Sondy pro monitorování provozu

FIT VUT Technický report

Jan Kořenek, Pavol Korček, Jan Kaštil

Technický report č. FIT-TR-2011-009
Fakulta informačních technologií, Vysoké učení technické v Brně

Poslední změna: 9. prosince 2011

Sondy pro monitorování provozu

Jan Kořenek, Pavol Korček a Jan Kaštil

Fakulta informačních technologií
Vysoké učení technické v Brně
Božetěchova 1/2, 612 66 Brno
{korenek,ikorcek,ikastil}@fit.vutbr.cz

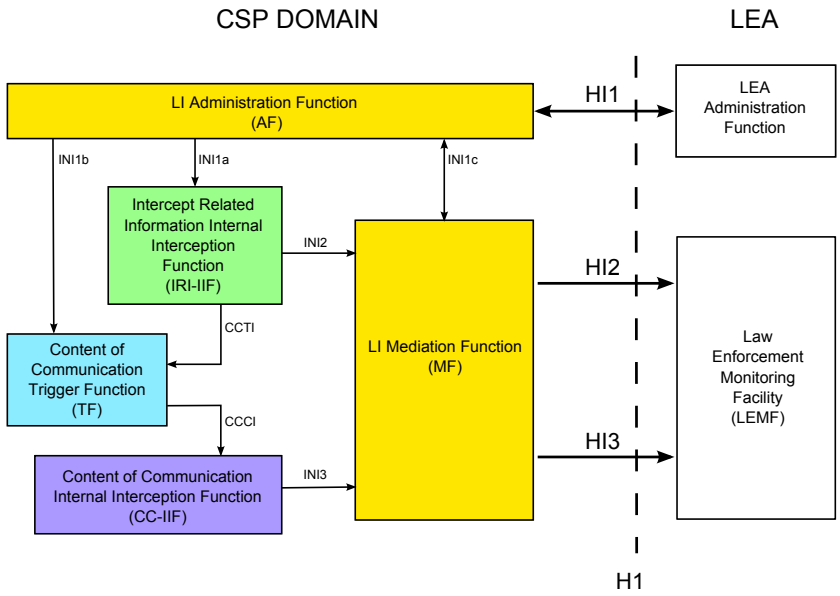
Abstrakt Technický report se zabývá návrhem vysokorychlostní sondy a mikrosondy, které jsou určeny pro sběr síťových dat v systémech zaměřených na zákonné odposlechy. Vysokorychlostní sonda je určena do sítí velkých ISP s propustnosti v řádech desítek gigabitů, zatímco mikrosonda je navržena pro menší koncové sítě s propustností v řádu jednotek gigabitů. V systémech zaměřených na zákonné odposlechy je nutné zajistit zachycení požadovaného provozu beze ztráty jediného paketu, což klade značné nároky na konstrukci sond. Při návrhu obou sond byla proto použita hardwarová akcelerace filtrace síťového provozu s využitím technologie FPGA. Díky hardwarové akceleraci bude možné dosáhnout nejen zpracování síťového provozu na plné propustnosti linky, ale v případě mikrosondy i malých rozměrů a malé spotřeby zařízení. Návrh architektury vysokorychlostní sondy vychází z vlastností platformy NetCOPE a akcelerační karty COMBO. V případě mikrosondy byla i na základě testu dostupných platforem navržena a vytvořena vlastní hardwarová deska.

1 Úvod

Zajištění bezpečnosti současných sítí vyžaduje neustálé monitorování síťového provozu s cílem získat relevantní informace pro odhalování kybernetických hrozeb. Kromě monitorováním síťového provozu je potřeba umožnit také odposlech uživatelů tak, aby bylo možné získat důkazní materiál pro objasnění trestných činů. Cílem projektu *Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace* je proto vytvořit prototypy zařízení, které umožní nejen monitorování sítě, ale i zákonné odposlechy v prostředí velkých i malých poskytovatelů Internetu (ISP, Internet Service Provider) s podporou protokolu IPv6.

Na obrázku 1 je zachycena struktura systému pro zákonné odposlechy (LI, Lawful Interception) podle evropských norem ETSI [5]. Základem systému je několik vzájemně propojených funkcí, které mohou být realizovány pomocí samostatných zařízení. Případně může být více funkcí sloučeno do jednoho zařízení. Sběr dat ze sítě zajišťuje CC (CC, Content of Comunication) funkce, která bývá nejčastěji realizována prostřednictvím specializovaných hardwarových sond. Sondy posílají odposlouchávaný provoz do mediační funkce (MF, Mediation Function), která získaná data transformuje do HI3 rozhraní a přenáší

do bezpečnostní agentury (LEA, Law Enforcement Agency). Současně mediační funkce posílá do agentury ve formě HI2 rozhraní informace z IRI (IRI, Intercept Related Information) funkce, která poskytuje doplňující informace k jednotlivým odposlechům. Například informace o přihlášení nebo odhlášení odposlouchávaného uživatele nebo informace o změně přidělené IP adresy. Celé řízení systému pro zákonné odposlechy pak zajišťuje administrační funkce (AF, Administration Funtion), se kterou komunikuje LEA prostřednictvím HI1 rozhraní.



Obrázek 1. Architektura systému pro zákonné odposlechy podle norem ETSI.

Pro monitorování sítě a sběr dat je nutné využít sondy, které jsou schopny zajistit odposlech požadovaného provozu. Aby mohly sondy využít orgány činné v trestním řízení, je nutné zajistit zaznamenání veškeré komunikace beze ztráty paketu. V opačném případě by mohly být získané informace zpochybněny a nebyly by použity při vyšetřování. Proto je nutné realizovat sondy jako speciální hardwarová zařízení, která pracují na rychlosti vstupních linek beze ztráty jediného paketu.

Tato technická zpráva je zaměřena na návrh vysokorychlostní sondy a mikrosondy (dále uSonda), které jsou vyvíjeny v rámci projektu *Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace*. Vysokorychlostní sonda je určena do sítě velkých ISP s propustnosti v řádech desítek gigabitů, zatímco uSonda je navržena pro menší koncové sítě s propustností v řádu jednotek gigabitů. Dokument je členěn celkem do 3 kapitol. Na úvod navazuje kapitola popisující vývojovou platformu a architekturu vysokorychlostní sondy. Ve

třetí kapitole je pak popsána hardwarová platforma uSondy a první architektura firmware pro její FPGA.

2 Vysokorychlostní sonda

Vysokorychlostní sonda je určena pro nasazení k velkým ISP a na páteřní linky, jejichž přenosová rychlost je velmi vysoká. Sledování a filtrování komunikace na takovéto rychlosti je výpočetně velmi náročná úloha. Dnešní počítače v kombinaci s dostupnými síťovými kartami neumožňují konstrukci sond, které by zajistily filtraci zájmového provozu beze ztráty jediného paketu pro zatížené 10 Gb/s linky. Problémem je nejen nízká přenosová rychlost síťových karet, ale i výpočetní výkon současných procesorů. Vysokorychlostní sonda musí být proto postavena na kartách umožňující hardwarovou akceleraci zpracování síťového provozu tak, aby bylo možné zaznamenávat veškerou komunikaci odposlouchávaných cílů.

2.1 Akcelerační karta a platforma NetCOPE

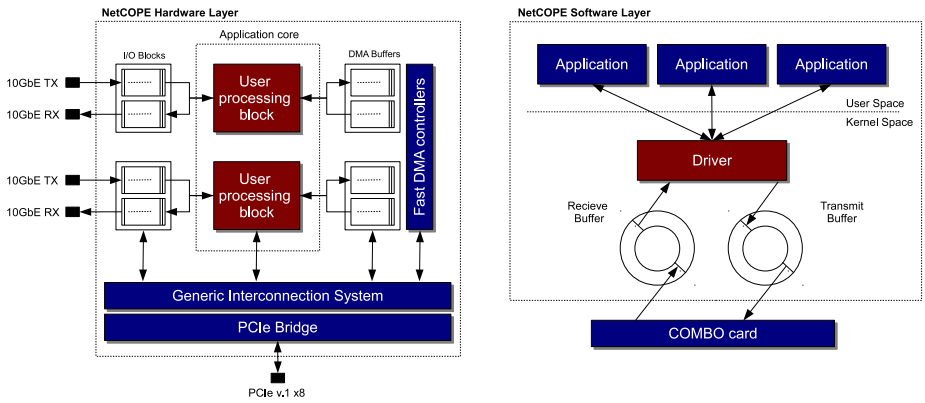
Pro realizaci vysokorychlostní sondy byly zvoleny akcelerační karty COMBO [8], které byly vyvinuty v rámci spolupráce FIT VUT se sdružením CESNET [1]. COMBO karta je zachycena na obrázku 2. Je osazena výkonným FPGA Virtex-5, rychlou pamětí QDR SRAM, slotem pro DRAM s kapacitou až 2 GB, dvěma 10 Gb/s síťovými porty a rychlým PCI express rozhraním (PCIe v.1 x8). Díky výkonnému FPGA je možné kartu využít pro hardwarovou akceleraci časově kritických úloh. Umožňuje tak konstrukci zařízení pro vysokorychlostní sítě, které i pro propustnosti v řádech 10 Gb/s umožní zpracovat veškerý síťový provoz bez ztráty paketu. Pro vývoj sondy byly zakoupeny celkem 3 karty včetně vývojových serverů a platformy NetCOPE [15,12], která umožňuje rychlý vývoj hardwarově akcelerovaných síťových aplikací.

