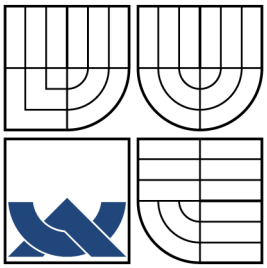
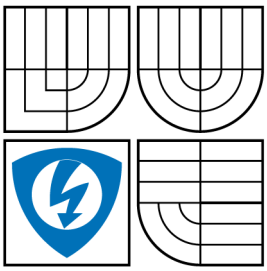


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND
COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

SBĚRNICE PCI EXPRESS MODUL LOGICKÉHO ANALYZÁTORU

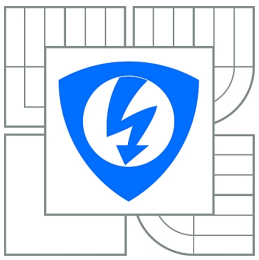
LOGIC ANALYZER MODULE BASED ON PCIE CARD

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — TOMÁŠ JUŘÍK
AUTHOR

VEDOUCÍ PRÁCE — ING. SOBĚSLAV VALACH
SUPERVISOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Bakalářská práce

bakalářský studijní obor
Automatizační a měřicí technika

Student:	Tomáš Juřík	***ID:***	109667
Ročník:	3	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Sběrnice PCI express modul logického analyzátoru

POKYNY PRO VYPRACOVÁNÍ:

Ve studijní fázi projektu se seznamte se sběrnicí PCIe a jejími základními charakteristikami. Ve druhé fázi projektu implementujte PCIe core do hradlového pole a prověřte chová systému zda odpovídá teoretickým předpokladům. V poslední fázi vytvořte jednoduchý modul, který bude sloužit jako logický analyzátor připojený k rozhraní PCIe. Funkci ověřte jak na reálných tak i na syntetických datech. Zadání bude realizováno na platformě FPGA Xilinx Spartan3 nebo Virtex5.

DOPORUČENÁ LITERATURA:

Termín zadání:	8.2.2010	***Termín odevzdání:***	31.5.2010

Vedoucí práce: Ing. Soběslav Valach

prof. Ing. Pavel Jura, CSc.
Předseda oborové rady

ABSTRAKT

Cílem této práce je implementace jednoduchého logického analyzátoru do hradlového pole připojeného k sběrnici PCI-Express. Dále jsou vytvořeny moduly čtyř čítačů pro generování testovacích dat. V práci je popisán princip funkce logického analyzátoru. Je také rozebrána vývojová vývojová karta Spartan-3 PCI Express Starter Kit a architektura hradlových polí Xilinx Spartan-3. Uvedeny jsou jednotlivé kroky vývoje součástí logického analyzátoru.

KLÍČOVÁ SLOVA

Logický analyzátor, FPGA, PCI Express, Link, Lane, Xilinx, Spartan-3, XC3S1000, ISE, CORE Generator, WinDriver

ABSTRACT

The goal of this bachelor's thesis is to implement simple FPGA-based logic analyzer connected to PCI-Express bus. Furthermore four counters are implemented to generate testing dataset. This thesis describes a fundamental priciple and use of logic analyzer. An overview of Spartan-3 PCI Express Starter Kit development board and Xilinx Spartan-3 field-programmable gate array anrchitecture is given. Stages of logic analyzer development are detailed as well.

KEYWORDS

Logic analyzer, FPGA, PCI Express, Link, Lane, Xilinx, Spartan-3, XC3S1000, ISE, CORE Generator, WinDriver

JUŘÍK, T. *Sběrnice PCI express modul logického analyzátoru*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 44 s. Vedoucí bakalářské práce Ing. Soběslav Valach.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Sběrnice PCI express modul logického analyzátoru“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 31. května 2010 .....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu bakalářské práce Ing. Soběslavu Valachovi za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 31. května 2010 .....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

Cílem této práce je vytvoření jednoduchého modulu logického analyzátoru do hradlového pole. Použití klasických logických analyzátorů nebo osciloskopů tu selhává. Tento fakt je dán tím, že většina děje se odehrává uvnitř čipu a sondám jsou tak požadované signály nepřístupné. Vyvedení signálů na výstupní piny je málo aplikovatelné řešení. Výstupních pinů je omezené množství a počet vnitřních propojek je o několik řádů vyšší. S některými pouzdry je připojení sond dokonce nemožné. Řešením je, naprogramovat logický analyzér přímo do intérní logiky hradlového pole a komunikačním kanálem monitorovat vnitřní stavy.

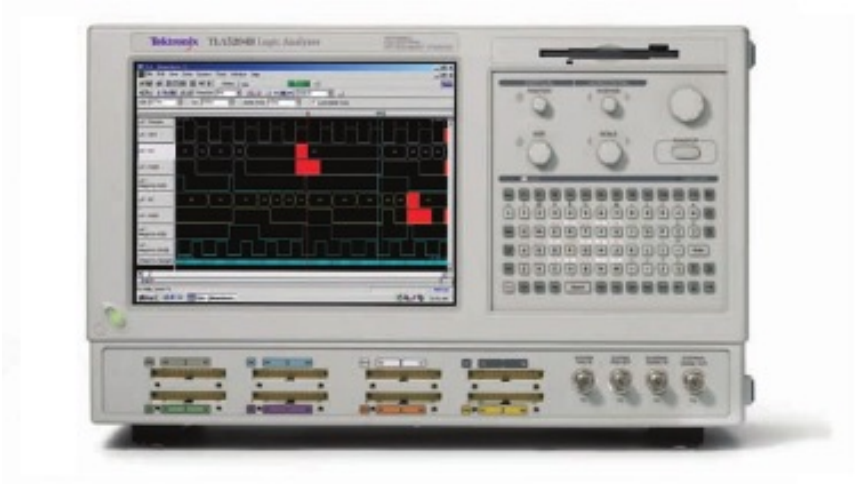
V následujících kapitolách jsou popsány jak hardwarové, tak softwarové nástroje použité při implementaci. Definovány jsou základní vlastnosti logického analyzátoru a jeho rozdíly s osciloskopem. Rozhraní PCI-Express je rozebrán v takové míře, abychom ho mohli implementovat s modulem logického analyzátoru.

1 LOGICKÝ ANALYZÁTOR

1.1 Definice

Logický analyzátor je zařízení, které se používá k zobrazení a analýze signálů. Jedná se o signály logických úrovní o kmitočtech zpravidla daleko přesahující pozovací schopnost člověka. Omezení pozorovatele se obchází pomocí realtime záznamu signálů do paměti a následujícím offline zpracování a vykreslení průběhů. Logický analyzátor je tedy vývojářův aparát určený k ladění chování integrovaných obvodů a desek plošných spojů.

Na první pohled může zdát, že logický analyzátor má stejnou funkci jako osciloskop, ale není tomu tak. Osciloskop je většinou vybaven dvěma či čtyřmi analogovými kanály. Logický analyzátor má naopak vstupy binární často v počtu 16, 32 a více. Existují i zařízení kombinující tyto funkce v podobě tzv. osciloskopů smíšených signálů.



Obrázek 1.1: Logický analyzátor Tektronix TLA5000 [7]

Logické analyzátory můžou být v podobě samostaného přístroje s obrazovkou nebo pouze jako vstupní zařízení připojené k PC např. přes USB. Zobrazení a rozbor signálů je proveden pomocí specifického programu.

Tato práce se ale zabývá implementací analyzátoru pro interní signály hradlového pole. Tyto signály nejsou vyvedeny na piny integrovaného obvodu a nedokážeme je tedy připojit sondami ani jinými fyzickými vodiči. Jedná se tedy čistě o softwarový nástroj.

1.2 Typy analýz

Při použití analyzátoru rozlišujeme typ analýzy:

Časová – Pozorování synchronnosti vícevodičové sběrnice, korelaci vůči hodinovému signálu a detekce vzniku poruchových impluzů (tzv. Glitch).

Stavová – Rozbor komunikace sběrnic (např. sledování reakce datové sběrnice na změnu řídící). Signály by měly být synchronní.

1.3 Trigger

Na sběrnicích v průběhu testování se obvykle mění obrovské množství stavů, nicméně vývojáře zajímá jeden určitý konkrétní. Logický analyzér data cyklicky zapisuje do paměti a přepisuje nejstarší zaznamenané vzorky. Pro zastavení zápisu se využije předem definované události v podobě tzv. triggeru (z angl. trigger = spoušť). Používají se dvě základní triggrovací podmínky (angl. trigger condition):

Pattern trigger – Spoušť je citlivá na logické úrovně kanálů podle daného vzoru.

Edge Trigger – Hranový trigger pozoruje výskyt nástupné či sestupné hrany.

Spojením základních podmínek, kombinační a sekvenční logiky vznikají sofistikované triggery. Příkladem může být například sledování TCP a HTTP protokolů.

Trigger je tedy událost, která při výskytu umožňuje logickému analyzéru dokončení zaplnění paměti a zastavení měření.

1.4 Existující řešení

Nejznámější nástroj pro analýzu interních signálů FPGA je software ChipScope od firmy Xilinx. Skládá se ze dvou hlavních částí:

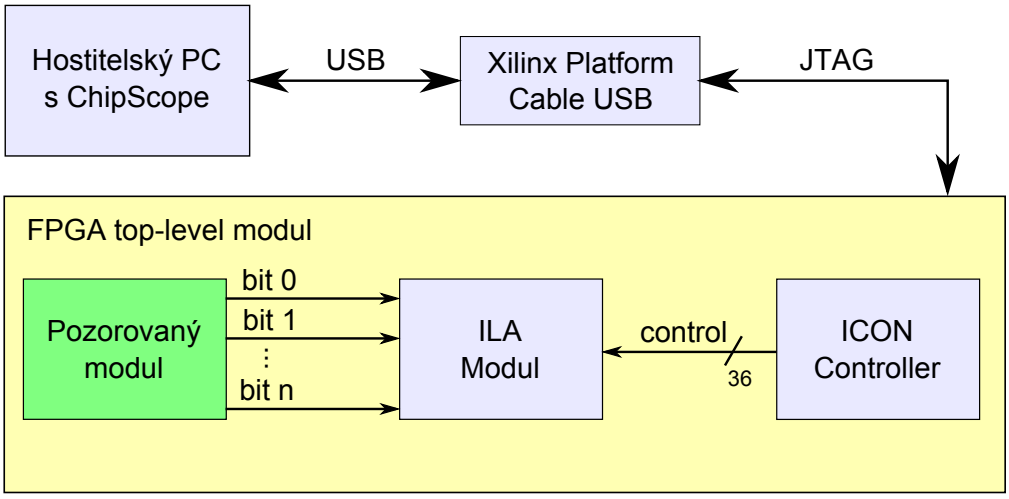
Embedded IP core – Vložený modul do návrhu zachytává hodnoty vzorků uvnitř FPGA nebo ovládá virtuální vstupy a výstupy (modul VIO – Virtual Input/Output core).

Software na hostitelském počítači – Načítá a vizualizuje uložené vzorky.

ChipScope obsahuje množství embedded modulů pro ladění různorodých typů sběrnic. Pro funkci univerzálního logického analyzéru je určena kombinace těchto modulů:

- ICON (Integrated CONtroller): Modul řadiče, který poskytuje komunikaci mezi hostitelským PC a embedded ChipScope moduly (např. VIO a ILA). V sysému je vždy jen jedna instance.
- ILA (Integrated Logic Analyzer): Modul, který umožňuje záznam hodnot vstupních signálů a umístění triggeru. Vzorky jsou ukládány do blokových RAM z prostředků FPGA. Modulů může být vloženo i několik, jsou vzájemně nezávislé.

