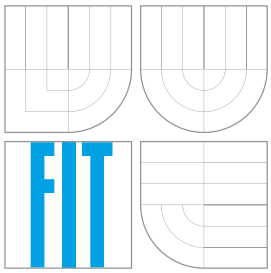
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

IMPLEMENTACE A VERIFIKACE VSTUPNÍCH A VÝSTUPNÍCH SÍŤOVÝCH BLOKŮ

IMPLEMENTATION AND VERIFICATION OF NETWORK INTERFACE BLOCKS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JIŘÍ MATOUŠEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ TOBOLA
SUPERVISOR

BRNO 2009

Abstrakt

V rámci platformy NetCOPE se vstupní a výstupní síťové bloky používají pro odstínění návrháře síťové aplikace od problémů s implementací linkové vstvy síťového modelu ISO/OSI, zvláště pak její MAC podvrstvy. Tato bakalářská práce se zabývá návrhem, implementací a verifikací takovýchto bloků pracujících na rychlosti 10 Gb/s. Navržený vstupní síťový blok provádí kontrolu příchozích rámců a umožňuje zahazování těchto rámců na základě výsledků prováděných kontrol. Výstupní síťový blok podporuje nahrazování zdrojové MAC adresy rámce a doplnění pole FCS. Součástí obou síťových bloků jsou také různé druhy čítačů rámců. Navržené síťové bloky byly otestovány na kartách COMBO v rámci platformy NetCOPE a bylo pro ně navrženo verifikační prostředí pro jazyk SystemVerilog.

Abstract

Network interface blocks are basic part of the NetCOPE platform where they help to the network application designers to deal with problems of implementing the Data Link Layer of the OSI Reference Model, especially the MAC sublayer. This thesis is focused on the design and implementation of such network interface blocks operating at speed 10 Gb/s. Designed input interface block provides checking of several parts of the Ethernet frame and allows discarding of this frame based on checking results. Output interface block supports replacing frame's Source Address by a pre-set value and provides frame's CRC computation. Both network interface blocks also include a set of frames counters. Implemented network interface blocks were tested on the COMBO card. SystemVerilog verification testbench was also designed for both network interface blocks.

Klíčová slova

vstupní a výstupní síťové bloky, XGMII, 10 Gigabit Ethernet, FrameLink, NetCOPE, FPGA, VHDL, SystemVerilog

Keywords

network interface blocks, XGMII, 10 Gigabit Ethernet, FrameLink, NetCOPE, FPGA, VHDL, SystemVerilog

Citace

Jiří Matoušek: Implementace a verifikace vstupních a výstupních síťových bloků, bakalářská práce, Brno, FIT VUT v Brně, 2009

Implementace a verifikace vstupních a výstupních síťových bloků

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jiřího Toboly. Další informace mi poskytli kolegové z projektu Liberouter. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jiří Matoušek
19. května 2009

Poděkování

Na tomto místě bych rád poděkoval panu Ing. Jiřímu Tobolovi za vedení při vypracování mé bakalářské práce. Mé poděkování patří také kolegům z projektu Liberouter, kteří mě v mém snažení podporovali po stránce odborné i technické.

Obsah

Kapitola 1

Úvod

Oblast počítačových sítí a zejména pak Internetu zažívá v posledních letech velký rozkvět. Objevují se nové technologie přenosu signálu, které umožňují připojit do počítačových sítí nejrůznější koncová zařízení. Připojení k Internetu je dnes díky bezdrátovým technologiím možné z mnoha dříve nedostupných míst. Počítačové sítě a Internet se stávají běžně používaným nástrojem pro komunikaci a jsou na ně kladeny stále vyšší nároky.

Velmi často zmiňovaným a měřeným parametrem počítačových sítí je rychlost přenosu dat. V současné době se v LAN (*Local Area Network*) sítích běžně používají zařízení schopná přenášet a zpracovávat data rychlostí až 1 Gb/s. Tato rychlost je definována standardem IEEE 802.3 [5], který definuje nejběžnější technologii používanou v sítích LAN — Ethernet. Poslední verze standardu IEEE 802.3 však definují technologii Ethernet také pro rychlost 10 Gb/s a v přípravné fázi je i standard Ethernetu pro rychlosti 40 Gb/s a 100 Gb/s.

Přenosové rychlosti v počítačových sítích se neustále zvyšují. Pro zpracování přijímaných a vysílaných dat již nepostačují možnosti procesorů a sotwarového vybavení dnešních zařízení a je tedy nutné zpracování síťových dat akcelerovat pomocí hardware. Tato technika se běžně používá u aktivních prvků síťové architektury, kterými jsou například přepínače a směrovače. V těchto zařízeních se pro hardwarovou akceleraci používají aplikačně specifické integrované obvody — ASIC (*Application-Specific Integrated Circuit*).

Hardwarovou akceleraci lze také postavit na technologii programovatelných hradlových polí FPGA (*Field Programmable Gate Array*). Touto oblastí se zabývá například projekt Programmable hardware [1]. V rámci této výzkumné aktivity zastřešené společností Cesnet z. s. p. o. mimo jiné vznikla platforma pro rychlý vývoj síťových aplikací — NetCOPE [12].

Cílem této práce je navrhnout, implementovat a verifikovat vstupní a výstupní síťové bloky pro platformu NetCOPE. Implementované síťové bloky musí být schopné přijímat a vysílat pakety na rychlosti 10 Gb/s přes standardní rozhraní XGMII [5]. Pro komunikaci s uživatelskou aplikací bude využit protokol FrameLink [12] a propojovací systém sběrnic platformy NetCOPE [8]. Funkčnost implementovaných síťových bloků bude ověřena verifikací pomocí jazyka SystemVerilog.

Práce je členěna do kapitol, které popisují jednotlivé etapy práce na zadaném problému. V kapitole 2 jsou popsány protokoly, se kterými síťové bloky pracují, a rozhraní, přes která komunikují s ostatními komponentami. Další kapitolu 3 tvoří popis navržených síťových bloků. Následuje kapitola 4 shrnující podstatné informace o výsledné implementaci. Popis verifikace výsledné implementace je uveden v kapitole 5. Závěrečná kapitola 6 srhnuje výsledky dosažené v rámci vypracování této práce. Diskutují se možnosti dalšího rozšíření a uplatnění implementovaných síťových bloků.

Kapitola 2

Teoretický rozbor

Témata, kterými se zabýva tato část práce, představují popis prostředí, do kterého budou navržené síťové bloky zasazeny a se kterým budou muset správným způsobem komunikovat. Pro správnou funkčnost navrhovaných síťových bloků je nezbytně nutné pochopit komunikační protokoly, které definují hodnoty a význam signálů na jednotlivých rozhraních, a také formát dat přenášených přes tato rozhraní. Pro lepší představu o požadavcích na funkce implementované ve vstupním a výstupním síťovém bloku je také dobré znát architekturu platformy NetCOPE, jejíž budou síťové bloky součástí.

2.1 Komunikační protokoly

Hlavním úkolem vstupních a výstupních síťových bloků je transformace signálů používaných technologií LAN sítě na signály protokolu pro přenos dat mezi jednotlivými funkčními bloky implementovanými na čipu FPGA. Implementované síťové bloky budou nazaseny v prostředí, kde je jako technologie LAN sítě použit Ethernet a přenos dat k dalším komponentám na čipu FPGA probíhá pomocí protokolu FrameLink [12].

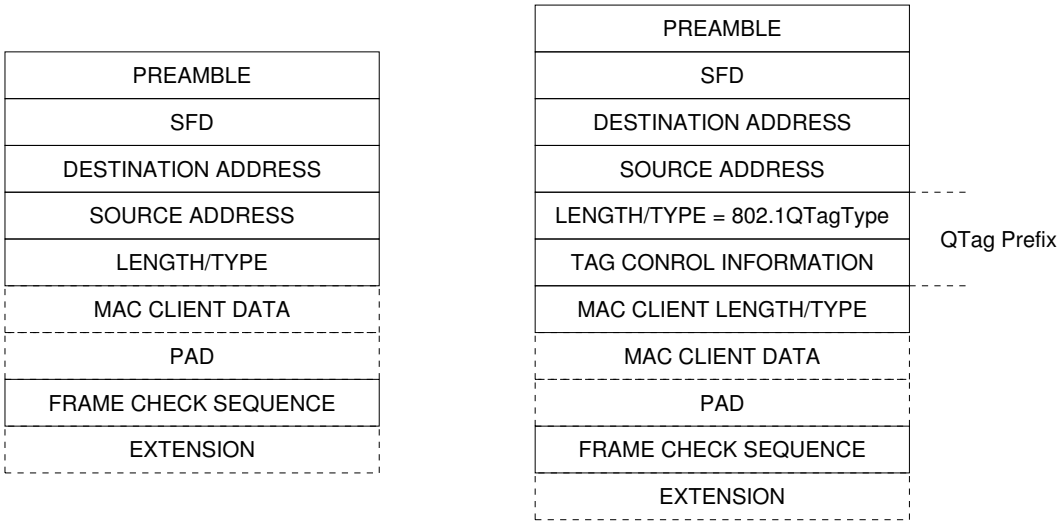
2.1.1 Ethernet

Ethernet je nejrozšířenější technologií používanou v LAN sítích. Standard Ethernetu [5] pokrývá oblasti spadající jak do fyzické, tak i do linkové vrstvy referenčního síťového modelu ISO/OSI. Na fyzické vrstvě jsou řešeny například typy propojovacích vodičů (koaxiální kabel, kroucená dvojlinka, optické vlákno) a jejich maximální délka, připojení ke komunikačnímu médiu a nebo kódování přenášených dat. Na linkové vrstvě se pak standard zabývá především řízením přístupu k médiu.

Pro přenos jsou data z vyšších vrstev ISO/OSI modelu zapouzdřována do datových jednotek, které se nazývají rámce. Formát ethernetového rámce je podobně jako u datových jednotek jiných vrstev síťového modelu přesně definován (viz obrázek 2.1). Význam jednotlivých polí rámce je popsán v následujícím seznamu.

Preamble (7 bytů) — tato část rámce signalizuje začátek vysílání a slouží především k synchronizaci časování vysílající a přijímající strany. V rámci pole Preamble jsou hodnoty každých dvou sousedních bitů různé.

Start Frame Delimiter (1 byte) — jde o poslední byte před vlastním obsahem rámce a označuje tedy jeho začátek. Obsah SFD je vždy 10101011 (od nejméně významného bitu po nejvýznamnější bit).



Obrázek 2.1: Standardní a tagovaný ethernetový rámec

Destination Address (6 bytů) — toto pole obsahuje MAC adresu [3] specifikující kam má být rámec zaslán. Tato adresa může být jak individuální (*unicast*), tak skupinová (*multicast* nebo *broadcast*).

Source Address (6 bytů) — položka obsahuje MAC adresu stanice, která rámec vysílá. Zdrojová adresa je vždy individuální (*unicast*).

Length/Type (2 byty) — hodnota v této části rámce může mít dva významy — buď udává počet bytů v poli MAC Client Data a nebo specifikuje, který protokol vyšší vrstvy ISO/OSI modelu je v části MAC Client Data zapouzdřen. Rozdělení, zda jde o velikost datové části rámce, nebo o identifikaci zapouzdřeného protokolu, se provádí na základě honoty tohoto pole. Je-li hodnota pole Length/Type menší nebo rovna 1500 (05DC hexadecimálně), pak toto pole udává počet bytů datové části rámce. Při hodnotě 1536 (0600 hexadecimálně) nebo vyšší je obsahem pole identifikátor zapouzdřeného protokolu.

MAC Client Data a Pad (46 – 1500 bytů) — velikost polí MAC Client Data a Pad je proměnná. Aby však byly dodrženy limity na minimální a maximální povolenou délku rámce, musí se součet velikostí polí MAC Client Data a Pad pohybovat v intervalu 46 až 1500 bytů. Nedosahuje-li délka pole MAC Client Data hodnoty 46 bytů, zbývající prostor je vyplněn libovolnými daty, která tvoří položku Pad.

Frame Check Sequence (4 byty) — pole Frame Check Sequence obsahuje kontrolní součet (*Cyclic Redundancy Check*, CRC) vypočítaný z obsahu polí Destination Address, Source Address, Length/Type, MAC Client Data a Pad. Obsah tohoto pole slouží ke kontrole integrity přenášeného rámce.

Extension — za standardem definovaných podmínek může pole Extension obsahovat rozšíření ethernetového rámce, které umožní splnění časových požadavků na délku přenosu rámce při vysokých rychlostech. Pokud přítomnost tohoto rozšíření není nutná, má pole Extension velikost 0 bytů.

Kromě výše popsaného typu ethernetového rámce zmiňuje standard také tzv. tagované ethernetové rámce, které jsou definovány v [6]. Struktura tagovaného ethernetového rámce je zobrazena na obrázku 2.1. Tento typ rámce se v LAN síti objevuje v případě, že se používají virtuální lokální sítě (*Virtual Local Area Network*, VLAN). Oproti klasickému ethernetovému rámci přibyla ve struktuře rámce dvě pole, každé po 2 bytech. Následuje popis polí zavedených společně s tagovaným ethernetovým rámcem.

Length/Type (2 byty) — ačkoliv pole se stejným názvem existuje také v klasickém ethernetovém rámci, v tagovaném rámci patří pole Length/Type do části QTag Prefix. Hodnota těchto dvou bytů je v tagovaném rámci vždy 8100 hexadecimálně, což je identifikátor typu 802.1Q.

Tag Control Information (2 byty) — toto pole obsahuje informace související s VLAN. Význam jednotlivých bitů je definován v [6].

MAC Client Length/Type (2 byty) — pole MAC Client Length/Type odpovídá poli Length/Type z klasického ethernetového rámce.

Jak pro klasický, tak i pro tagovaný ethernetový rámec platí, že jeho velikost je omezena. Tento limit se vztahuje na část rámce začínající polem Destination Address a polem Frame Check Sequence končícím. Spodním limitem velikosti této části rámce je hodnota 64 bytů (pro tagovaný i klasický rámec) a maximální přípustná hodnota je 1518 bytů pro klasický rámec, respektive 1522 bytů pro tagovaný ethernetový rámec.