Obrázek 2. Karta COMBO se dvěma 10 Gb/s porty.

Hardwarová akcelerace je většinou rozdělena na akcelerační jádro a řídicí software. Akcelerační jádro je umístěno do FPGA a implementuje časově kritické části aplikace, jako je extrakce položek z hlaviček paketů, klasifikace paketů nebo rychlé hledání regulárních výrazů. Řídicí software zajišťuje zejména správu a řízení akceleračního jádra. V případě vysokorychlostní sondy se předpokládá zejména hardwarová akcelerace filtrace zájmového provozu a řídicí software pro konfiguraci filtru a komunikaci s mediačním zařízením.

Platforma NetCOPE poskytuje prostředky pro zjednodušení vývoje aplikačního jádra, ale i řídicího software. Současně přesně definuje vzájemná rozhraní mez hardware a software. Architektura platformy je zachycena na obrázku 3. Kromě aplikačního jádra se skládá z I/O bloků, propojovacího systému a rychlých DMA řadičů.

- **I/O blok** implementuje příjem a vysílání paketů na plné rychlosti linky podle standardu IEEE 802.3.
- **Propojovací systém (Generic Interconnection System)** umožňuje rychlou komunikaci mezi komponenty v FPGA a systémovou sběrnicí. Podporuje režimy bus master a slave. V režimu slave zpracovává transakce systémové sběrnice a posílá je jednotlivým komponentám v FPGA. V režimu bus master umožňuje komponentám iniciovat na systémové sběrnici čtecí a zápisové transakce a přistupovat tak zcela autonomně do paměti hostitelského počítače.
- **Rychlé DMA řadiče** zajišťují rychlé přenosy mezi akceleračním jádrem a pamětí hostitelského počítače.



Obrázek 3. Architektura platformy NetCOPE.

Díky rychlým DMA přenosům poskytuje platforma NetCOPE možnost efektivně rozdělit činnost zařízení mezi hardwarové a softwarové prostředky. Je tak

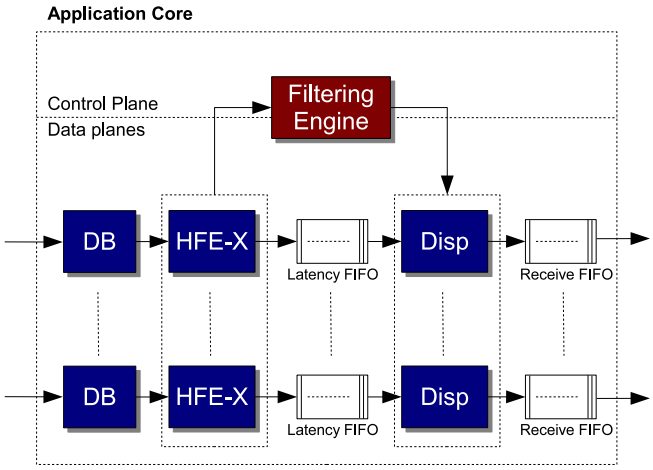
možné akcelerovat pouze časově kritické části aplikace a šetřit plochu na čipu. DMA přenosy jsou realizovány prostřednictvím datových přenosů mezi dvojicí kruhových bufferů. Jeden kruhový buffer je umístěn v paměti hostitelského počítače a poskytuje data softwarové aplikace nebo vláknům. Druhý buffer je implementovaný v FPGA na akcelerační kartě, kde poskytuje data akceleračnímu jádru. Buffer v paměti je fyzicky rozdělen do stránek o velikosti 4 kB, které jsou souvisle mapovány do adresového prostoru aplikace pomocí systémového volání `nmap()`. Součásti platformy NetCOPE jsou ovladače a knihovny, které umožňují aplikaci jednoduše přistupovat k datům přenášených do nebo z akcelerační karty. Kruhové buffery jsou v návaznosti na aplikaci a akcelerační kartu zachyceny na obrázku 3.

2.2 Architektura

Architektura vysokorychlostní sondy vychází z vlastností akceleračních karet COMBO a platformy NetCOPE. Úloha odposlechu zájmového provozu je rozdělena na hardwarovou a softwarovou část. V hardware je realizována časově kritická operace filtrace síťového provozu, zatímco software zajišťuje management filtru a komunikaci s mediační a administrační funkcí LI systému. Je tak využita rychlost hardware a zároveň i flexibilita software v duchu myšlenky HW&SW codesign. Díky vysokým přenosovým rychlostem mezi akcelerační kartou a hostitelským počítačem je navíc u navrženého rozdělení úlohy mezi hardware a software možné vybírat ze sítě i relativně velký objem zájmového provozu.

Aby filtrace provozu byla prováděna na rychlosti linky i pro multi-gigabitové propustnosti a nedocházelo k nekontrolované ztrátě paketů, je nutné věnovat zvláštní pozornost návrhu hardwarové architektury firmware akcelerační karty. Na základě analýzy problematiky zákonných odposlechů byla proto navržena architektura znázorněna na obrázku 4.

Architektura firmware je založena na zřetězené lince procesních jednotek. Pro každý vstupní port je přijetí paketu zabezpečeno platformou NetCOPE, která jej následně pošle do jednotky zpožďovacího bufferu (DB, Delay Buffer). DB realizuje prostřednictvím externí paměti definované zpoždění síťového provozu. Z jednotky DB pokračují pakety do jednotky HFE-X (HFE-X, Header Field Extraction XML), kde jsou z hlaviček paketu získány položky potřebné k provedení filtrace. Jednotka HFE-X současně rozděluje paketový tok na datovou a řídicí cestu. Extrahované položky jsou posílány řídicí cestou do filtrační jednotky, zatímco pakety pokračují po datové cestě do FIFO paměti, která slouží pouze k překrytí latence výpočtu ve filtrační jednotce. Filtrační jednotka na základě položek z hlaviček paketů rozhodne, jestli se má paket zahodit nebo propustit. Výsledek filtrace je odeslán do jednotky Disp (Disp, Dispatcher), která zajišťuje fyzické zahazování nepotřebných paketů tak, aby bylo možné propagovat dále pouze zájmový provoz. V jednotce Disp se tak opět spojuje datová a řídicí cesta. Nepotřebné pakety jsou zahozený a zájmový provoz je propagován do přijímací FIFO paměti, odkud je DMA přenosy platformy NetCOPE přenášen do paměti hostitelského počítače a po zpracování softwarovou aplikací odesílán na mediační zařízení.



Obrázek 4. Navržená rchitektura firmware vysokorychlostní sondy.

V návrhu architektury systému jsou klíčovými jednotkami zejména zpožďovací buffer (DB), jednotka pro extrakci položek z hlaviček paketů (HFE-X) a jednotka pro filtraci paketů (Filtr). Je proto vhodné uvést charakteristiku uvedených komponent.

Jednotka pro analýzu a extrakci položek z hlaviček paketů - (HFE-X) slouží k analýze a extrakci položek z hlaviček paketů. Implementace jednotky vznikla již v rámci dřívější spolupráce se sdružením CESNET. Je navržena pro zpracování síťového provozu až do rychlosti 40 Gb/s a byla publikována v roce 2009 na konferenci DDECS [9].

Základem jednotky je automat, který analyzuje vstupní data a rozpoznává pozice hlaviček paketů. Automat není vytvářen manuálně, ale je generován na základě definice protokolů popsaných v jazyce XML. Při generování je možné zadat nejen vlastní XML soubor s definicí podporovaných protokolů, ale je možné specifikovat i datovou šířku vstupu, která pak přímo určuje propustnost jednotky. Výsledkem generování je automat, který je optimalizován pro zadanou množinu protokolů a datovou šířku vstupu tak, aby se šetřily zdroje na čipu.

Na základě analýzy vstupních dat a rozpoznání konkrétních protokolů provádí HFE-X extrakci položek z hlavičky paketů. Položky, které mají být extrahovány a poslány do filtrační jednotky je možné specifikovat ve speciálním XML souboru, ze kterého je možné vygenerovat pomocí připravených skriptů správnou konfiguraci jednotky. Konfiguraci extrakční části je možné nastavit nejen při vytváření jednotky, ale i za běhu systému.

Jednotka pro zpoždění vstupního provozu - (DB) má za úkol vytvořit zpoždění vstupního provozu o přesně definovaný čas t_d přičemž samotnou dobu

zpoždění bude možné nastavovat za běhu například změnou obsahu registru. Vstupním i výstupním rozhraním jednotky je tok paketů. V komponentě nedochází k modifikaci obsahu paketů, pouze k zachycení provozu na dobu t_d.

Aby bylo možné realizovat zpoždění síťového provozu, je potřeba pakety přechodně ukládat do paměti. Kapacita paměti pak přímo určuje maximální délku zpoždění, kterou lze s touto pamětí realizovat. Pro realizaci zpoždění 1 s na plně saturované lince s propustností 10 Gb/s je potřeba paměť o velikosti 1,25 GB, pro 1 ms je to pak 1.25 MB. Pro různé paměti dostupné na kartě COMBO tak dostáváme:

- SRAM (8 MB na port): 6.4 ms - paměť QDR SRAM má propustnost 10 Gb/s pro zápis a 10 Gb/s pro čtení s tím, že pro každý port je k dispozici jedna paměť.
- DRAM (1 GB na port): 800 ms - na kartě COMBO je pouze jedna paměť DRAM s teoretickou propustností 32 Gb/s, což nestačí pro zachycen (čtení i zápis) síťového provozu ze dvou plně zatížených 10 Gb/s portů.