Výhoda použití ChipScope modulů leží v možnosti zaznamenávat i vstupně/výstupní signály z hardwaru při běhu zařízení. Dosaženo je to tím, že tyto moduly jsou vlastně standardní HDL moduly, které jsou syntetizovány a implementovány zároveň s vlastním návrhem. Simulací jsme sice schopni dosáhnou vytvoření umělých signálů pro top-level modul, ale je to velmi obtížné. Na obr. 1.2 je zobrazen propojovací žetězec při použití jednoho ILA modulu.

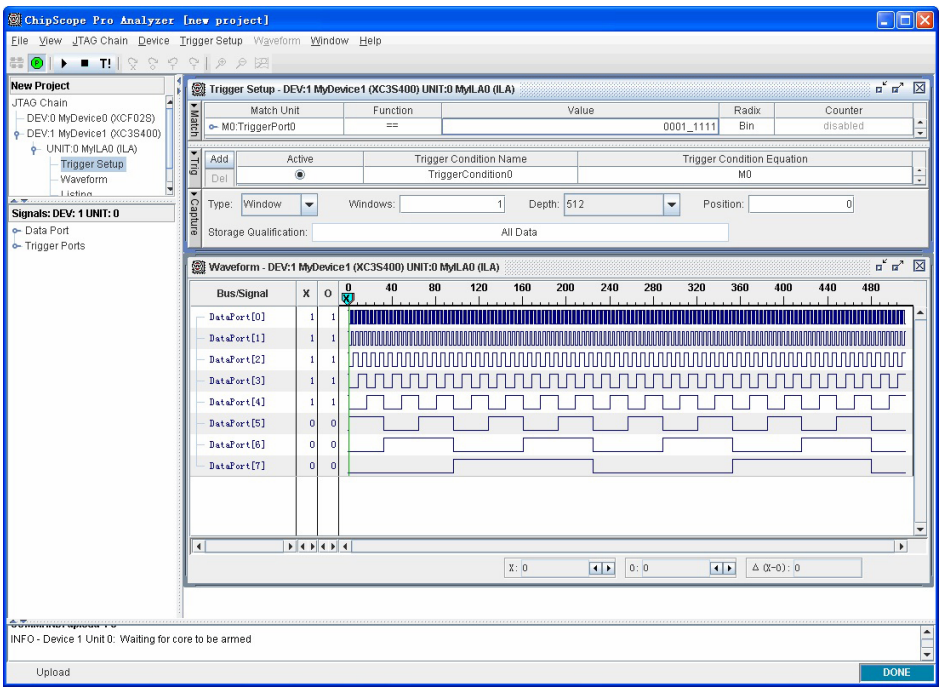


Obrázek 1.2: Diagram propojení ChipScope modulů

Pro připojení k PC je použit USB JTAG debugger, který se souběžně použije jako programátor (viz 2.1.3). Moduly ILA a ICON jsou generovány v programu ChipScope Core Generator pro danou architekturu hradlového pole. Před generováním se volí velikost paměti vzorků, počet vstupních kanálů a typ triggeru. Do top-level modulu jsou vloženy jako uzavřené (tzv. blackbox) instance.

Vlastnosti ChipScope Pro [8]

- 1 až 4096 vstupních kanálů.
- Pamět na 256 až 131 072 vstupních vzorků.
- Šířka triggeru až 256 kanálů.
- Všechny funkce triggeru a ukládání dat jsou synchronní s frekvencí až do 500 MHz.
- Vložení až 15 instancí ILA modulu.
- Signál pro potvrzení triggrovací podmínky má zpoždění 10 hodinových cyklů od odpovídajících vstupních dat.



Obrázek 1.3: Grafické rozhraní analyzéru ChipScope Pro

2 HARDWAROVÉ PROSTŘEDKY

Pro implementaci logického analyzátor byla vybrána platforma Xilinx Spartan-3 v podobě vývojové karty Spartan-3 PCI Express Starter Kit. Ta je přímo určena k vývoji aplikací založených na rozhraní PCI Express. Je možnost ji využít ale i k obecnému vývoji pro FPGA.

Na testování i vývoj byl použit počítač se základní deskou ASRock A330ION. Tato deska byla vybrána pro svůj malý formát Mini-ITX a vyvedenému rozhraní PCI Express 2.0 x16, což u tohoto formátu není běžné. Procesor je integrovaný typu Intel Dual-Core Atom 330 se základním taktem 1.6 GHz. Paměti jsou použity v konfiguraci 2x 2 GB DDR3 taktovaných na frekvenci 1066 MHz. Volba vývojového počítače má podstatný význam pro rychlost sestavení programu. V popsané konfiguraci sestavení finálního bitstreamu pro FPGA tvrvá přibližně 15 minut.

K programování hradlového pole byl použit originální programátor od firmy Xilinx s názvem Platform Cable USB. K počítači je připojen pomocí rozhraní USB 2.0 a k vývojové desce přes standardizované rozhraní JTAG. Primární použití rozhraní JTAG je testování integrovaných obvodů a plošných spojů, ale používá se i k programování pamětí pomocí přidružených standardů (např. IEEE 1532). Identifikátor modelu použitého programátoru je DLC9.

2.1 Spartan-3 PCI Express Starter Kit

Vybraná vývojová deska poskytuje vhodné prostředí k okamžitému vývoji rozmanitých zařízení s aplikační logikou umístěnou v FPGA a komunikujících přes rozhraní PCI Express. Tato komunikace je umožněna pomocí integrovaného obvodu Philips PX1011A PCI Express PHY, který obsluhuje nízkoúrovňové signály a část protokolu sběrnice PCI Express. Podrobějšímu popisu tohoto obvodu a komunikačního modelu je věnována kapitola 3.2.

Typické cílové aplikace vývojového kitu

- Komunikační karty, VGA grafické adaptéry, instrumentace

- Digital Signal Processing (DSP) a další přídavné PC karty

2.1.1 Relevantní komponenty vývojového kitu

- Xilinx Spartan-3 XC3S1000 FPGA [2]
  - 676-pin FBGA pouzdro
  - 391 uživatelských I/O pinů
  - 1920 CLB bloků
  - 54 kB blokové RAM
  - 4 DCM bloky
- Paměť konfigurace
  - Xilinx 8 Mbit Platform Flash configuration PROM
- Zdroje hodinového signálu
  - PCIe RX Clock
  - 50 MHz crystal clock oscillator
- Konektory a rozhraní
  - Philips Semiconductors 2.5 Gbps PCI Express single lane PHY
  - PCI Express X1 Card Edge

Pro implementaci logického analyzátoru není tedy potřebný žádný zákrok ze strany hardwaru. Jedinou nezbytností je korektní nastavení vstupu napájení a způsobu programování FPGA. Tyto kroky jsou popsány v následujících podkapitolách.

Prosotrové uspořádání vývojové karty Spartan-3 PCI Express Starter Kit je zobrazeno na obr. 2.1. Relevantní komponenty jsou pro zvýraznění vykresleny červenou barvou.

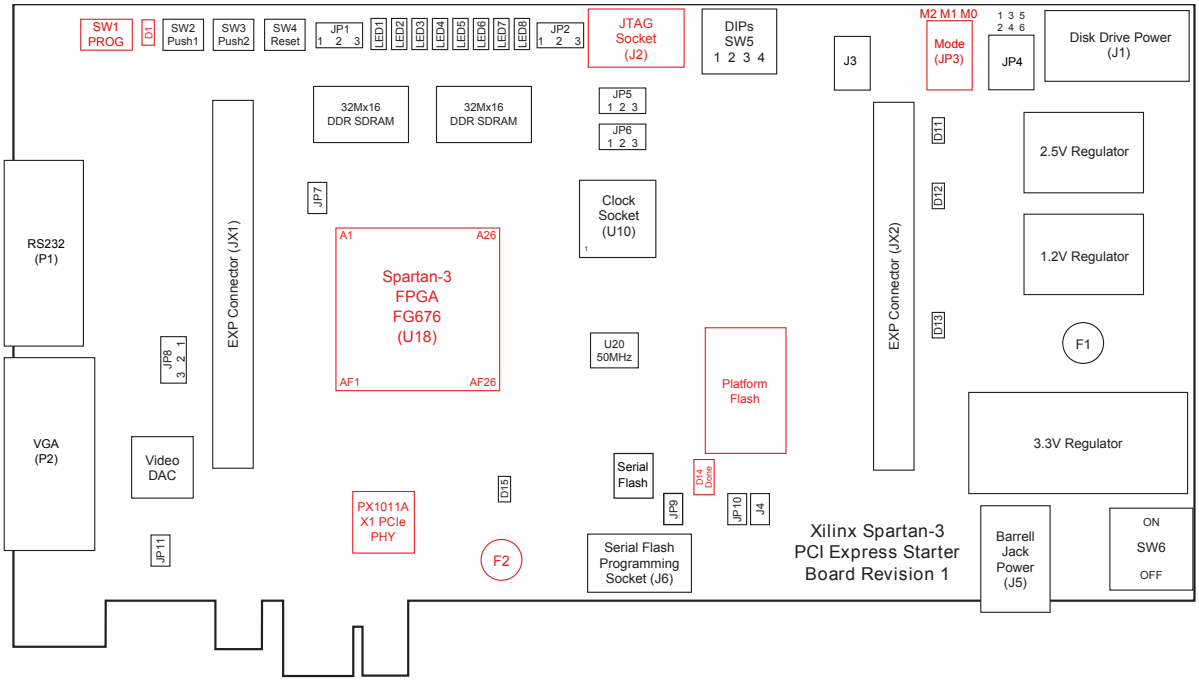
2.1.2 Zdroje napájení

Vývojová deska může být napájena několika možnými způsoby:

- 3.3V PCI Express edge konektor
- 5V ATX standardní konektor pro jednotky o velikosti 5,25 palců
- 5V kulatý konektor

Při volbě způsobu napájení je nutné zkontrolovat zapojení pojistek F1 a F2 podle uvedených pravidel v [1]. Při nesprávném zapojení těchto pojistek a použitém napájením dojde k nenávratnému poškození součástí karty.

Pro realizaci je zvoleno napájení pomocí PCIe konektoru. Pojistka je tedy umístěna pouze do konektoru F2. V F1 nesmí být umístěna. Napájením z PCIe rozhraní je i zajištěn souběžný start a vypnutí karty s počítačem. Poloha přepímače ON/OFF (SW6) při použití tohoto typu napájení nemá žádný vliv. O správném napájení jsme informováni LED diodou D1, případně diodami stabilizátorů vstupního napětí D11, D12 a D13.



Obrázek 2.1: Prosotrové uspořádání vývojové karty [1]

Relevantní komponenty jsou pro zvýraznění vykresleny červenou barvou.

2.1.3 Konfigurace FPGA

Hradlové pole Spartan-3 XC3S1000 používá ke konfiguraci statickou paměť RAM. To znamená, že po připojení napájení je nutné vždy nahrát znovu konfiguraci. Vývojová deska nám nabízí dvě možné řešení:

- Stáhnutí konfigurace programátorem přes JTAG rozhraní. Použitím této metody je ale nutné mít stále připojenený programátora a spustit konfiguraci po každém zapnutí hradlového pole.
- Naprogramovat Platform Flash PROM a nastavit konfigurační mód tak, aby se hradlové pole konfigurovalo z této paměti. Při každém dalším restartu se použije obraz této paměti a není potřeba programátoru.