Omezení na minimální velikost rámce vychází z historického vývoje Ethernetu. V době, kdy byl přístup ke komunikačnímu médiu řízen algoritmem CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*), bylo nutné, aby doba zasílání rámce o minimální povolené velikosti byla větší než doba, za kterou vysílaný signál obsadí celé médium (tzv. kolizní okénko). V opačném případě by vysílající stanice nebyly schopé detekovat kolizi. Maximální velikost rámce je pak objem dat, který byl ještě v historických sítích přenesen za rozumnou dobu.

Vysílání rámce začíná polem Preamble a končí položkou Frame Check Sequence, případně polem Extension (je-li tato položka přítomná). Jednotlivá pole jsou vysílána od nejméně významného bitu (LSB) po nejvýznamnější bit (MSB). Mezi vysíláním dvou rámců musí být přítomná mezera (*Interframe Gap*), která odpovídá době vyslání 12 bytů.

2.1.2 FrameLink

Pro přenos dat mezi funkčními bloky na čipu FPGA byl v rámci platformy NetCOPE navržen protokol FrameLink. Jde o paketově orientovaný protokol pro synchronní point-to-point komunikaci. Data jsou přenášena ve formě rámců rozdělených na obecně nedefinovaný počet částí, přičemž význam jednotlivých částí je dán kontextem. Typicky se však používají rámce obsahující až tři části — kontrolní data před paketem (*header*), vlastní paketová data (*payload*) a kontrolní data za koncem paketu (*footer*). Přenos probíhá po slovech generické šířky s uspořádáním bytů ve slově formou *Little Endian*.

Rozhraní protokolu umožňuje komunikaci jen jedním dopředu zvoleným směrem. Kromě datových a hodinových vodičů jsou součástí také řídicí vodiče, sloužící mimo jiné k označení začátku a konce rámce a jednotlivých částí v rámci, nebo signály pro řízení přenosu vyvedené pro každou komunikující stranu. Následující tabulka 2.1 shrnuje informace o šířce a směru signálů rozhraní FrameLink. Je také uveden podrobný popis jednotlivých signálů.

Signál	Šířka (v bitech)	Směr
CLK	1	vstup
DATA	$n \times 8$	zdroj → cíl
SOF_N	1	zdroj → cíl
EOF_N	1	zdroj → cíl
SOP_N	1	zdroj → cíl
EOP_N	1	zdroj → cíl
REM	$\log_2 n$	zdroj → cíl
SRC_RDY_N	1	zdroj → cíl
DST_RDY_N	1	cíl → zdroj

Tabulka 2.1: Popis signálů protokolu FrameLink

Clock (CLK) — hodinový signál pro synchronizaci komunikace.

Data Bus (DATA) — datová sběrnice o generické šířce. Šířku sběrnice lze nastavovat jen po celých bytech.

Start of Frame (SOF_N) — signál určující začátek vysílání (první slovo) rámce. Signál je aktivní v logické nule.

End of Frame (EOF_N) — signál pro označení konce vysílání (poslední slovo) rámce. Signál je aktivní v logické nule.

Start of Part (SOP_N) — určuje začátek části rámce. Při první části v rámci je aktivní společně se signálem SOF_N. Signál je aktivní v logické nule.

End of Part (EOP_N) — signál pro označení konce části rámce. U poslední části v rámci je aktivní společně s EOF_N. Je-li některá část rámce menší než je šířka datové sběrnice, pak při vysílání této části jsou signály SOP_N a EOP_N aktivní současně. Signál je aktivní v logické nule.

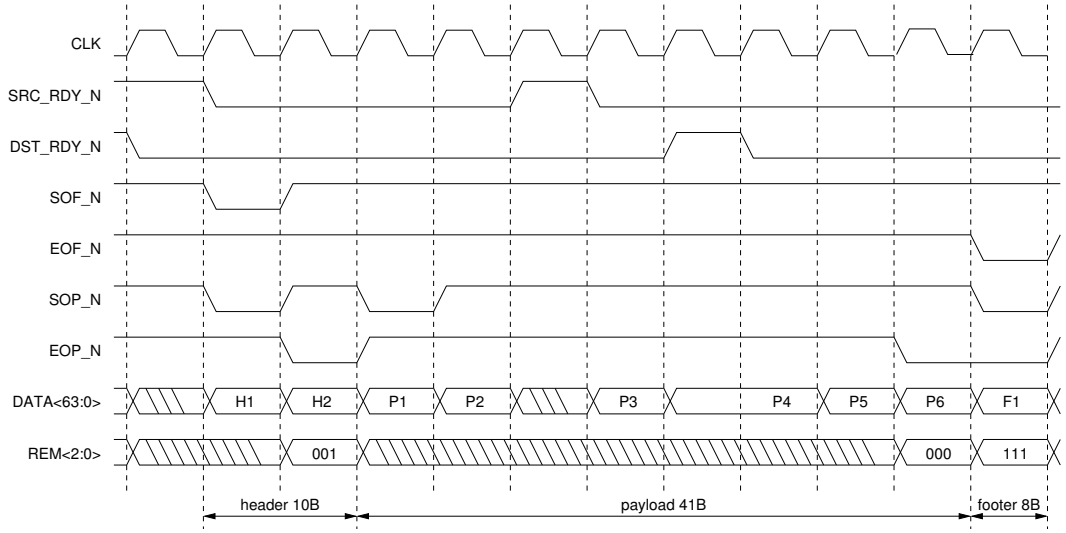
Reminder Bus (REM) — šířka této sběrnice je dána dvojkovým logaritmem šířky datové sběrnice a binární hodnota na této sběrnici určuje pořadí posledního platného bytu v aktuálně přenášeném slově. Signál je platný pouze se společně aktivním signálem EOP_N.

Source Ready (SRC_RDY_N) — indikuje připravenost zdroje dat vysílat data na datovou sběrnici. Přenos slova dat proběhne pouze v případě, že SRC_RDY_N je aktivní společně s DST_RDY_N. Signál je aktivní v logické nule.

Destination Ready (DST_RDY_N) — signál indikuje připravenost cíle přijímat data na datové sběrnici. Přenosy po sběrnici probíhají pouze v případě současné aktivity SRC_RDY_N a DST_RDY_N. Signál je aktivní v logické nule.

Význam jednotlivých signálů a průběh komunikace podle tohoto protokolu ukazuje časový diagramu na obrázku 2.2. Ten zobrazuje přenos jednoho rámce, který obsahuje tři části. Délka jednotlivých částí není násobkem šířky datové sběrnice, a tak se do hry dostává signál REM. U poslední části je dokonce délka kratší než jedno datové slovo a lze tedy sledovat současnou aktivitu SOP_N a EOP_N (a dokonce i EOF_N). Během přenosu dojde dvakrát

k jeho pozastavení z důvodu nepřipravenosti komunikujících stran indikované neaktivním signálem SRC_RDY_N v prvním případě a DST_RDY_N v případě druhém.



Obrázek 2.2: Časový diagram protokolu FrameLink

2.2 Rozhraní síťových bloků

Základními rozhraními vstupních a výstupních síťových bloků jsou rozhraní směrem k LAN síti a rozhraní k ostatním funkčním blokům na čipu FPGA. Komunikace směrem k LAN síti probíhá přes rozhraní XGMII (*10 Gigabit Media Independent Interface*). Ve směru k ostatním blokům na čipu FPGA se používá rozhraní protokolu FrameLink.

Kromě přenosu dat z LAN sítě na čip FPGA a naopak je také potřeba zajistit řízení zpracování datového toku síťovými bloky. Funkce síťových bloků je dána nastavením uloženým v konfiguračních registrech. K jejich nastavení se využívá systému sběrnic platformy NetCOPE, který je k síťovým blokům připojen přes rozhraní MI32 (*32-bit Memory Interface*). Ke vstupnímu síťovému bloku jsou navíc připojeny pomocné jednotky pro vzorkování datového toku a generování kontrolních dat pro příchozí pakety. Řízení vzorkování se děje přes *Sampling Unit* rozhraní a vygenerovaná kontrolní data jsou do vstupního síťového bloku zasílána přes rozhraní PACODAG (*PAcket COntrol DAta Generator*).

2.2.1 XGMII

Rozhraní XGMII je definováno jako součást fyzické vrstvy referenčního ISO/OSI modelu. Cílem je poskytnout jednoduché a snadno implementovatelné rozhraní mezi fyzickou vrstvou a MAC vrstvou síťového modelu, které by bylo nezávislé na použitém přenosovém médiu. Hlavní charakteristiky tohoto rozhraní jsou následující

- podpora rychlosti 10 Gb/s
- plně duplexní provoz

- přenos dat a kontrolních znaků synchronně s hodinovým signálem (samostatný pro vysílání a příjem)
- 32bitové datové rozhraní oddělené pro vysílání a příjem

Ve vysílacím (*transmit*, TX) i přijímacím (*receive*, RX) směru obsahuje rozhraní 32bitový datový signál (TXD<31:0> a RXD<31:0>), dále 4bitový řídicí signál (TXC<3:0> a RXC<3:0>) a jeden bit hodinového signálu (TX_CLK a RX_CLK). Datové a řídicí vodiče jsou v obou směrech organizovány do 4 nezávislých cest obsahujících vždy 8 datových vodičů a jeden řídicí vodič. Hodnota řídicího vodiče pak udává, zda datové vodiče příslušné cesty obsahují data, nebo řídicí znak (viz dále). Hodinový signál je společný pro všechny 4 cesty v daném směru. Podrobně ukazuje organizaci rozhraní XGMII tabulka 2.2

Směr	**TXD**	**TXC**	**Cesta**	**Směr**	**RXD**	**RXC**	**Cesta**
Transmit	<7:0>	<0>	0	Receive	<7:0>	<0>	0
	<15:8>	<1>	1		<15:8>	<1>	1
	<23:16>	<2>	2		<23:16>	<2>	2
	<31:24>	<3>	3		<31:24>	<3>	3
	TX_CLK				RX_CLK		

Tabulka 2.2: Organizace rozhraní XGMII

Je-li hodnota řídicího vodiče 0, pak byte dat přenášený ve stejném hodinovém taktu přes datové vodiče odpovídající cesty obsahuje data, která by měla být interpretována ve vyšších vrstvách ISO/OSI modelu. V případě, že se na řídicím vodiči vyskytuje hodnota 1, pak datové vodiče obsahují kontrolní znak sloužící k řízení zpracování datového toku na XGMII rozhraní. Tuto situaci a přehled řídicích znaků zachycuje tabulka 2.3

TXC / RXC	**TXD / RXD**	**Význam**
0	00 až FF	Data
1	07	Idle
1	FB	Start
1	FD	Terminate
1	FE	Error

Tabulka 2.3: Význam řídicího vodiče XGMII rozhraní a přehled řídicích znaků

Organizace přenosu dat přes XGMII rozhraní má podobnou strukturu jako rámec Ethernetu, který byl popsán v kapitole 2.1.1. Tato organizace dat se nazývá *XGMII Data Stream* a její forma je následující

<inter-frame><preamble><sfd><data><efd>

<inter-frame> — během vysílání této části nejsou na datových vodičích přítomna žádná data a tato část odpovídá *Interframe Gap* mezi ethernetovými rámci. <inter-frame> začíná kontrolním znakem *Terminate*, za nímž následují kontrolní znaky *Idle*. Posledním bytem <inter-frame> je kontrolní znak *Idle* předcházející prvnímu bytu části <preamble>. Tato část zajišťuje splnění požadavků na časování a zarovnání začátku části <preamble> na cestu 0.

<**preamble**> — obsahuje 7 bytů dat, ve kterých se na sousedních bitech střídá hodnota 1 a 0. Tato část datového toku slouží především pro synchronizaci vysílající a přijímající strany.

<**sfd**> – svým obsahem i funkcí plně odpovídá části SFD (Start Frame Delimiter) popsané v kapitole 2.1.1.

<**data**> — v této části datového toku by se neměl vyskytovat žádný řídicí znak. Přenášená data by měla odpovídat formátu ethernetového rámce, respektive jeho částem Destination Address až Frame Check Sequence (viz kapitola 2.1.1).

<**efd**> — je vyvolán vysláním kontrolního znaku *Terminate* na kterékoliv ze čtyř cest.

2.2.2 FrameLink

FrameLink rozhraní slouží pro přenos dat mezi vstupními a výstupními síťovými bloky a ostatními funkčními bloky implementovanými na čipu FPGA. Popis tohoto rozhraní je součástí kapitoly 2.1.2, která popisuje protokol používaný při komunikaci přes toto rozhraní.

2.2.3 MI32

Rozhraní MI32 slouží v rámci platformy NetCOPE k připojení uživatelské komponenty k propojovacímu sytému sběrnic, konkrétně k lokální sběrnici (*Local Bus*). V roli uživatelské komponenty bude zapojen vstupní nebo výstupní síťový blok a přes rozhraní MI32 bude probíhat nastavování a čtení hodnot konfiguračních a stavových registrů síťových bloků. Seznam signálů rozhraní MI32 je uveden tabulce 2.4. Kromě označení je u každého signálu uveden také jeho směr na rozhraní uživatelské komponenty a datová šířka.

Signál	**Šířka (v bitech)**	Směr
CLK	1	výstup
DWR	32	vstup
ADDR	32	vstup
RD	1	vstup
WR	1	vstup
BE	4	vstup
DRD	32	výstup
ARDY	1	výstup
DRDY	1	výstup

Tabulka 2.4: Signály rozhraní MI32

Clock (CLK) — hodinový signál pro rozhraní MI32.

Data for Write Bus (DWR) — data zapisovaná do uživatelské komponenty.

Address Bus (ADDR) — adresa zapisovaného nebo čteného paměťového pole. Význam se rozlišuje podle hodnot signálů RD a WR.

Read Request (RD) — požadavek na čtení z nastavené adresy.

Write Request (WR) — požadavek na zápis na nastavenou adresu.

