VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

JEDNODUCHÁ HRA PRO PLATFORMU FITKIT

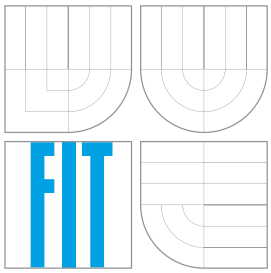
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JIŘÍ ŠUBR

BRNO 2009

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

JEDNODUCHÁ HRA PRO PLATFORMU FITKIT

SIMPLE GAME FOR FITKIT PLATFORM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JIŘÍ ŠUBR
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÁCLAV ŠIMEK
SUPERVISOR

BRNO 2009

Abstrakt

Tato bakalářská práce se zabývá implementací jednoduché hry. Tato hra je implementována na platformě FITkit. Je zde popsána implementace softwaru pro mikrokontrolér a implementace hardwaru v programovatelném hradlovém poli.

Abstract

This bachelor thesis deals with the implementation of simple game. This game is implemented on the FITkit platform. It described the implementation of software for the microcontroller and implementation hardware in the programmable gate array.

Klíčová slova

FITkit, FPGA, MCU, vestavěný systém, Sokoban, hra

Keywords

FITkit, FPGA, MCU, embedded system, Sokoban, game

Citace

Jiří Šubr: Jednoduchá hra pro platformu FitKit, bakalářská práce, Brno, FIT VUT v Brně, 2009

Jednoduchá hra pro platformu FitKit

Prohlášení

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

.....
Jiří Šubr
27. května 2009

Poděkování

Děkuji vedoucímu Ing. Václavu Šimkovi za odbornou pomoc při realizaci této práce.

Obsah

Kapitola 1

Úvod

Tématem práce je návrh a implementace jednoduché hry pro platformu FITkit. Tato platforma byla vyvinuta na Fakultě informačních technologií Vysokého učení v Brně. Poskytuje studentům snadno dostupný prostředek pro vývoj praktických školních, ale i osobních projektů. Realizované projekty mohou být softwarové, hardwarové nebo lze navrhnout software, a k němu podpůrný hardware. Aplikace, které jsou na této platformě vyvíjeny, se nazývají vestavěné systémy(embedded systems).

Vestavěné systémy mají již dnes významné zastoupení v každodenním životě a jejich význam ještě více v budoucnu poroste. Můžeme se s nimi setkat v mnoha zařízeních (mobilní telefony, televizní přijímače, automobily, atd.). Jedná se o zařízení, která vykonávají po celou dobu svojí životnosti jednu a tu samou úlohu. Při jejich vývoji je kladen důraz na bezchybný návrh a vypořádání se s kritickými situacemi.

Obsahem práce je návrh a implementace jednoduché hry, která by demonstrovala možnosti platformy a rozšířila počet aplikací, které jsou pro ni dostupné na jejich oficiálních stránkách. Tím, že návrhem je hra, je do jisté míry také snaha o získání dalších zájemců o tuto platformu. Už během prvních konzultací s vedoucím práce vznikla myšlenka, navrhnout a k ovládání hry využít modul herního ovladače. Tento ovladač by poskytoval lepší uživatelský komfort při ovládání her a k FITkitu by se připojoval přes jedno z jeho rozhraní.

V druhé kapitole je zpřesnění obecného zadání práce. Třetí kapitola obsahuje seznámení se s platformou FITkit, některými jejími periferiemi a komunikačními rozhraními, které lze využít pro připojení herního ovladače. Dále jsou zde informace o vybrané hře Sokoban, která byla vybrána k implementaci. Čtvrtá kapitola obsahuje návrh řešení aplikace jako je rozložení částí aplikace na hardwarovou a softwarovou část, návrh těchto částí a návrh desky herního ovladače, kde je kladen důraz na rozložení jednotlivých ovládacích prvků. Pátá kapitola popisuje vlastní implementaci aplikace. Závěrečná šestá kapitola je věnována závěru, zhodnocení a případným dalším rozšířením projektu.

Kapitola 2

Specifikace zadání

Zadání této práce bylo uvedeno v obecné formě, proto se v této kapitole nachází detailní specifikace zadání. Cílem této práce je vytvořit aplikaci v podobě jednoduché hry pro platformu FITkit [7]. V zadání je už přímo uvedena cílová platforma, pro kterou bude hra vyvíjena, takže se nemusím zabývat výběrem vhodné platformy. Protože se tato platforma nachází ve dvou částečně odlišných verzích, bude snahou vyvinout hru vhodnou pro obě verze. Aplikace by měla být spustitelná i bez připojení osobního počítače k FITkitu.

Vzhled aplikace by neměl obsahovat jen okno se samotnou hrou, ale měl by obsahovat úvodní menu s položkami. Po jednotlivých položkách se bude moci uživatel pohybovat pomocí herního ovladače nebo klávesnice, která je součástí FITkitu. Položka, na které se uživatel nachází, bude od ostatních dostatečně vizuálně odlišena. Obsahem aplikace bude i nápověda pro ovládání a hraní hry.

Na základě dohody s vedoucím práce byla vybrána logická hra Sokoban, její popis je uveden v sekci 3.3. Hra bude vycházet z původní hry a nebude obsahovat žádné modifikace. Hráč bude mít na výběr několik úrovní (levelů) hry. V průběhu hraní bude možno hru přerušit. Pokud hráč hru úspěšně dokončí, bude mu zobrazen jeho výsledek (skóre). Výsledek bude tvořen časem hry.

Nad rámec zadání bude snaha o návrh a oživení herního ovladače, který nebude obsahovat žádné mechanické snímače. Ovladač bude tvořit jedna deska plošných spojů (DPS), na které budou umístěna kapacitní tlačítka a snímač pohybu.

Kapitola 3

Teoretický úvod

Tato kapitola se věnuje podrobnějšímu popisu platformy FITkit a součástkám použitých při realizaci herního ovladače. Kapitola je rozdělena do tří částí, první je věnována platformě FITkit, druhá součástkám použitým při návrhu herního ovladače. Poslední část je věnována hře, která má být implementována v praktické části.

3.1 Platforma FITkit

Platforma FITkit je studentům k dispozici od roku 2006 (FITkit 1.x). Od této doby je pomocí školních projektů, týkajících se především hardwarové oblasti ale i vestavěných systémů, součástí výuky v bakalářském studiu na FIT. FITkit obsahuje výkonný mikrokontrolér (Microcontroller - MCU) a programovatelné hradlové pole (Field-programmable gate array - FPGA). Pro komunikaci mezi komponentami v FPGA a MCU je využita sběrnice SPI (Serial Peripheral Interface Bus). Kromě těchto dvou hlavních komponent disponuje mnoha periferiemi jako jsou řádkový displej, maticová klávesnice, VGA (Video Graphics Array) rozhraní, SDRAM (Synchronous dynamic random access memory) paměť, Flash paměť. informace o dalších periferiích lze nalézt na [19]. Od roku 2008 je studentům k dispozici nová verze FITkitu (FITkit 2.0), která obsahuje řadu vylepšení oproti předcházející verzi.

Obsahem této části je seznámení se s komponentami, které mohou být použity při návrhu a realizaci práce. Je zde podrobněji popsána platforma FITkit a její základní komponenty, kterými jsou MCU a FPGA. Obsahuje popis také dalších komponent, které mohou být použity při realizaci praktické části práce.

3.1.1 Mikrokontrolér

Monolitický integrovaný obvod obsahující mikroprocesor (Central Processing Unit - CPU), paměť ROM, paměť RAM a příslušné podpůrné obvody. Mezi podpůrné obvody patří například: I/O komponenty, sériové rozhraní, řadiče přerušení, čítače/časovače, A/D a D/A převodníky. Mikrokontroléry jsou vhodné pro řízení vnějších zařízení, v aplikacích požadujících minimum komponent a ve všech aplikacích, které jsou řízeny pomocí neměnných programů. Vyznačují se velkou spolehlivostí a kompaktností. Často jsou MCU součástí vestavěných systémů. MCU můžeme dělit podle toho zda používají odlišnou paměť pro data a program nebo používají pro data i program stejnou paměť na:

- **Von Neumanova** architektura má společnou paměť pro data i program. Jedná se o architekturu typickou pro univerzální počítače mezi které patří osobní počítače,

PDA, mobilní telefony a další. Architektura pro přístup do paměti využívá jeden paměťový prostor. Programátor nemusí rozlišovat do jakého paměťového prostoru přistupuje. Nevýhodou této architektury je, že se musí postupně data i instrukce číst z jedné paměti místo načítání dat a instrukce zároveň z rozdílných pamětí. Dnes se tento problém částečně řeší využitím rychlých vyrovnávacích pamětí(cache), které načítají postupně další části vnější paměti.

- **Harvardská architektura** je architektura, ve které jsou oddělena data od programu. Má oddělený adresový prostor pro data a pro program. K datům a instrukcím lze přistupovat paralelně. MCU obsahuje dvě sběrnice. Jedna sběrnice je určená pro zpracovávaná data, druhá sběrnice je určena pro načítání instrukcí. Každá sběrnice může mít jinou šířku. Pro paměť programu se používá nevolatilní paměť typu Flash, ROM, PROM, EPROM, EEPROM a další. Tato paměť je měněna jen při programování. Pro data se používá volatilní paměť typu RAM. Našla uplatnění v jednočipových počítačích a jednoúčelových obvodech. Tato architektura je vhodná pro takzvané vestavěné systémy, ve kterých se program nemění a mění se jen zpracovávaná data.