Uvědomíme-li si, že hradlové pole musí být schopno komunikaci ihned po zapnutí počítače (napájení), aby se karta zapsala v systému a byl jí vyhrazen paměťový prostor, tak se musí použít možnost druhá. K tomu je využit obvod Xilinx XCF08P Platform Flash PROM. To už je permanentní paměť o dostatenčné velikosti 8 Mb. Je připojena do JTAG řetězce společně s hradlovým polem a pro připojení programátoru slouží konektor J2. Na desce je umístěn indetický konektor J6, ten je ale

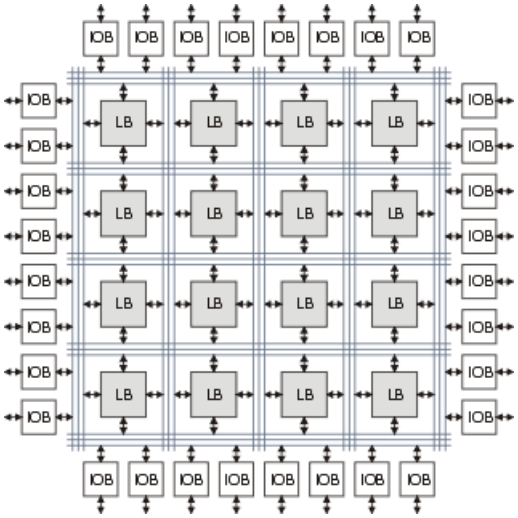
Pro nastavení požadovaného konfiguračního módu jsou na desce umístěny příslušné piny M2, M1 a M0 konektoru JP3. Dle [1] je vybrán mód Master Serial uzemněním všech pinů toho konektoru. Tedy M2=M1=M0=0.

Pokud je potřeba vyvolat konfiguraci ručně, tak je možné použít tlačítko SW1 (PROG), které spustí znovunačtení obrazu. Po stlačení tohoto tlačítka zhasne modrá LED dioda D14 DONE a je vyvolán nucený restart. Po úspěšné konfiguraci se opětovně LED dioda rozsvítí.

2.2 Obecná architektura FPGA

Field Programmable Gate Array - specifická třída IO obvodů, která je navržena tak, aby umožnila realizovat elektronický systém v jediném IO nastavením propojení mezi jednotlivými logickými bloky. Propojení logických bloků je možné mnohokrát měnit a to je taky hlavní výhodou proti klasickým integrovaným obvodům ASIC (Application-Specific Integrated Circuit). Na Obr. 2.2 je znázorněna typická struktura FPGA.

Kapacita FPGA se udává v počtu systémových hradel. Tento počet je ale pouze orientační, neboť jejich počet zahrnuje i kompletní rozsah interní logiky včetně konfiguračních prostředků a navíc výrobcům půjde často i o marketing a z toho důvodu se snaží vypočítat co největší číslo. Při znalosti struktury má pro vyšší vypovídající hodnotu počet logických bloků.

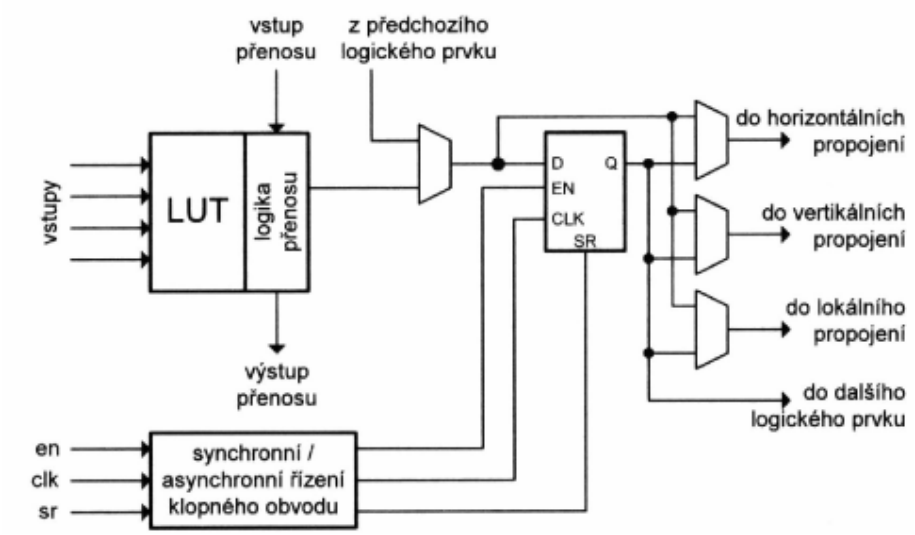


Obrázek 2.2: Typická architektura obvodu FPGA [9]

Základem každého FPGA je

- Matice Logických Bloků - Tyto bloky provádí jednoduché logické funkce. Složitější funkce se realizují ve více LB (Logický Blok) propojených najednou.
- Vstupně/výstupní bloky (IOB) – obvod pro každý pin FPGA.
- Propojovací matice – Slouží jednak k propojení LB mezi sebou a také k propojení mezi LB a IOB.

Všechny bloky mohou být různě propojeny globální propojovací maticí. Nejpoužívanější struktura konfigurovatelného logického bloku je znázorněna na obr. 2.3.



Obrázek 2.3: Typická struktura logického bloku

Na LUT (Look-Up Table) jsou přímo připojeny vstupní linky, které fungují jako adresová sběrnice. Výstup LUT je1-bitová hodnota. Pomocí LUT se nechá vytvořit jakákoli funkce čtyř nezávislých proměnných. LUT také může být konfigurována jako distribuovaná paměť RAM velikosti 16 x 1 bit nebo jako 16-bitový posuvný registr.

FPGA obvykle umožňují propojit některé signály logických bloků přímo se sousedním bez nutnosti využívat globální propojovací matici. Takovéto spoje mají mnohem menší zpoždění.

2.3 Architektura Spartan-3 XC3S1000

2.3.1 Hlavní rysy hradlového pole XC3S1000

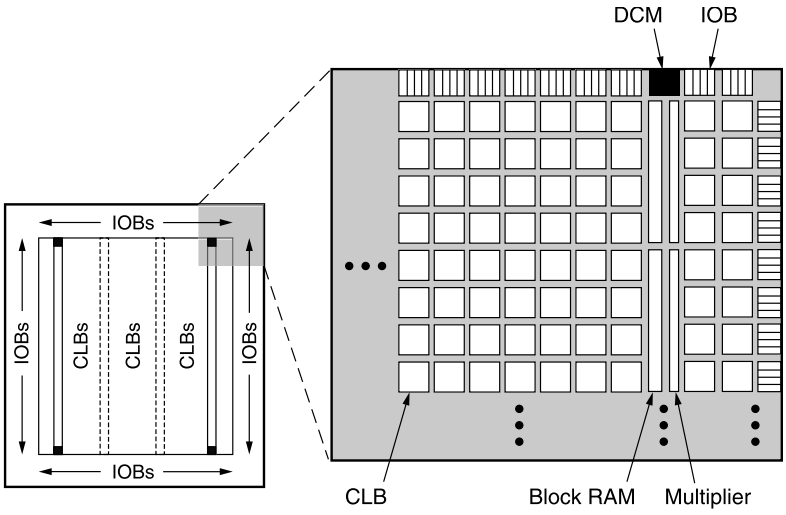
- jeden milión systémových hradel, 17280 logických buněk, 1920 CLBs
- 391 I/O pinů s maximem 175 diferenčních párů
- 18 jednopinových (single-ended) a osm rozdílových (differential) I/O
- 432Kb blokové RAM
- 120Kb distribuované RAM

- osm globálních hodinových linek, čtyři bloky Digital Clock Manager (DCM)
- Vestavěné násobičky 18x18 bitů

Podobně jako je tomu u jiných obvodů FPGA, jsou obvody řady Spartan-3 organizovány jako pole konfigurovatelných logických bloků (CLB) obklopených vstupně/výstupními bloky (IOB). Část plochy čipu zabírají specializované strukturní prvky pro zrychlení činnosti obvodu a rozšíření funkčnosti.

Architektura je složena z pěti funkčních/logických bloků:

- IOBs (Input/Output Blocks) řídí tok dat mezi I/O pinem a vnitřní logikou
- CLBs (Configurable Logic Blocks) obsahuje LUT tabulky
- Block RAM je blok určený k ukládání dat
- Násobičky umožňující vynásobení dvou 18-bitových čísel
- DCM (Digital Clock Manager) poskytuje autokalibraci, plně digitální řešení distribuce zpoždění, násobení, dělení a fázový posun hodinového signálu

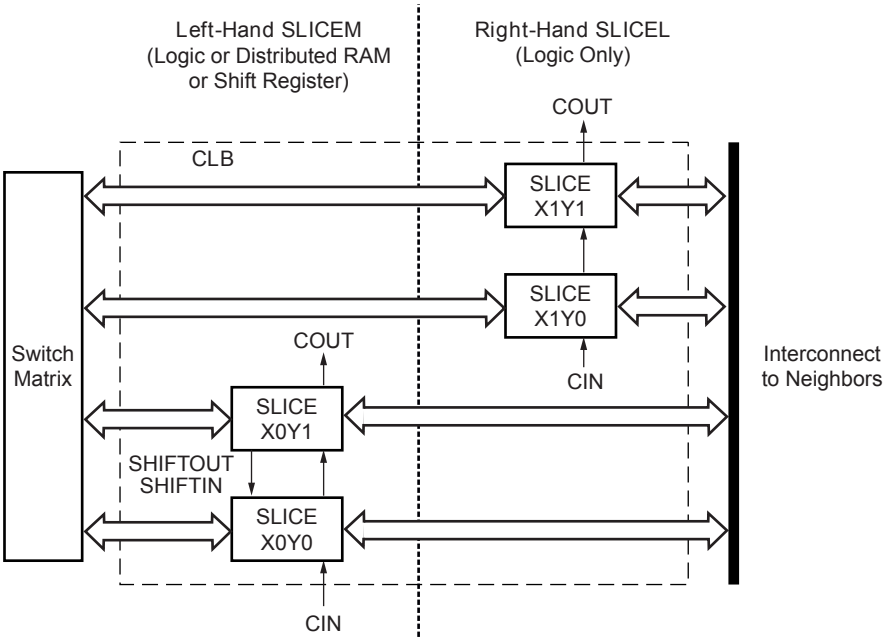


Obrázek 2.4: Organizace rozložení dílčích bloků pro architekturu rodiny Spartan-3 [3]

2.3.2 CLBs (Configurable Logic Blocks)

CLB je tvořena čtyřmi LC (Logic Cell), které jsou uspořádány do dvojic (řezy) tak, že daná dvojce má společný carry přenos. Oba řezy obsahují dva logické funkční generátory

LUT, dva ukládací elementy, multiplexer, carry logiku. Těmito bloky jsou tvořeny logické, aritmetické a paměťové funkce. Levý pár má navíc další dvě funkce použití: ukládaní dat pomocí distribuované RAM a posun dat s 16-bitovým registrem.