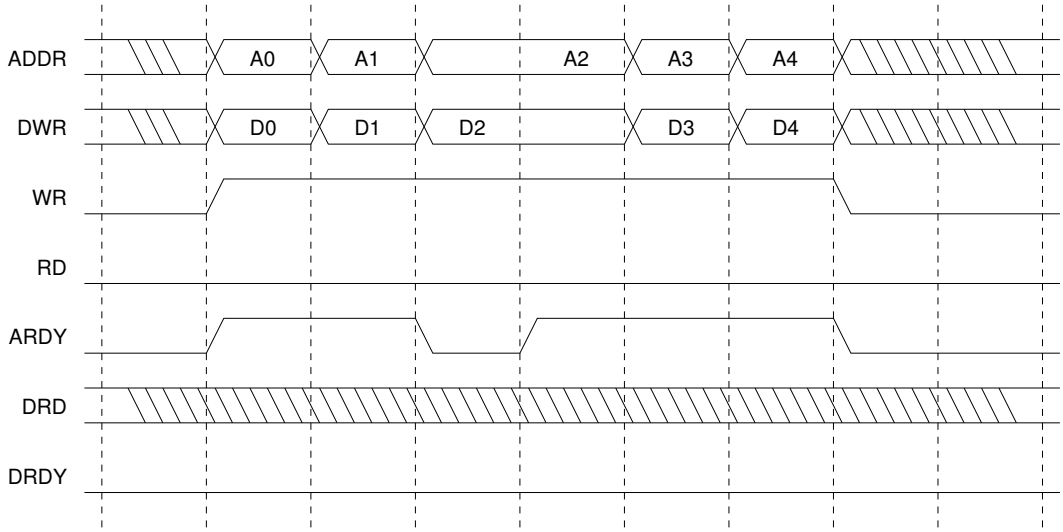
Byte Enable Bus (BE) — 4bitový signál pro určení platnosti jen některých bytů 32bitového zapisovaného nebo čteného slova.

Data for Read Bus (DRD) — data vyčtená z uživatelské komponenty.

Address Ready (ARDY) — potvrzení přijetí signálu RD nebo WR a převzetí adresy na vodičích ADDR.

Data Ready (DRDY) — signál potvrzující přítomnost přečtených dat na vodičích DRD.

Komunikace přes rozhraní probíhá synchronně s hodinovým signálem CLK. Při zápisové transakci obsahuje signál ADDR adresu, na kterou má být proveden zápis a přijetí této adresy je potvrzeno signálem ARDY. S přechodem signálu WR do aktivní úrovně pak začíná zápis dat obsažených na vodičích DWR. Časový diagram zápisové transakce je na obrázku 2.3. Čtecí transakce na rozhraní MI32 opět začíná vystavením adresy na vodičích ADDR a přijetím této hodnoty signálem ARDY. Vlastní čtení začíná s přechodem signálu RD do aktivní úrovně. Po jisté době se na vodičích DRD objeví vyčtená data a jejich platnost je potvrzena aktivní úrovní signálu DRDY. Čtecí transakci zachycuje obrázek 2.4.

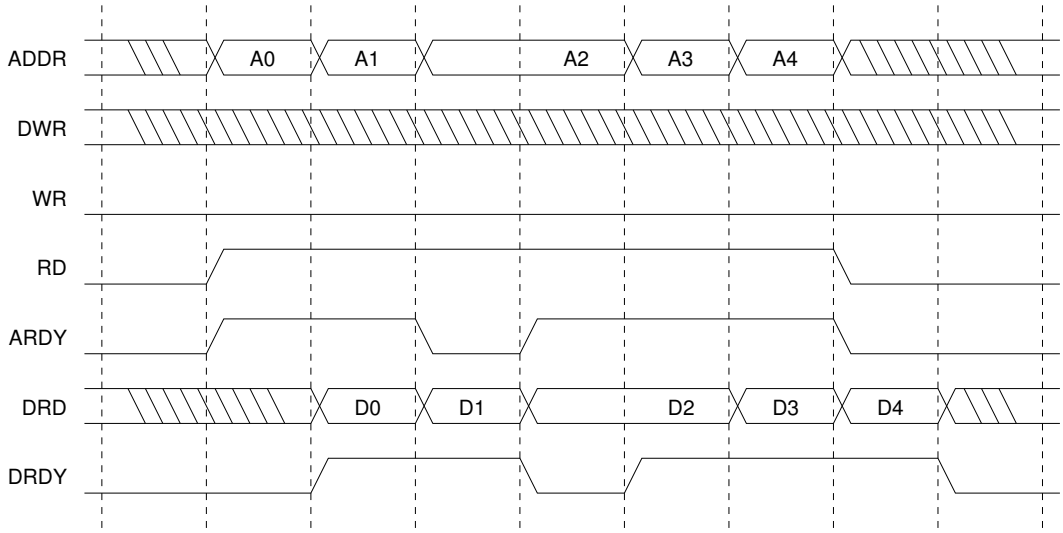


Obrázek 2.3: Zápisová transakce na rozhraní MI32

2.2.4 PACODAG

Komponenta PACODAG slouží ke generování kontrolních dat pro příchozí ethernetové rámce. Tato kontrolní data jsou pak připojena na začátek odpovídajícího rámce protokolu FrameLink (*header*) a nebo na jeho konec (*footer*). Přenos vygenerovaných kontrolních dat z PACODAG komponenty do vstupního síťového bloku probíhá přes stejnojmenné rozhraní. Přehled signálů tohoto rozhraní je uveden v tabulce 2.5.

Clock (CLK) — synchronizační hodinový signál nastavovaný vstupním síťovým blokem.



Obrázek 2.4: Čtecí transakce na rozhraní MI32

Signál	**Šířka (v bitech)**	Směr
CLK	1	výstup
DATA	$n \times 8$	vstup
DREM	$\log_2 n$	vstup
SOP_N	1	vstup
EOP_N	1	vstup
SRC_RDY_N	1	vstup
DST_RDY_N	1	výstup
SOP	1	výstup
STAT	21	výstup
STAT_DV	1	výstup
PACODAG_RDY	1	vstup

Tabulka 2.5: Signály rozhraní PACODAG

Control Data Bus (DATA) — datová sběrnice pro vygenerovaná kontrolní data. Datová sběrnice může mít různou šířku, podobně jako ve FrameLink rozhraní (viz kapitola 2.1.2).

Data Reminder Bus (DREM) — význam a funkce této sběrnice je stejná jako u sběrnice Reminder Bus (REM) v rozhraní FrameLink (kapitola 2.1.2).

Start of Part (SOP_N) — tento signál udává začátek vysílání vygenerovaných kontrolních dat na datové sběrnici. Signál je aktivní v logické nule.

End of Part (EOP_N) — signál udávající konec vysílání kontrolních dat na sběrnici DATA. Signál je aktivní v logické nule.

Source Ready (SRC_RDY_N) — signalizace připravenosti zdroje přenášet data. Za zdroj je v rámci tohoto rozhraní považována PACODAG komponenta. Signál je ak-

tivní v logické nule.

Destination Ready (DST_RDY_N) — udává připravenost cíle přijímat data na datové sběrnici. Roli cíle plní vstupní síťový blok. Signál je aktivní v logické nule.

Start of Operation (SOP) — požadavek vstupního síťového bloku na vygenerování kontrolních dat PACODAG komponentou.

Statistic Bus (STAT) — sběrnice obsahující vodiče nesoucí statistické informace. Na základě těchto statistických informací generuje PACODAG komponenta kontrolní data. Přehled signálů této sběrnice je uveden v tabulce 2.6.

Signál	**Šířka (v bitech)**	Směr
PAYLOAD_LEN	16	výstup
FRAME_ERROR	1	výstup
CRC_CHECK_FAILED	1	výstup
MAC_CHECK_FAILED	1	výstup
LEN_BELOW_MIN	1	výstup
LEN_OVER_MTU	1	výstup

Tabulka 2.6: Signály sběrnice Statistic Bus (STAT)

Statistic Data Valid (STAT_DV) — signál potvrzující platnost dat na sběrnici STAT. Po přijetí tohoto signálu začíná PACODAG komponenta generovat kontrolní data.

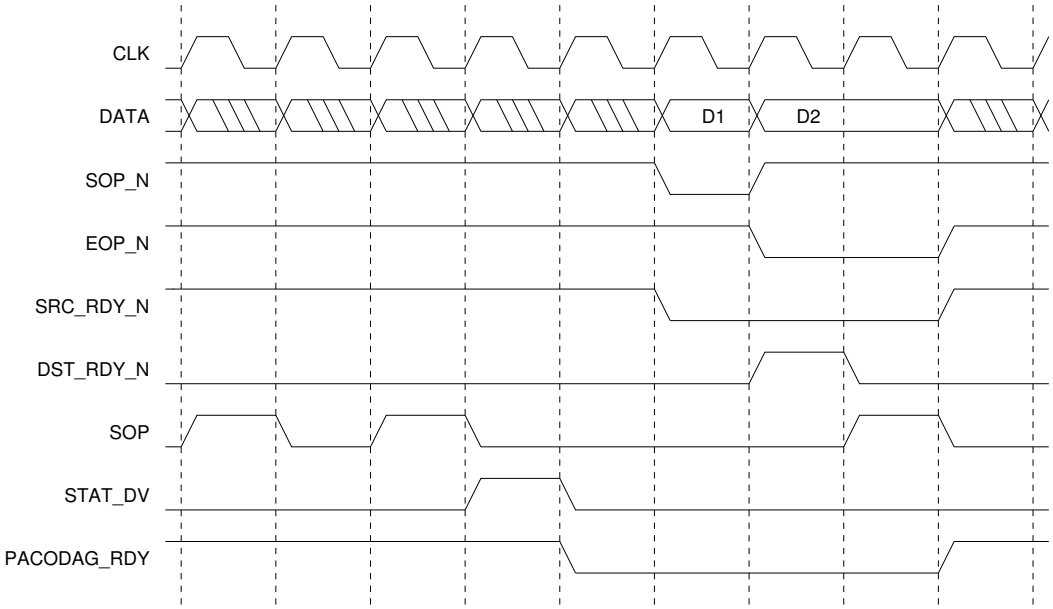
PACODAG Ready (PACODAG_RDY) — signalizace připravenosti PACODAG komponenty zpracovávat požadavky na generování kontrolních dat.

Způsob komunikace na PACODAG rozhraní je dobře patrný z časového diagramu na obrázku 2.5. Diagram nezachycuje situaci na sběrnicích STAT a DREM.

Požadavek na generování kontrolních dat je představován signálem SOP. Generování však začne pouze v případě, že jsou platná statistická data, na základě kterých je generování prováděno. Tento případ nastal až po druhém požadavku na generování. PACODAG komponenta bude nyní generovat a odesílat kontrolní data pro ethernetový rámec, a proto nebude schopna zpracovávat další požadavky na generování. Tento stav dá najevo nastavením signálu PACODAG_RDY do logické nuly. Jakmile je první část dat k dispozici, PACODAG komponenta nastaví signály SRC_RDY_N a SOP_N do logické nuly. Vstupní síťový blok je připraven data přijímat (DST_RDY_N je v nule) a přenos kontrolních dat proto začne. Druhé slovo kontrolních dat je poslední, a proto PACODAG komponenta nastaví signál EOP_N do nuly. Vstupní síťový blok však není v tomto taktu schopen data přijmout (DST_RDY_N v logické jedničce) a přenos je tak dokončen až v následujícím taktu. Ještě před dokončením přenosu však byl zaslán další požadavek na generování. Tento požadavek nebude zpracován, jelikož přišel v době, kdy nebyla PACODAG komponenta připravena jej přijmout (PACODAG_RDY v logické nule).

2.2.5 Sampling Unit

Jednotka *Sampling Unit* implementovaná na čipu FPGA a připojená ke vstupnímu síťovému bloku pomocí signálů v tabulce 2.7 umožňuje vzorkování vstupního datového toku.



Obrázek 2.5: Časový diagram komunikace na PACODAG rozhraní

Způsob vzorkování závisí na nastavení *Sampling Unit*. Lze nastavit vzorkování podle pořadí v datovém toku, délky příchozí datové jednotky a nebo pravděpodobnosti.

Signál	**Šířka (v bitech)**	Směr
SAU_REQ	1	výstup
SAU_ACCEPT	1	vstup
SAU_DV	1	vstup

Tabulka 2.7: Signály pro připojení Sampling Unit

Sampling Request (SAU_REQ) — tento signál představuje dotaz na Sampling Unit, zda má být aktuální rámec ponechán nebo zahozen.

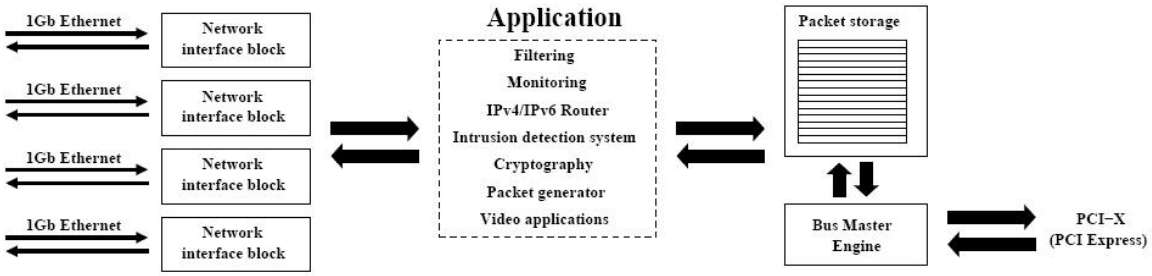
Sampling Accept (SAU_ACCEPT) — aktuální rámec má být podle nastaveného vzorkování přijat.

Sampling Information Valid (SAU_DV) — signalizace platnosti vzorkovací informace (signál SAU_ACCEPT).

2.3 Platforma NetCOPE

Platforma NetCOPE byla navržena s cílem usnadnit a urychlit vývoj hardwarově akcelerovaných síťových aplikací v rámci projektu Liberouter [1]. Snahou bylo odstínit vyvojáře síťové aplikace od nízkoúrovňových problémů, kterými jsou například příjem a vysílání dat na síťových rozhraních, rychlý přenos dat do paměti počítače a nebo řízení komponent umístěných mimo čip FPGA.