Dále můžeme MCU dělit podle toho jakou instrukční sadu používá:

- **Kopmletní sada instrukcí** (Complex Instruction Set Computer - CISC) se vyznačuje velkým množstvím istrukcí (více než 100). CPU mají malý počet registrů a tak instrukce pracují přímo s operandy uloženými v paměti což má negativní vliv na rychlost zpracování instrukcí. Instrukce nemají stejnou délku a trvají různý počet cyklů. Velké množství adresovacích módů umožňuje efektivně využít paměť.

- **Redukovaná sada instrukcí** (Reduced Instruction Set Computer - RISC) obsahuje 80 - 150 instrukcí, kde jednotlivé instrukce se provedou v jednom cyklu. Pro práci s pamětí jsou k dispozici pouze instrukce LOAD a STORE. Operandy prováděných operací jsou uloženy v registrech, což zvyšuje rychlost zpracování operací. Instrukce mají stejnou délku, to umožňuje jejich jednodušší zpracování na hardwarové úrovni. Kompilátory optimalizují kód pro konkrétní CPU, vyvíjí se tedy společně s CPU a tvoří spolu propojený celek. CPU obsahují rychlé vyrovnávací paměti pro uložení programů a operandů. Mají větší výpočetní výkon než CPU s CISC a jsou tedy vhodné pro složitější aplikace.

Na FITkitu 1.x se nachází 16-bitový MCU MSP430F168IPM firmy Texas Instruments s redukovanou instrukční sadou (RISC). Programátor má k dispozici stejný adresový prostor pro data i pro program, takže se jedná o Von-Neumanovu architekturu (CPU). FITkit 2.0 obsahuje výkonnější MCU se stejnou instrukční sadou i stejnou architekturou jako FITkit 1.x. Jedná se o mikrokontrolér s těmito parametry:

- frekvence 8 MHz
- nízké napájecí napětí (1,8 V - 3,6 V)
- nízký příkon (330 uA v aktivním režimu při 1 MHz a 2,2 V, 1,1 uA ve stand-by režimu, 0,2 uA v režimu vypnuto)
- paměť programu typu Flash 48 kB
- paměť pro data typu RAM 2 kB

- obsahuje moduly:
  - 3-kanálové DMA (Direct Memory Access) umožňující přímý přístup do paměti bez účasti CPU
  - 12-bitové A/D a D/A převodníky
  - 2x 16-bitové časovače
  - 2x sériová rozhraní (Universal Serial Communication Interface - USART/USCI), podrobnější popis uveden v sekci 3.1.2

Další informace ke zde uvedenému MCU naleznete na [3].

FITkit 2.0 obsahuje MCU MSP430F2616, které se od MCU použitém na FITkitu 1.x liší v těchto parametrech:

- frekvence 16 MHz
- paměť programu typu Flash 92 kB
- paměť pro data typu RAM 4 kB
- obsahuje moduly:
  - 4x sériová rozhraní (Universal Serial Communication Interface - USCI), podrobnější popis uveden uveden v sekci 3.1.2

další informace o tomto MCU lze nalézt na [2]. MCU je připojeno k SPI sběrnici FITkitu a je konfigurováno jako řídící prvek sběrnice (master). V další části textu je myšleno pod pojmem MCU mikrokontrolér, který se nachází na FITkitu.

3.1.2 Sériové rozhraní MCU

MCU, které se nachází na FITkitu 1.x, obsahuje dvě seriová rozhraní USART0, která lze využít v módech UART, SPI, a I2C. Rozhraní USART1 obsahuje módy SPI a UART. Na FITkitu 2.0 se nachází čtyři seriová rozhraní USCI_A0 a USCI_A1 s módy UART, IrDA a SPI. Rozhraní USCI_B0 a USCI_B1 obsahují módy SPI a I2C. Rozhraní funguje vždy jen ve zvoleném módu, nelze využívat více módů na jednom rozhraní.

3.1.3 Programovatelné logické obvody

Programovatelné logické obvody (Programmable logic device - PLD) jsou elektronické součástky používané k vytváření logických obvodů. Na rozdíl od logických hradel, registrů a jiných integrovaných obvodů není dána jejich funkce už z výroby. Mezi hlavní výhody patří, že si uživatel může naprogramovat svůj vlastní logický obvod sám. Dnešní PLD umožňují realizovat i složité obvody. Mezi PLD obvody lze zahrnout i obvody FPGA 3.1, které mají z této skupiny obvodů nejobecnější strukturu a obsahují nejvíce logiky.

Konfigurace FPGA obvodu je umístěna ve volatilní paměti typu SRAM, proto je nutné je po každém zapnutí znovu nahrát konfiguraci do jejich obvodu. U zařízení, která jsou vyráběna ve velkých sériích a nepředpokládá se jejich dodatečná změna, se hradlová pole nahrazují zákaznickými obvody ASIC, což jsou hradlová pole naprogramována již při výrobě a nelze je dodatečně změnit. Pro programování hardwaru je na FITkitu umístěn programovatelný logický obvod FPGA.

Programovatelné hradlové pole FPGA XC3S50-4PQ208C řady Spartan 3, které se nachází na FITkitu je od firmy Xilinx. Tento typ FPGA je určen pro nejširší použití. Je použito i na FITkitu 2. FPGA obsahuje:

- 124 uživatelských vstupně/výstupních obvodů pro každý pin (Input/Output blocks - IOB), podporují až 23 různých standardů,
- 192 programovatelných logických bloků (Configurable Logic Block - CLB) uspořádaných v matici o 16 řádcích a 12 sloupcích,
- 1728 logických buněk,
- 50000 logických hradel,
- 4x násobičky 18x18 bitů,
- 4x dvoukanálové blokové paměti (Block RAM - BRAM), každá o velikosti 2 kB
- 12kbitů distribuované paměti RAM,
- 2x jednotky pro správu hodin (Digital Clock Manager - DCM) a bloky DLL (Delay Locked Loop), které slouží k rekonstrukci a případnému násobení nebo dělení vnějších taktovacích signálů

Další informace lze nalézt na [8]. Konfigurace FPGA je uložena ve statické paměti RAM. To znamená, že po připojení napětí je nutné nahrát znovu konfiguraci do obvodu. Toto je řešeno na FITkitu použitím paměti Flash, kde je uložena konfigurace FPGA. Při každém spuštění je přes MCU nahrána konfigurace do FPGA.

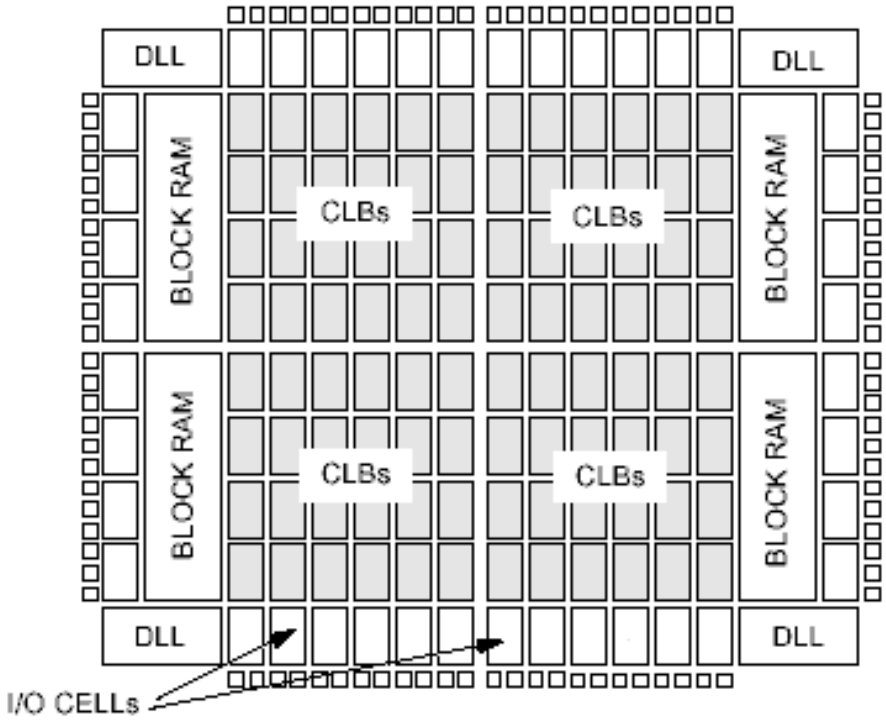
Od této chvíle se pod pojmem FPGA rozumí FPGA umístěné na FITkitu.

3.1.4 Periferie

FITkit obsahuje mnoho periferií, které lze využít při mnoha projektech. Většina z nich je připojena přímo k FPGA a některé jsou připojeny k SPI sběrnici. Zde jsou popsány periferie, které mohou být využity v této práci.

Na obou FITkitech se nachází :

- **Displej**, který se nachází na FITkitu 1.x, je displej šestnácti znakový jednořádkový displej CM1610NR-J2. Fitkit 2.0 je osazen dvouřádkovým displejem CM160224 s možností zobrazení až 32 znaků.
- **Klávesnice** s šestnácti tlačítky AK-1604-A-WWB zapojených do matice 4x4.
- **VGA port** je jednosměrný port sloužící k připojení zobrazovacích periferií. Rozhraní používá pro zobrazení dat tři základní barevné složky (červená, modrá, zelená) a dva typy synchronizačních pulsů (vertikální a horizontální).
- **SDRAM paměť**, která slouží pro dočasné ukládání dat má označení GM72V66841ET7K. Paměť je složena z kondenzátorů, proto se musí data po určitém čase obnovovat. Po vypnutí napájení se data uložená v paměti ztrácí. Jedná se o paměť se synchronním přístupem. Paměť je veliká 8 MB rozdělená do 4 banků, nejvyšší její frekvence je 100 MHz.