Nejdelší doby zpoždění je možné dosáhnout pomocí paměti DRAM. Nicméně paměť DRAM neposkytuje dostatečnou propustnost pro zápis a následně čtení síťového provozu ze dvou plně saturovaných 10 Gb/s linek. Proto budou pro realizaci zpožďovacího bufferu použity paměti QDR SRAM, případně kombinace QDR SRAM a DRAM.

Samotné zpoždění bude realizováno prostřednictvím kruhového bufferu se zápisovým a čtecím ukazovátkem, které jsou od sebe posunuty o velikost odpovídající definovanému zpoždění t_d. Rozestup vzdáleností čtecího a zápisového ukazovátka pro zpoždění t_d je možné vyjádřit v MB jako $D = t_d * 1250$. Aby nebylo potřeba v hardware zbytečně implementovat přepočet, bude změna zpoždění nastavována přímo ve formě rozestupu zápisového a čtecího ukazovátka. Hodnotu rozestupu vypočítá konfigurační software a nahraje výsledek do odpovídajících registrů. Po změně zpoždění je nutné provést inicializaci, jinak mohou být data v paměti špatně interpretována.

Pro kruhový buffer je nutné nějakým způsobem uchovávat informaci o začátcích a koncích paketů. Nejjednodušším způsosobem, jak identifikovat začátek a konec paketů je využít paritní bity paměti, což vede ke snížení kapacity o 1/9. Paritní bity jsou k dispozici u COMBO SRAM i DRAM. Ukládání dat do paritních bitů sebou přináší nutnost použití válcového posouvače (barell shifter). Na 128-bitové sběrnici (10 Gb/s $\approx$ 100MHz) představuje barell shifter nemalé zdroje FPGA, takže se většinou paritní bity pro ukládání paketů nepoužívají. Signalizaci začátku nebo konce paketu pomocí paritních bitů lze tedy chápat spíše jako efektivní využití parity než jako režii.

Protože po startu systému mohou být v paměti uloženy na paritních bitech hodnoty signalizující začátek nebo konec paketu, je potřeba po startu systému počkat dobu odpovídající definované délce zpoždění a až pak začít odesílat pakety z bufferu. V opačném případě mohou být náhodná data v paměti reprezentována jako pakety, které budou zaslány do systému v záporném čase vzhledem ke startu systému. Fáze inicializace by se měla provést při každé změně hodnoty

zpoždění. Fyzicky lze tuto funkci realizovat tak, že stavový automat po resetu zařízení uloží hodnotu zápisového ukazovátka a povolí odesílání paketů až dosáhne čtecí ukazovátko této pozice.

Filtrační jednotka - cílem filtrační jednotky je identifikovat na základě pravidel zájmový provoz a poskytnout informaci, které pakety se mají propagovat do mediačního centra a naopak které pakety se mají v jednotce Disp zahodit. Aby bylo možné identifikovat zájmový provoz musí filtrační jednotka provádět filtraci paketů na základě

- IP adres identifikujících počítače uživatelů,
- TCP/UDP spojení (síťových toků) a
- TCP/UDP portů, které určují důležité služby (SIP, IMAP a další).

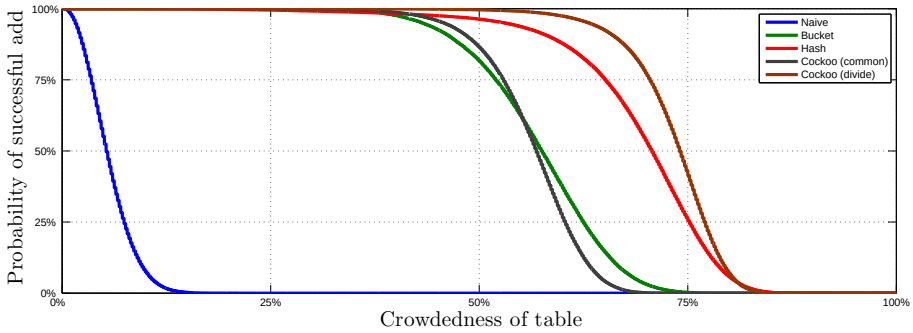
Filtrační pravidla je tak možné rozdělit do tří množin dle rozdělení do uvedených kategorií a pro každou množinu vytvořit samostatný filtr. Za zájmový provoz je pak považován každý paket, který je označen alespoň jedním z těchto tří filtrů. Vlastnosti a struktura pravidel jednotlivých filtrů se značně liší:

- **Filtr IP adres** - v případě filtru IP adres jsou porovnávány zdrojové a cílové IP adresy paketů vůči zadané množině IP adres s tím, že je nutné zajistit podporu protokolů IPv4 i IPv6. V případě obou protokolů se předpokládá velikost množiny pravidel maximálně do 1000 položek. To znamená, že filtr musí podporovat filtraci pro 1000 IPv4 a 1000 IPv6 prefixů adres. Při realizaci filtru se předpokládá, že maximální rozdíl dvou různých délek prefixů bude 8 bitů.
- **Filtr TCP/UDP spojení** - u filtru TCP/UDP spojení není potřeba uvažovat prefixy IP adres. TCP/UDP spojení je identifikováno na základě pětice: zdrojová a cílová IP adresa, zdrojový a cílový port, číslo protokolu. Stačí pouze porovnat uvedené položky z hlavičky paketů s množinou pravidel, přičemž velikost množiny pravidel bude maximálně 1000 položek.
- **Filtr služeb** - filtrace konkrétních služeb (DNS, SIP, SSH a podobně) je možné řešit na základě množiny TCP/UDP portů. Předpokládá se filtrace podle až 64 služeb. V počáteční fázi realizace nepředpokládáme, že by filtr prováděl detekci tunelovaných služeb na základě předem definovaných vzorů chování sítě. Nicméně v pozdějších fází projektu může být tato funkce do architektury doplněna.

Jednou z klíčových vlastností filtru pro zákonné odposlechy je přidávání a odebírání pravidel. Aby byl zachycen veškerý provoz uživatel ihned po přihlášení k rádius serveru, je nutné zajistit přidání filtračního pravidla do karty v řádu jednotek milisekund. Navíc změna jednoho pravidla (přidání i odebrání) nesmí nikterak ovlivnit filtraci provozu podle ostatních pravidel. Konfigurace filtru bude vycházet z obou uvedených předpokladů.

Architektura jednotky musí být navržena tak, aby umožňovala zpracovávat na rychlosti linky provoz až ze dvou 10 Gb/s rozhraní. Vzhledem k předpokládané velikosti množin filtračních pravidel a požadované rychlosti zpracování je

vhodné uchovávat pravidla přímo na čipu (v paměti BlockRAM). Počet položek v jednotlivých množinách bude nastavitelný pomocí generického parametru komponenty tak, aby bylo možné optimalizovat použité prostředky na čipu.



Obrázek 5. Graf využití paměti pro různé hash funkce.

Pro zpracování síťového provozu na rychlosti linky je důležité rychlé nalezení filtračního pravidla. Byla proto provedena analýza algoritmů pro rychlé vyhledání záznamu v zadané množině. Nejrychlejší vyhledání je možné dosáhnout prostřednictvím hash funkce, kdy v jednom kroku (jednom paměťovém přístupu) je možné najít odpovídající filtrační pravidlo. Problém je ale najít hash funkci, která umožňuje vysoký stupeň zaplnění paměti aniž by docházelo ke vzájemným kolizím jednotlivých položek. Provedli jsme proto porovnání různých přístupů používající hash funkce a výsledky vynesli do grafu na obrázku 5, kde na ose x je zaplněnost tabulky v procenttech, a na ose y pravděpodobnost umístění nové položky do paměti. Celkem bylo anaylzováno 5 přístupů:

- **Naivní přístup (Naive):** spočíta se jediná hash funkce a na místo určené hash funkcí se přidá záznam. Pokud je místo již obsazeno, nastává kolize.
- **Bucket:** spočíta se jediná hash funkce, projde se N míst (bucket) bezprostředně navazující na místo určené hash funkcí. Záznam se přidá na první volné místo. V grafu je zvolena velikost bucketu N = 12.
- **Hash:** prochází se N míst v paměti podobně jako v přípdě Bucket přístupu. Nicméně pozice v paměti jsou určeny prostřednictvím dvou hash funkcí. První hash funkce určuje první pozici v paměti. Další pozice v paměti jsou určeny postupným přičítáním druhé hash funkce. V grafu je zvolena hodnota N = 12.
- **Cockoo (common):** Využití principu kukaččí hash funkce, kdy se všechny hash funkce odkazují do společné tabulky. V grafu jsou použity 2 hash funkce.
- **Cockoo (divide):** Využití principu kukaččí hash funkce, kdy se každá hash funkce odkazují jiné části tabulky. V grafu jsou použity 2 hash funkce. To znamená, že paměť je rozdělena na dvě části.

Z obrázku 5 je vidět, že vysokou rychlost vyhledání s malou paměťovou režií poskytuje zejména algoritmus kukaččí hashing (Cockoo hashing) [16], který bude také použít jako základ filtrační jednotky.