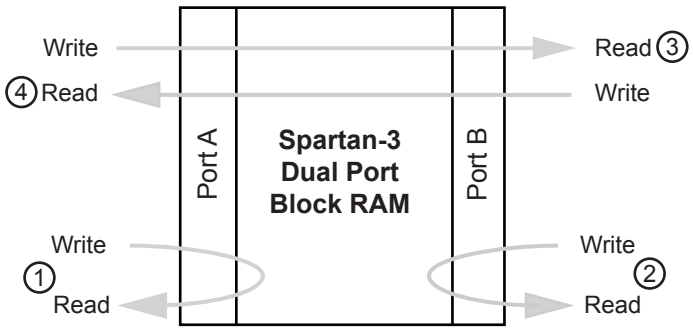


Obrázek 2.5: Uspořádání řezů uvnitř CLB [3]

Součástí CLB jsou také sběrnice propojující jednotlivé funkční bloky. Existují čtyři druhy propojení: long lines, hex lines, double lines a direct lines. Tyto propojení umožňují vytvoření složitějších sekvenční/kombinačních funkcí. Do CLB také vstupuje osm globálních hodinových signálů z globální sítě hodinového signálu. Propojení pro rozvod vysokofrekvenčního hodinového signálu jsou speciálně vytvořeny s minimální kapacitou a zpožděním (low-skew).

2.3.3 Block RAM

Všechny obvody rodiny Spartan-3 obsahují blokovou RAM uspořádanou po blocích velikosti 18Kbit. Pro uložení velkého množství dat je efektivnější použít blokovou RAM než distribuovanou RAM. Spartan-3 XC3S1000 obsahuje 24 RAM bloků ve dvou sloupcích s celkovým počtem 442 368 adresovatelných bitů. Paměť také umožňuje dvouportový přístup – může být konfigurována jako Single-Port nebo Dual-Port RAM. Paměť je na čipu rozmístěna po blocích tvořící sloupce. Paměť lze konfigurovat do různých „rozměrů“. Dle šířky datového slova odpovídá šířka adresové sběrnice. Dva identické porty A a B dovolují nezávislý přístup do sdílené blokové RAM: (1) zápis a čtení z portu A, (2) zápis a čtení z portu B, (3) přenos dat z portu A do B, (4) přenos dat z portu B do A.



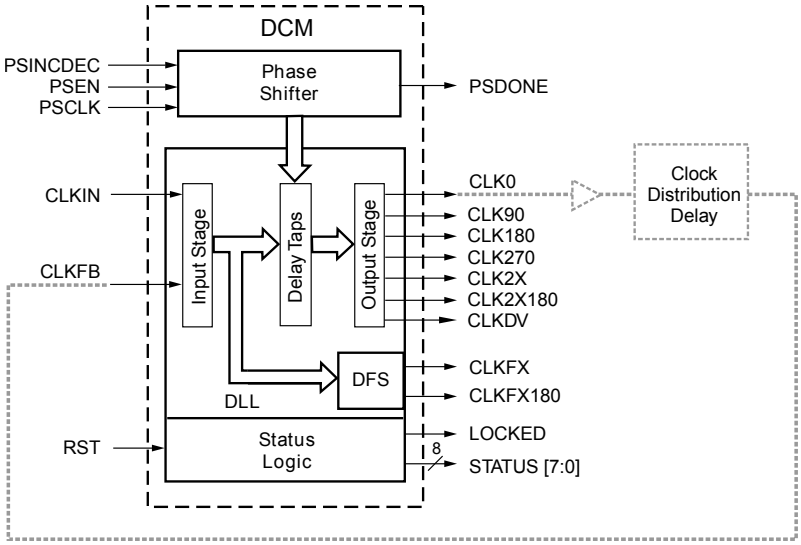
Obrázek 2.6: Datové cesty dual-port blokové RAM [3]

2.3.4 IOBs (Input/Output Blocks)

IOB poskytuje programovatelné obousměrné rozhranní mezi I/O pinem a interní logikou FPGA. Architekrura Spartan-3 nabízí nastavení jednoho z 18 možných úrovňových standardů pro jednotlivé piny (single-ended) a jedenoho z osmi pro rozdílový výstup (differential).

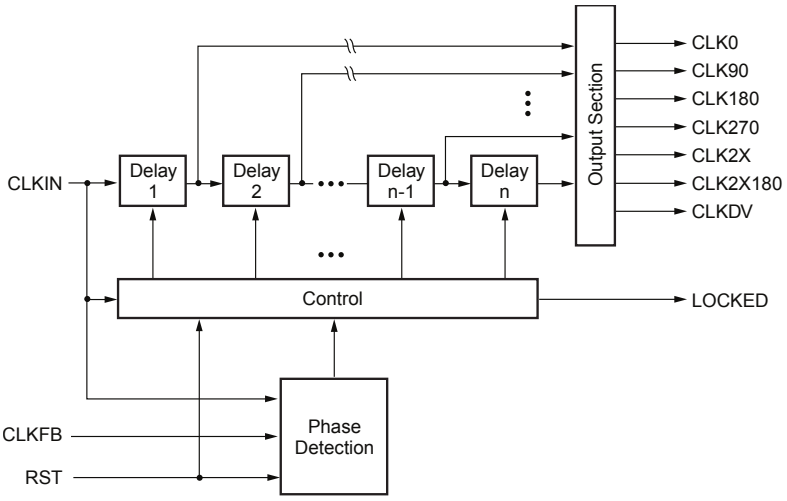
2.3.5 DCM (Digital Clock Manager)

DCM řídí, upravuje a distribuuje hodinový signál po celém čipu. Plní tři základní funkce: eliminace časového zpoždění, frekvenční syntéza, fázový posun signálu. Obvod Spartan-3 XC3S1000 obsahuje čtyři DCM funkční bloky.



Obrázek 2.7: DCM funkční blok [3]

DCM se skládá ze čtyř základních bloků: DLL (Delay-Locked Loop), DFS (Digital Frequency Synthesizer), PS (Phase Shifter), Status Logic. Základní funkce DLL je vyrovnávání časového zpoždění a generování fázově posunutých signálů. DLL má dva hodinové vstupy a sedm hodinových výstupů.

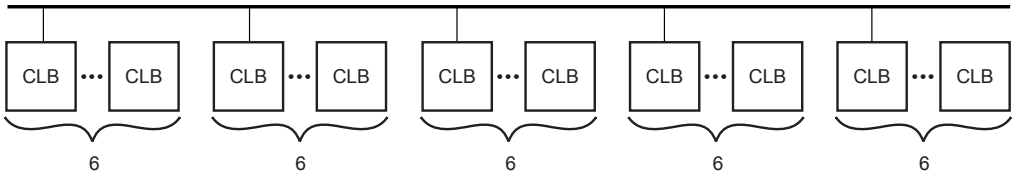


Obrázek 2.8: Zjednodušené schéma DLL [3]

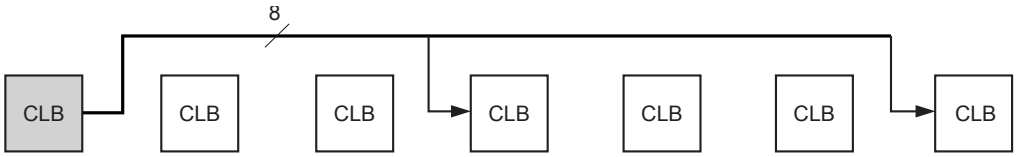
2.3.6 Propojovací síť

Propojovací síť (matice) je tvořena čtyřmi druhy propojovacích linek

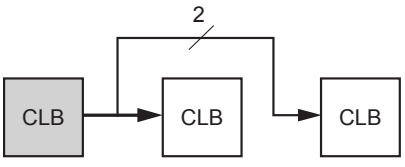
- Long line – vhodné pro distribuci globálního vysokofrekvenčního signálu s malým zpožděním, propojuje každou šestou CLB –2.9
- Hex line – také vhodné pro vedení rychlých signálů s malým zpožděním, navíc umožňuje efektivnější propojení, protože jsou spojeny s každou třetí CLB –2.10
- Double line – spojují každou druhou CLB s vysokou flexibilitou propojení -2.11
- Direct line – propojuje nejbližších sousedních osm CLB –2.12



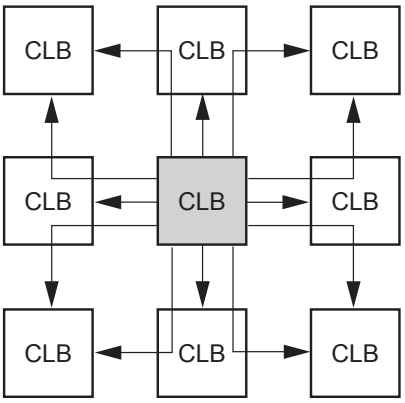
Obrázek 2.9: Long Line [3]



Obrázek 2.10: Hex Line [3]



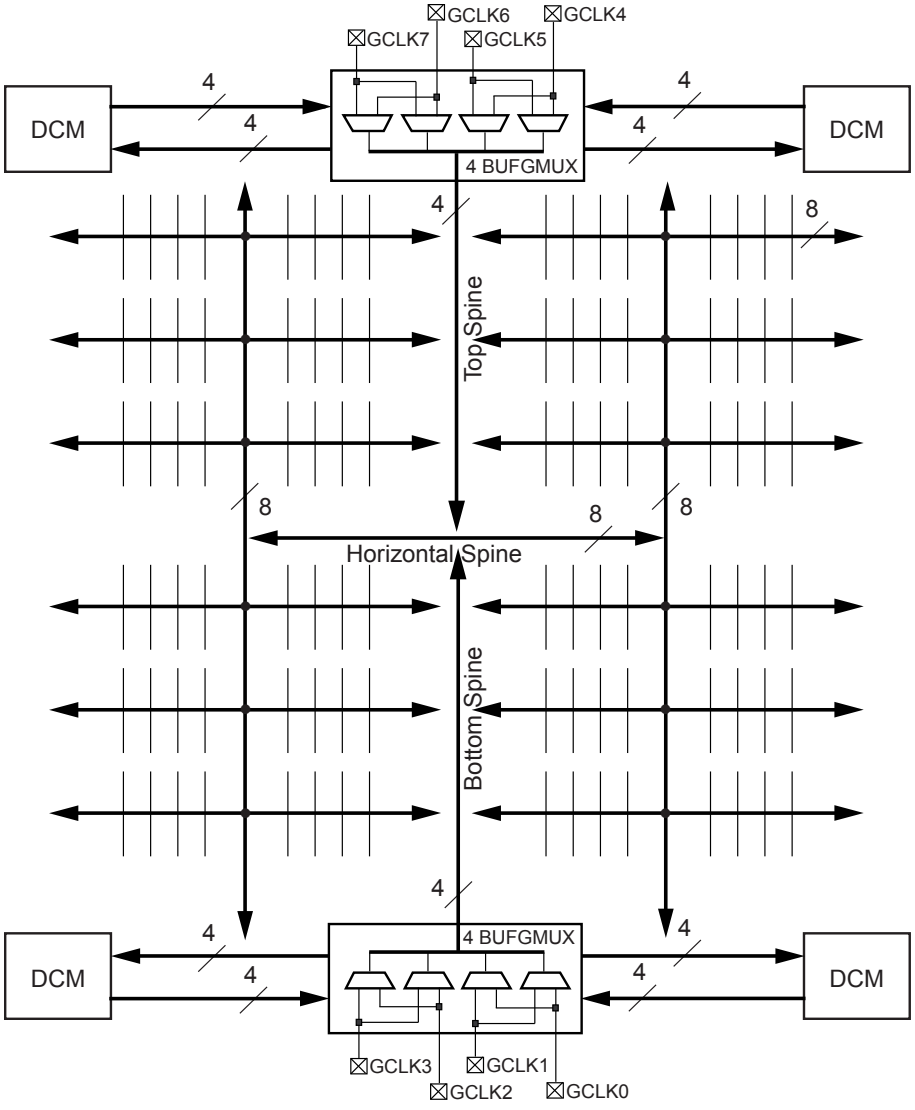
Obrázek 2.11: Double Line [3]



Obrázek 2.12: Direct Line [3]

2.3.7 Distribuční síť hodinového signálu

FPGA Spartan-3 obsahují osm globálních hodinových signálů GCLK0-GCLK7. Ty vstupují do pole multiplexerů kde jsou přepnuty a pokračují do DCM nebo do globální sítě hodinového signálu. V DCM jsou upraveny a vedou zpět do pole multiplexerů a pokračují do globální sítě hodinového signálu. Tato propojovací síť je speciálně vytvořena s minimální kapacitou a zpožděním signálu.