Obrázek 3.1: Obecná struktura FPGA.
[18]

- **Flash paměť** slouží pro uchování dat i při odpojení napětí. Jedná se o paměť AT45DB021B od firmy ATMEL. Velikost paměti je 2Mb, má 1024 stránek, v každé stránce 268 bytů. Mazání této paměti je po blocích. Paměť je připojena k SPI sběrnici FITkitu. Slouží k uložení konfigurace FPGA. Zbytek paměti lze libovolně využít.

Mimo paměti Flash jsou tyto komponenty propojeny s FPGA a pro práci s nimi je nutné použití předpřipravených řadičů, které jsou již pro FPGA napsány.

3.1.5 Komunikační rozhraní MCU

Pro připojení různých periferií k MCU jsou na vstupně výstupních portech k dispozici různé sběrnice. Pokud přenášíme do MCU například pouze informaci o tom, zda bylo stisknuto tlačítko, nepotřebujeme k tomu využívat žádnou specializovanou sběrnici, ale stačí nám připojit tlačítko na V/V port MCU, který je schopen detekovat na svém vstupu úroveň signálu a případně vyvolá při změně signálu přerušení. Jestliže potřebujeme přenášet komplexnější informace tvořené sekvencí několika bajtů, je vhodné připojit tento druh signálu na porty MCU, které dokáží tuto posloupnost bajtů autonomně zpracovat. Takový druh spojení se využívá při propojení dvou mikrokontrolérů, nebo třeba při propojení MCU s nějakou inteligentní periferií (teplotní čidlo, snímač pohybu, displej atd.). Je vhodné tyto informace přenášet po co nejmenším počtu vodičů, protože počet vývodů MCU je omezený.

Podle počtu vodičů, které se při přenosu využívají rozdělujeme přenos na:

- **Sériový přenos dat** – Při sériovém přenosu dat se data přenášejí v podobě jednotlivých bitů po jednom vodiči. Výhodou tohoto přenosu je menší náchylnost proti rušení. Tím, že jsou data přenášena po jednom vodiči, není zde problém tzv. clock skew.

- **Paralelní přenos dat** – Paralelní přenos dat je přenos dat, kde se data přenáší po více vodičích současně. Je určen pro přenos dat na krátké vzdálenosti. Protože se jedná o přenos dat po více vodičích, je zde větší problém s rušením než u sériového přenosu dat. Vzniká zde také problém se synchronizací signálu tzv. clock skew. V dnešní době za využití moderních technologií je na ústupu.

Dále se přenos dat dělí podle synchronizace na:

- **Synchronní přenos dat** – Sběrnice využívající synchronní přenos dat jsou sběrnice, které spolu s daty přenáší synchronizační signál. Od jeho hran jsou pak snímána data na datových vodičích.

- **Asynchronní přenos dat** – Po této sběrnici není přenášen synchronizační signál. Synchronizační pulsy si přijímač dat generuje sám. Zde jsou kladeny nároky na dostatečně přesný generátor hodinového signálu na straně přijímače.

Sběrnice I2C [11] je synchronní sériová sběrnice vyvinutá společností Philips. Sběrnice poskytuje jednoduchý a levný prostředek pro propojování integrovaných obvodů na krátkou vzdálenost. Původně měla sběrnice sloužit k propojení obvodů v rámci většího systému (např. nastavování periferí pomocí MCU). Postupem času se sběrnice rozšířila a dnes je jedním z nejpoužívanějších prostředků pro komunikaci mezi zařízeními v rámci vestavěných systémů. Je vhodná pro zařízení, která mezi sebou nepotřebují posílat velké objemy dat.

Na sběrnici jsou připojena zařízení, která řídí sběrnici (master) a zařízení, která jsou sběrnicí řízena (slave). Protože na sběrnici může být připojeno více obvodů, jsou adresovány 7-bitově (až 128 zařízení) nebo v rozšířené verzi 10-bitově(až 1024 zařízeni).

Po sběrnici se přenášejí 8-bitová data. Sběrnice používá pro přenos dva obousměrné vodiče z toho jeden datový (Serial Data - SDA) a druhý hodinový (Serial Clock - SCL). Díky malému počtu signálních vodičů je vhodné její použití v MCU. Zařízení umístěná na sběrnici mohou být buď **master** nebo **slave**. Na oba vodiče musí být připojeny pull-up odpory v řádech kiloohmů. Přenos dat na sběrnici se realizuje v poloviční duplexním režimu. To znamená, že zařízení na sběrnici mohou buď data přijímat nebo vysílat. Sběrnice může pracovat ve třech režimech rychlosti:

- Standard mode – do 100kb/s
- Fast mode – 400kb/s
- Fast mode plus – do 1000kb/s
- High speed – do 3,4 Mb/s

Rychlost přenosu je přizpůsobena nejpomalejšímu zařízení na sběrnici. Pro rychlost 100kb/s je doporučeno použít pull-up odpory o hodnotě 4,7 kiloohmů.

SPI [13] sběrnice patří mezi synchronní sériové sběrnice. Stejně jako I2C sběrnice slouží pro propojení obvodů uvnitř vestavěného systému. Zařízení spolu sdílí komunikační sběrnici. Na sběrnici se nachází zařízení typu slave, kterých může být více a zařízení typu master, které řídí sběrnici a na sběrnici se nachází pouze jedno. Zařízení Master generuje hodinový

signál sběrnice a rozhoduje o tom, jaké zařízení typu slave s ním bude komunikovat. Master zařízení je zpravidla MCU a slave zařízení jsou jeho periferie.

Sběrnice má dva datové vodiče MISO (Master input, Slave output) a MOSI (Master output, Slave input). Názvy signálů určují tok dat podle toho v jakém režimu se zařízení nachází. Hodinový signál obsahuje vodič SPSCK (SPI Serial Clock). Tato sběrnice je plně duplexní tzn. zařízení může souběžně s vysíláním dat na sběrnici i data přijímat. Maximální přenosová rychlost této sběrnice je 70 MHz, u sběrnice I2C jen 3,4 MHz. Nevýhodou může být vyšší počet vodičů než u I2C.

3.2 Herní ovladač

Herní ovladač slouží k pohodlnému ovládání her. Je součástí herních konzolí [14]. Herní konzole jsou vestavěné systémy s grafickým výstupem určené pro hraní her. V dnešní době herní ovladače úspěšně nahradily další ovládací zařízení kterým je klávesnice. Je spousta druhů herních ovladačů, některé se používají jen pro určitý typ her, jiné jsou univerzální a je možné je použít pro hraní většiny her. Herní ovladač, který bude navrhován, neobsahuje žádné mechanické snímače. Ovladač bude obsahovat akcelerometr a kapacitní tlačítka.

3.2.1 Kapacitní senzory

Kapacitní senzory (Touch Pads) jsou senzory, které jsou využívány obdobně jako klasická mechanická tlačítka. Na rozdíl od mechanických tlačítek nemají kapacitní senzory žádné mechanické prvky tzn., že u nich nedochází k mechanickému opotřebení a k jejich sepnutí nemusíme vyvíjet žádnou sílu. Pracují na principu detekce změny kapacity buď mezi senzory nebo mezi senzorem a např. prstem.

Senzory MPR08X

Kapacitní senzory s rozhraním I2C firmy Freescale, přes které je lze konfigurovat. Obsahují jeden konfigurovatelný výstup pro využití přerušení. Můžeme s nimi realizovat tlačítka různých tvarů. K detekci dochází při pouhém přiblížení. Detekuje se nejenom prst člověka, ale i tužka, kov, guma, atd.. můžeme použít až 8 snímacích ploch.

Senzor MPR083 vyhodnocuje nejen stisk, ale i v reálném čase detekuje souvislý pohyb po snímacím kruhu nebo tzv. slideru. Nemusí se v tomto případě znovu přibližovat a oddalovat prst, dochází k vyhodnocení přechodů mezi ploškami. Více informací o senzoru MPR083 je uvedeno na [4]. Senzor MPR084 umožňuje detekci stisku jednotlivých tlačítek a je možné použít jako klasickou klávesnici. Více informací o senzoru MPR084 je uvedeno na [5].

3.2.2 Akcelerometr

Jedná se o snímač pohybu. Pracuje na principu využití gravitačního zrychlení. Existují akcelerometry, které snímají jednu a nebo více os. Pro snímání pohybu existují technologie jako gyroskop [10] nebo MEMS (Micro-Electro-Mechanical Systems) [12].

U MEMS technologie zjednodušeně řečeno je snímání realizováno pomocí měření kapacity kondenzátorů, kde jedna elektroda se pohybuje. Pohyb elektrody je dán působením zrychlení na snímač. Je zde využito základního vztahu působení síly při zrychlení tělesa $F = m * a$.