3 Mikrosonda (uSonda)

Na rozdíl od vysokorychlostní sondy je mikrosonda (uSonda) určena pro nasazení k menším ISP. Předpokládá se tedy její využití na linkách do rychlosti 1 Gb/s, přičemž se počítá s monitorováním obou směrů komunikace. Cílem uSondy je vyfiltrovat na základě konfigurace provoz odposlouchávaných uživatelů a poslat jej na mediační zařízení, nejlépe zabezpečeně a spolehlivě. V případě výpadku spojení s mediačním zařízením musí uSonda přechodně ukládat veškerý vyfiltrovaný provoz a to do doby, než bude spojení opětovně navázáno. Dočasné ukládání provozu je potřeba i na monitorovaných rozhraních. Podobně jako u vysokorychlostní sondy, je potřeba zpozdit vstupní provoz tak, aby při změně konfigurace nedocházelo ke ztrátě paketů. Například po přihlášení uživatele k rádius serveru nebo při získání nové IP adresy, mohou být odvysílány emaily nebo jiný důležitý provoz uživatele ještě před vložením IP adresy uživatele do konfigurace filtru a uSonda tak nezachytí část požadovaného provozu. S využitím dostatečně velkého zpožďovacího bufferu, umístěného na vstupních rozhraních, můžeme tyto problémy snadno eliminovat. Kromě uvedených funkcí by uSonda měla být snadno konfigurovatelná a mít i relativně nízkou spotřebu, neboť může být v některých případech nasazena do prostředí, které vyžaduje napájení z baterií nebo z Ethernetového kabelu (standard PoE).

Na základě plánovaného využití uSondy jsme definovali několik klíčových požadavků, které je nutné splnit pro reálné nasazení. Základním požadavkem pro uSondu je zajistit zpracování monitorovaného provozu na rychlosti linky beze ztráty paketu i při maximálním zatížení (wire-speed propustnost). Zpracování provozu beze ztráty paketu je důležité nejen k získání veškerého zájmového provozu, ale i pro utajení uSondy v infrastruktuře sítě, zejména pokud je uSonda provozována v tzv. inline režimu, kdy monitorovaná linka prochází skrze uSondu. Dalším klíčovým požadavkem je zajistit při výpadu linky s mediačním zařízením ukládání zájmového provozu na externí záznamové médium tak, aby nedocházelo ke ztrátě dat. Ukládání vyfiltrovaných dat ale klade požadavky nejen na rychlost zápisu na externí médium ale i na rychlost rozhraní mezi uSondou a samotným záznamovým zařízením. Důležité je také zajistit rychlou konfiguraci uSondy zejména v případech, kdy se dynamicky mění požadavky na zájmový provoz. Protože uSonda může být nasazena i ve špatně přístupných podmínkách, je nutné kromě lokální konfigurace zajistit i vzdálený přístup přes síť. Vzdálený přístup sebou ale přináší další požadavky na vlastnosti síťových portů, které kromě monitorování musí být použitelné i pro běžnou komunikaci jako je například navázání TCP spojení nebo přihlášení přes SSH. Klíčovým požadavkem je také nízká spotřeba vycházející z případu použití, kdy je nutné uSondu napájet z baterie nebo s využitím PoE, které mimo jiné specifikuje maximální

dodávaný výkon. Byly specifikovány a analyzovány i další požadavky na hardwarovou platformu uSondy, které lze rozdělit do následujících tří kategorií:

- požadavky vycházející ze specifikace cílů projektu *Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace*,
- obecné požadavky zaměřené hlavně na použitelnost platformy pro vědecké účely a nakonec
- požadavky aplikačně specifické.

Všechny požadavky byly zohledněny při testování deseti vybraných běžně dostupných platforem s cílem vybrat vhodnou hardwarovou platformu pro uSondu. Během testovaní se však ukázalo [10], že ani jedno z testovaných zařízení není schopno splnit všechny klíčové požadavky. Asi největší problém nastal u měření propustnosti a výkonnosti platforem, kde žádné z testovaných zařízení nedosahovalo plné propustnosti linky. Téměř vždy docházelo k zahazovaní paketů i při jednoduchém přeposílání paketů z jednoho portu na druhý. Problém se samozřejmě stupňoval se složitostí zpracování a v některých případech bylo dokonce možné pozorovat chování, kdy na výstupu přicházely pakety mimo pořadí. Takové chování mohlo být způsobeno buď špatnou softwarovou implementací TCP/IP vrstev nebo špatnou implementací priorit správy hardwarových front. I na základě výsledků těchto testů jsme proto přistoupili k návrhu vlastní HW platformy pro uSondu.

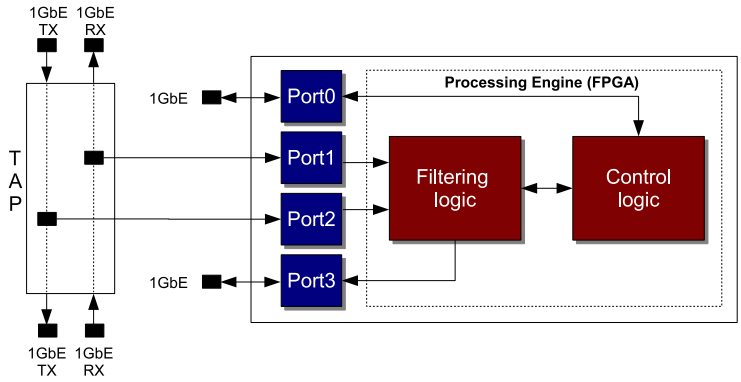
3.1 Návrh HW platformy

Z pohledu návrhu hardwarové platformy pro uSondu je klíčová volba hlavního výpočetního prvku. Platforma by neměla mít jen wire-speed propustnost, jak bylo diskutováno v předchozí kapitole 3, ale i dostatečnou výkonnost pro realizaci filtru. Provedené testy [10] ukázaly, že žádný z testovaných vestavěných procesorů neposkytuje dostatečný výkon. Jako hlavní výpočetní prvek bylo vybráno FPGA, které umožňuje běh softwarových aplikací na vestavěném procesoru MicroBlaze [26] a současně umožňuje akceleraci časově kritických operací v programovatelné logice. Navíc FPGA díky své flexibilitě poskytuje možnost jednoduše přidávat nebo modifikovat funkci zařízení, aniž by bylo potřeba měnit hardware desky. Kvůli odhadu velkosti samotného FPGA byla provedená analýza komponent (tzv. IP cores), které je potřeba integrovat do FPGA. Je to zejména IP core pro realizaci MAC vrstvy Ethernetu (viz dále) spolu s DMA řadiči pro rychlý přenos paketů do paměti, dále pak řadič dynamické paměti, procesor MicroBlaze, který se může s různými konfiguracemi vyskytovat i ve dvou instancích (a přiblížit se tak připravované architektuře ZynQ [3], viz kapitola 3.4), výpočetní jednotky pro filtraci provozu a ostatní komponenty nezbytné pro monitorování síťového provozu v FPGA na plné rychlosti linky. Na základě této analýzy bylo zjištěno, že by měl stačit FPGA čip střední velikosti. Nicméně výsledky nezahrnují další pomocné komponenty, které jsou potřeba například pro zajištění komunikace mezi procesorem a interními nebo externími jednotkami. Bez detailního zapojení není možno u těchto pomocných komponent dopředu

přesně odhadnout požadované množství zdrojů (např. u sběrnicového systému AXI, viz kapitola 3.4). Abychom zajistitli dobrou rozšiřitelnost a zamezili možným problémům s frekvencí implementovaného firmware pro FPGA, byl pro uSondu vybrán největší model ve své řadě (XC6SLX150-3FGG484C) [27]. Při návrhu desky plošných spojů se však bral zřetel na pinovou kompatibilitu tak, aby v budoucnosti bylo možné nahradit FPGA menší a tedy i levnější variantou.

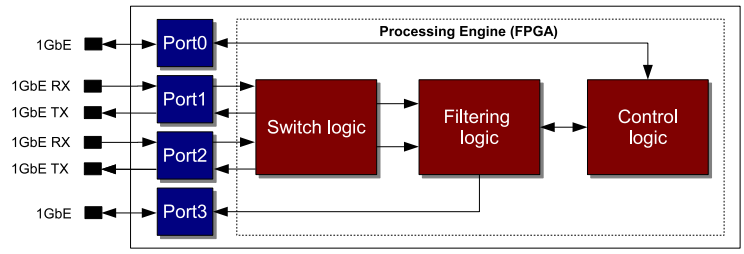
Aby bylo možné monitorovat obousměrně gigabitovou linku a současně bylo možné zajistit vzdálený přístup na uSondu přes síť, jsou k FPGA připojena celkem čtyři síťová rozhraní, každé o rychlosti 1 Gb/s. Dvě z těchto rozhraní jsou určeny pro monitorování RX a TX směru monitorované linky dle obrázků 6 a 7. Dále jedno z rozhraní slouží jako konfigurační (management) port, což umožní i vzdálenou konfiguraci s využitím existující síťové infrastruktury. Na obrázcích 6 a 7 je konfigurace znázorněna samostatným blokem Control logic. Poslední síťové rozhraní je určeno pro zasílaní zájmového provozu pomocí zvoleného protokolu do mediačního centra.

Na fyzické vrstvě pracují všechny čtyři rozhraní se standardem Ethernet a využívají metalické konektory RJ45. Zapojení uSondy do sítě je možné prostřednictvím síťového rozbočovače (TAP) podle obrázku 6. Další možnost je zapojit sondu inline do rozpojené linky podle obrázku 7 a využít pro přeposílání provozu mezi porty logiku FPGA. Ušetří se tím nutnost instalace síťového rozbočovače avšak vznikne potřeba správně přesměrovávat provoz síťových portů, což ukazuje blok Switch logic na obrázku 7.



Obrázek 6. Zapojení uSondy se sítovým rozbočovačem (TAP).