Architektura platformy je modulární a je naznačena na obrázku 2.6. Jednotlivé funkční bloky, ze který se platforma skládá, je možné samostatně vyvíjet a inovovat. Pro uživatele — návrháře síťové aplikace — představuje vývoj síťové aplikace seznamení se s rozhraním, které mu platforma NetCOPE nabízí a zasazení logiky aplikace do tohoto prostředí.



Obrázek 2.6: Architektura platformy NetCOPE

Mezi klíčové prvky této platformy patří vstupní a výstupní síťové bloky, vyrovnávací buffery a řadič přímého přístupu do paměti (*Direct Memory Access*, DMA), propojovací systém sběrnic pro rychlý přenos dat do paměti počítače, protokol FrameLink pro přenos dat mezi funkčními bloky platformy a uživatelskou aplikací, sada nástrojů pro práci s protokolem FrameLink a další. Navrhované vstupní a výstupní síťové bloky mají rozšířit možnosti platformy NetCOPE o příjem a vysílání paketů na rychlosti 10 Gb/s. Doplní tak již implementované síťové bloky, které zajišťovaly komunikaci na rychlostech definovaných pro gigabitový Ethernet.

Kapitola 3

Návrh vstupních a výstupních síťových bloků

Kapitola návrhu vstupních a výstupních síťových bloků je stěžejní jak v rámci této bakalářské práce, tak i v samotném procesu tvorby těchto jednotek a obecně jakéhokoliv produktu. Zanedbání nebo podcenění této fáze může mít katastrofální následky. Proto je této oblasti věnována zvýšená pozornost. Do návrhu jsou také zaneseny prvky, které usnadňují implementaci a udržování vytvořených vstupních a výstupních síťových bloků a pro případ chyby v návrhu minimalizují dopad tohoto problému na vyvíjený systém. Takovým prvkem je především modulárnost vyvíjeného systému. Návrh i implementace se tímto přístupem výrazně zjednodušují. Velmi podstatný je však přínos tohoto postupu při hledání a opravování chyb, které je snadné a rychlé.

Při návrhu bylo dále vycházeno z parametrů prostředí, ve kterém vstupní a výstupní síťové bloky získají uplatnění. Tímto prostředím je platforma NetCOPE, stručně popsaná v kapitole 2.3. Kromě této platformy je také nutná znalost síťových technologií používaných v sítích LAN a jejich standardů. Navrhované vstupní a výstupní síťové bloky totiž budou nasazeny právě na rozhraní těchto dvou poměrně odlišných prostředí a jejich hlavním úkolem bude zajišťovat komunikaci mezi těmito prostředími.

Kapitola o návrhu vstupních a výstupních síťových bloků v této práci se v části 3.1 zabývá identifikací požadavků na vyvíjené komponenty a to jak těch společných, tak i požadavků specifických pro každý ze síťových bloků. V kapitolách 3.2 a 3.3 jsou pak návrhy vstupních a výstupních síťových bloků detailně popsány. Popis se na každý ze síťových bloků dívá nejdříve jako na soubor vzájemně propojených a komunikujících podkomponent. V druhé části se pak popis zabývá návrhem funkcionality, která může být zajišťována i několika různými částmi vstupního nebo výstupního síťového bloku.

3.1 Požadavky na navrhované komponenty

Požadavky na vstupní a výstupní síťové bloky již byly naznačeny — tyto komponenty by měly sloužit k propojení prostředí LAN sítě a platformy NetCOPE implementované na čipu FPGA. Tento obecný požadavek musí být samozřejmě zachován, ale je důležité jej podrobně rozpracovat do dílčích požadavků.

Některé z požadavků se budou hodit jak pro vstupní, tak i pro výstupní síťový blok. Většina se jich však bude vztahovat k jednomu konkrétnímu síťovému bloku. Ačkoliv je jejich funkce do jisté míry opačná (příjem × vysílání), nelze jeden z druhého vytvořit jen

otočením pořadí podkomponent. Na každý ze vstupních a výstupních síťových bloků jsou kladeny specifické požadavky a v rámci této práce se jimi proto zabývají dvě oddělené kapitoly.

3.1.1 Společné požadavky

Požadavky společné pro oba navrhované síťové bloky se týkají především rozhraní těchto bloků a zajištění možnosti propojení s dalšími komponentami umístěnými na čipu FPGA. Mezi požadavky společné pro oba síťové bloky patří

- standardní XGMII rozhraní směrem k LAN síti
- FrameLink rozhraní ke komponentám na čipu FPGA
- přístup ke konfiguračním a stavovým registrům přes lokální sběrnici a rozhraní MI32
- synchronizace externím hodinovým signálem (společný s dalšími komponentami platformy NetCOPE)
- bufferování přijímaných a odesílaných dat

Popis zmiňovaných rozhraní síťových bloků a důvody pro jejich použití byly uvedeny již v kapitole 2.2. Požadavek na sychronizaci hodinovým signálem sdíleným s ostatními komponentami platformy NetCOPE jednak umožňuje synchronní komunikaci mezi jednotlivými komponentami a také dovoluje použití jednoho kvalitního zdroje hodinového signálu. Díky dobré infrastruktuře pro rozvod hodinového signálu po čipu FPGA je možné takto generovaný hodinový signál bez větších problémů rozvést ke všem komponentám platformy NetCOPE. Konečně zajištění bufferování přijímaných a odesílaných dat je požadavek vyplývající z nutnosti dalšího zpracování přijímaných a odesílaných dat a je běžnou praxí ve všech síťových zařízeních.

3.1.2 Požadavky na vstupní síťový blok

Vstupní síťové bloky jsou specifické prováděním množství kontrol nad příchozími ethernetovými rámci. Také jsou první filtrační jednotkou, která k dalšímu zpracování propouští pouze bezchybná a požadovaná data. Pro další zpracování jsou navíc k těmto datům přidávány informace o výsledcích některých prováděných kontrol. Základní požadavky specifické pro vstupní síťové bloky jsou shrnuté v následujícím seznamu

- maskovatelná kontrola příchozích rámců na
  - správnou hodnotu Destination Address
  - správnost hodnoty Frame Check Sequence
  - dodržení limitů pro velikost rámce
  - výskyt chyby na XGMII rozhraní
- různé módy kontroly pole Destination Address
  - povolení všech MAC adres
  - povolení MAC adres uložených v paměti

  - povolení MAC adres z paměti a *broadcastové*
  - povolení MAC adres z paměti, *broadcastové* a *multicastových*
- zahazování příchozích rámců na základě maskovatelných výsledků kontrol nebo vzorkování
- čítače rámců
  - všech přijatých
  - správných
  - zahozených
  - zahozených na základě přeplnění bufferů
- volitelné odstranění pole Frame Check Sequence
- přidání kontrolních dat vygenerovaných komponentou PACODAG

Kontrola příchozích rámců je jedním z hlavních úkolů vstupního síťového bloku. Nad příchozími rámci je možné provádět množství kontrol, které následně ovlivňují to, zda bude rámec propuštěn do dalšího zpracování či nikoliv. Výsledky těchto kontrol by mělo být možné pro potřeby zahazování maskovat a umožnit tak propuštění také chybných rámců (např. kvůli jejich další analýze).

Samostatně nastavitelná by měla být kontrola správné hodnoty v poli Destination Address. Vstupní síťový blok by tak mělo být možné nastavit například do tzv. promiskuitního módu, při kterém není brán na hodnotu cílové MAC adresy zřetel a lze tedy přijímaty i rámce, které byly zaslány někomu jinému.

Důležitou informací o fungování vstupního síťového bloku jsou hodnoty čítačů rámců. Kvůli možnosti zahazování rámců je však vhodné používat čítačů několik a uchovávat tak samostatně informace o celkovém počtu přijatých rámců, o počtu správných a zahozených rámců a také o počtu rámců zahozených z důvodu přeplnění bufferů. Tyto hodnoty mohou následně pomoct při odhalování problémů se síťovou komunikací.

Kromě již výše zmíněných požadavků by měly implementované vstupní síťové bloky být schopné volitelně (na základě generického parametru) odstranit pole Frame Check Sequence. Toho může být využito v opačném významu, kdy bude žádoucí zachovat toto pole pro analýzu dalšími komponentami, případě softwarovým nástrojem. Vstupní síťový blok by také měl být schopen do výstupního rámce protokolu FrameLink přidat kontrolní data vygenerovaná komponentou PACODAG.

3.1.3 Požadavky na výstupní síťový blok

Oproti vstupním síťovým blokům jsou požadavky na výstupní síťové bloky jednodušší. Se vstupními daty již není prováděno takové množství operací jako u vstupního síťového bloku a tato „jednoduchost“ se projevuje jak v uvedeném seznamu požadavků, tak i v návrhu struktury výstupního síťového bloku. Požadavky na výstupní síťový blok jsou následující

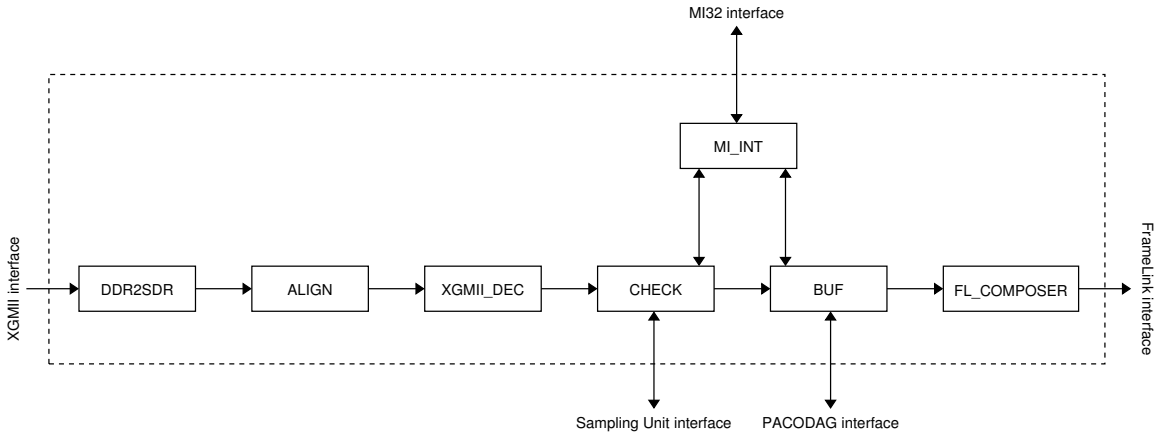
- nahrazení zdrojové MAC adresy či zahození rámce podle FrameLink *footer*
- konfigurovatelná MAC adresa pro odchozí rámce
- čítače rámců

  - všech zpracovaných
  - odeslaných
  - zahozených
- nastavení správné hodnoty pole Frame Check Sequence

Zpracování rámců, které jsou do výstupního síťového bloku dopravovány přes FrameLink rozhraní, by se mělo řídit kontrolními daty, které jsou v rámci FrameLink rámce uvedena za vlastními daty rámce (*footer*). Na základě těchto dat může být rámec zahozen, nebo mu může být nastavena předkonfigurovaná zdrojová MAC adresa. Odchozímu rámci je také vypočítána a nastavena hodnota pole Frame Check Sequence. Stejně jako u vstupního síťového bloku, i tady budou využity čítače rámců reflektující fungování výstupního síťového bloku.

3.2 Architektura vstupního síťového bloku

Podle požadavků, které vyplývají z kapitol 3.1.1 a 3.1.2, byl navržen vstupní síťový blok, jehož popis je obsahem této kapitoly. Navržená struktura je modulární, skládající se z několika podkomponent (viz obrázek 3.1). Každá podkomponenta zajišťuje provádění stanovených operací, které budou popsány dále.



Obrázek 3.1: Bloková struktura vstupního síťového bloku

3.2.1 Rozhraní a generické parametry

Jednotlivá rozhraní a způsoby komunikace přes tato rozhraní byly popsány již v kapitole 2.2. Na vstupním síťovém bloku — narozdíl od výstupního síťového bloku — se uplatní všech pět popsaných rozhraní. Kromě těchto signálů bude do vstupního síťového bloku přiveden ještě synchronizační hodinový signál a resetovací signál synchronní s použitým hodinovým signálem.

Za určitý typ rozhraní lze také považovat generické parametry jazyka VHDL — *VHSIC* (Very High Speed Integrated Circuits) *Hardware Description Language*. Pomocí těchto parametrů lze ovlivňovat architekturu výsledné syntetizované komponenty a lze si je tedy

představit jako parametry softwarových produktů používané při jejich spouštění. Koncept generických parametrů bude využit u vstupních i výstupních síťových bloků.

U vstupního síťového bloku slouží generické parametry především k ovlivnění velikosti používaných bufferů, indikaci struktury používaných rámců protokolu FrameLink, ovlivnění odstraňení nebo ponechání pole Frame Check Sequence a nebo k nastavení šířky datové sběrnice protokolu FrameLink. Přehled generických parametrů vstupního síťového bloku je uveden v tabulce 3.1, která je následována podrobnějším popisem významu jednotlivých parametrů.

Parametr	Typ	Defaultní hodnota
DFIFO_SIZE	integer	1024
HFIFO_SIZE	integer	128
HFIFO_MEMTYPE	mem_type	LUT
CTRL_HDR_EN	boolean	true
CTRL_FTR_EN	boolean	false
MACS	integer	16
INBANDFCS	boolean	true
FL_WIDTH_TX	integer	64
FL_RELAY	boolean	true

Tabulka 3.1: Generické parametry vstupního síťového bloku

DFIFO_SIZE — tento parametr určuje počet položek, které je možné uložit do datového bufferu typu FIFO použitého ve vstupním síťovém bloku. Velikost jedné položky je dána součtem šířky datové sběrnice rozhraní FrameLink a šířky řídicích signálů. Parametr musí mít hodnotu 2^n pro $n \geq 2$.