Akcelerometry se využívajší v mnoha oborech jako jsou např. strojírenství, zdravotnictví, stavebnictví, doprava. Akcelerometry pro svůj výstup využívají buď digitální nebo

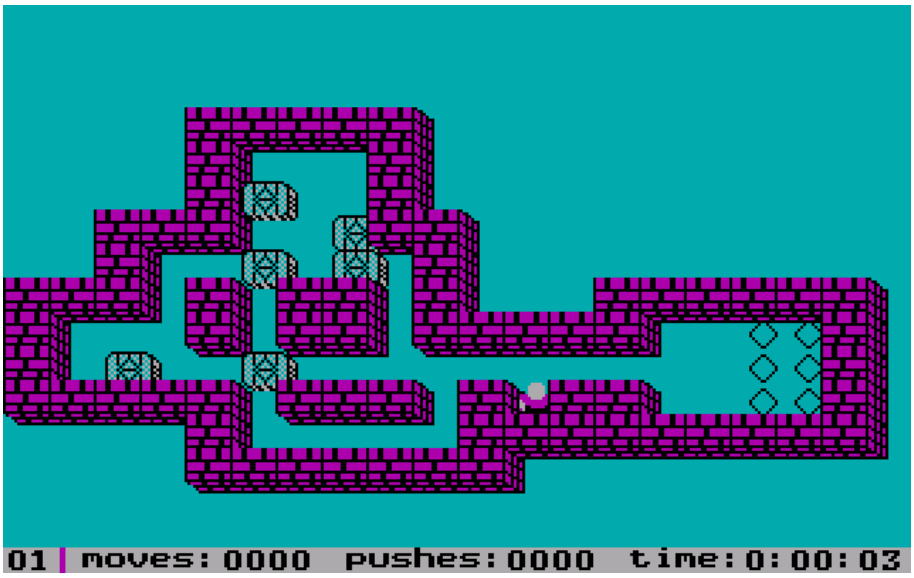
analogový výstup. Digitální výstup nevyžaduje žádné další zařízení/rozhraní v podobě A/D převodníku, který by zpracovával jeho výstup.

Akcelerometr MMA7455L

Jedná se o miniaturní, akcelerometr s nízkou spotřebou využívající MEMS technologie. Tento akcelerometr, vyvinutý firmou Freescale, má digitální výstup využívající sběrnici I2C nebo SPI. Přes toto rozhraní se i konfiguruje. Pohyb snímá ve třech osách. Má sebetestovací mechanismus. Detekuje různé typy pohybu (pád, vibrace, náraz). Umožňuje volbu měřícího rozsahu a citlivosti (2/4/8 g). Více informací o tomto akcelerometru lze nalézt na [6]

3.3 Sokoban

Hra Sokoban patří mezi logické video hry a je určena pro jednoho hráče. Patří mezi tzv. offline hry, které jsou určeny pro hraní na jedné platformě. Hra simuluje činnost z reálného života. Můžeme ji přirovnat k jednoduché simulaci dění ve skladě, kde skladník přesouvá bedny z jednoho místa na druhé. Původní hru 3.2 vytvořil v roce 1980 Hiroyuki Imabayashi. Vydána byla v roce 1982 pro Commodore 64 a IBM-PC. Úkolem hráče je přemístit bedny



Obrázek 3.2: Původní verze hry

na vyznačené místo. Ve hře platí jednoduchá pravidla:

- bedny je možné sunout jen před sebou
- posunovat lze jen jednu bednu
- pokud se bedna dostane do rohu hracího pole nelze s ní dále pohybovat

Ve hře se sleduje čas potřebný k přemístění všech beden na vyznačená místa. Čím kratší čas, tím lepší výsledek.

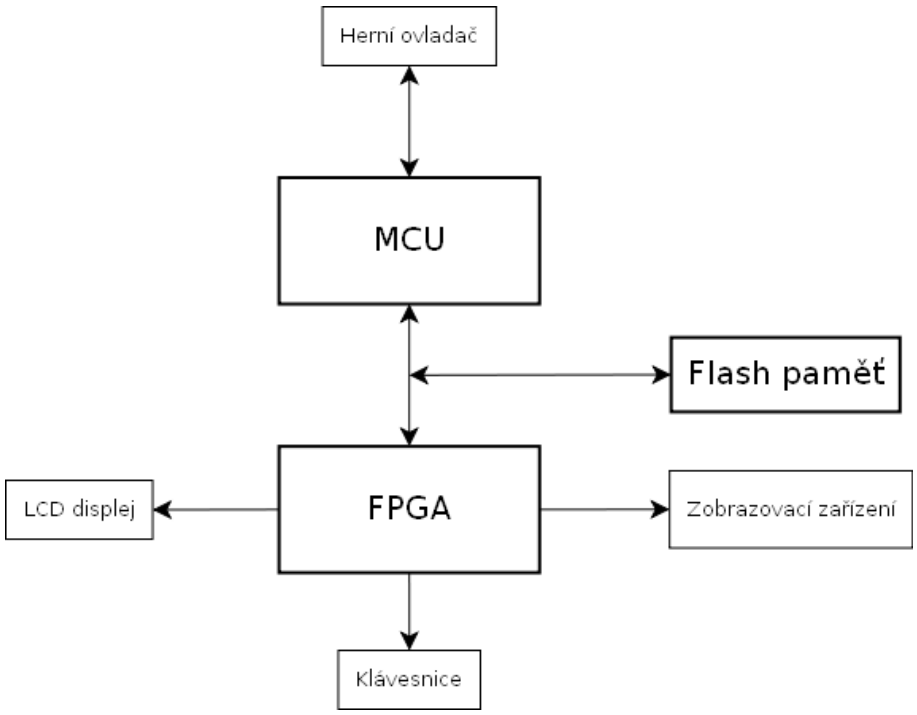
Hra je velmi populární, a tak vzniklo mnoho implementací hry na mnoho počítačových platforem včetně osobních počítačů(Personal Computer - PC). Existují verze pro herní konzole, mobilní telefony, MP3 přehrávače, digitální fotoaparáty a podobně.

Dnes existuje spousta variant této hry. Grafika původní hry byla dvourozměrná (2D) a měla čtvercové obklady. Dnes je spousta verzí třírozměrné hry (3D) a obklady mohou být ve formě trojúhelníku, šestiúhelníku, kruhu a jiných. Další varianta hry obsahuje více pohyblivých postav, takže hráč neovládá jednu (jako v původní hře), ale i více postav. Přesunované bedny mohou mít různé barvy a tím určeno na jakou cílovou pozici, která je také barevně odlišena, se mají přesunout. Existují také rozšíření v podobě teleportů, děr, pohybujících se stěn nebo jednosměrných cest. Mezi rozšíření, které popírá základní pravidla hry, je například možnost nejen sunutí bedny před sebou ale i její táhnutí. Množina zde vyjmenovaných rozšíření není konečná, tvoří jen základní část modifikací této hry. Existují také hry, které ze hry Sokoban vychází, takže přebírají principy z původní hry.

Kapitola 4

Návrh řešení

Po seznámení se s prvky, které je možno použít v této práci, následuje návrh řešení. Návrh vychází z dostupných prostředků na FITkitu a z již vytvořených aplikací. Tato kapitola obsahuje návrh rozložení aplikace mezi hardwarovou (FPGA) část a softwarovou část (MCU). Nachází se zde rozdělení úloh mezi úlohy vykonávané v FPGA a úlohy vykonávané v MCU. Je zde uveden detailní popis využití jednotlivých periferií. Na obrázku 4.1 je zobrazeno blokové schéma celé práce.



Obrázek 4.1: Celkové blokové schéma projektu.

4.1 Rozdělení aplikace

Obecně lze aplikaci implementovat v FPGA nebo v MCU. Dále je možné část aplikace implementovat v MCU a část v FPGA. Takovouto implementaci nám umožňuje propojení

obou částí sběrnicí SPI. Při použití obou částí je třeba brát ohled na omezenou rychlost sběrnice SPI.

Tato aplikace nemůže být implementována pouze v MCU, jelikož MCU neobsahuje potřebné periferie. Veškeré periferie využívané v této aplikaci jsou připojeny k FPGA. Periferie v podobě herního ovladače jako jediná nemá dostupný řadič pro FPGA. To znamená, že pro komunikaci s touto periferí by musel být navržen nový řadič. Pokud by se řadič navrhl, šla by celá aplikace navrhnout jen pro FPGA. FPGA umístěné na FITkitu, ale není velké a při implementaci by mohl vzniknout problém s velikostí navrženého obvodu. Proto bylo zvoleno řešení využívající jak MCU tak FPGA.

Po prozkoumání již realizovaných projektů pro FITkit bylo vybráno rozložení aplikace vycházející z projektů *Piškvorky* [9] a *Textový režim* [15]. FPGA je u těchto projektů využito pro poskytování dat VGA rozhraní. MCU má na starost aktualizaci grafických dat uložených v blokové paměti FPGA. Další činností MCU je zpracování uživatelských vstupů a řízení celé aplikace.

4.2 Návrh FPGA

FPGA se bude starat o poskytování dat VGA rozhraní. Budou v něm uložena zobrazovaná data a zpřístupní pro MCU periferie klávesnice a LCD displeje.

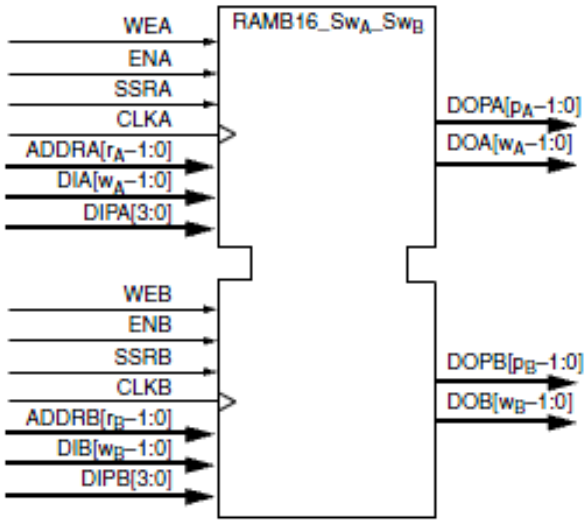
Aplikace bude používat již navržený řadič VGA rozhraní. Základní informace o VGA řadiči jsou uvedeny na [16]. Bude použito rozlišení obrazovky 640x480 bodů s obnovovací frekvencí 60 Hz. Toto rozlišení je pro tuto aplikaci dostatečné. Zvolení většího rozlišení by znamenalo větší nároky na paměť pro ukládání grafických dat.

4.2.1 Výběr paměti

Pro data bude nutné použít paměť dostatečné rychlosti a velikosti. Rychlost přístupu k paměti je dána požadovaným rozlišením a obnovovací frekvencí VGA rozhraní. Na FITkitu je k tomuto účelu možné využít blokovou paměť umístěnou v FPGA. Tuto paměť lze jednoduše využívat v aplikacích. Nevýhodou této paměti je její velikost 8kB. Další možností je využití SDRAM paměť, která je připojena přímo k FPGA. Tato paměť má kapacitu 8MB, ale její nevýhodou je nutnost použití řadiče. Pokud si aplikace vystačí s blokovou pamětí, je výhodné tuto paměť použít. Práce s ní je jednodušší než práce s SDRAM.

4.2.2 Bloková paměť

Jako video paměť a paměť pro uložení textur a fontu bude použita dvouportová bloková paměť. Jeden port bude připojen k vnitřní sběrnici FPGA a bude tak dostupný pro MCU, druhý bude poskytovat data VGA rozhraní. Bude použita dvouportová paměť RAMB16_S9_S9 obrázek 4.2. Všechny zde použité blokové paměti budou mít datovou šířku obou portů 9 bitů (8 bitů datových, 1 bit paritní). Paritní bity mohou být využity k uchování dat. Pro adresování 9-bitových slov se používá 11 adresních bitů. V aplikaci budou zobrazovány jak textové obrazovky, tak obrazovky s texturami. Z hlediska rychlosti přepínání mezi zobrazováním textu a zobrazováním textur je vhodné umístit do blokových pamětí jak textury, tak i font.



Obrázek 4.2: Dvouportová bloková paměť.

4.2.3 Obsah blokových pamětí

Font bude využit z projektu [15]. Jednotlivé znaky fontu mají velikost 8 bodů na šířku a 16 bodů na výšku. Na obrazovku se tedy vejde 30 řádků o 80-ti znacích. Font obsahuje 128 znaků. Z těchto údajů lze vypočítat paměťovou náročnost pro uložení fontu: $8 \cdot 16 \cdot 128 = 16384\text{b} = 2\text{kB}$. Takže celý font zabírá akorát jednu blokovou paměť.

Obrazovka zobrazující textury vyžaduje použití pěti textur. Jedna textura bude vytvořena invertováním textury hracího pole. Invertace se provede nastavením příslušného bitu indexu textury ve video paměti. V blokové paměti budou proto uloženy jen čtyři textury o velikosti 32x32 bodů. Každý bod textury bude uložen na devíti bitech. Bloková paměť umožňuje ukládat data v 8-bitové formě. Navíc ještě poskytuje pro každých 8 bitů jeden paritní bit. V tomto případě bude bit použit právě pro uložení nejnižšího bitu bodu textury. Paměťová náročnost textur o rozměru 32x32 bodů: $9 \cdot 32 \cdot 32 \cdot 4 = 36864\,\text{b} = 4,5\,\text{kB}$. Maximální obsah blokové paměti s využitím paritního bitu: $9 \cdot 1024 = 18432\,\text{b} = 2,25\,\text{kB}$. Z toho vyplývá, že kdyby se do paměti ukládaly celé textury, bylo by možné uložit do jedné blokové paměti pouze 2 textury. Textury ale potřebujeme umístit do jedné blokové paměti z důvodu omezeného prostoru. Jelikož jsou textury souměrné, stačí nám uložit pouze půlku textury. Adresace druhé půlky textury se provede invertováním nejnižších tří bitů adresy textury.

Video paměť budou tvořit dvě blokové paměti, které budou použity pro uložení indexů znaků v textovém režimu a pro uložení indexu textury v grafickém režimu. Použití dvou blokových pamětí vychází z paměťové náročnosti indexů použitých v textovém režimu. Indexy mohou dosahovat hodnoty až 127, proto bude pro jejich uložení potřeba minimálně sedm bitů. Na obrazovku se celkem vejde $30 \cdot 80 = 2400\,\text{znak}$. Indexy jsou sice 7 bitové, ale použitá bloková paměť je 8-bitová. Do jedné blokové paměti se vejde maximálně 2048 znaků (11-bitová adresace). Takže proto je nutné použití dvou blokových pamětí. Nevyužitý bit může být vhodné využít při zvýraznění položek v menu hry. Pokud bude nastaven, dojde k invertaci jak barvy znaku tak jeho pozadí.

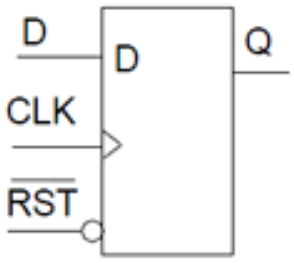
4.2.4 Vnitřní sběrnice FPGA

Pro přístup MCU k periferiím připojeným k FPGA byl navrhnut tzv. *Propojovací systém*. Detailní popis je uvedený na [17]. Hlavním úkolem tohoto systému je vytvoření sběrnice uvnitř FPGA, která by dovolovala komunikaci MCU se zařízením připojeným k této sběrnici. Základem je SPI řadič, který provádí konverzi SPI protokolu na vnitřní sériový protokol. K této sběrnici se prostřednictvím libovolného počtu SPI dekodérů mohou připojovat zařízení vytvořená v FPGA. Úkolem dekodéru je dekódovat adresu cílového zařízení a poskytnutí paralelního rozhraní dané periferii.

Pomocí tohoto dekodéru budou MCU zpřístupněny: klávesnice, LCD displej, řídící registr a blokové paměti.

4.2.5 Řídící registr

O tom, která data budou posílána na vstup VGA řadiče, bude rozhodovat tzv. řídící registr. Registr bude muset být pomocí vnitřní sběrnice FPGA zpřístupněný MCU. Tento registr bude nastavován MCU a bude rozhodovat o tom zda se bude zobrazovat textura nebo font. Řídící registr bude implementován jako klopný obvod typu D obrázek 4.3. Zápis do registru bude umožněn při náběžné hraně synchronizačního signálu vnitřní sběrnice. Čtení z registru bude umožněno asynchronně.



Obrázek 4.3: Klopný obvod typu D.

4.2.6 Procesy

Výběr dat z video paměti a z paměti fontu nebo textur bude řešen pomocí procesů, které budou vybírat požadovaná data. Dále budou zajišťovat adresaci v obou pamětech. Budou rozhodovat o tom, která data se mají vložit na vstup VGA řadiče. Procesy budou ovládány řídícím registrem.

4.3 Návrh MCU

V MCU bude aplikace, která bude řídit celou hru. Hra se skládá z více obrazovek, takže bude vhodné, když každou obrazovku bude zpracovávat samostatná funkce. Textové obrazovky budou využívat funkce určené pro práci s textem. Vlastní hra bude obsahovat funkce pro přesun hráče a kontrolu konce hry.

Aplikace je celkem komplikovaná, takže bude vhodné použít pro ukládání dat datové struktury. Tyto datové struktury v sobě budou zapouzdřovat úzce související proměnné.

4.3.1 Datové struktury

Pohyby hráče do všech směrů budou definovány v poli. Toto pole bude obsahovat strukturu, ve které budou uvedeny informace o směru pohybu hráče. Hodnota stisknuté směrové klávesy bude odpovídat hodnotě indexu v poli. V aplikaci je počítáno s pohybem do čtyř stran. Takže pole bude obsahovat čtyři struktury. Takto navržený systém pohybu umožňuje případné rozšíření pohybů po hrací ploše. Stačí přidat do pole další struktury, které budou odpovídat novým pohybům. Funkce pro pohyb hráče budou pracovat s touto strukturou. Výhodou také je, že změnu pohybových vlastností hráče bude možné provést na jednom místě. Obsah struktury:

- posun v ose x
- posun v ose y

Položky menu budou uloženy také v poli. Pokud se bude aplikace dále rozšiřovat o nové funkce, stačí přidat do pole nové položky. Takto definované položky umožňují snadnou editaci na jednom místě. Jednotlivé položky menu budou používat strukturu, ve které budou uvedeny:

- souřadnice na ose x
- souřadnice na ose y
- zobrazovaný text
- délka textu

Struktura pro uložení času sdružuje úzce související proměnné. Obsahuje položky:

- počet sekund
- počet minut
- počet hodin

4.3.2 Úkoly MCU

MCU bude mít tyto úkoly:

- obsluha ovladače
- nastavování řídícího registru
- aktualizaci dat ve video paměti
- detekci konce hry
- odměřování času hry
- detekci stisknuté klávesy
- nahrávání her z Flash paměti

4.3.3 Měření času hry

Odměřování času hry bude realizováno v podobě časovače v režimu výstupní komparace. Bude využito přerušení od časovače. V tomto přerušení dojde k aktualizaci struktury, ve které je uchováván aktuální čas. Aktuální čas se bude zobrazovat na LCD displeji. Aktualizace displeje bude realizována v obsluze přerušení.