Realizací fyzické vrstvy ethernetového rozhraní mají na starosti čipy (phytry) společnosti Micrel s označením KSZ9021RN [13]. Jde o phytry s nízkým příkonem a zjednodušeným rozhraním RGMII, které proti GMII rozhraní ušetří až polovinu externích pinů čipu FPGA. Na straně čipu FPGA je toto rozhraní zpra-



Obrázek 7. Zapojení uSondy v režimu inline.

cováno pomocí IP core Tri Mode Ethernet MAC komponenty (TEMAC), který je možné získat po zaplacení licenčních poplatků společnosti Xilinx [25].

Pro dočasné ukládaní vstupního, případně výstupního provozu je nutné mít na hardwarovém přípravku dostatečně velkou a rychlou vyrovnávací paměť. Protože FPGA čipy obecně nedisponují dostatečnou kapacitou interní paměti, je potřeba připojit k FPGA externí čip dynamické paměti (DRAM). Paměť DRAM může být určená nejen jako odkládací prostor (buffer) pro vstup i výstup, ale i jako paměť RAM pro procesor MicroBlaze. Abychom dosáhli dostatečné propustnosti a kapacity pro realizaci bufferu a paměti procesoru MicroBlaze, rozhodli jsme se osadit na hardwarovou desku celkem dvě paměti DRAM. Architektura uSondy může dvojici pamětí využít tak, že jak paměť procesoru tak i vstupně/výstupní buffer budou realizovány v samostatných DRAM pamětech, případně mohou obě paměti realizovat buffer i paměť procesoru zároveň. Při tom zvolená kapacita paměti 256 MB by měla být dostatečná i pro realizaci zpoždění provozu na dobu delší než 1 sekunda, a to i při plné saturaci vstupní gigabitové linky a při využití poloviny paměti procesorem MicroBlaze. Na vybraném typu FPGA jsou k dispozici dva pomocné obvody (MCB) [23], které umožňují zabezpečit řízení signálů u obou paměti. To výrazně zjednodušuje výslednou implementaci a snižuje spotřebu zdrojů v FPGA. Na druhou stranu MCB je možné použít jen s vybranými typy pamětí, přesněji s pamětí DRAM typu DDR, DDR2, DDR3 nebo LPDDR. Při výběru typu paměti jsme se zaměřili nejen na kapacitu, ale i na propustnost mezi pamětí a FPGA. Provedli jsme studii propustnosti a zjistili, že propustnost závisí nejen na typu paměti, ale i na režimu napájení (normální a rozšířený), frekvenci rozhraní (250 MHz, 300 MHz, 333 MHz nebo 400 MHz), speedgrade FPGA (-2, -3), konfiguraci paměti s ohledem na počet banků (x4, x8, x16), konfigurovatelné arbitraci na pěti 32-bitových fyzických portech MCB a samozřejmě uživatelské frekvenci. Na základě této studie a na základě požadavku zajistit nízkou spotřebu výsledného zařízení byla vybrána paměť typu DDR3 (konkrétně Micron MT41J256M8HX-187E:D) [14] o maximální frekvenci 400 MHz (resp. 800 MHz DDR) a konfiguraci x8. Maximální teoretická propustnost na rozhraní této paměti je 5336 Mb/s při normálním módu a až 6400 Mb/s při rozšířeném módu napájení FPGA.

Jedním z požadavků na uSondu je umožnit lokální záznam velkého množství paketů nebo jiných dat. Proto bylo na uSondu přidáno rozhraní USB, ke kterému je možné připojit rychlý externí disk. Aby bylo možné využít potenciálu nejrychlejších disků na trhu, kde se zápisové rychlosti média pohybují v rozmezí okolo 1,5 Gb/s, je nutné využít nejnovější z dostupných specifikací rozhraní a to USB 3.0 s teoretickou propustností až 5 Gb/s. Existuje však nemalý problém s dostupností tohoto řešení pro FPGA. Implementace řadiče pro rozhraní s USB procesorem v FPGA by byla náročná nejen z pohledu doby vývoje, ale i z pohledu spotřebovaných zdrojů na čipu. Proto byl experimentálně vybrán čip EZ-USB FX3 od společnosti Cypress [2], který řeší fyzickou vrstvu USB 3.0 rozhraní a jako jediný svého druhu poskytuje vhodné rozhraní na FPGA. Tím je 32-bitové datové rozhraní s frekvencí 100 MHz, na které jsme ve VHDL vyvinuli řídicí komponenty do FPGA. Ze strany USB 3.0 rozhraní pak poskytuje zvolený čip následující funkcionalitu:

- USB 3.0 device (Super Speed),
- USB 2.0/3.0 device (High Speed),
- USB 2.0/3.0 device (Full Speed),
- USB OTG 2.0 Dual-role (High Speed),
- USB OTG 2.0 Dual-role (Full Speed) a
- USB OTG 2.0 Host (Low Speed).

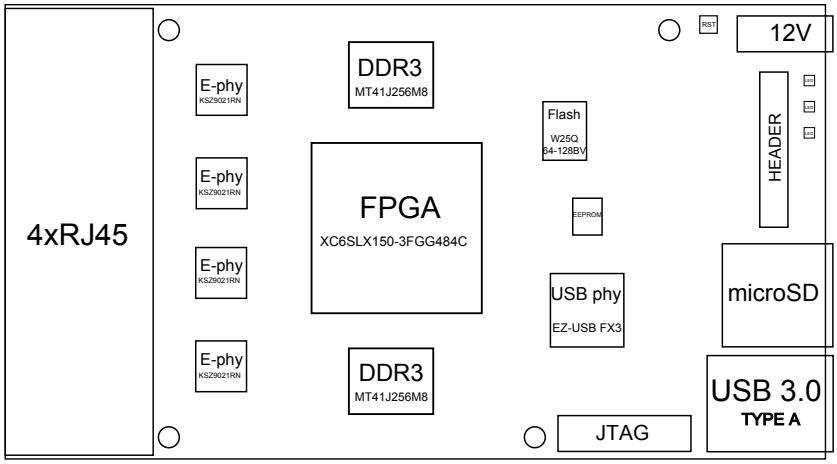
Čip je plně programovatelný a zároveň je možné jej ladit přes JTAG rozhraní. Jako pomalejší variantu lokálního úložiště je zvolen SD slot pro připojení (micro) SD karty. Prostor je možno využít např. pro spustitelný kód procesorů nebo pro konfiguraci. Jednoduché sériové rozhraní neumožňuje dosáhnout velkých rychlostí, avšak použitím specifikace SD/SDHC bude možno připojit karty s poměrně velikou kapacitou (32 GB).

Kromě základních komponent bylo potřeba uSondu osadit ještě dalšími pomocnými komponentami, které jsou nezbytné pro činnost zařízení. Mezi tyto patří zejména paměť typu flash určená pro uložení konfigurace FPGA (bitstream). Jde o model 64 Mb [20] nebo až 128 Mb [19] SPI kompatibilní flash, která svoji kapacitou umožňuje uložení hned několika konfigurací pro FPGA. Kromě základní konfigurace tak bude možné do paměti uložit i tzv. golden verzi firmware, která se použije v případě výskytu jakékoliv chyby bitstreamu [22]. Navíc bude možné dodatečný paměťový prostor využit pro načítaní programu procesoru nebo jiných dat pomocí SPI rozhraní, jako je tomu například u standardních PROM [17,21]. Dále je uSonda osazena pamětí typu EEPROM s rozhraním I2C, která je určená primárně pro uložení spustitelného kódu pro USB procesor. Její velikost se odvíjí od doporučení výrobce, tj. 512 kB. Obsah obou pamětí bude nahráván před použitím uSondy (viz dále) a pro ochranu pamětí před zápisem za běhu zařízení budou využity speciální mechanizmy. Dále uSonda obsahuje zdroje pro různé úrovně napájení, které budou vytvářeny ze standardního 12 V externího adaptéru na 230 V / 50 MHz, sadu signalizačních diod typu LED (signalizující stav napájení, stav FPGA, stav USB procesoru, stav aplikace a pod.), JTAG konektor pro nahrávaní obsahu flash či ladění a tzv. pin hea-

der pro připojení dalších externích komponent jakým je např. GPS přijímač pro přesné časové značky.

3.2 Rozmístění součástek

Navrhované rozložení součástek desky plošných spojů (PCB, Printed Circuit Board) je zřejmé z obrázku 8 (velikosti jednotlivých součástek jsou v poměru 1:1).



Obrázek 8. Navrhované rozložení součástek desky plošných spojů pro hardvérovou platformu uSondy.

Uprostřed PCB je FPGA tak, aby mělo snadný přístup ke všem komponentám na desce. Z komponent jsou asi nejdůležitější paměti DRAM a čtyři síťová rozhraní. Obě paměti jsou umístěny po stranách FPGA tak, aby byly co nejblíž vestavěným MCB blokům (viz kapitola 3.1). To výrazně zjednodušuje návrh PCB, protože tyto paměti vyžadují velice krátké spoje. Síťová rozhraní jsou umístěna na jedné ze stran PCB pro jednoduchý přístup a pro jednoduché připojení kabelů. V cestě mezi konektory a FPGA se nacházejí ethernetové phytry, což přesně odpovídá postupu zpracování síťových dat. Na PCB je také možné vidět paměti určené k bootování některých čipů a rozhraní JTAG, které je jednoduše přístupné z okraje desky. Slot na microSD kartu, tlačítka, signalizační LED diody a USB 3.0 rozhraní spolu s napájecím konektorem jsou pak přístupné na zadní straně desky. Takto orientovaný návrh umožnil snížit problémy při routováni spojů po desce a současně umožnil dosáhnout relativně malých rozměrů výsledné desky (odhadem 1100x610 [mm]). Kromě těchto součástek je také potřeba počítat s pasivními komponentami (kondenzátory, rezistory, cívky, ...), oscilátory, stabilizátory a podobně.