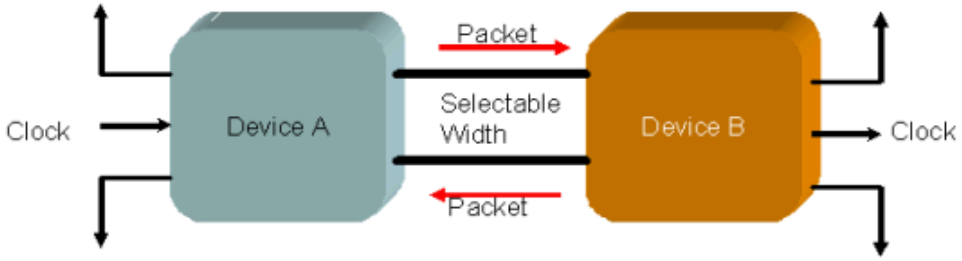


Obrázek 2.13: Globální propojovací síť hodinového signálu [3]

3 SBĚRNICE PCI EXPRESS

Sběrnice PCI-Express (je známá také jako 3GIO = 3rd Generation I/O) je pokračováním implementace sběrnice PCI. Používá existující komunikační standardy – je však založena na mnohem rychlejší sériové komunikaci. Na standardu PCI-Express začala jako první pracovat společnost Intel.

Komunikační kanál mezi dvěma zařízeními sběrnice PCI Express reprezentuje PCI Express Link. Základní link je tvořen ze dvou diferenciálních párů a to vysílajícího a přijímacího označovaného jako Lane. Činnost vysílače i přijímače je na sobě nezávislá a link tvoří plně duplexní komunikační kanál.



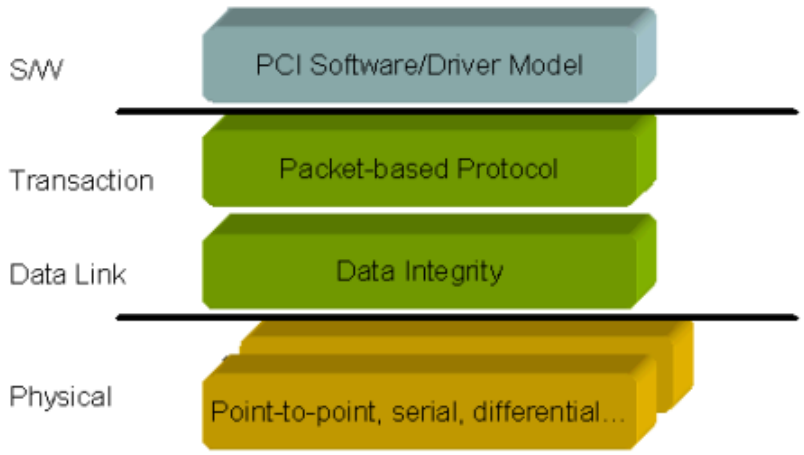
Obrázek 3.1: PCI Express Link [5]

Základní vlastnosti komunikačního kanálu Link:

- Základní link se skládá ze dvou jednosměrných diferenciálních párů v každém směru, reprezentující přijímací a vysílací pár.
- Každý link musí podporovat alespoň jeden Lane. Pro zvýšení přenosové rychlosti je možné využít sdružování lanes do linků v povolené šířce. Obvykle se jedná o hodnoty x1, x2, x4, x8, x12, x16 a x32. Stejná šířka musí byt dodržena jak pro přijímací, tak vysílací část. Během hardwarové inicializace linku se vyjedná pracovní frekvence a počet lanes sestavujících link.

3.1 Vrstvy sběrnice PCI Express

PCI Express je jako protokol založený na vrstvách skládající se z Transaction Layer (transakční vrstva), Data Link Layer (linková vrstva) a Physical Layer (fyzická vrstva). Linková vrstva obsahuje ještě MAC (media access control) vrstvu. Fyzická vrstva je rozdělena na logickou a elektrickou podvrstvu. Názvosloví je podobné jako v ISO-OSI modelu.



Obrázek 3.2: Vrstvy sběrnice PCI Express [5]

Transakční vrstva

Nejvyšší vrstvou architektury je transakční vrstva. Tato vrstva je zodpovědná za zpracování (kompozici a dekompozici) paketů TLP (Transaction Layer Packet). Tyto pakety nesou informaci o typu prováděné operace, jako je čtení, zápis, zpráva nebo operace s IO prostorem. Pakety, které výžadují potvrzení jsou implementovány jako dvě transakce (request/completion). Paket transakční vrstvy se skládá z hlavičky (TLP Header) a vlastních dat. Hlavička se skládá z řídících informací o typu přenosu.

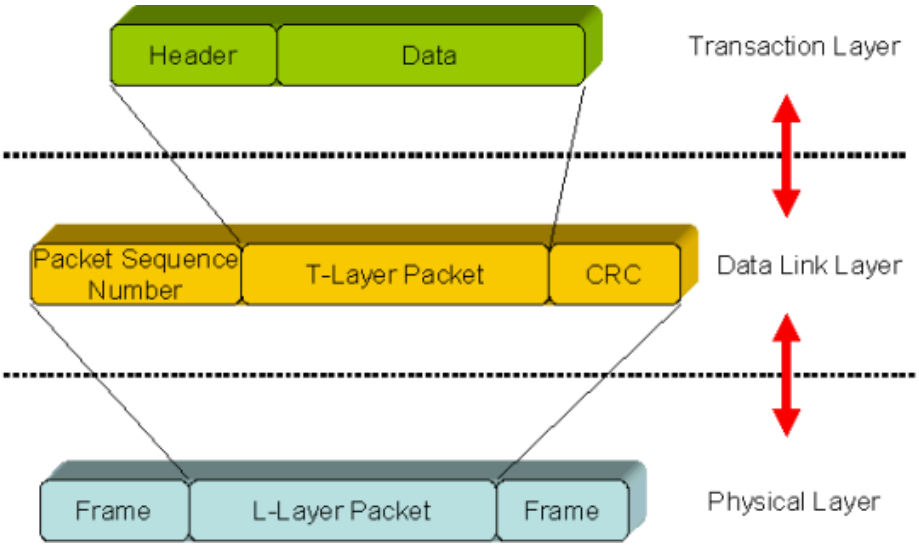
Linková vrstva

Linková vrstva je vložená mezi transakční a fyzickou vrstvu. Jejím úkolem je zajišťování integrity dat, detekce a oprava chyb. Data přijatá z transakční vrstvy jsou opatřena kon-

trolním kódem, identifikačním číslem a poslána do fyzické vrstvy. Naopak data přijatá z fyzické vrstvy jsou otestována, zda neobsahují nějakou chybu a jsou poslána do transakční vrstvy. V případě výskytu chyby, vrstva zajišťuje opakovaný požadavek na data, dokud nejsou požadovaná data přítomná.

Fyzická vrstva

Fyzická vrstva zajišťuje veškeré obvody nutné pro připojení k linku. Jsou to buffery, sério-paralelní a paralelně sériové převodníky a logiku pro inicializaci a udržování spojení na linku (vyjednání přenosové rychlosti, formátu přenosu dat). Paket přijatý z linkové vrstvy je doplněn o kódy začátku konce paketu podobně, jako je tomu u síťových paketů (Ethernet). Dále jsou do paketu doplněny další informace zajišťující synchronizaci. Potom je paket převeden na sériový kód a odvysílán do příslušného Lanu. Přijímací část fyzické vrstvy postupuje opačným způsobem. Dekóduje přijatý paket na řídící kódy a data. Pokud je rámec paketu v pořádku a odpovídá kontrolní součet, je odeslán do linkové vrstvy, dále je také odesláno potvrzení o přijetí dat zdroji transakce. Pokud přijdou data s chybou, odesílá se do zdrojového portu požadavek na opakování transakce.



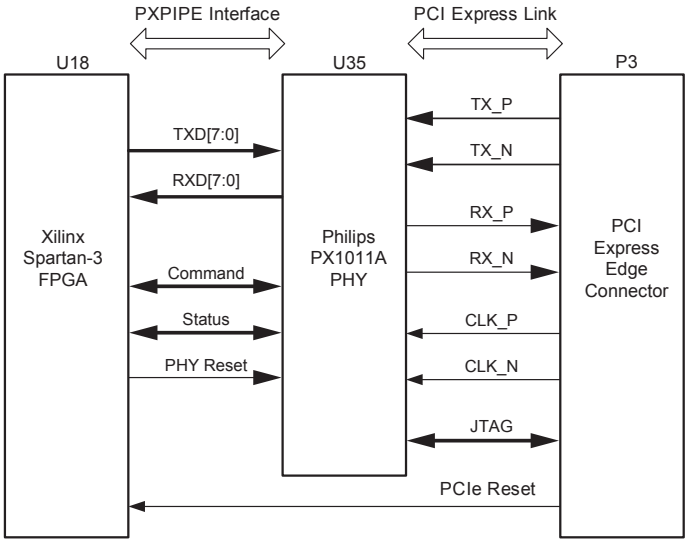
Obrázek 3.3: Grafické znázornění kompozice a dekompozice součástí paketu [5]

3.2 Propojení FPGA a PX1011A PHY

Jak už bylo dříve zmíněno deska je osazena integrovaným obvodem Philips PX1011A PCI Express PHY. Tento obvod nám zprostředkovává převod mezi MAC vrstvou a fyzickou vrstvou PCIe rozhraní. Obsluhuje tedy low level signály a komunikační protokol PCIe. Jsou to funkce jako serializaci a de-serializaci dat, enkódování, detektor příchozích dat, obsluha modulace a demodulace vstupních a výstupních párů. Aby komunikace správně probíhala, tak musí být FPGA nakonfigurováno tak, aby obsahovalo PCIe PIPE (PHY Interface for PCI Express) endpoint core. Je to vlastně obslužný blok obvodů, který umí zpracovat komunikaci na PXPIPE rozhraní.

Philips PX1011A také obsahuje zdroj synchronních hodin pro vysílání a přijímání dat. 8-bitové datové rozhraní pracuje na 250MHz se standardem napěťových hladin SSTL_2 (Stub Series Terminated Logic). Napěťové úrovně SSTL_2 jsou podporovány i na straně Xilinx Spartan-3 FPGA pomocí nastavení vstupních portů IOB. Čerpáno z [1] a [4].