4.3.4 Popis vlastní aplikace

Při spuštění FITkitu dojde k inicializaci blokových pamětí v FPGA. Nastaví se řídící registr FPGA na textový režim. Dojde k inicializaci klávesnice umístěné na FITkitu. Nastaví se proměnná, ve které se nachází aktuální obrazovka, na úvodní menu a dojde k zavolání funkce pro inicializaci úvodního menu.

Jádrem aplikace bude nekonečná smyčka. Smyčka bude obsahovat funkci pro čtení dat z klávesnice. Bude se tedy jednat o dotazovací režim. Dále se ve smyčce bude nacházet čekací procedura, která bude zajišťovat rychlost skenování klávesnice. V hlavní smyčce se také bude rozhodovat o tom, jaká funkce se použije pro zpracování příkazu od klávesnice. Funkce se vybere na základě hodnoty uložené v proměnné s informací o aktuální obrazovce.

Tyto funkce budou obsahovat obsluhu jednotlivých obrazovek. Budou jednoduché a tato koncepce umožní snadné rozšíření aplikace o více obrazovek.

4.3.5 Přechod mezi obrazovkami

V aplikaci dochází ke dvěma různým přechodům mezi obrazovkami. Jeden přechod je mezi textovou částí a částí s hrou, další přechod je mezi dvěma textovými obrazovkami. Oba přechody mají společný jeden krok. Tím krokem je inicializace video paměti. To znamená, že při každém přechodu mezi obrazovkami se musí inicializovat video paměť. Při přechodu mezi odlišnými zobrazovacími částmi je dalším krokem nastavení řídícího registru do příslušného módu.

4.4 Využití periferí v aplikaci

V této sekci je popsáno konkrétní využití jednotlivých periferií v aplikaci.

4.4.1 Zobrazovací zařízení

V zadání práce je uvedeno, že vyvíjenou aplikací je jednoduchá hra. Tato hra se musí uživateli zobrazovat na dostatečně kvalitním zobrazovacím zařízení. FITkit sám o sobě má zobrazovací zařízení v podobě jednořádkového LCD displeje, ale toto zařízení není vhodné pro grafické aplikace. LCD displej je vhodný pro výpis krátkých textů. Navrhovaná aplikace ho bude využívat jako vedlejší zobrazovací zařízení. Budou na něm zobrazovány informace jako například herní čas nebo aktuální položka v menu. Na FITkitu se dále vyskytuje grafické rozhraní VGA, které poskytuje dostatečně kvalitní grafický výstup pro tuto aplikaci. Aplikace ho tedy bude využívat jako hlavní zobrazovací zařízení. Obě zařízení jsou připojena k FPGA.

4.4.2 Ovládání hry

K ovládání aplikace je možné využít na FITkitu se nacházející klávesnici, která je připojena také k FPGA. Dalším ovládacím prvkem bude herní ovladač, který vzniká současně s touto

prací. Herní ovladač lze připojit k MCU, které má komunikační rozhraní potřebné pro komunikaci s herním ovladačem. Ovladač je možné připojit také na V/V porty FPGA.

4.4.3 Ukládání perzistentních dat aplikace

Jelikož má být aplikace spustitelná i bez použití osobního počítače je nutné, aby veškerá data potřebná pro běh aplikace byla uložen v paměti Flash. Tato paměť je energeticky nezávislá, takže po odpojení napájení v ní zůstanou uložená data. Paměť bude využita pro ukládání textur fontů a grafiky hry. Dále se v ní budou nacházet jednotlivé herní úrovně a nejlepší výsledky v těchto úrovních.

4.5 Návrh herního ovladače

Tato sekce se zabývá návrhem herního ovladače. Herní ovladač bude využívat senzory popsané v sekci 3.2. Návrh herního ovladače se skládá z realizace schématu zapojení jednotlivých součástek a z realizace návrhu desky plošných spojů (DPS). Při těchto činnostech byl použit návrhový systém Eagle [1]. Práce v tomto programu se skládá z návrhu schématu zapojení a následném návrhu desky plošných spojů.

4.5.1 Schéma zapojení

Pro navržení schématu je nutné mít k dispozici v programu všechny součástky. Součástky je možné najít v připravených knihovnách, které jsou součástí programu. Pokud nejsou součástky k dispozici, je nutné si potřebnou knihovnu se součástkami vytvořit. Všechny součástky pro herní ovladač nebyly v knihovnách nalezeny, a proto byla navržena knihovna chybějících součástek. Součástky jsou ve schématu znázorněny pomocí svých značek a jsou propojeny podle jejich doporučeného zapojení, které je uvedeno v dokumentech k jednotlivým součástkám. Použité senzory používají ke komunikaci s řídícím prvkem systému (v tomto případě MCU), sběrnici I2C a tak lze s výhodou využít jejich připojení ke stejné sběrnici. Tím se výrazně zjednoduší propojení ovladače s MCU.

Dále je potřeba zajistit napájení součástek na DPS. Při návrhu schématu byl do schématu přidán stabilizátor napětí, který zajišťuje stabilní napájení součástek 3,3 V při připojení vnějšího zdroje napětí v rozmezí 2,4 V až 24 V.

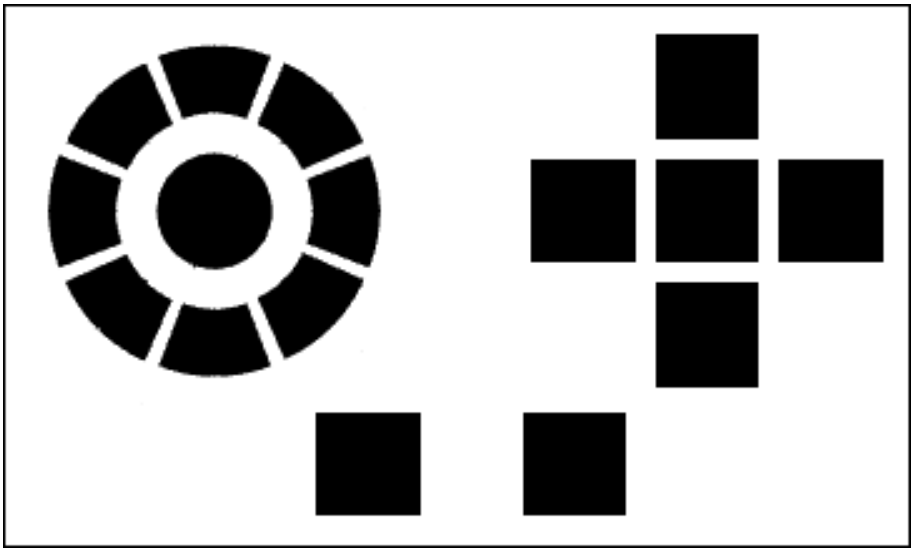
Jednotlivé senzory umožňují při změně jejich stavu vyvolat přerušení. Proto byly na konektor herního ovladače vyvedeny samostatně od každého senzoru signály vyvolávající přerušení. Informace z ovladače nemusíme díky tomu získávat pouhým dotazováním na jednotlivé senzory, ale můžeme využít i přerušení od senzorů. Díky vyvedení každého přerušení zvlášť přesně víme, na kterém se změnil stav a můžeme z něho přečíst data.

Pro nastavení chování senzorů konektor obsahuje i tyto vývody. Tohoto lze využít pokud budeme chtít během běhu programu změnit chování senzorů.

K zachování zpětné kompatibility s FITkitem 1.x byly na všechny vývody připojeny pull-up odpory. FITkit 1.x na svých V/V portech toto chování neumožňuje nastavit.

4.5.2 Deska plošných spojů

Další částí návrhu herního ovladače je návrh desky plošných spojů. Tento návrh vychází ze schématu zapojení. Na herním ovladači je uprostřed DPS umístěn akcelerometr MMA7455L. Toto umístění je nutné z hlediska minimálního zkreslení snímání pohybu ovladače. Rozložení tlačítek na DPS je znázorněno na obrázku 4.4.



Obrázek 4.4: Rozložení dotykových tlačítek na DPS.

Tlačítka, která jsou umístěna v kruhu, jsou připojena ke kapacitnímu senzoru MPR083, ten umožňuje snímání jak stisknutých tlačítek tak i směr, kterým se uživatel po kruhu pohybuje.

Další tlačítka jsou napojena k senzoru MPR084, který umožňuje detekci stisku jednotlivých tlačítek nebo skupiny tlačítek.

Deska je obdélníkového tvaru. Při případných modifikacích bude vhodnější tvar desky změnit. Desku je vhodné umístit do nějakého obalu, aby nedošlo k její poškození.

Kapitola 5

Implementace

V této kapitole se nachází implementace celé aplikace. Implementace vychází z předcházejícího návrhu aplikace. Je využito rozdělení aplikace na softwarovou a hardwarovou část. Implementace těchto dvou částí může probíhat zcela nezávisle na sobě. Jsou zde popsány konkrétní problémy řešené při implementaci.