3.3 Proces oživení a bootovaní

Při návrhu systému uSondy je velmi důležité, jakým způsobem bude prováděno oživení hardwarové platformy a následné bootování procesorů. Pro oživení byly navrženy a implementovány ve VHDL následující 4 testy:

- **LED test** pro ověření správného nabootování FPGA, správného připojení hodin do FPGA a správného nastavení pinů,
- **Ethernet test** pro ověření funkčnosti všech 4 portů síťového rozhraní,
- **USB test** pro ověření správné činnosti a zapojení rozhraní s USB procesorem a
- **DDR DRAM test** pro ověření správné činnosti a zapojení DDR3 pamětí.

Všechny testy budou následně nahrazeny jediným kompletním testem, který bude řízený procesorem MicroBlaze. Po úspěšném otestování a připojení napájení k uSondě bude postup oživení pokračovat následujícími kroky:

1. Přeložený aplikační firmware (.bit) pro FPGA se pomocí kabelu připojeného na jedné straně přes JTAG k uSonde a na straně druhé skrz USB rozhraní k počítači nahraje do obsahu paměti pro FPGA.
2. Přeložená softwarová aplikace (.elf) pro USB procesor se nahraje do paměti pro procesor pomocí:
   - stejného kabelu jako v předchozím bodě,
   - pomocí speciálního softwarového vybavení od společnosti Cypress a USB 3.0 kabelu připojeného k počítači a uSondě

Po oživení a nahrání obsahů pamětí je možno přejít k normálnímu běhu zařízení. Navržený systém bootování se skládá z následujících kroků:

1. uSonda je resetováná nebo je odpojena a opět připojená k napájení (paměti drží svůj obsah).
2. FPGA i USB procesor nabootují každý ze svých pamětí, přičemž není možno spoléhat na to, které zařízení bude připraveno k činnosti jako první.
3. Hlavní procesor MicroBlaze (v FPGA) si pomocí zavaděče, který je uložený v interní paměti FPGA, nahraje do paměti operační systém i s aplikací. Zdrojem přitom může být (podle preferencí):
   - SD karta - pokud je přítomná,
   - sériová linka - pokud je přítomná,
   - TFTP server na síti, pokud máme konektivitu a nakonfigurovaný server,
   - paměť USB procesoru - pokud to její zvyšná kapacita dovolí,
   - připojený USB disk, jen pokud bude připojen a USB procesor už bude běžet,
4. Pokud je přítomen, tak obdobným způsobem nabootouje i druhý procesor MicroBlaze a to jen když má odemčen zámek synchronizace zamknutý hlavním procesorem.
5. Zavaděč instruuje procesor(y) MicroBlaze na správnou adresu do paměti RAM a spustí tak operační systém s aplikaci.

6. Po odpojení uSondy od napájení a opětovném připojení se pokračuje krokem č. 2)

Na základě navrženého systému bootovaní vyplynuly požadavky na připojená zařízení (viz kapitola 3.1). Z popisu (bod 3) je mimo jiné zřejmé, že v rámci bootování nebude možné využít dnes poměrně často používaného zavaděče u-Boot [4], který existuje i pro procesory MicroBlaze [18]. Hlavním důvodem je omezení velikosti binárního kódu zavaděče v FPGA maximálně na 64 kB, což u-Boot ani v základní konfiguraci nesplňuje. Dalším důvodem je omezená funkcionalita u-Boot, která neposkytuje možnost zavést aplikaci/OS z disku SD, což je implicitní zdroj dat pro uSondu. Znamenalo by to uvedenou funkci doprogramovat a tím ještě zvětšit velikost binárního kódu. Namísto u-Boot byla proto zvolená možnost zavaděče tzv. "first-stage", který přímo z disku SD nabootuje aplikaci/nebo OS pro procesor [11]. V případě potřeby bude možno takto zavést i u-Boot ("second-stage"), který může nahrát aplikaci/OS v základní konfiguraci pomocí sériového portu nebo TFTP serveru. Implementovaný zavaděč je tedy určen pro zavedení programu nebo operačního systému obecně pro procesory v FPGA (např. MicroBlaze) z karet typu SD, SDHC a SDXC využívajíc jednoduchého SPI módu. Je plně integrovatelný do vývojového nástroje Xilinx EDK/SDK (viz dále). Volitelně podporuje souborový systém FAT16 nebo FAT32, šifrovaní pomocí XXTEA, testování paměti a v případě minimální konfigurace (bez kontrolních výpisů) je možné jej uložit do necelých 16 kB.

3.4 Architektura

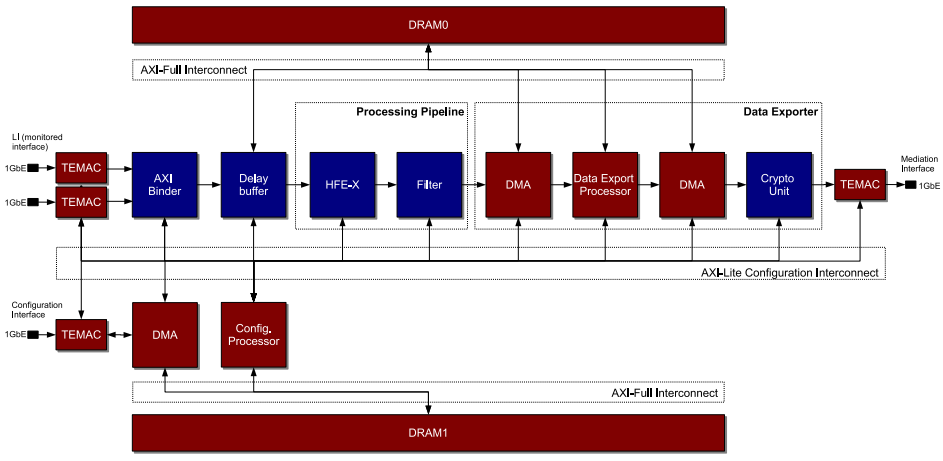
Již při návrhu architektury je důležitá znalost vývojového prostředí. V rámci návrhu a implementace LI sondy jsme se rozhodli využít nástroje EDK od firmy Xilinx [24]. Tento nástroj obsahuje velké množství univerzálně použitelných hardwarových komponent vhodných pro rychlé prototypování. Díky využití nástroje EDK je možné rychle implementovat první verzi uSondy a ověřit tak funkčnost jednotlivých komponent v simulacích, popřípadě v reálném hardware. Díky vysoké univerzalitě hardwarových komponent obsažených v EDK je možné rychle vyhodnotit a srovnat vliv jednotlivých částí návrhu na výkon celého systému a na základě tohoto srovnání zlepšovat navrženou architekturu. Tato univerzalita však způsobuje velkou spotřebu zdrojů v logice FPGA, což je také hlavní nevýhodou použití systému EDK. Pokud by se podařilo snížit potřebné množství zdrojů FPGA bylo by možné osadit sondu menším a tím i levnějším FPGA. Po implementaci a vyhodnocení veškeré funkcionality uSondy bude možné vyhodnotit spotřebu zdrojů FPGA a případně vybrané komponenty implementovat jako proprietární jednoúčelové jednotky.

Na základě diskuze s potenciálními uživateli LI systému jsme upřesnili požadavky na uSondu definované v předchozí kapitole následujícím způsobem:

- zařízení musí být schopné přijímat a filtrovat provoz dvou plně vytížených 1 Gb/s ethernetových rozhraní,
- musí podporovat 1000 filtrovacích pravidel,

– výstupní část systému musí zpracovat provoz 200 Mb/s,
– nesmí dojít ke ztrátě dat při výpadku spojení s mediačním centrem nebo agenturou a
– musí být schopna aplikovat nová filtrovací pravidla i na provoz, který byl na síti před 1 s.

Na základě vyjmenovaných požadavků byla navržena architektura, která využívá komponenty dostupné ve vývojovém prostředí EDK řízené procesorem MicroBlaze. Vývoj programového vybavení pro procesor MicroBlaze je v porovnání s vývojem hardwarových jednotek rychlý a proto je při využití procesoru možné vytvořit poměrně rychle implementaci základní funkcionality uSondy. Problémem architektur založených na procesorech je jejich celková propustnost. Proto byla provedena analýza propostnosti dostupných TCP/IP implementací pro procesor MicroBlaze. Nejrychlejší implementace dosahují rychlosti v průměru 250 Mb/s, a to i při zpracování nejdelších paketů. Z požadavků kladených na navrhovanou architekturu však vyplývá, že je nutné dosáhnout propustnost 2 Gb/s. Pro zvýšení propustnosti na požadovanou hranici byly aplikovány techniky HW&SW codesign a zpracování příchozích paketů spolu s filtrací bylo přesunuto do hardware. Procesor tak zpracovává pouze vyfiltrovaný provoz, který dle požadavků může dosahovat rychlosti maximálně 200 Mb/s. Sonda musí podporovat také vzdálenou konfiguraci. Aby nedocházelo ke konfliktům mezi procesem vysílajícím vyfiltrované pakety a procesem udržujícím aktuální konfiguraci, byla konfigurační část přesunuta do samostatného procesoru MicroBlaze. Výsledná architektura tedy bude obsahovat 2 procesory – jeden pro zpracování vyfiltrovaného provozu a druhý pro konfiguraci. Navržená architektura je zobrazena na obrázku 9.