Na Obr. 3.4 je vyobrazen blokový diagram propojení Spartan-3 FPGA a PX1011A PHY.



Obrázek 3.4: Propojení Spartan-3 FPGA a PX1011A PHY [1]

V našem případě je použito propojení PCIe x1. Link tedy obsahuje jeden lane, který se skládá ze dvou párů diferenciálně zapojených vodičů. Podle specifikace může rychlost jednom směru dosáhnou až 2,5Gbitu/s (s kódováním přibližně 250MB/s), přičemž komunikace může nezávisle probíhat oběma směry, rozhraní je plně duplexní. [6]

4 APLIKACE FPGA

Pro implementaci logického analyzátoru byla použita referenční aplikace dodávaná s vývojovou kartou, ve které je vytvořen základní engine pro řízení odesílání a přijímaní informací pomocí PCIe endpoint core. Komunikační RX a TX moduly dokáží obsluhovat TLP pakety s velikostí obsahu (payload) jednoho dvojslova. Přenos je umožněn pouze jako neblokový. Každý přístup tedy vždy přenese přesně 32bitů.

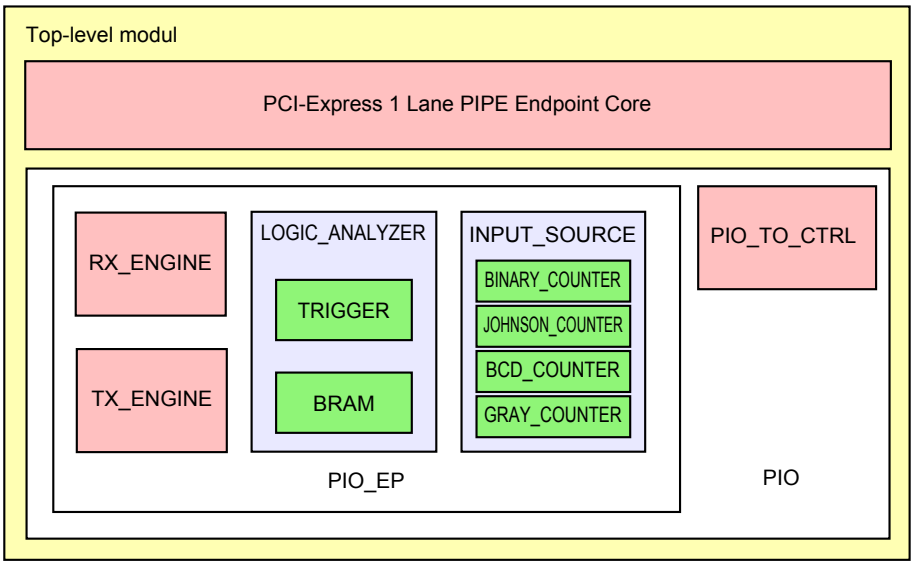
Pro aplikaci logického analyzátoru byl vybrán programovací jazyk Verilog pro jeho podobnost s jazykem C a taky fakt, že referenční aplikace je také vytvořená ve Verilogu. Byl vytvořen modul logického analyzátoru a modul pro generování demostračních dat. Ty jsou použita jako vstupní data do logického analyzátoru. Generované jsou pomocí čtyř druhů čítačů: binární, BCD, Grayův a Johnsonův.

4.1 Hierarchie modulů

V top-level modulu je umístěna instance modulu PCI-Express 1 Lane PIPE Endpoint Core. Pro uživatele vytváří transakční rozhraní nad PCI-Express sběrnicí. Tento modul je vygenerován pomocí nástroje CORE Generátor ze softwarového balíku ISE (viz 4.3). Vstupy a výstupy jsou namapované IO piny v souboru `xilinx_1_lane_epipe_ep-XC3S1000-FG676-4.ucf` a přesně definované rozsáhlé rozhraní pro aplikační logiku. Modul se pro uživatele jeví jako tzv. blackbox a jeho vnitřní algoritmy nezměníme.

Na rozhraní Endpoint Core jsou napojeny moduly PIO, PIO_EP. Jsou to komponenty, které odlehčují níže položeným modulům komunikaci s Endpoint Core a rozdělují kompetence. Použití rozdělení kompetencí umožňuje vytvářet jednoduší a specifičtější moduly, což vede na přehlednější vývoj. Modul PIO_TO_CTRL (Programmable Input Output Turn Off Control) kontroluje zda jsou přenesena všechny data a jednotka se může vypnout. Moduly RX s TX ENGINE sestavují hlavičku a data pro transakční rozhraní Endpoint Core.

Funkce logického analyzátoru je umístěna do modulu LOGIC_ANALYZER. Vstupují do něj 16-ti bitová data z vybraného čítače modulu INPUT_SOURCE. Analyzátor je tedy vytvořen se 16 vstupními kanály. Tyto data jsou cyklicky ukládány do dvouportové blokové RAM a současně přivedeny do modulu TRIGGER, kde je zjištěno zda je

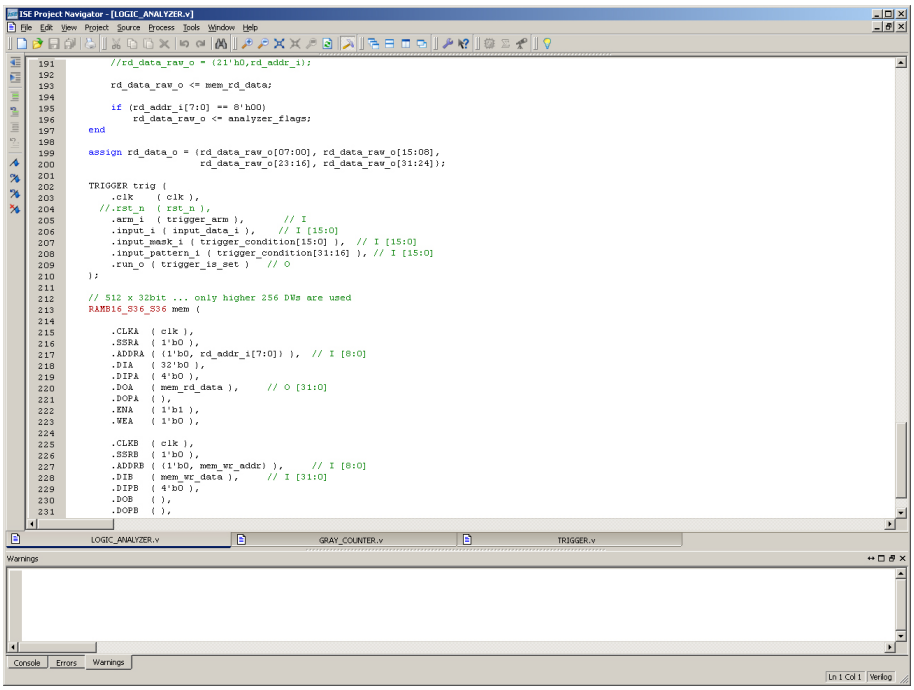


Obrázek 4.1: Zjednodušená hierarchie modulů logického analyzéru

splněna triggrovací podmínka (viz 4.5). Bloková RAM je použita s bitovou šířkou 32 bitů a velikostí 2kB. Počet uložitelných vzorků je dán následujícími skutečnostmi. V systému má karta mapován 1kB paměťový prostor, využijeme tedy pouze horní polovinu blokové RAM. Na jednotlivou adresu zapisujeme vždy po dvou vstupních vzorcích. Adresa 0x00h je využita pro specifický stavový registr logického analyzátoru. Maximální počet uložitelných vzorků je tímpádem 511. Tento počet není daný omezením kapacity harwaru, ale použitým návrhem pro demonstrativní implementaci. Limit by šel velmi jednosuše zvýšit použitím většího adresovacího prostoru a zapojením vícero multiplexovaných BRAM.

4.2 ISE WebPack 11.1

Pro vývoj aplikace FPGA bylo použito návrhové prostředí ISE v11.1 od firmy Xilinx. Toto prostředí je uvolněno i ve verzi WebPack, které je zdarma k použití. Protože ISE je produktem firmy Xilinx, tak umí nejenom provést syntézu na obecné bloky, ale také dokáže převést na bloky specifického hradlového pole od téže firmy. Výstupem je tedy bitstream optimalizovaný na rozmístění a propojení. Tento bitstream můžeme nahrát přímo pomocí prgramu iMPACT do hradlového pole, nebo ho zkonvertovat a nahrát do Platform Flash.

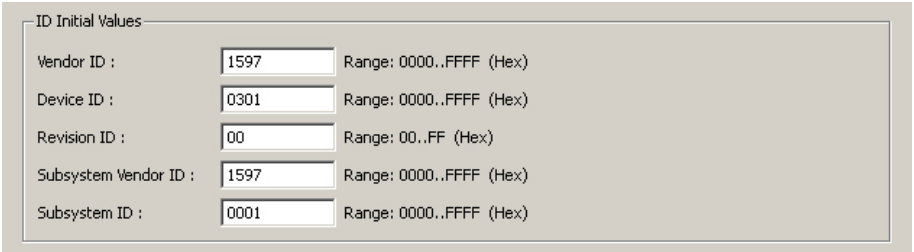


Obrázek 4.2: Vývojové prostředí ISE WebPack 11.1

4.3 CORE Generator

CORE Generátor je jedna z součástí vývojového systému ISE. Obsahuje velké množství předpřipravených bloků. Jejich vložením do návrhu můžeme okamžitě získat potřebnou funkcionalitu. Jedná se o bloky matematických funkcí, pamětí, DSP a sběrnic.

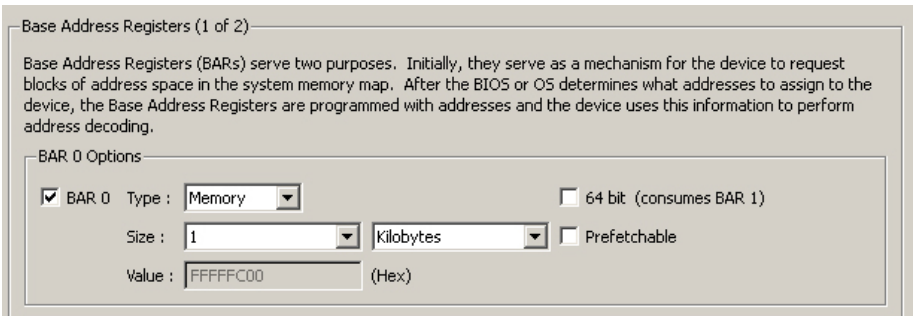
V našem návrhu logického analyzátoru je použit právě jeden takový blok. Jedná se o PCI Express PIPE verze 1.7. Blok je propojen s externím integrovaným obvodem Philips PX1011A, jehož signály spracovává a uživateli nabízí transakční rozhraní sběrnice PCI-Express. Pro vygenerování tohoto bloku pro použití v hardware je potřeba minimálně evaluation licence. Tu je zapotřebí získat z webu firmy Xilinx.



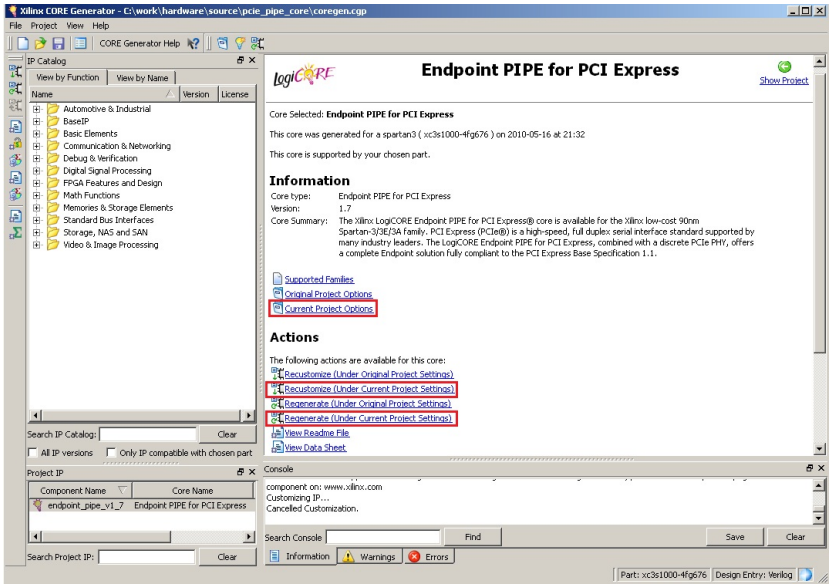
Obrázek 4.3: Nastavení identifikace karty

Při generování core volíme tak, jak chceme, aby se karta identifikovala v systému. Tato situace je zachycena na obr. 4.3. Dva hlavní parametry jsou `Vendor ID` a `Device ID`. `Vendor ID` je identifikátor výrobce zařízení a je přidělován společenstvím skupiny PCI-SIG. `Device ID` je identifikátor zařízení a je volen výrobcem. Pro logický analyzátor jsou identifikátory (`Vendor ID=0x1597; Device ID=0x0301`) přeneseny z referenční aplikace pro vývojovou kartu. Zmenšíme tak riziko případné kolize.

Na obr. 4.4 je zachyceno nastavování mapování adresovacích prostorů. V našem případě vyhrazujeme 1kB adresovacího prostoru typu memory. Nastavení prefetchable, určuje zda můžou být data načítány i bez přímého uživatelova požadavku do mezipaměti. Pokud se data mění na základě čtení, tak nesmí mít adresovací prostor příznak nastaven.



Obrázek 4.4: Vývojové prostředí ISE WebPack 11.1



Obrázek 4.5: Hlavní okno CORE Generátoru

4.4 Stavové registry

Pro komunikaci se softwarem na hostitelské počítači je zapotřebí přenášet kromě vzorků i informace a příkazy pro řízení analyzátoru. Je k tomu použit 32-bitový registr na adrese 0x00h.

[0x00h] **Horních 16 bitů je určeno ke čtení a mají následující strukturu:**

bit č.	31	30	29 – 25	24	23 – 16
název	trigger_is_set	mem_is_full	volné	addr0_hi	addr0

Význam jednotlivých pozic:

`trigger_is_set` – bit je nastavený v log. 1 při splnění triggerovací podmínky

`mem_is_full` – paměť je zaplněná vzorky a připravena ke čtení

`addr0_hi` – definuje, zda je na `addr0` nultý vzorek v horních nebo dolních 16-ti bitech

`addr0` – adresa nultého vzorku v paměti (0x01 – 0xFF)

[0x00h] **Dolních 16 bitů je určeno k zápisu a mají následující strukturu:**

bit č.	15 – 10	9 – 8	7 – 2	0
název	volné	input_choice_o	volné	trigger_arm

Význam jednotlivých pozic:

`input_choice_o` – dvoubitová hodnota nastavuje který čítač má být použitý jako vstupní

`trigger_arm` – zápisem log. 1 spustí povolí kontrolování triggerovací podmínky

Pro zápis tiggrovacího vzoru a masky je použitá adresa 0x01h. Při čtení je ale navrácen klasický vzorek. Tento register musí být zapsán před bitem `trigger_arm`.

[0x01h] **Při zápisu 32 bitů je použita následující struktura:**

bit č.	31 – 16	15 – 0
název	input_pattern_i	input_mask_i

Význam jednotlivých pozic:

`input_pattern_i` – stav vstupních kanálů, který nastavý trigger

`input_mask_i` – maska vybírá bity, které chceme porovnávat

4.5 Implementace triggeru

Vytvořen je jednoduchý trigger, který porovnává logické úrovně vstupní 16-ti bitové hodnoty s nastavenou šablonou. Pokud chceme porovnávat pouze některé bity, tak můžeme použít masku a do ignorovaných pozic bitů nastavit nulu. Zapnutí triggeru je řízeno vstupním signálem `arm_i`. Pokuď je v logické 1, vezme se vstupní hodnota a provede se bitová operace XOR s nastaveným vzorem. Výsledkem je opět 16-ti bitová hodnota s log. 1 na pozicích, kde se vyskytoval rozdíl. Provedením logického součinu s maskou dosáhneme toho, že pozice, která nás nezajímá je vždy v log. 0. Pokud jsou všechny bity rovny log. 0, nastavíme výstup `run_o`. Dáme tím najevo logickému analyzátoru, provedl jeden necelý cyklus zápisu. Tím že nepřepíšeme celou pamět 511 vzorků, ale pouze např. 501 získáme 10 vzorků před triggerem. Výstup se resetuje, až když je `arm_i` v log. 0.

Příklad jednoduchého modulu triggeru

```
1  module TRIGGER (
2      clk,
3      arm_i,                  // I
4      input_i,                // I [15:0]
5      input_mask_i,           // I [15:0]
6      input_pattern_i ,       // I [15:0]
7
8      run_o                   // O
9  );
10     input                  clk;
11     input                  arm_i;
12     input   [15:0]         input_i;
13     input   [15:0]         input_mask_i;
14     input   [15:0]         input_pattern_i ;
15
16     output reg             run_o;
17
18     always @(posedge clk) begin
19         if (arm_i)
20             run_o <= run_o || ((( input_i ^ input_pattern_i ) & input_mask_i) == 16'b0);
21         else
22             run_o <= 1'b0;
23     end
24 endmodule
```

Definice modulu začíná deklarací vstupních, výstupních sítí (angl. `net`) a jejich šířek. Vstupy jsou implicitně typu `wire`. Výstup si nastavíme na typ `reg`. Rozdílem mezi těmito typy je v možnosti použití sekvenčního anebo kobinačního přiřazení. V cyklicky prováděném bloku `allways` se používá podmínka a tudíž se jedná o sekvenční proces, který proběhne s náběžnou hranou hodinového signálu `clk`.

4.6 Rychlost běhu a využití zdrojů

Při syntéze a sestavování bitstreamu určeného k běhu na FPGA je určena maximální frekvence běhu programu. Pro implementovaný logický analyzér je to 63,299 MHz. Tato frekvence se může porovnat s frekvencí transakčního rozhraní PCIe Core. Ta je pevně dána jako 62,50 MHz. Je vidět, že tyto frekvence si jsou velmi blízké. Tato skutečnost by se dala vysvětlit tím, že časová optimalizace se ukončí v momentu, kdy je požadovaná frekvence dosažena a už není potřeba dále optimalizovat. Kdyby optimalizace selhala, tak nezbývá nic jiného než upravit návrh. Například se mohou rozložit dlouhotrvající sekvenční kroky rozložit do kratších synchronních.

Přehled využitých zdrojů FPGA je vypsán v tabulce 4.1.

Typ bloku	Využito	Celkem	V procentech
DCM	1	4	25%
IOB	34	391	8%
RAMB16	9	24	37%
Slices	4602	7680	59%

Tabulka 4.1: Přehled využitých zdrojů FPGA

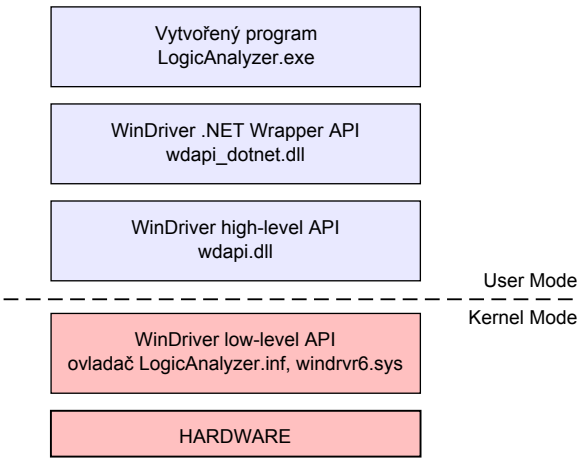
5 ŘÍDÍCÍ SOFTWARE

Pro vývoj uživatelského řídícího programu logického analyzátoru byl zvolen framework .NET a vývojové prostředí Visual Studio 2008 společnosti Microsoft. Proramovacím jazykem byl zvolen C#, který je pro .NET framework jedním z nejrozšířenějších v užití. Zvolením těchto nástrojů se velmi zefektivnil vývoj uživatelského rozhraní a logického analyzátoru jako celku. Důležité je podotknout, že běh programu vyžaduje přítomnost WinDriveru a přítomnost vývojové karty.

5.1 Model komunikace s WinDriver

Při vytváření ovladačů pro hardware počítače je potřeba, aby vývojář znal vnitřní chování systému, uměl naprogramovat ovladač pro kernel mode a teprvé poté vytvořit aplikaci komunikující s ovladačem. Celý vývoj je potřeba neboť uživatelské programy nemají dostatečná oprávnění od operačního systému, aby přistupovaly přímo k hardwaru. Pro zjednodušení tohoto procesu se využije univerzálního ovladače WinDriver firmy Jungo.

Tento ovladač je v plné verzi velmi drahý, a tak použijeme trial WinDriver ve verzi v10.11. Nejdříve je programem DriverWizard vytvořen ovladač a základní diagnostická aplikace ve zvoleném programovacím jazyku. Při programování nad .NET frameworkem se do komunikace s WinDriverem vsouvá další vrstva rozhraní. Model použité komunikace je znázorněn na obr. 5.1. Čerpáno z [10].

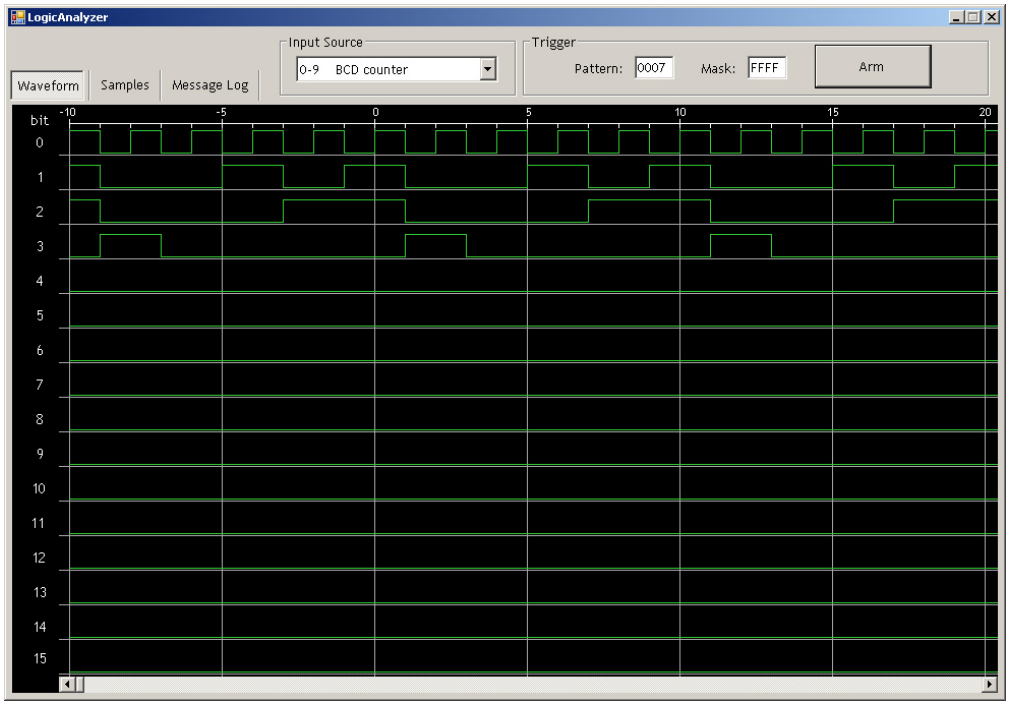


Obrázek 5.1: Model komunikace s použitím WinDriveru

Na vrcholu modelu sedí naše uživatelská aplikace běžící nad .NET frameworkem. Pomocí proxy knihovny `wdapi_dotnet.dll` se přistupuje k systémové knihovně `wdapi.dll`. Toto je již komponenta, která svým API nabízí přístup k informacím, jako jaké PCI zařízení jsou připojeny na sběrnici. Zjišťovat velikosti I/O a paměťových adresovací prostorů. V těchto prostorech číst a zapisovat. Jedinou podmínkou je zavedení ovladače `windriver.sys` k hardware, se kterým chceme komunikovat. Na nejnižší úrovni komunikace již leží hardware počítače.

