro platformu FITKit, BP 2007 FIT VUT Brno.

[7] Xilinx. *Spartan 3E FPGA Family.* Verze 3.7 [online]. , poslední aktualizace 4. 18. 2008. Dostupné z WWW: <http://www.xilinx.com/support/documentation/data_sheets/ds312.pdf>

[8] Xilinx. *PicoBlaze 8-bit Embedded Microcontroller.* Verze 1.1.2 [online]. , poslední aktualizace 6. 24. 2008. Dostupné z WWW: <http://www.xilinx.com/support/documentation/ip_documentation/ug129.pdf>

Seznam příloh