Obrázek 9. Navržená rchitektura firmware uSondy.

Architektura se skládá ze čtyř ethernetových rozhraní, které jsou řízeny prostřednictvím IP cores TEMAC (viz kapitola 3.1). U dvou TEMAC bloků jsou výstupy spojeny do jednoho proudu AXI-Stream [7] pomocí komponety AXI-Binder tak, aby bylo možno použít společnou extrakci hlaviček (jednotka HFE-X) a filtrování síťového provozu (jednotka Filtr). Před zpracováním jednotkami HFE-X a Filtr (tzv. procesní linkou) je možno v případě potřeby implementovat zpoždění vstupního provozu (komponenta Dellay Buffer). HFE-X slouží k extrakci informací pro filtrování z hlaviček paketů. Tyto informace jsou přidány před samotný paket přenášený pomocí AXI-Stream rozhraní. Filtr pak na základě těchto informací rozhodne, zda aktuální paket odpovídá filtrovacím pravidlům. Pokud ano, je před paket vložena informace o odpovídajícím pravidlu. V opačném případě je paket zahozen. Formát výstupu Filtru je tedy totožný s formátem výstupu komponenty TEMAC, což umožňuje příchozí pakety zpracovat pomocí DMA řadiče pro síťové rozhraní. DMA řadič uloží přijatý paket do paměti DRAM0 a vygeneruje přerušení pro Data Export procesor. Přerušení je generováno pouze jednou za stanovený počet příchozích paketů tak, aby bylo dosaženo požadované propustnosti. Po přijetí přerušení rozdělí procesor přijatá data do bloků o délce MTU, doplní je o hlavičku přenosového protokolu a inicializuje přenos paketu přes výstupní rozhraní do mediačního centra nebo přímo do agentury. Data jsou z paměti čtena pomocí druhého DMA řadiče a převedena na AXI-Stream. Ještě před odesláním mohou být zašifrována pomocí hardwarové šifrovací jednotky (komponenta Crypto Unit). O výpočet a korektní nastavení kontrolních součtů jednotlivých paketů se stará komponenta TEMAC. Díky podpoře scatter/gather DMA přenosů nemusí procesor provádět žádné posuny v paměti při přidávání hlaviček k jednotlivým blokům dat.

V architektuře systému jsou použity dva procesory. Jeden je zodpovědný za realizaci konfiguračního rozhraní (Config. Processor) a druhý za bezchybné a případně zabezpečené odesílání paketů (Data Export Processor). Díky tomuto rozdělení je garantováno, že žádné externí zásahy nemohou ovlivnit propustnost filtrovací části systému. Po přijetí konfigračních příkazů přes zabezpečené rozhraní provede procesor výpočet nastavení jednotlivých komponent a jednotek, které následně nahraje do jejich interních registrů. Hardwarové prvky jsou navrženy tak, aby bylo možné změnit jejich nastavení bez ztráty jediného paketu. Po celou dobu zpracování konfiguračních příkazů uSonda filtruje na plné rychlosti podle předchozích pravidel.

Výstupní část uSondy se skládá z ethernetového subsystému obsaženého v knihovně nástroje EDK rozšířeného o šifrovací jednotku (Crypto Unit). Nejdůležitější součástí vysílací části je software udržující jednoduchý zásobník typu FIFO (first-in first-out) přijatých paketů v paměti. FIFO lze implementovat pomocí kruhového bufferu. Na začátku bude veškerá volná paměť přiřazena k DMA řadiči přijímajícímu data z filtru. Jakmile dojde k vygenerování přerušení, které signalizuje načtení paketů do paměti, dojde k přeřazení bloku paměti s přijatým paketem do kruhového bufferu spravovaného vysílacím DMA řadičem. Díky podpoře scatter/gather funkcionality lze rozdělit přijatá data na bloky a opatřit je hlavičkami aniž by bylo nutné kopírovat data. Stačí použít pouze ope-

race s ukazateli. Pro zabezpečení spolehlivého a bezpečného přenosu dat bude nutné také implementovat řízení protokolu vyšší vrstvy. Se vzrůstající složitostí systému narůstá množství operací které je třeba provádět s každým odeslaným paketem a tím klesá propustnost vysílací části. Zajištění bezpečnosti přenosu paketů bude realizováno kombinací asymetrického a symetrického šifrování. Pomocí asymetrické šifry dojde k výměně šifrovacího klíče, který bude použit pro symetrické šifrování odchozích paketů. Výměna šifrovacích klíčů bude realizována plně v režiji procesoru vysílací části, protože se nejedná o časově kritickou operaci. Symetrické šifrování odesílaného obsahu bude realizováno hardwarovou jednotkou, která bude řízena procesorem. Spolehlivý přenos vyfiltrovaných a zašifrovaných dat je možné zajistit více způsoby. Při implementaci se soustředíme na následující tři varianty, které budeme více rozpracovávat a dále analyzovat.

- Použití TCP protokolu. TCP je rozšířený standart, který je podporován velkou většinou zařízení. Jedná se o robusní protokol, který podporuje řadu nastavení, díky čemuž je jeho implementace náročná jak z pohledu lidských zdrojů, tak i z pohledu výpočetní kapacity.
- Implementace vlastního zabezpečovacího systému nad UDP protokolem. Protokol UDP je velmi jednoduchý a je plně podporován v rámci HW komponent nástroje EDK. Spolehlivý přenos je však nutno implementovat na aplikační úrovni. Výhodou tohoto přístupu je možnost zaintegrovat spolehlivost a bezpečnost do jednoho aplikačního protokolu a tím zrychlit výpočet.
- Implementace nového nebo úprava standardního síťového protokolu. Výhody tohoto přístupu kopírují výhody předchozího bodu a navíc dochází k odstranění režije UDP protokolu. Nevýhodou je nutnost implementovat vlastní řešení na mediačním zařízení.

V rámci implementace zvolené architektury je možné vyjít z komponent dostupných v rámci vývojového prostředí EDK, ale současně je třeba vyvinout i několik nových komponent nebo jednotek. Jedná se o AXI-Binder, Delay buffer, jednotka HFE-X a jednotka Filtr. Také je potřeba vyvinout sadu převodních VHDL komponent mezi novou třídou sběrnic AXI (AXI-Full, AXI-Lite, AXI-Stream) a starším typem sběrnic MI32 (paměťovo orientované 32 bitové rozhraní) a FrameLink (proudové rozhraní o libovolné dátové šířce):

- AXI-Binder má na vstupu dvě rozhraní AXI-Stream generované komponentami TEMAC. AXI-Binder provádí jejich spojení do jediného AXI-Streamu takovým způsobem, aby výsledný proud obsahoval vždy celý přijatý paket. Komponenta není v současné době implementována.
- Zpožďovací buffer (Delay buffer). Tato jednotka slouží k pozdržení vstupního provozu o 1 s. Komponenta není v současné době implementována.
- Jednotka HFE-X slouží k extrakci položek z hlaviček paketů různých síťových protokolů. Vstupem jednotky HFE-X je AXI-Stream obsahující celé síťové pakety. Výstupem je také AXI-Stream rozšířený o extrahované položky. Pro realizaci sondy byla využita již dříve vyvintá jednotka, která byla modifikována a přidána do repozitáře EDK. Funkcionalita HFE-X je detailně posána v části zabývající se vysokorychlostní sondou v kapitole 2.2.

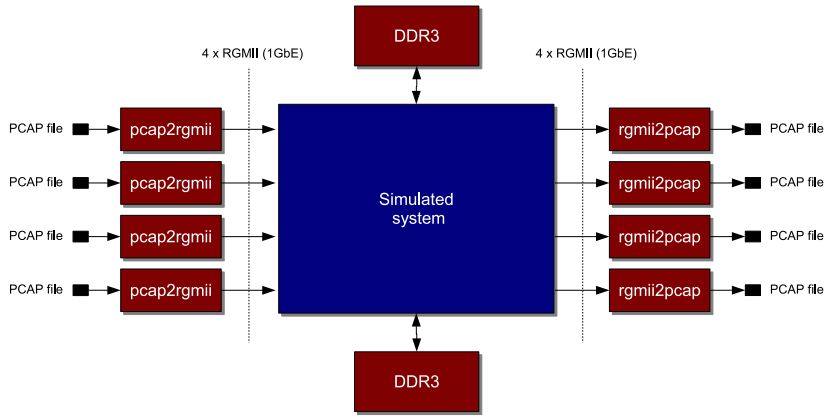
– Klíčovou jednotkou v architektuře sondy je Filtr, který zodpovídá za výběr požadovaného provozu podle zadaných pravidel. Výstup této jednotky je opět AXI-Stream, který před jednotlivé pakety přidává přesnou identifikaci pravidla, které daný paket akceptovalo. Tato jednotka je implementována a také přidána do repozitáře EDK.
– Vstupně výstupní komponenty. Jedná se o komponenty, které budou zajišťovat komunikaci FPGA s okolím (řadič USB, SPI, I2C, ...). Tyto komponenty jsou již implementovány nebo dostupné ve formě IP cores v EDK.
– Jako poslední jsou převodní komponenty MI32 → AXI-Lite, AXI-Lite → MI32, FrameLink → AXI-Sream a AXI-Stream → FrameLink. Implementace těchto komponent je nutná pro efektivní využití již dříve vyvinutých funkčních jednotek.