5.2 Uživatelské rozhraní

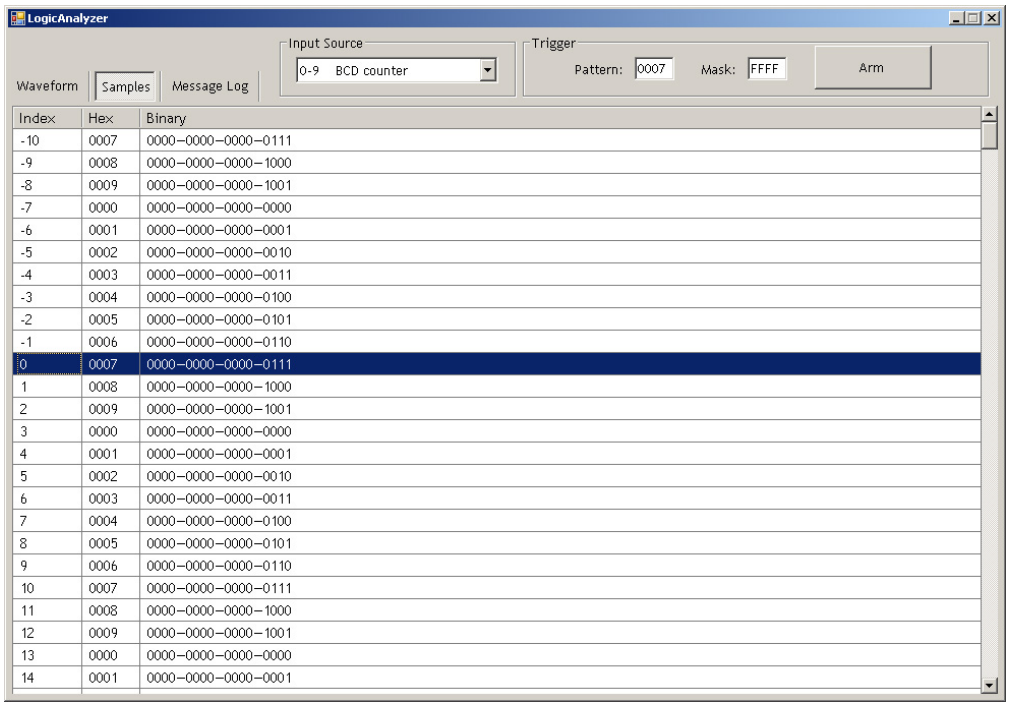
Uživatelské rozhraní programu je vytvořen pomocí WinForms, jednoho z grafických rozhraní, které .NET framework nabízí. Okno se skládá z vrchního pásu, kde je možnost vybrat čítač, který bude generovat vstupní data, předlohu a masku pro trigger. Parametry triggeru se zadávají jako 16-ti bitová čísla v šestnáctkové soustavě (čtyři znaky 0–F).



Obrázek 5.2: Grafické znázornění zaznamenaných průběhů pro BCD čítač

Při zmáčknutí tlačítka `Arm` se zapíše do bitu `trigger_arm` log. 1, nastavení šablony a masky do příslušných registrů v hradlovém poli (viz 4.4). V tomto momentě se také nastaví příznak, že se má kontrolovat bit `mem_is_full`. Tímto příznakem je probuzeno vlánko běžící na pozadí aplikace. Registr `mem_is_full` je cyklicky každou sekundu vyčítán (angl. je tato metoda označována jako pooling). V okamžiku načtení log. 1 je vlákno uspáno a je načten obsah paměti blokové ram hradlového pole. Adresovací rozsah pro vyčtení vzorků je 0x004h až 0x3FFh. Následně jsou data zobrazeny a je zapsána log. 0 do registru `trigger_arm`. Triggrovací podmínka je resetována a logický analyzér je připraven k dalšímu použití.

Vlánkno běžící na pozadí programu a synchronizační události jsou nutné, aby při cyklickém kontrolování registru `mem_is_full` nedošlo k zamrznutí GUI.



Obrázek 5.3: Tabulka zaznamenaných vzorků pro BCD čítač

6 ZÁVĚR

Úvod práce se zabývá popisem základních principů logického analyzátoru. Je zde vysvětlen základní princip triggeru a nastíněny stavové a časové analýzy. Dále je popsáno existující řešení v podobě modůlů ChipScope ILA od firmy Xilinx. Rozbor komunikačního řetězce a hiearchie modulů načrtla podobu vlastní implementace logického analyzéru.

Další část je věnována hardwarovým prostředkům. Jako vývojová platforma byla zvolena karta s hradlovým polem z rodiny Xilinx Spartan-3. U vývojové karty jsou vhodně vybrány režimy napájení a konfigurace. Toto společně s podrobným popisem architektury rodiny Spartan-3 vytvořilo dobrý základ pro vývoj s HDL jazykem a prochopení vnitřních mechanismů hradlových polí.

Pro seznámení se sběrnicí PCI-Express jsou definovány základní pojmy a vysvětleny úlohy jednotlivých vstev. Celkový pohled na rozhraní komunikace nám vyjasní vysvětlení účelu integrovaného obvodu Philips PX1011A.

Další dvě kapitoly již popisují samotnou implementaci logického analyzátoru. Nejdříve bylo nakonfigurováno a vygenerováno PCI-Express Endpoint core. Byl použit 1 kB adresovací paměťový prostor. Analyzátor má 16 vstupních kanálů a podporuje trigger reagující na logické úrovně. Vzorky jsou cyklicky ukládány do blokové ram s celkovou kapacitou 511 vzorků. Pro ověření správného chodu uživatelské aplikace a programu uvnitř FPGA byly vytvořeny čtyři čítače. Na jejich známých posloupnostech se dá jednoduše ověřit funkčnost triggerua záznamu vzorků. Vlastním testováním byla plně ověřena. Myslím si tedy, že zadání práce bylo splněno.

Pokračování práce by mělo být zaměřeno na vytvoření pokročilých triggerovacích podmínek. Dále by bylo vhodné upravit hierarchii aby se daly analyzovat signály v top-level modulu. Inspirací může být například hierarchie použitá u analyzátoru ChipScope.

LITERATURA

[1] Xilinx, Inc. *Xilinx UG256, Spartan-3 Starter Kit Board for PCI-Express User Guide v1.3* [online]. UG256 May 23, 2007 [cit. květen 2010]. Dostupné z URL: <http://www.xilinx.com/support/documentation/boards_and_kits/ug256.pdf>.

[2] Xilinx, Inc. *Xilinx UG331, Spartan-3 Generation FPGA User Guide* [online]. UG331 (v1.6) December 3, 2009 [cit. květen 2010]. Dostupné z URL: <http://www.xilinx.com/support/documentation/user_guides/ug331.pdf>.

[3] Xilinx, Inc. *Xilinx DS099, Spartan-3 FPGA Family data sheet* [online]. DS099 December 4, 2009 [cit. květen 2010]. Dostupné z URL: <http://www.xilinx.com/support/documentation/data_sheets/ds099.pdf>.

[4] Philips Semiconductors *PX1011A/PX1012A PCI Express stand-alone X1 PHY* [online]. Rev. 02 – 18 May 2006 [cit. květen 2010]. Dostupné z URL: <http://ics.nxp.com/products/px/datasheet/px1011a.px1012a.pdf>.

[5] National Instruments *PCI Express – An Overview of the PCI Express Standard* [online]. 13. 8. 2009 [cit. květen 2010]. Dostupné z URL: <http://zone.ni.com/devzone/cda/tut/p/id/3767>.

[6] VALACH, Soběslav. *Technologie: Sběrnice PCI Express* [online] 18. 5. 2005 [cit. květen 2010]. Dostupné z URL: <http://www.svethardware.cz/art_doc-66405A60712A7A6CC12570000046F535.html>.

[7] Tektronix, Inc. *Your Guide to Selecting the Right Logic Analyzer* [online] 3. 2. 2009 [cit. květen 2010]. Dostupné z URL: <http://www2.tek.com/cmswpt/pidetails.lotr?ct=PI&cs=pse&ci=4440&lc=EN>.

[8] Xilinx, Inc. *Xilinx UG029, ChipScope Pro 12.1 Software and Cores* [online]. UG029 (v12.1) April 19, 2010 [cit. květen 2010]. Dostupné z URL: <http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_1/chipscope_pro_sw_cores_ug029.pdf>.

[9] PECH, Jan. *Nebojte se FPGA* [online]. 5. 1. 2004 [cit. květen 2010]. Dostupné z URL: <http://hw.cz/Teorie-a-praxe/Dokumentace/ART365-Nebojte-se-FPGA.html>.

[10] Jungo Software Technologies Ltd. *WinDriver^{TM} PCI/ISA/CardBus User's Manual* [online]. Version 10.1.1, 17. 1. 2010 [cit. květen 2010]. Dostupné z URL: <http://www.jungo.com/st/support/documentation/windriver/1011/wdpci_man.pdf>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

ASIC – Application Specific Integrated Circuit

CLB – Configurable Logic Block

DCM – Digital Clock Manager

DDL – Dekay-Locked Loop

DDR – Double Data Rate

FPGA – Field-Programmable Gate Array

GUI – Graphical User Interface

HDL – Hardware Description Language

I/O – Inputs and Outputs

IEEE – Institute of Electrical and Electronics Engineers

IOB – Input Output Block

IP – Intellectual property

ISE – Integrated Software Environment

JTAG – Joint Test Advisory Group

LUT – Look Up Table

MAC – Media Access Control

PCI Express – Peripheral Component Interconnect Express

PHY – PHYsical layer

PIPE – PHY Interface for the PCI Express

SSTL_2 – Stub Series Terminated Logic for 2.5 Volts

SEZNAM PŘÍLOH

PŘÍLOHY

Příloha 1

CD vloženo do kapsy výtisku práce. Obsahuje tuto práci v elektronické verzi, zdrojové kódy aplikace pro FPGA a grafického rozhraní.

Příloha 2