HFIFO_SIZE — představuje počet položek, které je možné uložit do bufferu pro kontrolní data generovaná komponentou PACODAG. Šířka jedné položky je stejná jako u datového bufferu.

HFIFO_MEMTYPE — určuje typ paměťových prvků na čipu FPGA, ze kterých je poskládán buffer pro kontrolní data. Přípustné hodnoty jsou LUT (*look-up table*), SRL16 (*16-bit shift register look-up table*) a BRAM (*block RAM*).

CTRL_HDR_EN — signalizuje použití kontrolních dat umístěných před vlastními daty v rámci protokolu FrameLink (*header*).

CTRL_FTR_EN — význam je podobný jako u parametru CTRL_HDR_EN. Tentokrát se však týká kontrolních dat umístěných za vlastními daty (*footer*).

MACS — hodnota tohoto parametru udává počet MAC adres, které lze uložit do paměti.

INBANDFCS — na základě tohoto parametru se provádí rozhodnutí, zda se má odstranit pole Frame Check Sequence. Je-li hodnota parametru true, pole FCS je ponecháno spolu s obsahem rámce pro analýzu dalšími komponentami. V opačném případě je pole FCS odstraněno.

FL_WIDTH_TX — tento parametr určuje šířku datové sběrnice výstupního FrameLink rozhraní. Šířka je udávána v bitech a hodnota parametru by měla být minimálně 64.

FL_RELAY — určuje, zda bude komponenta FrameLink Relay (viz [12]) ve vstupním síťovém bloku umístěna (true) a nebo ne (false).

3.2.2 Popis podkomponent

Jednotlivé funkční bloky, ze kterých se vstupní síťový blok skládá, jsou zobrazeny na obrázku 3.1. Každý z bloků implementuje část operací nad příchozím datovým tokem, které jsou od vstupního síťového bloku vyžadovány. Náročnost vyžadovaných operací se však liší a složitost některých funkčních bloků je tak vyšší. K této rozdílnosti přispívá také fakt, že některé bloky zajišťují více než jednu požadovanou funkci. Následující seznam proto poskytuje přehled funkcí zajišťovaných jednotlivými bloky z obrázku 3.1.

DDR2SDR — úkolem tohoto bloku je převod XGMII rozhraní z 32bitového DDR (*Double Data Rate*) na 64bitové SDR (*Single Data Rate*). Tento převod provádí jednotka implementovaná společností Xilinx. Popis této implementace lze nalézt v [9].

ALIGN — hlavní motivací pro nasazení tohoto bloku byla snaha usnadnit práci s datovým tokem produkovaným jednotkou DDR2SDR. Po převodu 32bitové datové šířky na 64 bitů je potřeba provést zarovnání na 64bitová datová slova. Společně s tímto úkolem zde také dochází k detekci počátku rámce, vzhledem ke kterému se zarovnání provádí a který bude s výhodou využit v následujících blocích.

XGMII_DEC — usnadňuje práci se vstupním datovým tokem, když ho převádí z formátu, který se používá na rozhraní XGMII, do protokolu podobného protokolu FrameLink. V tomto formátu jsou data přenášena a používána v dalších částech vstupního síťového bloku. Podrobnější popis převodu je součástí kapitoly 3.2.5.

CHECK — tento blok lze označit za jeden z klíčových v rámci vstupního síťového bloku. Zde se provádí veškeré kontroly datového toku. K tomu je nutné propojení také se *Sampling Unit* a blokem MI_INT, který obsahuje některé registry ovlivňující kontroly a také paměť CAM (*Content-Addressable Memory*), ve které jsou uloženy MAC adresy. Výsledky jednotlivých kontrol jsou představovány hodnotami bitů statistického signálu, který spolu s daty putuje do dalšího funkčního bloku. Podrobněji se kontrolám datového toku věnuje kapitola 3.2.4. Další funkcí tohoto bloku je volitelné odstraňování pole Frame Check Sequence na základě generického parametru INBANDFCS (viz kapitola 3.2.1).

MI_INT — zajišťuje především připojení vstupního síťového bloku k rozhraní lokální sběrnice MI32. Přes toto rozhraní lze nastavovat a vyčítat hodnoty konfiguračních a stavových registrů. Hodnoty uložené v těchto registrech ovlivňují a monitorují fungování bloků CHECK a BUF, se kterými je jednotka MI_INT propojena. Podrobný popis jednotlivých registrů je uveden v kapitole 3.2.3. Součástí tohoto bloku je také paměť CAM na MAC adresy, které se porovnávají s polem Destination Address ethernetového rámce.

BUF — druhou ze dvou klíčových podkomponent vstupního síťového bloku je BUF. Podobně jako v části CHECK, i zde je soustředěno více funkcí do jednoho bloku. Na základě statistik z části CHECK je rozhodnuto, zda bude rámec propuštěn k dalšímu zpracování, nebo zahozen. Hodnoty statistik však lze vymaskovat a je tedy možné dosáhnout i zpracování dat obsahujících chybu. Výsledek rozhodování je zaznamenán

do čítačů v bloku MI_INT a rámce určené pro další zpracování jsou uloženy do bufferů. Kontrolní data vygenerovaná externí komponentou PACODAG jsou uložena do samostatného bufferu.

FL_COMPOSER — tento poslední blok slouží k vytvoření datového toku odpovídajícího protokolu FrameLink. Vstupem jsou vlastní data z ethernetového rámce a také kontrolní data vygenerovaná komponentou PACODAG. Na základě generických parametrů CTRL_HDR_EN a CTRL_FTR_EN (viz kapitola 3.2.1) je z těchto dvou vstupů vytvořen správný rámec protokolu FrameLink, jsou vygenerovány chybějící signály a výsledek je zaslán na výstupní FrameLink rozhraní.

3.2.3 Registry a adresový prostor

Sada konfiguračních a stavových registrů je součástí jednotky MI_INT. Hodnoty uložené v konfiguračních registrech ovlivňují fungování vstupního síťového bloku, kontroly prováděné nad datovým tokem a maskování výsledků kontrol při zahazování rámců. Stavové registry zobrazují důležité informace o fungování vstupního síťového bloku. Hlavní stavové registry uchovávají hodnoty čítačů rámců. Následující přehled popisuje konfigurační a stavové registry vstupního síťového bloku.

Total Received Frames Counter – spodní část (TRFCL) a horní část (TRFCH) hodnota čítače všech přijatých rámců je uložena v registrovém párů složeném ze dvou 32bitových registrů. Spodní část (TRFCL) uchovává bity <31:0> a v horní části (TRFCH) se nacházejí bity <63:32>.

Correct Frames Counter – spodní část (CFCL) a horní část (CFCH) — čítač zaznamenaný v těchto registrech je také 64bitový a organizace uložených hodnot je obdobná jako u předešlých registrů. Hodnota uložená v těchto registrech vyjadřuje počet rámců, u kterých nebyla zjištěna chyba.

Discarded Frames Counter – spodní část (DFCL) a horní část (DFCH) — v této dvojici registrů je obvyklým způsobem uložena hodnota čítače rámců, které byly zahozeny.

Counter of Frames Discarded due to Buffer Overflow – spodní a horní část — poslední dvojice registrů sloužící k uložení hodnoty 64bitového čítače vyjadřuje počet rámců, které musely být zahozeny z důvodu zaplnění bufferů. Uložená hodnota je obvyklým zůsobem rozdělena na dvě části.

Enable Register — v tomto registru je význam přiřazen pouze jeho nejnižšímu bitu. Tímto bitem je řízeno povolení (hodnota 1) nebo zakázání (hodnota 0) činnosti vstupního síťového bloku.

Error Mask Register — jednotlivé bity tohoto registru určují, zda chyba nalezená při odpovídající kontrole příchozího rámce způsobí jeho zahození (hodnota 1) a nebo bude tato chyba vymaskována (hodnota 0) a rámec projde do dalšího zpracování. Významné bity tohoto registru jsou popsané v tabulce 3.2.

Status Register — bity tohoto registru signalizují aktuální stav vstupního síťového bloku. Jejich význam je patrný z tabulky 3.3.

Pozice	Název	Kontrola
0	XGMII_ERROR	chyba na XGMII rozhraní
1	CRC_ERROR	CRC v poli Frame Check Sequence
2	MIN_LEN_CHECK	minimální délka rámce (viz Minimum Frame Length Allowed)
3	MTU_CHECK	maximální délka rámce (viz Frame MTU)
4	MAC_CHECK	povolená MAC adresa v poli Destination Address

Tabulka 3.2: Význam bitů v Error Mask Register

Pozice	Název	Význam
0	PACODAG_OVF	některý z požadavků na generování kontrolních dat nebyl zpracován
1	DFIFO_OVF	došlo k přeplnění datového bufferu
4 – 5	IBUF speed	udává podporovanou rychlost (11 = 10 Gb/s)

Tabulka 3.3: Význam bitů ve Status Register

Command register — nejnižší byte tohoto registru slouží k zápisu speciálních hodnot — příkazů — které ovlivňují hodnoty registrů čítačů, případně hodnoty samotných čítačů. Přehled těchto speciálních hodnot ukazuje tabulka 3.4.

Název	Hodnota	Význam
IBUFCMD_STROBE_COUNTERS	0x01	uložení aktuální hodnoty čítačů rámců do registrů
IBUFCMD_RESET_COUNTERS	0x02	vynulování aktuální hodnoty čítačů rámců

Tabulka 3.4: Příkazy pro práci s čítači rámců

Minimum Frame Length Allowed — hodnota na bitech <15:0> tohoto registru udává minimální povolenou velikost přijatého ethernetového rámce v bytech. Defaultní nastavená hodnota je 64 bytů.

Frame MTU — na spodních dvou bytech tohoto registru je možné nastavit hodnotu MTU (*Maximum Transmission Unit*) pro příchozí rámce. Defaultně je nastavena hodnota 1526 (bytů).

MAC Address Check Mode — v tomto registru se nastavuje způsob kontroly pole Destination Address. K výběru módu slouží nejnižší dva bity registru. Přesný význam hodnot zapsaných na tyto pozice shrnuje tabulka 3.5.

Memory of Available MAC Addresses — toto paměťové místo slouží k ukládání MAC adres, které se porovnávají s polem Destination Address při kontrole MAC adresy v módu 1, 2 a 3. Každá zapsaná adresa je uložená ve dvou 32bitových paměťových buňkách. Na nižší adrese jsou uloženy nižší 4 byty MAC adresy a vyšší adresa obsahuje horní dva byty MAC adresy. Na dalším bitu je pak uložen tzv. valid bit, který udává

Hodnota	Mód kontroly	Význam
0x0	mód 0 (promiskuitní)	všechny MAC adresy projdou
0x1	mód 1	platné jsou MAC adresy uložené v paměti (Memory of Available MAC Addresses)
0x2	mód 2	mód 1 + broadcastová adresa
0x3	mód 3	mód 2 + multicastové adresy

Tabulka 3.5: Módy kontroly Destination Address

platnost MAC adresy a určuje také její zapojení do porovnávání. Zápis MAC adresy do paměti musí proběhnout ve dvou fázích, kdy v první fázi se zapisuje spodních 32 bitů a ve druhé fázi je zapisováno zbývajících 16 bitů a valid bit. Maximální počet MAC adres v Memory of Available MAC Addresses je dán hodnotou generického parametru MACS (viz kapitola 3.2.1).

Pro zápis a čtení hodnot konfiguračních a stavových registrů se používá rozhraní MI32. Šířka datové sběrnice tohoto rozhraní je 32 bitů, a proto jsou také registry organizovány do paměťových polí po 32 bitech. Přesná adresa registrů vstupního síťového bloku záleží na volbě bázové adresy, na které bude začínat adresový prosto této komponenty. Adresa registrů v rámci tohoto adresového prostoru je však vždy stejná a ukazuje ji tabulka 3.6

Adresa	Registr
0x00	Total Received Frames Counter – spodní část (TRFCL)
0x04	Correct Frames Counter – spodní část (CFCL)
0x08	Discarded Frames Counter – spodní část (DFCL)
0x0C	Counter of Frames Discarded due to Buffer Overflow – spodní část
0x10	Total Received Frames Counter – horní část (TRFCH)
0x14	Correct Frames Counter – horní část (CFCH)
0x18	Discarded Frames Counter – horní část (DFCH)
0x1C	Counter of Frames Discarded due to Buffer Overflow – horní část
0x20	Enable Register
0x24	Error Mask Register
0x28	Status Register
0x2C	Command Register
0x30	Minimum Frame Length Allowed
0x34	Frame MTU
0x38	MAC Address Check Mode
0x80	Memory of Available MAC Addresses

Tabulka 3.6: Adresový prostor vstupního síťového bloku

Jak bylo uvedeno u popisu Command Register, pro práci s hodnotami registrů čítačů se používá speciální příkaz zapisovaný právě do Command Register. Tento přístup je nutný proto, že hodnoty registrů čítačů lze při přístupu přes rozhraní MI32 pouze číst. Druhou podobnou výjimkou je právě Command Register, který je přes rozhraní MI32 dostupný

pouze pro zápis. Všechny ostatní registry jsou dosputné jak pro čtení, tak pro zápis.

3.2.4 Kontroly datového toku

Kontrola příchozích rámců a jejich případné zahazování je hlavní funkcí, kterou vstupní síťový blok vykonává. Tato část zpracování příchozího datového toku je poměrně propracovaná. Takto složitý úkol však nemůže být vykonáván najednou a v jediné části vstupního síťového bloku.

Kontrola příchozích rámců začíná již v části XGMII_DEC, kde jsou detekovány případné chyby XGMII rozhraní. V části CHECK lze nalézt jádro kontrolního mechanismu. Zde probíhají kontroly následujících parametrů ethernetových rámců

- pole Destination Address
- minimální vyžadovaná délka rámce
- maximální povolená délka rámce
- správná hodnota pole Frame Check Sequence

Hodnota v poli Destination Address je kontrolována v závislosti na módu zvoleném v registru MAC Address Check Mode. Může být zvolen promiskuitní mód (všechny adresy jsou platné), nebo některý z dalších módů, které množinu přípustných MAC adres různým způsobem omezují (viz kapitola 3.2.3). Minimální a maximální délka rámce jsou porovnávány s hodnotami nastavenými v registrech Minimum Frame Length Allowed a Frame MTU. Kontrola pole FCS je prováděna vlastním výpočtem hodnoty CRC, která se následně porovná s hodnotou v poli FCS.

Poslední kontrolou v části CHECK je dotaz na *Sampling Unit*, zda má být aktuální rámec zahozen či nikoliv. Informace o provedených kontrolách jsou následně společně s ethernetovými rámci zaslány do části BUF, kde probíhá jejich zahazování, případně uložení do bufferů. Zahazování se děje na základě informací o provedených kontrolách. Některé informace je však možné vymaskovat (podrobněji viz tabulka 3.7) nastavením správných bitů v Error Mask Register. Do bufferů tak může být uložen i rámec obsahující chybu, kterou uživatel vymaskoval.

Problém	**Maskovatelné**
přetečení bufferů	ne
vzorkování datového toku	ne
chyba XGMII rozhraní	ano
kontrola pole FCS	ano
kontrola MAC adresy	ano
kratší rámec než povolený	ano
rámec větší než povolený	ano

Tabulka 3.7: Maskovatelnost prováděných kontrol

Při zpracování rámců v části BUF (zahození nebo uložení do bufferů) jsou také upravovány hodnoty čítačů rámců. S každým novým rámcem je o 1 zvýšena hodnota čítače Total Received Frames Counter. Je-li rámec v pořádku, zvýší se také hodnota Correct Frames

Counter. V opačném případě je navýšen čítač Discarded Frames Counter. Speciálním případem je situace, kdy dojde k zahození rámce kvůli zaplnění bufferů. V takovýchto případech se navyšuje hodnota čítače Counter of Frames Discarded due to Buffer Overflow.

3.2.5 Transformace na protokol FrameLink

Druhou základní funkcí vstupního síťového bloku, která je kvůli své náročnosti rozdělena na několik dílčích úloh a implementována ve více blocích, je převod dat z formátu na rozhraní XGMII do fomátu platného pro rozhraní protokolu FrameLink.

Převod začíná již v prvním bloku za vstupním XGMII rozhraním. Blok DDR2SDR provádí konverzi komunikace DDR na SDR, což s sebou přináší také zdvojnásobení výstupní datové šířky tohoto bloku. Datová sběrnice rozhraní XGMII má 32 bitů (viz kapitola 2.2.1) a na výstupu bloku DDR2SDR tak musí být datová šířka 64 bitů. Tímto převodem je tedy dosaženo také převedení datové šířky vstupního rozhraní vstupního síťového bloku na šířku datové sběrnice výstupního FrameLink rozhraní (v případě hodnoty generického parametru FL_WIDTH_TX = 64).

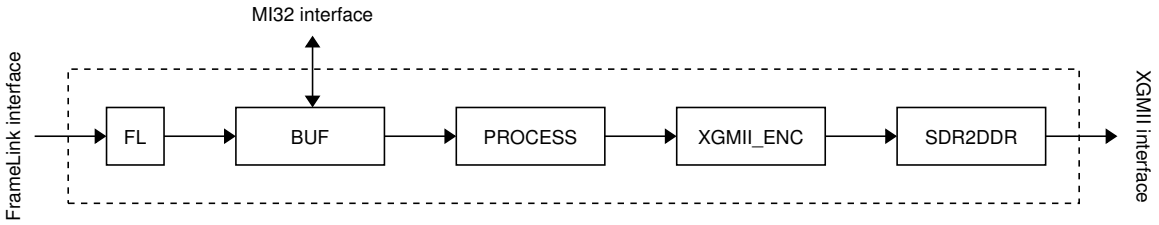
V následujících blocích jsou formovány signály, které se podobají některým signálům FrameLink rozhraní. V bloku ALIGN je v rámci zarovnání na 64bitová slova detekován začátek rámce a odpovídajícím způsobem je generován signál SOP. Část XGMII_DEC přidává k signálu SOP signály EOP a EOP_POS. EOP značí konec rámce a až na převrácený význam hodnot odpovídá signálu EOP_N protokolu FrameLink. Signál EOP_POS má podobný význam jako REM a je platný pouze se současně aktivním signálem EOP. Na výstupu bloku XGMII_DEC tak nalézáme signály, které odpovídají zjednodušené variantě protokolu FrameLink. Rámec tohoto upraveného protokolu má vždy jen jednu část (postačují signály SOP a EOP) a součástí protokolu nejsou signály SRC_RDY_N a DST_RDY_N.

Další transformace probíhá v bloku BUF vstupního síťového bloku. Signály zjednodušeného protokolu FrameLink, které můžeme nelézt na výstupu XGMII_DEC, jsou převedeny na signály plně odpovídající protokolu FrameLink. Pro SOP a EOP to tedy znamená inverzi významu logických hodnot a vznik SOP_N a EOP_N. Signál EOP_POS stačí pouze přejmenovat na REM. Volitelně ja také v bloku BUF možné upravovat šířku datové sběrnice. Tato transformace se však uplatní pouze v případě, že výstupní datová šířka protokolu FrameLink je větší než šířka datové sběrnice vedoucí z bloku DDR2SDR.

Konečnou fázi transformace na výstupní FrameLink protokol zajišťuje blok FL_COMPOSER. Ten se stará o zformování platného FrameLink rámce, jehož datová část se skládá z ethernetového rámce, kterou předchází nebo následuje část obsahující kontrolní data vygenerovaná komponentou PACODAG. Pro správnou funkci protokolu FrameLink je také generován signál SRC_RDY_N a signály označující začátek a konec rámce (SOF_N, EOF_N), respektive jeho částí (SOP_N, EOP_N).

3.3 Architektura výstupního síťového bloku

Oproti vstupnímu síťovému bloku je architektura výstupního síťového bloku jednodušší. Na výstupní síťový blok není kladeno tolik požadavků (viz kapitola 3.1.3), a proto není jeho struktura tak složitá. To ukazuje také obrázek 3.2, na kterém je zachycena bloková struktura výstupního síťového bloku. Návrh je tedy opět modulární a jednotlivé části provádějí nad datovým tokem stanovené operace.



Obrázek 3.2: Bloková struktura výstupního síťového bloku

3.3.1 Rozhraní a generické parametry

Jak je vidět z blokové stuktury výstupního síťového bloku na obrázku 3.2, tato komponenta má pouze tři rozhraní. Jedná se o vstupní FrameLink rozhraní, výstupní XGMII rozhraní a rozhraní MI32 sloužící pro připojení k lokální sběrnici platformy NetCOPE. Popis všech těchto rozhraní lze nalézt v kapitole 2.2.

Stejně jako u vstupního síťového bloku, i zde byl uplatněn koncept generických parametrů, pomocí kterých lze jednoduše nastavovat základní parametry výstupního síťového bloku a ovlivňovat tak výslednou strukturu této jednotky. Přehled generických parametrů je uveden v tabulce 3.8, za kterou následuje podrobný popis jednotlivých generických parametrů. V porovnání se vstupním síťovým blokem je zde počet generických parametrů nižší. To však jen dokládá rozdíl ve složitosti mezi vstupním a výstupním síťovým blokem.

Parametr	Typ	Defaultní hodnota
CTRL_CMD	boolean	true
FL_WIDTH_RX	integer	64
DFIFO_SIZE	integer	1024
HFIFO_SIZE	integer	256
HFIFO_MEMTYPE	mem_type	LUT

Tabulka 3.8: Generické parametry výstupního síťového bloku

CTRL_CMD — hodnota tohoto parametru určuje, zda rámce na vstupním FrameLink rozhraní obsahují kontrolní data za datovou částí rámce (*footer*), nebo jsou přenášena pouze vlastní data. V případě, že FrameLink rámec obsahuje také kontrolní data, tato jsou využita pro určení operací, které mají být s daným rámcem provedeny.

FL_WIDTH_RX — tento generický parametr určuje šířku datové sběrnice vstupního XGMII rozhraní. Šířka se udává v bitech a povolené hodnoty se řídí pravidly stanovenými v rámci protokolu FrameLink (viz kapitola 2.1.2). Nastavená hodnota by také neměla být menší než nastavená defaultní hodnota.

DFIFO_SIZE — udává počet položek, které je možné uložit do datového bufferu umístěného ve výstupním síťovém bloku. Pro velikost jedné položky a povolené hodnoty velikosti tohoto bufferu viz kapitolu 3.2.1.

HFIFO_SIZE — parametr stanovuje maximální počet položek, které lze uložit do bufferu pro kontrolní data přenášená ve FrameLink rámci, v případě nastaveného parametru CTRL_CMD. Šířka jedné ploţky je 1 bit.

HFIFO_MEMTYPE — hodnota tohoto parametru udává typ paměťových prvků na čipu FPGA, ze kterých bude poskládán buffer pro kontrolní data z FrameLink rámce. Typ tohoto parametru a přípustné hodnoty jsou popsané v kapitole 3.2.1.

3.3.2 Popis podkomponent

Výstupní síťový blok se skládá z několika dílčích bloků, které vykonávají jednotlivé funkce požadované od tohoto bloku jako celku. Tento modulární přístup výrazně usnadňuje návrh, implementaci i údržbu výstupního síťového bloku a přispívá také ke snadnějšímu řešení chyb. Význam jednotlivých částí výstupního síťového bloku je popsán v následujícím přehledu.

FL — první část výstupního síťového bloku má za úkol na základě hodnoty generického prametru CTRL_CMD (viz kapitola 3.3.1) rozdělit příchozí FrameLink rámce na datovou část a část obsahující kontrolní data. Tyto části jsou následně v bloku BUF uloženy do bufferů, které jsou řízeny automatem umístěným v části FL.

BUF — hlavním úkolem bloku BUF je poskytnout buffery pro uložení dat před jejich odesláním. Jak již bylo zmíněno, řízení těchto bufferů obstarává automat umístěný v předchozí části výstupního síťového bloku. Kromě bufferování zajišťuje tato část také transformaci datového toku na výstupní datovou šířku 64 bitů. Transformace se však uplatní pouze v případě, je-li šířka datové sběrnice na vstupním FrameLink rozhraní jiná než 64 bitů. Posledním úkolem tohoto bloku je zajištění připojení rozhraní MI32 k výstupnímu síťovému bloku. S tím je také spojeno umístění konfiguračních a stavových registrů. Tyto registry se nacházejí právě v této podkomponentě.

PROCESS — tento blok je klíčovou podkomponentou výstupního síťového bloku. Nejprve je vypočítána délka příchozího rámce a v případě potřeby je k rámci přidána položka Pad (viz kapitola 2.1.1), která jeho velikost doplní na hodnotu 60 bytů. Následně může být změněna zdrojová MAC adresa rámce na hodnotu nastavenou v registru MAC Address Register. V poslední fázi je pro zpracovávaný rámec vypočítán kontrolní součet CRC.

XGMII_ENC — v předposlední části výstupního síťového bloku dochází k převodu dat do formy předepsané rozhraním XGMII (viz kapitola 2.2.1). K vysílanému ethernetovému rámci je správným způsobem připojeno pole Frame Check Sequence obsahující kontrolní součet CRC vypočítaný v bloku PROCESS. Tento kompletní ethernetový rámec je poté pomocí automatu překódován na sekvenci datových a řídicích znaků na datové sběrnici rozhraní XGMII a odpovídajícím způsobem jsou nastaveny také hodnoty bitů řídicí sběrnice.

SDR2DDR — poslední úpravou zformovaného *XGMII Data Stream* je převod ze 64bitové SDR komunikace na 32bitovou DDR komunikaci. Pro převod opačným směrem byla ve vstupním síťovém bloku použita jednotka implementovaná společností Xilinx. Nejinak tomu je u výstupního síťového bloku, kde je použita vysílací jednotka ukázkového designu popsaného v [9].

3.3.3 Registry a adresový prostor

V rámci výstupního síťového bloku jsou konfigurační a stavové registry sloužící k nastavení a monitorování funkce komponenty umístěny v bloku BUF. Přístup k těmto registrům je zajištěn pomocí rozhraní MI32. Následuje přehled registrů výstupního síťového bloku.

Total Frames Counter – spodní část a horní část — tato dvojice registrů slouží pro uložení hodnoty čítače všech zpracovaných rámců. Do registru spodní části se ukládají bity čítače <31:0> a registr horní části uchovává bity <63:32>.

Transmitted Frames Counter – spodní část a horní část — slouží pro uložení hodnot čítače, který počítá všechny rámce úspěšně odeslané na výstupní XGMII rozhraní. Hodnota čítače se opět ukládá do dvou registrů, způsobem popsaným u Total Frames Counter.

Discarded Frames Counter – spodní část a horní část — prostor, který je poskytován těmito registry, slouží k uložení hodnoty čítače zaznamenávajícího počet rámců zahozených na základě kontrolních dat FrameLink rámce. Podrobněji je popis zahazování rámců výstupním síťovým blokem popsán v části 3.3.4. Hodnota čítače se do dvojice registrů ukládá stejným způsobem, jako u předešlých dvou registrových párů.

Enable Register — nejnižší bit tohoto registru určuje, zda je činnost výstupního síťového bloku povolena (hodnota 1) nebo zakázána (hodnota 0). V případě zakázání činnosti nedochází k vysílání rámců přes výstupní XGMII rozhraní, ale příchozí rámce jsou ukládány do přítomných bufferů.

MAC Address Register – spodní část a horní část — paměťová pole tohoto registrového páru slouží k uložení MAC adresy, která může být v části PROCESS vložena do pole Source Address rámce. V registru spodní části jsou uchovávány bity <31:0> MAC adresy a zbývající část — bity <47:32>) — je uložena v registru horní části.

Command Register — na nejnižších 8 bitů tohoto registru je možné zapsat speciální hodnoty, které představují příkazy pro práci s čítači rámců. Přehled těchto příkazů je v tabulce 3.9.

Název	Hodnota	Význam
OBUFCMD_STROBE_COUNTERS	0x01	uložení aktuální hodnoty čítačů rámců do registrů
OBUFCMD_RESET_COUNTERS	0x02	vynulování aktuální hodnoty čítačů rámců

Tabulka 3.9: Příkazy pro práci s čítači rámců

Status Register — bity nebo skupiny bitů tohoto registru reflektují nastavení výstupního síťového bloku. Popis významných bitů je uveden v tabulce 3.10.

Pozice	Název	Význam
4 – 5	OBUF speed	udává podporovanou rychlost (11 = 10 Gb/s)

Tabulka 3.10: Význam bitů ve Status Register

Přístup k jednotlivým registrům je zajišťován přes rozhraní MI32. Šířka datové sběrnice tohoto rozhraní má samozřejmě vliv na organizaci adesového prostoru registrů výstupního síťového bloku. Tento adresový prostor je proto organizován do slov po 32 bitech. Bázová

Adresa	Registr
0x00	Total Frames Counter – spodní část
0x04	Transmitted Frames Counter – spodní část
0x08	Discarded Frames Counter – spodní část
0x10	Total Frames Counter – horní část
0x14	Transmitted Frames Counter – horní část
0x18	Discarded Frames Counter – horní část
0x20	Enable Register
0x24	MAC Address Register – spodní část
0x28	MAC Address Register – horní část
0x2C	Command Register
0x30	Status Register

Tabulka 3.11: Adresový prostor výstupního síťového bloku

adresa je opět volitelná, posunutí adres jednotlivých registrů vůči této bázové adrese je však pevné. Adresový prostor výstupního síťového bloku přehledně zachycuje tabulka 3.11

Různé registry se liší také ražimem, ve kterém se dá přistupovat k jejich hodnotám. Bez omezení — pro čtení i pro zápis — jsou přístupné pouze registry Enable Register a obě části MAC Address Register. Pouze pro zápis je přístupný Command Register a pouze pro čtení jsou přístupné všechny registry čítačů a také Status Register. Hodnoty ve Status Register jsou nastavovány podle vnitřního stavu výstupního síťového bloku a hodnoty čítačů registrů se nastaví na aktuální hodnotu čítačů rámců ve chvíli zápisu příkazu OBUFCMD_STROBE_COUNTERS do Command Register.

3.3.4 Operace se vstupním datovým tokem

Výstupní síťový blok nezajišťuje tolik funkcí jako vstupní síťový blok. Přesto však nejde o pouhé přeposílání dat z jednoho rozhraní na druhé. Minimálně by bylo nutné provést překódování z protokolu FrameLink na *XGMII Data Stream*, které v podstatě celé zajišťuje blok XGMII_ENC. Kromě toho jsou však prováděny také další operace.

Řízení prováděných operací je založeno na zasílání kontrolních bitů ve FrameLink rámci, v části následující po datech. Tato část FrameLink rámce však není vždy přítomná a její výskyt se řídí generickým parametrem CTRL_CMD (viz kapitola 3.3.1). Pokud nejsou kontrolní data přítomná, pak má FrameLink rámec následující tvar

<SOP><FRAME_DATA><EOP>

Je-li však kontrolní část rámce přítomná, pak se vyskytuje za vlastními daty rámce (*footer*) a příchozí datový tok na rozhraní FrameLink má tvar

<SOP><FRAME_DATA><EOP><SOP><FOOTER_DATA><EOP>

V takovém případě má část FOOTER_DATA velikost 2 bity a právě tyto dva bity řídí zpracování rámce.

Frame Valid (bit 0) — má-li tento bit hodnotu 1, pak je rámec v pořádku. Pokud je však hodnota tohoto bitu rovna 0, rámec bude zahozen a nebude odeslán na výstupní rozhraní.

Replace Source MAC (bit 1) — nastavení hodnoty 1 tohoto bitu znamená nahrazování pole Source Address rámce MAC adresou uloženou v registru MAC Address Register (viz kapitola 3.3.3).

Kromě zahazování rámců a náhrady zdrojové MAC adresy provádí výstupní síťový blok také zarovnání rámců na minimální hodnotu délky (60 bytů bez pole Frame Check Sequence) a výpočet hodnoty CRC, která je přidána jako pole Frame Check Sequence.

Kapitola 4

Implementace navržených síťových bloků

Vstupní a výstupní síťové bloky, jejichž návrh byl popsán v kapitole 3, byly implementovány pomocí jazyka VHDL. Při implementaci byly použity některé komponenty (*IP cores*) vyvinuté v rámci projektu Liberouter. Jedná se především o paměťové komponenty (FIFO, posuvné registry, CAM paměť), dále nástroje pro práci s protokolem FramLink (Transformer, Pipe), modul pro výpočet CRC a další. Pro převod komunikace typu DDR na SDR (příjem) a naopak (vysílání) na rozhraní XGMII byly použity komponenty vytvořené společností Xilinx.

4.1 Výsledky syntézy

Kromě syntézy za účelem ověření výsledné implementace vstupních a výstupních síťových bloků na kartách rodiny COMBO byla provedena také samostatná syntéza jednotlivých implementovaných bloků. Cílem této syntézy bylo zjistit zdroje, které výsledná implementace zabírá na čipu FPGA, a maximální možnou frekvenci hodinového signálu.

Syntéza byla provedena nástrojem XST (*Xilinx Synthesis Technology*) verze 10.1.03. Za cílovou technologii byl zvolen čip Virtex-5 LXT, konkrétně varianta XC5VLX110T. Tento čip je osazen na nové generaci katret COMBO. Získané hodnoty shrnuje tabulka 4.1. Hodnoty v závorkách vyjadřují vztah k celkovým dostupným zdrojům na zvoleném čipu FPGA.

Virtex-5 LXT (XC5VLX110T)	**Vstupní blok**	**Výstupní blok**
Slice Registers	2266 (3 %)	1443 (2 %)
Slice LUTs	3323 (4 %)	1223 (1 %)
Maximální frekvence	275,406 MHz	290,957 MHz

Tabulka 4.1: Využité zdroje a maximální frekvence na čipu XC5VLX110T

Kapitola 5

Verifikace vstupních a výstupních síťových bloků

Pro ověření správného fungování implementovaných vstupních a výstupních síťových bloků lze použít několik metod. V rámci projektu Liberouter se používají 4 přístupy. Jednotlivé přístupy se liší časovou náročností, úrovní abstrakce použité při testování a úplností prováděných testů.

Simulace činnosti pomocí testu popsaného v jazyce VHDL. Testovaná komponenta je zapojena do prostředí testu (*testbench*) a lze tak nastavovat hodnoty na jejích vstupech a sledovat hodnoty výstupních signálů. Výsledkem je časový diagram vybraných signálů, který je třeba analyzovat a určit, zda testovaná komponenta v simulaci uspěla či nikoliv.

Verifikace chování sytému pomocí jazyka rodiny HVL (*Hardware Verification Language*). Úroveň abstrakce a postup verifikace je podobný jako v případě simulace. Narozdíl od simulace však HVL jazyky zavádí konstrukce, které umožňují náhodné generování testovacích hodnot. Verifikace je tak oproti simulaci variabilnější a testuje větší množství možných stavů verifikované komponenty. Stále však nejde o formální verifikaci, která může dokázanou bezchybnost podložit matematickým důkazem.

Hardwarové testování představuje zapojení testované komponenty do designu pro čip FPGA umístěný na kartě COMBO, vytvoření konfiguračního řetězce (*bit stream*) a nahrání této konfigurace do FPGA. Podle povahy testovaného systému pak testování probíhá buď pomocí softwarových nástrojů a nebo za použití hardwarového testeru schopného generování, zachytávání a analýzy síťového provozu.

Formální verifikace zahrnují skupinu metod, které mohou dokáznou bezchybnost podložit matematickým důkazem. Jejich problémem je však nemožnost plně automatického provedení důkazu u většiny metod a také velká výpočetní náročnost. V rámci projektu Liberouter se používá metoda tzv. ověřování modelu (*model checking*) [2], která je jako jediná z formálních verifikačních metod plně automatizovatelná. Náročné je však převedení testované komponenty do požadované reprezentace, a tak se tato metoda používá pouze u nejdůležitějších komponent.

Implementované síťové bloky byly otestovány v rámci simulace a hardwarového testování. Obsahem této kapitoly je popis aplikace principů verifikace pro otestování implementovaných síťových bloků. Z rodiny HVL jazyků byl zvolen SystemVerilog.

5.1 Možnosti jazyka SystemVerilog

Jazyk SystemVerilog může sloužit jak k návrhu a implementaci hardware, tak i k jeho verifikaci. Lze ho tedy zařadit jak do rodiny HDL (*Hardware Description Language*) jazyků, tak i mezy HVL (*Hardware Verification Language*). V [11] se proto uvádí, že SystemVerilog patří do nové rodiny jazyků — HDVL (*Hardware Decription and Verification Language*).

Standard [4] tohoto jazyka pochází z roku 2005, a tak mohl SystemVerilog převzít kontrukce a principy od mnoha jazyků, ať už obecných (např. C nebo C++) či specializovaných (Verilog, VERA a další). Hlavní základ tvoří jazyk Verilog podle standardu z roku 2005 [7]. Tento fakt předurčuje SystemVerilog k použití v oblasti návrhu hardware. Z ostatních jazyků si však SystemVerilog přinesl konstrukce, které návrh hardware výrazně ulehčují

- nové datové typy
- procedurální bloky
- rozhraní
- ...

Jazyk SystemVerilog má však daleko širší uplatnění než jeho hlavní předchůdce — Verilog. Kromě návrhu hardware lze tohoto jazyka s výhodou využít také pro jeho verifikaci. To umožňují principy a programové konstrukce, jejichž inspirací byly především jazyky rodiny HVL. Některé principy používané v rámci verifikací navrhovaných systému však mají svůj původ u obecných jazyků

- zavedení tříd a objektově orientovaný přístup
- constrained random generation
- assertions
- coverage
- ...

Všechny uvedené vlastnosti a konstrukce jazyka SystemVerilog budou využity při návrhu verifikačního prostředí pro vstupní a výstupní síťové bloky. Použití tříd a objektů usnadní návrh a umožní vytvoření obecnějšího a lépe parametrizovatelného verifikačního prostředí. Princip constrained random generation se uplatní při získávání testovacích hodnot. Jde o generování náhodných hodnot vstupních signálů, které lze omezit takovým způsobem, aby byly generovány pouze přípustné hodnoty. Při správném nastavení tak lze automaticky ověřit všechny přípustné kombinace vstupních hodnot. Assertions jsou představovány podmínkami, jejichž nesplnění vyvolá požadovanou akci (výpis chybového hlášení, ukončení verifikace). Tímto způsobem lze ověřovat například dodržování předepsaných protokolů na rozhraních testované komponenty. Konečně coverage slouží ke sledování úplnosti verifikace. Lze sledovat například pokrytí všech řádků zdrojového kódu testované komponenty (*code coverage*) nebo pokrytí všech předepsaných funkcí (*fuctional coverage*).

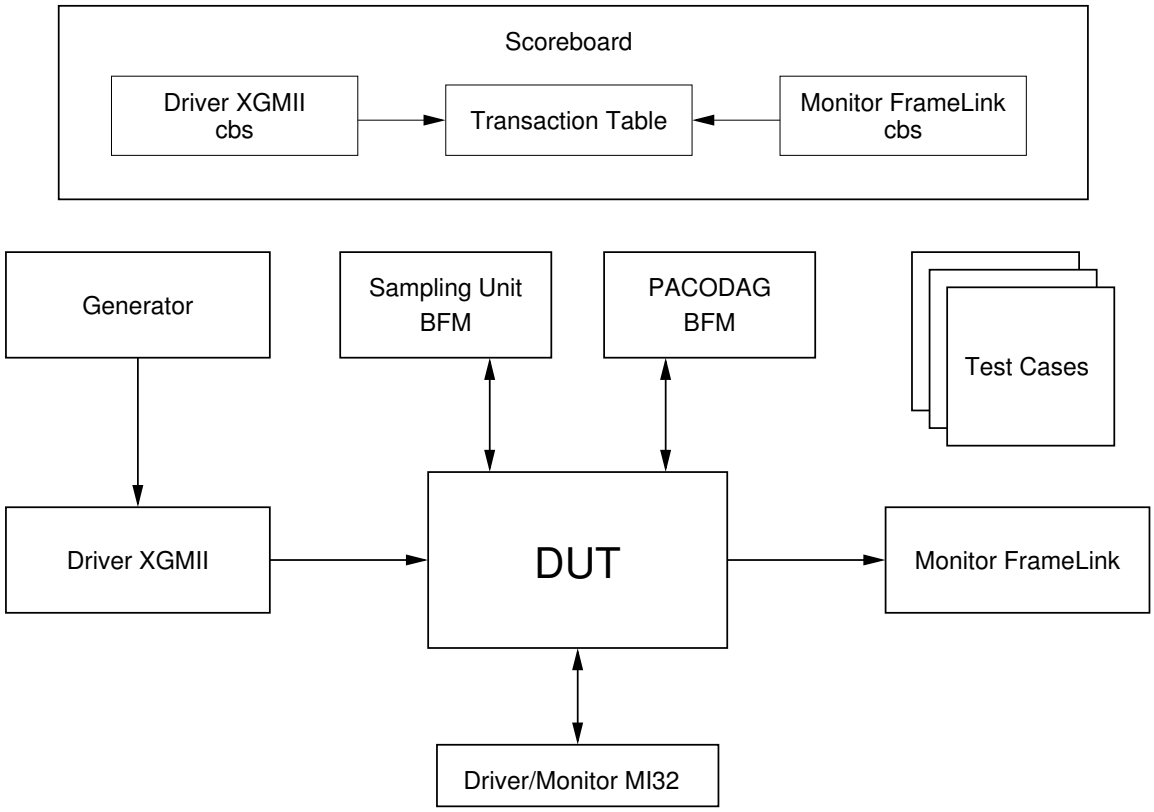
5.2 Návrh verifikačního prostředí

Vlastností jazyka SystemVerilog, popsaných v kapitole 5.1, bylo využito při návrhu verifikačních prostředí (*testbench*) pro vstupní a výstupní síťové bloky. Jelikož se testované komponenty liší jak v implementovaných rozhraních, tak i v poskytované funkcionalitě, je pro každý ze síťových bloků navrženo vlastní verifikační prostředí. Při návrhu byly aplikovány principy popsané v [10], zvláště pak rozdělení popisu testovacího prostředí do několika vstev a objektů (*layered testbench*). Díky tomu lze zajistit maximální znovupoužitelnost kódu a tudíž rychlý vývoj druhého verifikačního prostředí. Navíc bude možné použít některé části již implementované v rámci projektu Liberouter.

5.2.1 Vstupní síťový blok

Podstatou verifikace vstupního síťového bloku bude zasílání XGMII transakcí na vstupní rozhraní testovaného bloku a sledování výstupního FrameLink rozhraní, na kterém by se měla objevit transformovaná transakce. Výsledek transformace je závislý na nastavení konfiguračních registrů vstupního síťového bloku. Tato konfigurace bude provedena před spuštěním testu podle hodnot, které budou parametrem testu.

Obrázek 5.1 zachycuje jednotlivé komponenty verifikačního prostředí vstupního síťového bloku a komunikaci mezi nimi. Význam jednotlivých částí je popsán v seznamu, který následuje za obrázkem.



Obrázek 5.1: Verifikační prostředí vstupního síťového bloku

DUT — představuje instanci testované komponenty v testovacím prostředí. DUT je zkratkou pro *Design Under Test*.

Generator — objekt této třídy zajišťuje generování transakcí s náhodnými hodnotami signálů. V případě vstupního síťového bloku slouží pro generování XGMII transakcí. Generování náhodných hodnot je prováděno technikou *constrained random generation*.

Driver XGMII — jde o přeposílací objekt (*transactor*). Od generátoru přijímá transakce, které se správným časováním odesílá na vstupní XGMII rozhraní testované komponenty a zároveň pomocí callback funkce informuje o této skutečnosti scoreboard.

Monitor FrameLink — monitoruje hodnoty na výstupním FrameLink rozhraní testované komponenty. Z příchozích dat formuje transakce a po přijetí celé transakce informuje přes callback funkci scoreboard.

Scoreboard — tento objekt slouží k automatickému vyhodnocování výsledků verifikace. Při odeslání transakce na vstupní rozhraní DUT je tato transakce zaslána také do scoreboard. V rámci callback funkce driveru je proveden převod na očekávanou výstupní transakci a výsledek je uložen do transaction table. Po dokončení zpracování transakce testovanou jednotkou spustí monitor provádění své callback funkce umístěné ve scoreboard. Pokud se dokončená transakce nachází v transaction table, je záznam z tabulky vymazán. V opačném případě je zahlášena chyba a verifikace končí neúspěchem. Tato část verifikačního prostředí je jednou z nejdůležitejších. Je zde nutné implementovat veškerou funkcionalitu testované jednotky. Do scoreboard se také musí dostat informace o nastavení konfiguračních registrů vstupního síťového bloku.

Driver/Monitor MI32 — slouží k nastavování a vyčítání hodnot vnitřních konfiguračních registrů vstupního síťového bloku přes rozhraní MI32. Před začátkem testu provádí zápis požadované konfigurace a po jeho ukončení může provést kontrolu hodnot čítačů.

PACODAG BFM, Sampling Unit BFM — tyto BFM (*Bus Functional Model*) jednotky slouží k simulaci PACODAG a *Sampling Unit* komponenty, které jsou nutné pro správnou funkci vstupního síťového bloku.

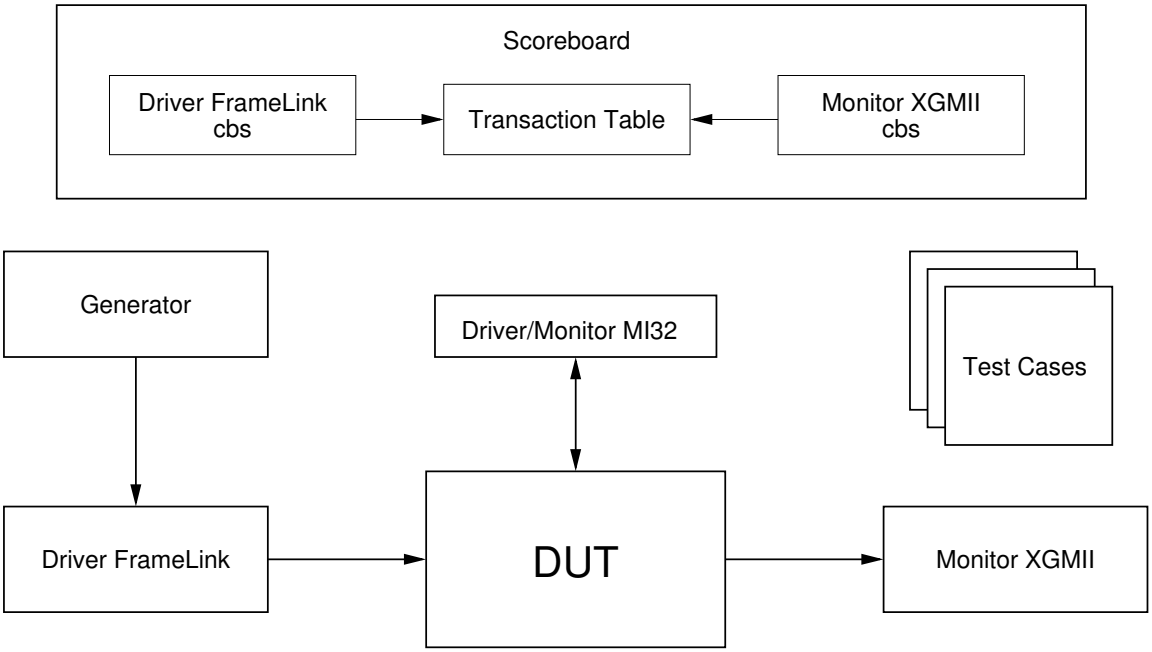
Test Cases — v návrhu verifikačního prostředí zastupuje tent blok popis různých testovacích případů, které mohou být se vstupním síťovým blokem provedeny.

5.2.2 Výstupní síťový blok

Základní principy verifikace jsou u obou síťových bloků stejné. I u výstupního síťového bloku budou hlavním předmětem verifikací transakce zasílané na vstupní rozhraní bloku (FrameLink) a monitorované na výstupním rozhraní (XGMII). Výsledek opět závisí na nastavení konfiguračních registrů, které bude provedeno před vlastním testem na základě parametrů testu. Rozdílem je však množství konfuguračních registrů u obou bloků. Oproti vstupnímu bloku se zde také neuplatní jednotky simulující činnost komponent PACODAG a *Sampling Unit* (PACODAG BFM a SAU BFM).

Strukturu verifikačního prostředí výstupního síťového bloku zobrazuje obrázek 5.2. Popis jednotlivých částí je uveden v následujícím seznamu.

DUT — instance testované komponenty ve verifikačním prostředí.



Obrázek 5.2: Verifikační prostředí výstupního síťového bloku

Generator — jako u vstupního síťového bloku, i zde slouží generátor k náhodnému generování testovacích transakcí metodou *constrained random generation*. Zde však jde o FrameLink transakce.

Driver FrameLink — zajišťuje vysílání transakcí přijatých od generátoru na vstupní FrameLink rozhraní výstupního síťového bloku. Po odeslání transakce informuje scoreboard.

Monitor XGMII — přijímá data na výstupním XGMII rozhraní testované komponenty a skládá z nich transakce. Jakmile je k dispozici celá transakce, monitor pomocí callback fukce informuje scoreboard.

Scoreboard — význam a fungování této komponenty je stejné jako ve verifikačním prostředí vstupního síťového bloku. Opět je zde callback funkce driveru a monitoru, které vkládají a vybírají záznamy z transaction table. Podstatným rozdílem je ale to, že zde scoreboard implementuje funkcionalitu výstupního síťového bloku.

Driver/Monitor MI32 — tento objekt může být stejný jako při verifikaci vstupního síťového bloku. Je však třeba brát v úvahu, že DUT obsahuje jinou sadu registrů a tudíž i adresní prostor je jiný.

Test Cases — opět zastupuje množinu testů, které lze s verifikačním prostředím provádět.

Kapitola 6

Závěr

Po analýze požadavků na příjem a vysílání paketů na rychlosti 10 Gb/s za pomoci FPGA byly navrženy vstupní a výstupní síťové bloky zajišťující tyto dvě úlohy v rámci platformy NetCOPE. Pro komunikaci směrem do LAN sítě používají navržené síťové bloky standardní XGMII rozhraní. Komunikace s dalšími komponentami na čipu FPGA je uskutečňována pomocí protokolu FrameLink. Vstupní síťový blok dále provádí kontroly příchozích dat a umožňuje zahazování chybných rámců. Výstupní síťový blok zajišťuje především kompletaci odchozího rámce a jeho případné úpravy. Operace prováděné vstupním a výstupním síťovým blokem lze ovlivnit nastavením hodnot vnitřních registrů přístupných přes rozhraní MI32.

Navržené vstupní a výstupní síťové bloky byly implementovány v jazyce VHDL a otestovány v rámci simulací i jako součást reálných aplikací nad platformou NetCOPE (síťová karta s hardwarovou filtrací a monitorovací sonda). Pro verifikaci implementovaných síťových bloků bylo navrženo verifikační prostředí pro jazyk SystemVerilog.

Uplatnění naleznou vstupní a výstupní síťové bloky především jako součást platformy NetCOPE. Zde budou podstatným přínosem pro tuto platformu, jelikož poskytnou návrháři aplikace přístup k síťovému rozhraní, které pracuje na rychlosti 10 Gb/s, pomocí jednoduchého protokolu FrameLink. Uplatnění vytvořených bloků mimo platformu NetCOPE se nepředpokládá. Omezení jsou však dána pouze použitými rozhraními.

Další významné rozšiřování implementovaných vstupních a výstupních síťových bloků očekávat nelze. V takovém okamžiku se však otevírá prostor pro verifikaci implementovaných komponent. Navržené verifikační prostředí by mělo být implementováno a použito k jejich důkladné verifikaci. Budoucnost těchto síťových bloků také závisí na rychlosti, s jakou bude standardizován a zaváděn Ethernet o rychlostech 40 Gb/s a 100 Gb/s. Pro podporu nového standardu by totiž musely být vytvořeny nové vstupní a výstupní síťové bloky.

Literatura

[1] Programmable hardware. [online]. ©2002-2006, [cit. 2009-04-24]. Dostupné na URL: http://www.liberouter.org/.

[2] Holeček, J.; et al: Verification Process of Hardware Design in Liberouter Project. Technická Zpráva 5, CESNET, 2004.

[3] The Institute of Electrical and Electronics Engineers, Inc., 3 Park Avenue, New York, NY 10016-5997, USA: *IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture*. IEEE std 802-2001 vydání, February 2002, ISBN 0-7381-2941-0.

[4] The Institute of Electrical and Electronics Engineers, Inc., 3 Park Avenue, New York, NY 10016-5997, USA: *IEEE Standard for SystemVerilog — Unified Hardware Design, Specification, and Verification Language*. IEEE std 1800-2005 vydání, November 2005, ISBN 0-7381-4811-3.

[5] The Institute of Electrical and Electronics Engineers, Inc., 3 Park Avenue, New York, NY 10016-5997, USA: *Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*. IEEE std 802.3-2005 vydání, December 2005, ISBN 0-7381-4741-9.

[6] The Institute of Electrical and Electronics Engineers, Inc., 3 Park Avenue, New York, NY 10016-5997, USA: *IEEE Standard for Local and metropolitan area networks: Virtual Bridged Local Area Networks*. IEEE std 802.1q-2005 vydání, May 2006, ISBN 0-7381-4877-6.

[7] The Institute of Electrical and Electronics Engineers, Inc., 3 Park Avenue, New York, NY 10016-5997, USA: *IEEE Standard for Verilog Hardware Description Language*. IEEE std 1364-2005 vydání, April 2006, ISBN 0-7381-4851-2.

[8] Martínek, T.; Tobola, J.: Interconnection System for the NetCOPE Platform. Technická Zpráva 34, CESNET, 2006.

[9] Rhodes, M.: XGMII Using the DDR Registers, DCM, and SelectI/O Features in Virtex-II Devices. Application Note xapp606, Xilinx, Inc., 2100 Logic Drive San Jose, CA 95124-3400, USA, October 2001.

[10] Spear, C.: *SystemVerilog for Verification: A Guide to Learning the Testbench Language Features*. Springer, 2006, ISBN 978-0-387-27036-4.

[11] Sutherland, S.; Davidmann, S.; Flake, P.: *SystemVerilog for Design: A Guide to Using SystemVerilog for Hardware Design and Modeling*. Springer, druhé vydání, 2006, ISBN 978-0-387-33399-1.

[12] Tobola, J.: *Platforma pro rychlý vývoj síťových zařízení.* Diplomová práce, FIT VUT v Brně, 2007.

Dodatek A

Obsah DVD

Adresář `source`

Obsahem adresáře `source` jsou zdrojové kódy implementovaných vstupních a výstupních síťových bloků. Složitá adresářová struktura je vynucená závislostmi mezi použitými komponentami vyvinutými v rámci projektu Liberouter. Podrobnosti k adresářové struktuře, umístění implementovaných síťových bloků a návod ke spuštění simulace jsou součástí souboru `README`.

Adresář `text`

Adresář `text` obsahuje zdrojové kódy a další soubory umožňující úpravu textu bakalářské práce a její opětovné vysázení. Pro sazbu bakalářské práce byl použit systém LaTeX a poskytovaná šablona.

`bp-xmatou06.pdf`

Soubor `bp-xmatou06.pdf` obsahuje vysázený text bakalářské práce začínající úvodním listem (bez desek). Součástí textu jsou také funkční odkazy.

`README`

Obsahuje informace o obsahu DVD a pokyny pro jeho použití (spuštění simulace, vysázení bakalářské práce, ...).