Implementace hardwarové části byla provedena pomocí jazyka VHDL. Skládá se z použití připravených komponent a jejich vzájemného propojení. Pouhé propojení komponent by k implementaci nestačilo a tak jsou vhodně využity i logické obvody, které jsou vkládány do cest signálních vodičů. Při implementaci jsem nejdříve navrhl aplikaci v FPGA a její funkčnost ověřil na již vytvořených projektech.

Část pro MCU byla napsána v jazyce C s využitím knihovny libfitkit. Při programování bylo využito modulárního přístupu programování. Jednotlivé řešené problémy jsou rozděleny do modulů (funkcí).

5.1 Implementace FPGA

Implementace části do FPGA byla provedena pomocí jazyka VHDL (VHSIC Hardware Description Language). V této části je uveden slovní popis implementace hardwarové části. Pro konkrétní implementaci v jazyce VHDL je k dispozici na přiloženém CD zdrojový kód s implementací pro FPGA.

5.1.1 Jazyk VHDL

Tento jazyk slouží k návrhu a simulaci digitálních integrovaných obvodů. Nejnovější verze umožňuje i návrh analogových obvodů. Jazyk není vázán na žádnou cílovou technologii. Obvody lze popsat behaviorálně nebo strukturálně.

- **Strukturální popis** obvodu je popis, při kterém se popíší jednotlivé komponenty obvodu a jejich propojení pomocí vodičů. Komponenty se mohou skládat z dalších subkomponent a ty mohou být samostatně popsány. Tímto je možné definovat hiearchii komponent. Tento popis je blízký finální obvodové realizaci.
- **Behaviorální popis** (popis chování) se používá k popisu obvodu algoritmem. Říká se mu také popis chování pomocí procesů. Při tomto popisu lze využít konstrukce běžné v jiných programovacích jazycích (funkce, cykly, atd.). Tento popis neuvažuje obvodové detaily. Všechny konstrukce vytvořené tímto popisem nemusí být syntetizovatelné tzn. nemusí existovat jejich ekvivalent v obvodové realizaci.

Komponenty se ve VHDL popisují pomocí:

- **Entita** definuje rozhraní komponenty. Pomocí entity jsou definovány signály komponenty. U signálů je nadefinována jejich šířka a jestli jsou vstupní nebo výstupní. Neobsahuje definici komponenty. Jedna entita může mít více architektur.
- **Architektura** určuje chování dané komponenty. Architektura je svázána s entitou. Je rozdělena na deklarační a příkazovou část
  - **Deklarační část** obsahuje deklaraci signálů a použitých komponent.
  - **Příkazová část** obsahuje zápis implementace pomocí strukturálního, behaviorálního nebo jiného popisu. To co je uvedeno v této části se vykonává paralelně.

5.1.2 Volba top-level entity

V obvodu vytvářeném pomocí VHDL existuje v hiearchii entit jedna nejvyšší top-level entita. Signály top-level entity na FITkitu korespondují s piny FPGA. V knihovně FITkitu existuje sada základních entit. Všechny entity obsahují signály SPI řadiče. Tyto signály jsou nutné z důvodu zabezpečení komunikace mezi MCU a komponentami v FPGA.

Při návrhu je nutné zvolit vhodnou top-level entitu a doplnit její architekturu. Jelikož navrhovaný obvod využívá VGA rozhraní, je nutné použít entitu umožňující přístup k tomuto rozhraní. Entita, která toto umožňuje je entita *tlv_pc_ifc*.

5.1.3 Zpřístupnění periferií MCU

MCU využívá periferie připojené k FPGA. FPGA pouze zprostředkovává přístup k těmto periferiím pro MCU. Aby mohlo tyto periferie MCU využívat, musejí být v FPGA pro tyto periferie umístěny řadiče. Řadiče jsou již implementovány v podobě komponent. Takže je v deklarační části top-level architektury použita tato komponenta a v části architektury jsou přiřazeny této komponentě signály příslušného SPI dekodéru, který tímto zpřístupní periferie MCU. Dále je SPI dekodéru přiřazena adresa, pomocí které je periferie adresovatelná z MCU. Podrobnější informace lze nalézt na [17]. Mezi takto zpřístupněné periferie patří klávesnice a LCD displej.

5.1.4 Bloková paměť

Bloková paměť je využívána pro ukládání zobrazovaných dat. Jak již bylo v sekci 4.2.2 návrhu řešení zmíněno, byla použita komponenta blokové paměti RAMB16_S9_S9. Při realizaci blokových pamětí je využito obou portů.

Textury a font využívají po jedné paměti. Implementace přístupu k těmto pamětem je jednodušší než k video paměti. Jeden port je připojen přes SPI dekodér k vnitřní sběrnici a je synchronizován synchronizačními pulsy vnitřní sběrnice. Povolení zápisu do této paměti je realizováno pomocí signálu WRITE_EN, který je přiveden z SPI dekodéru. Druhý port je určen pouze pro čtení. Tento port je synchronizován frekvencí 25 MHz. To je stejná frekvence jakou využívá VGA řadič. Využívá tedy stejný synchronizační signál jako VGA řadič. K této části jsou připojeny dva procesy, jeden je připojen k adresovému portu blokové paměti a stará se o adresaci dat. Druhý proces je připojen k datovému portu blokové paměti a zpracovává adresovaná data. Zpoždění, ke kterému dochází při práci s touto pamětí, je dlouhé jeden synchronizační puls. Video paměť využívá dvě blokové paměti. K těmto pamětem se přistupuje jako by to byla jedna bloková paměť. Stejně jako u předešlých

blokových pamětí je jeden port připojen k SPI dekodéru. Tento SPI dekodér umožňuje přístup k oběma pamětem. Má o jeden bit širší adresovou sběrnici. Tento jeden bit rozhoduje o tom, se kterou pamětí se bude pracovat. Druhý port je také podobný jako u předcházejících blokových pamětí. Tady je pro výběr blokové paměti použit devátý bit signálu ADDR_ROW, který určuje horizontální polohu vykreslovaného bodu VGA řadiče. Tímto je obrazovka přibližně rozdělena na dvě půlky. Jedna půlka je uložena v první blokové paměti a druhá půlka v druhé blokové paměti. O adresaci dat uložených v této paměti se stará proces, který skládá adresu ze signálů ADDR_ROW a ADDR_COLUMN VGA řadiče. Druhý proces je připojen na datovou část video paměti a pomocí získaných dat adresuje jednu z pamětí textur nebo fontu.

5.1.5 Procesy

O poskytování dat VGA řadiči se starají tři procesy. Všechny tři procesy jsou závislé na nastavení řídícího registru.

- **create_bram_addr** – Výstupem z procesu je adresa, která adresuje místo ve video paměti. Pro vytvoření adresy se používají signály ADDR_ROW a ADDR_COLUMN z VGA řadiče.
- **process_bram_data** – Tento proces zpracovává indexy získané z video paměti. Ze získaných dat proces složí adresu, která je poslána na adresový signál buď paměti textur nebo paměti fontu. Výběr paměti záleží na nastavení řídícího registru.
- **gen_graphic_data** – Tento proces se stará o zpracování dat z paměti textur a z paměti fontu. Zpracovaná data pošle na vstup VGA řadiče.

5.1.6 VGA řadič

Na synchronizační vstup VGA řadiče je připojen synchronizační signál o frekvenci 25 MHz. Pro adresaci videopaměti budou sloužit dva signály z VGA řadiče ADDR_COLUMN a ADDR_ROW. Tyto signály určují pozici zobrazovaného bodu. Podle těchto hodnot je proveden výběr indexu z videopaměti a následný výběr bodu v textuře nebo fontu.

Zpoždění dat z blokových pamětí bylo zohledněno v nastavení VGA řadiče. Je nutné mu nastavit zpoždění vstupního signálu o dva takty. Důvodem toho zpoždění je práce s blokovou pamětí. Pro získání všech informací o zobrazovaném bodě potřebujeme postupně získat data ze dvou blokových pamětí.

5.2 Implementace MCU

Aplikace pro MCU byla vytvořena v jazyce C. Při implementaci byla využita knihovna libfitkit. Zde je slovně popsána struktura aplikace a její funkce. Pro podrobnější seznámení se s jednotlivými funkcemi je k dispozici na přiloženém CD celý zdrojový kód aplikace.

5.2.1 Inicializace programu

Po zapnutí FITkitu dojde k nahrání konfigurace do FPGA a zavolá se funkce pro inicializaci LCD displeje a klávesnice. Dojde k inicializaci blokových pamětí texturami a fontem. Dále se inicializuje úvodní obrazovka hry, nastaví se proměnná uchovávající informaci o právě zobrazované obrazovce a dojde k inicializaci globálních proměnných. Tato obrazovka je

udělána v podobě menu. Nastaví se řídící registr v FPGA na režim zobrazení textové obrazovky.

5.2.2 Hlavní smyčka

Hlavní smyčka programu je tvořena nekonečným cyklem. Tato smyčka patří mezi typické programové konstrukce vestavěných systémů. Ve smyčce je volána funkce pro zjištění stavu klávesnice. Dále je zde čekací rutina, která ovlivňuje rychlost skenování stisknutých kláves. Pro zpracování příkazů z terminálu se zde nachází funkce pro obsluhu terminálu.

5.2.3 Obsluha terminálu

S aplikací je možné komunikovat přes terminál. Mimo základních funkcí, které terminál nabízí, je možné si definovat vlastní uživatelské funkce. Protože je potřeba nějakým způsobem nahrát textury, font a jednotlivé hry do Flash paměti, je nahrání implementováno pomocí příkazů zadaných v terminálu. Po zadání příkazu je zavolána funkce, která umožní nahrát data z PC pomocí terminálu do Flash paměti.

5.2.4 Ovládání aplikace

Pro pohyb po textové části hry jsou použity stejné klávesy jako pro pohyb ve hře. Pohyb po aplikaci je umožněn klávesami 2, 4, 6, 8. V textové části slouží pro výběr položky menu klávesa 5. Ve hře slouží klávesa # pro předčasné ukončení hry.

5.2.5 Obsluhy textových obrazovek

Obsluhy textových obrazovek fungují na stejném principu. Pokud se na obrazovce nachází menu, tak definují pohyb po tomto menu. Vybráním položky menu se zavolají inicializační funkce, která nastaví další obrazovku.

5.2.6 Výběr úrovně hry

Hra obsahuje více úrovní, takže musí být implementován i mechanismus výběru jednotlivých úrovní. Tento problém je řešen pomocí obrazovky, která umožňuje zvolit úroveň. Obrazovka s výběrem úrovní hry je složená z položek, mezi kterými lze snadno přecházet pomocí pohybových kláves. Po stisknutí klávesy pro výběr hry dojde k načtení požadované hry z Flash paměti do blokové paměti a nastavení řídícího registru na režim zobrazování textur.

5.2.7 Vlastní hra

Jediná obrazovka, která není textová, je obrazovka s hrou. Po výběru levelu hry následuje načtení hry z Flash paměti do video paměti FPGA. Spustí se časovač, který ve svojí obsluze přerušení měří čas hraní hry. Čas hry je zobrazován na LCD displeji FITkitu. Pokud uživatel stiskne pohybovou klávesu, dojde k ověření, zda je možné se na toto políčko přesunout. Pokud je to možné, aktualizuje se uživatelova pozice na hrací ploše. Dále dojde ke kontrole konce hry. Pokud uživatel přesunul všechny bedny na své pozice, je indikován konec hry. Uživateli se zobrazí textová obrazovka s dobou hry. Může také nastat situace, že uživatel nemůže dohrát hru. Tato situace není ve hře automaticky kontrolována, ale je možné, aby hráč předčasně ukončil hru tlačítkem # na klávesnici.

5.2.8 Kontrola přesunu hráče

Funkce využívá k výpočtu následující polohy hráče pole struktur. Podle stisknuté klávesy vybere z pole příslušnou strukturu. Ve struktuře je uložen vektor pohybu hráče. Tento vektor je aplikován na současnou souřadnici hráče. Co je na nové souřadnici zjistíme tak, že pomocí těchto souřadnic adresujeme místo v blokové paměti v FPGA. Tato paměť obsahuje informace o tom, co se na dané souřadnici nachází. Pokud je to místo na hrací ploše a je volné, hráč se tam přesune tím, že funkce zapíše do blokové paměti index textury hráče a původní nahradí volným místem. Pokud je to místo mimo hrací plochu, hráč zůstává na původní pozici. Poslední případ, který může nastat je ten, že ve směru pohybu hráče se nachází bedna. Potřebujeme zjistit, zda je před bednou volné místo. To jednoduše provedeme aplikací vektoru pohybu hráče na souřadnici bedny. Co je před bednou opět zjistíme ze stejné blokové paměti, která byla použita při přesunu samotného hráče s jednou výjimkou. Pro posun bedny jsou aplikovány stejné podmínky jako pro posun hráče. Na rozdíl od hráče bedna nemůže pohybovat jinou bednou. Pokud lze bednu i hráče přesunout, funkce provede aktualizaci příslušných dat v blokové paměti.

Kapitola 6

Závěr

6.1 Splnění cílů

Cílem práce byl návrh a implementace jednoduché hry pro platformu FITkit. Při návrhu hry byla zvolena implementace v MCU s využitím FPGA. Bylo tedy důležité realizovat spolupráci obou obvodů. Podle zvoleného návrhu byla úspěšně implementována hra Sokoban. Navíc bylo ke hře implementováno uživatelské menu a statistika hry. Tato statistika umožňuje hráči porovnat svůj výsledek s nejlepším výsledkem na daném FITkitu. Hra je funkční na obou verzích FITkitu.

Nad rámec zadání byl navrhnut a zkompletován herní ovladač. Po komplikacích s oživením herního ovladače se bohužel nepodařilo ovladač včas zprovoznit a tak nemohl být použit pro ovládání této hry. Výsledkem práce je tedy hra, která je ovládána pouze pomocí klávesnice na FITkitu.

6.2 Přínos

Tato práce ukázala možnost vývoje her na FITkitu. V práci byla ukázána implementace jednoduché hry s textovým menu. Doposud vytvořené projekty obsahovaly pouze vlastní obrazovku aplikace. V projektu byla ukázána možnost jednoduché implementace dvou zobrazovacích režimů. Dále byla ukázána komunikace mezi různými částmi FITkitu jako jsou FPGA, MCU, Flash atd.. Zveřejnění zdrojových souborů práce by mohlo přispět k větší popularizaci FITkitu mezi studenty.

6.3 Možnosti dalšího vývoje

Jelikož nebyl zprovozněn herní ovladač, nabízí se v rámci jiného projektu zprovoznit tento ovladač a vyvinout aplikaci, která by demonstrovala jeho využití. Znamenalo by to i napsání knihoven, které by s ním po sběrnici I2C komunikovaly. Další zajímavou možností je pokusit se realizovat celou hru v FPGA. Pro ukládání zobrazovaných dat by mohla být využita paměť SDRAM. Má větší kapacitu a je přímo připojená k FPGA, takže by nezatěžovala sběrnici SPI. V tomto projektu je použito zobrazování textur nebo fontu. Oba dva typy zobrazeni se na jedné obrazovce nevyskytují. Bylo by tedy vhodné navrhnout v FPGA obvod, který by umožňoval zobrazování jak textu, tak grafiky. Hra je bez zvukového doprovodu, takže by bylo možné rozšířit hru o zvukový doprovod.

Literatura

[1] Eagle Software [online]. http://www.elcad.cz/eagle/, 2005 [cit. 2009-05-17].

[2] MSP430F241x, MSP430F261x MIXED SIGNAL MICROCONTROLLER [online]. http://focus.ti.com/docs/prod/folders/print/msp430f2616.html, 2006 [cit. 2009-05-15].

[3] MSP430x15x, MSP430x16x, MSP430x161x Mixed Signal Microcontroller (Rev. E)[online]. http://focus.ti.com/docs/prod/folders/print/msp430f168.html, 2006 [cit. 2009-05-15].

[4] MPR083, Proximity Capacitive Touch Sensor Controller [online]. www.freescale.com/files/sensors/doc/data_sheet/MPR083.pdf, 2008 [cit. 2009-05-17].

[5] MPR084, Proximity Capacitive Touch Sensor Controller [online]. www.freescale.com/files/sensors/doc/data_sheet/MPR084.pdf, 2008 [cit. 2009-05-17].

[6] Three Axis Low-g Digital Output Accelerometer [online]. www.freescale.com/files/sensors/doc/data_sheet/MMA7455L.pdf, 2008 [cit. 2009-05-17].

[7] FITkit: Úvod [online]. http://merlin.fit.vutbr.cz/FITkit/, 2009 [cit. 2009-05-10].

[8] Spartan-3 Generation FPGA User Guide [online]. http://www.xilinx.com/support/documentation/user_guides/ug331.pdf, 2009 [cit. 2009-05-12].

[9] FITkit - Hra piškvorky [online]. https://merlin.fit.vutbr.cz/FITkit/private/web/index.php?pg=aplikace&cl=apps_games_tictactoe, 2009 [cit. 2009-05-16].

[10] Gyroscope [online]. http://en.wikipedia.org/wiki/Gyroscope, 2009 [cit. 2009-05-17].

[11] I2C. http://en.wikipedia.org/wiki/I%C2%B2C, 2009 [cit. 2009-05-17].

[12] Microelectromechanical systems [online]. http://en.wikipedia.org/wiki/Microelectromechanical_systems, 2009 [cit. 2009-05-17].

[13] SPI. http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus, 2009 [cit. 2009-05-17].

[14] Video game console [online]. http://en.wikipedia.org/wiki/Video_game_console, 2009 [cit. 2009-05-17].

[15] FITkit - Textový režim [online]. https://merlin.fit.vutbr.cz/FITkit/private/web/index.php?pg=aplikace&cl=apps_vga_textmode, 2009 [cit. 2009-05-18].

[16] FITkit - VGA rozhraní [online]. http://merlin.fit.vutbr.cz/FITkit/docs/firmware/fpga_vga.html, 2009 [cit. 2009-05-18].

[17] FITkit - Komunikační systém [online]. http://merlin.fit.vutbr.cz/FITkit/docs/firmware/fpga_interconect.html#propojovac_syst_m, 2009 [cit. 2009-05-19].

[18] Pech, I. J.: Nebojte se FPGA [online]. http://hw.cz/Teorie-a-praxe/Dokumentace/ART365-Nebojte-se-FPGA.html, 2002 [cit. 2009-05-12].

[19] Vašíček, Z.: FITkit - Hardware [online]. http://merlin.fit.vutbr.cz/FITkit/hardware.html, 2009 [cit. 2009-05-10].