Architektura uSondy je navržena způsobem, který podporuje inkrementální vývoj. Prvním krokem implementace je propojení procesní linky s procesorem MicroBlaze. V rámci implementace tohoto kroku bude procesor odesílat zachycené a přefiltrované pakety pomocí UDP protokolu bez podpory šifrování ani jiné formy zabezpečení. Dalším krokem implementace bude přidávání jednotlivých funkcí výstupní části, dokud nebudou splněny veškeré požadavky. Současně předpokládáme, že výpočetně náročné funkce výstupní části budou implementovány v logice FPGA.

Tvorba simulačního prostředí V prvním roce projektu ještě nebyla dostupná hardwarová platforma pro uSondu. Aby bylo možné implementovat a ladit funkci navrhovaného systému, bylo nutné vytvořit simulační prostředí, které svým chováním bude odpovídat navrhované hardwarové platformě. Z obrázku 10 je patrné, že námi navržené a vytvořené simulační prostředí se skládá z modelu testovaného systému v FPGA a z modelů okolí, které se skládá z generátorů a receptorů ethernetového provozu a simulačního modelu dynamické paměti.

Nástroj EDK umožňuje vygenerování simulačního modelu pro libovolný systém v FPGA s tím, že model obsahuje i software pro všechny instancované procesory. Výsledkem generování simulačního modelu je pak jediná komponenta simulující chování celého FPGA. Abychom mohli vygenerovaný simulační model testovat, bylo navrženo a implementováno testovací prostředí, které umožňuje do simulačního modelu posílat a příjmat pakety.

Nejdůležitějšími vstupy uSondy jsou ethernetová rozhraní. uSonda bude osazena čtyřmi ethernetovými porty. Simulační model tedy instancuje čtyři komponenty pcap2rgmii, které načítají soubory paketů uložené ve formátu pcap a vysílají zaznamenané pakety přes rozhraní RGMII. Komponenta pcap2rgmii je navržena tak, aby se vysílání neřídilo mezipaketovými mezerami v pcap souboru, ale signálem enable. V případě, že je podle specifikace RGMII přípustné vyslat paket (tedy neprobíhá vysílání žádného paketu a od ukončení vysílání uběhla dostatečně dlouhá doba) a enable signál je nastaven do jedničky, komponenta okamžitě zahájí vysílání dalšího ethernetového paketu. Touto technikou je umožněna snadná regulace rychlosti vstupního toku bez nutnosti měnit samotné pcap soubory.



Obrázek 10. Simulační prostředí.

Komponenta pcap2rgmii umožňuje generovat RGMII signály z pcap souboru. Simulační prostředí ji využívá ke generování síťových stimulů pro simulovaný systém. Rychlost generování paketů není závislá na pcap souboru, ale je řízena logikou simulačního prostředí. Díky tomu je možné simulovat různé režimy zatížení odposlouchávané linky. K zaznamenávání výstupů symulovaného systému slouží komponenta rgmii2pcap. Tato komponenta generuje pcap soubor obsahující pakety vytvořené simulačním modelem uSondy a současně vytváří časové značky s přesností na nanosekundy, což již nedpovídá pcap formátu. Cenou za takto vysokou přesnost je snížení doby, po kterou lze data zaznamenávat. Tato nevýhoda je ale pouze teoretická protože simulace je příliš náročná a není možné simulovat více než několik sekund simulačního času. Vygenerované pcap soubory je pak možno prohlížet např. v programu wireshark [6] a dále je zpracovávat tak, jako by byly zaznamenány z běžného síťového rozhraní.

Ke korektní funkci simulačního modelu je důležité mít připojeny modely dynamické paměti. Z internetových stránek výrobce DDR pamětí byl stažen simulační model paměti, která bude dle návrhu použita na uSondě. Tento model je instancován v simulačním prostředí pro obě paměti, kterými bude uSonda osazena. Spolu se simulačním prostředím byl také implementován jednoduchý program posílající zpět jednotlivé přijaté pakety a byla tak oveřena základní architektura uSondy.

Reference

1. CESNET; z.s.p.o.: Projekt Liberouter. 2011.
   URL `http://www.liberouter.org`
2. Cypress; Inc.: EZ-USB?? FX3 SuperSpeed USB Controller (CYUSB3014) - Datasheet. 2011, revision 1.
   URL `http://www.cypress.com/?docID=30512`

3. DeHaven, K.: Extensible Processing Platform: Ideal Solution for a Wide Range of Embedded Systems - Xilinx White Paper WP369. 2010, version 1.0.
URL http://www.xilinx.com/support/documentation/white_papers/wp369_Extensible_Processing_Platform_Overview.pdf
4. Denx: Das U-Boot - Universal Bootloader. 2011.
URL http://www.denx.de/wiki/U-Boot/
5. European Telecommunications Standards Institute: *ETSI TR 102 528: Lawful Interception (LI); Interception domain Architecture for IP networks*. 10 2006, version 1.1.1.
6. Foundation, W.: WIRESHARK: the world's foremost network protocol analyzer. 2011.
URL http://www.wireshark.org/
7. amd Inc., X.: AXI Reference Guide - Xilinx User Guide UG761. 2011, version 13.1.
URL http://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf
8. Invea-tech; a.s.: COMBO LXT a COMBO 10G2. 2011.
URL http://www.invea.cz/fpga-reseni/fpga-karty
9. Kobierský, P.; Kořenek, J.; Polčák, L.: Packet Header Analysis and Field Extraction for Multigigabit Networks. In *Proceedings of the 2009 IEEE Symphosium on Design and Diagnostics of Electronic Circuits and Systems*, IEEE CS, 2009, s. 96 – 101.
10. Korček, P.; Žádník, M.; Kaštil, J.: Leightweight benchmarking of platforms for network traffic processing, 2011.
11. Košar, V.; Korček, P.: Zaváděč z karty SD pro procesory v FPGA. 2011.
URL http://www.fit.vutbr.cz/research/view_product.php?id=219
12. Martínek, T.; Košek, M.: NetCOPE: Platform for Rapid Development of Network Applications. In *Proceedings of 2008 IEEE Design and Diagnostics of Electronic Circuits and Systems Workshop*, IEEE CS, 2008, s. 219 – 224.
13. Micrel; Inc.: KSZ9021RL/RN Gigabit Ethernet Transciever with RGMII Support - Datasheet. 2009.
URL
http://www.micrel.com/_PDF/Ethernet/datasheets/ksz9021rl-rn_ds.pdf
14. Micron; Inc.: DDR3 SRAM, 2Gb: x4, x8, x16 DDR3 SDRAM Features - Product Specification. 2006, revision K 04/10 EN.
URL http://download.micron.com/pdf/datasheets/dram/ddr3/2Gb_DDR3_SDRAM.pdf
15. Málek, T.; Martínek, T.; Kořenek, J.: GICS: Generic Interconnection System. In *Proceedings of 2008 International Conference on Field Programmable Logic and Applications*, IEEE CS, 2008, s. 263 – 268.
16. Pagh, L. R.; Rodler, F. F.: Cuckoo Hashing. In *Proceedings of the 9th Annual European Symposium on Algorithms*, Springer-Verlag, 2001, s. 121 – 133.
17. Sheth, S.: MicroBlaze Platform Flash/PROM Boot Loader and User Data Storage - Xilinx Application Note APP482. 2005, version 2.0.
URL http://www.xilinx.com/support/documentation/application_notes/xapp482.pdf
18. Simek, M.: U-BOOT, eCos, Device Tree, Qemu and Linux for Xilinx Microblaze, PowerPC and ZynQ. 2011.
URL http://www.monstr.eu/wiki/doku.php

19. Winbond; Inc.: 3V 128M-bit Serial Flash Memory with Dual and Quad SPI - Datasheet. 2010, revision E.
URL http://www.winbond.com/NR/rdonlyres/1C7E7EF8-ACBF-47EC-BE16-D668A7933A21/0/W25Q128BV.pdf
20. Winbond; Inc.: 64M-bit Serial Flash Memory with Dual and Quad SPI - Datasheet. 2010, revision E.
URL http://www.winbond.com.tw/NR/rdonlyres/591A37FF-007C-4E99-956C-F7EE4A6D9A8F/0/W25Q64BV.pdf
21. Xilinx; Inc.: Reading User Data from Configuration PROMs - Xilinx Application Note APP694. 2007, version 1.1.1.
URL http://www.xilinx.com/support/documentation/application_notes/xapp694.pdf
22. Xilinx; Inc.: SP606 Multiboot design - Xilinx Presentation XTP059. 2009.
URL http://www.xilinx.com/support/documentation/boards_and_kits/xtp059.pdf
23. Xilinx; Inc.: Spartan-6 FPGA Memory Controller - Xilinx User Guide UG388. 2010, version 2.3.
URL http://www.xilinx.com/support/documentation/user_guides/ug388.pdf
24. Xilinx; Inc.: Embedded System Tools Reference Manual (EDK) - Xilinx User Guide UG111. 2011, version 13.3.
URL http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_3/est_rm.pdf
25. Xilinx; Inc.: LogiCORE IP Tri-Mode Ethernet MAC - Xilinx Product Specification DS818. 2011, version 5.1.
URL http://www.xilinx.com/support/documentation/ip_documentation/ds818_tri_mode_eth_mac.pdf
26. Xilinx; Inc.: MicroBlaze Processor Reference Guide - Xilinx User Guide UG081. 2011, version 13.3.
URL http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_3/mb_ref_guide.pdf
27. Xilinx; Inc.: Spartan 6 Family Overview - Xilinx Product Specification DS160. 2011, version 2.0.
URL http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf