VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2019 Bc. Jaroslav Kolařík

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV MIKROELEKTRONIKY

DEPARTMENT OF MICROELECTRONICS

IMPLEMENTACE PCS PODVRSTVY 400 GB/S ETHERNETU V FPGA

IMPLEMENTATION OF 400 GB/S ETHERNET PCS LAYER TO FPGA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. Jaroslav Kolařík

VEDOUCÍ PRÁCE
SUPERVISOR

doc. Ing. Lukáš Fujcik, Ph.D.

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Mikroelektronika**
Ústav mikroelektroniky

Student: Bc. Jaroslav Kolařík ***ID:*** 177546
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Implementace PCS podvrstvy 400 Gb/s Ethernetu v FPGA

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se standardem IEEE 802.3bs-2017 definujícím 400 Gb/s Ethernet a s FPGA architekturami vhodnými pro implementaci režimu 400GBASE-R dle tohoto standardu. Podrobně prostudujte požadavky na PCS podvrstvu a proveďte její návrh a implementaci pro vhodnou FPGA platformu. Při realizaci vycházejte ze stávajících 100GBASE-R řešení. Implementace bloků řešících RS-FEC kodér a dekodér není vyžadována, ale diskutujte ji v teoretické části práce. Funkčnost implementované jednotky ověřte v simulaci. V závěru diskutujte dosažené výsledky a zamyslete se nad možnostmi optimalizace spotřeby zdrojů na FPGA.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce.

Termín zadání: 4.2.2019 ***Termín odevzdání:*** 21.5.2019

Vedoucí práce: doc. Ing. Lukáš Fujcik, Ph.D.
Konzultant:

doc. Ing. Lukáš Fujcik, Ph.D.
předseda oborové rady

ABSTRAKT

Tato práce se zabývá návrhem jednotky PCS v režimu 400GBASE-R podle standardu IEEE 802.3bs-2017 definující 400 Gb/s Ethernet. První část práce je zaměřena na obecnou architekturu FPGA, možné varianty FPGA pro implementaci 400 Gb/s Ethernet, popisem těchto architektur a zdrojů v FPGA. Další část popisuje vývoj Ethernetu a spojitost s referenčním modelem ISO/OSI. Následně je podrobně popsána fyzická vrstva Ethernetu pro rychlost 400 Gb/s za čímž následuje návrh jednotky PCS a popis její implementace, včetně využití zdrojů ve zvoleném FPGA. V poslední části je popsána simulace implementované jednotky PCS. V závěru jsou shrnuty dosažené výsledky a přínos této práce.

KLÍČOVÁ SLOVA

FPGA, VHDL, Ethernet, PCS, 400GBASE-R

ABSTRACT

This master thesis deals with the design of the 400GBASE-R PCS in accordance with the IEEE 802.3bs-2017 standard which defines 400 Gbps Ethernet. The first part of this thesis focuses on general architecture of FPGA and its possible variants for implementation for 400 Gbps Ethernet communication, therefore there is description of those architectures and its resources. The next part describes progression of the Ethernet and its connection to the ISO/OSI reference model. The next section of this thesis is about description of physical layer of Ethernet for 400 Gbps version, after which follows design of PCS unit and its implementation with use of resources of selected FPGA. In the last part of this thesis is description of the simulation of the implemented unit. Achieved results and outcomes of this master thesis are evaluated in a conclusion.

KEYWORDS

FPGA, VHDL, Ethernet, PCS, 400GBASE-R

KOLAŘÍK, Jaroslav. *Implementace PCS podvrstvy 400 Gb/s Ethernetu v FPGA*. Brno, 2019, 57 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav Mikroelektroniky. Vedoucí práce: doc. Ing. Lukáš Fujcik, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Implementace PCS podvrstvy 400 Gb/s Ethernetu v FPGA“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu doc. Ing. Lukáši Fujcikovi, Ph.D. za vedení, konzultace, trpělivost a podnětné návrhy k práci. Dále bych rád poděkoval odbornému konzultantovi Ing. Štěpánu Friedlovi za odborné vedení, konzultace a užitečné rady k práci. V neposlední řadě bych poděkoval Ing. Jakubu Cabalovi za konzultace a užitečné rady k práci.

Brno ....

podpis autora

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno ....

podpis autora

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Vznik Ethernetu se datuje od 70. let 20. století. První standardizovaná verze Ethernetu vyšla v roce 1983 v rámci standardizace IEEE802.3 pod označením 10BASE5, kde číslo 10 značí rychlost (10 Mbit/s) a číslice 5 značí dosah (ve stovkách metrů, tedy 500m). Tato verze Ethernetu pracovala na přístupové metodě CSMA/CD (Carrier Sense Multiple Access/Collision Detection). [1]

Od doby, kdy byla představena první verze Ethernetu, dnes již uběhlo 35 let. Jelikož se stále kladou nároky na větší datovou rychlost, byl dnes již vyvinut Ethernet pro rychlost 400 Gb/s. Za tuto dobu se neměnila jenom rychlost, ale i použité médium pro přenos. U nejstarších verzí Ethernetu býval použit koaxiální kabel, postupně se přešlo ke kroucené dvoulince a dnes již pro tak vysoké rychlosti jako 400 Gb/s je zapotřebí optický kabel. Další inovací od gigabitového Ethernetu bylo použití tzv. plně duplexního Ethernetu. Plně duplexní Ethernet nevyužívá žádnou přístupovou metodu, jelikož využívá dvoubodových spojů. [1]

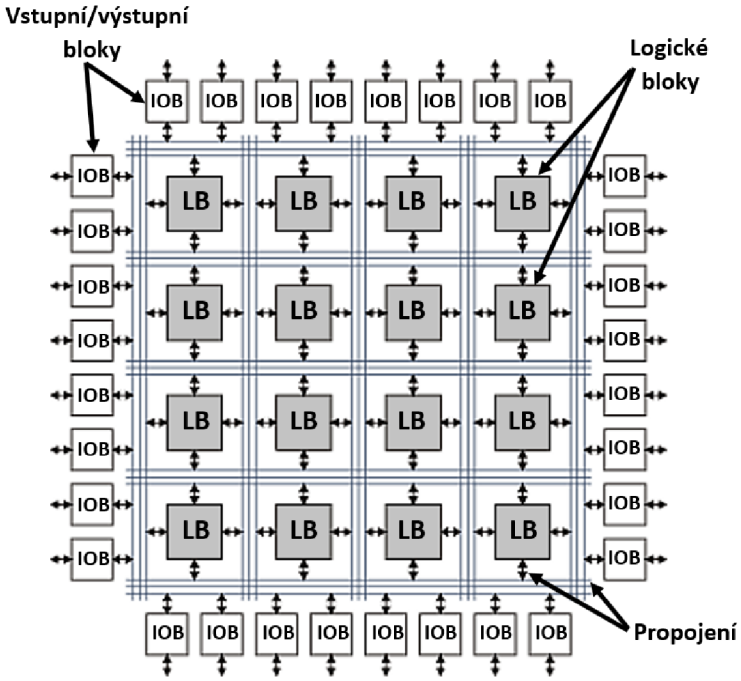
K implementaci Ethernetu s takto velkými datovými rychlostmi bývají používány akcelerační karty. Tyto karty obsahují FPGA s rychlým datovým rozhraním, které jsou nezbytné pro tak vysoké přenosové rychlosti. Akcelerační karty s přenosovou rychlostí dat v řádech stovek gigabitů za sekundu nachází uplatnění především v datových centrech. Pro domácnosti se v současné době uplatňují maximální rychlosti kolem 1 Gb/s. [1]

Ethernet pro rychlost 400 Gb/s dle specifikace nabízí čtyři varianty fyzické vrstvy: 400GBASE-SR16, 400GBASE-DR4, 400GBASE-FR8 a 400GBASE-LR8. V této práci je uvažováno použití varianty fyzické vrstvy 400GBASE-LR8, které používá kódování přes osm fyzických linek s jednovidovým optickým médiem a umožňuje dosah nejméně deset kilometrů. Tato varianta fyzické vrstvy předpokládá použití 58Gb/s transceiverů PAM-4. [2]

Kapitola 1 se zabývá problematikou FPGA. Je zde popsána základní architektura FPGA a možné varianty použití FPGA pro 400 Gb/s Ethernet s fyzickou vrstvou 400GBASE-LR8. Kapitola 2 popisuje Ethernet a jeho spojitost s modelem ISO/OSI. V další kapitole 3 je podrobně rozebrána fyzická vrstva. Kapitola 4 se zabývá návrhem jednotky PCS. Následující kapitola 5 popisuje implementaci jednotky PCS do cílového FPGA. Kapitola 6 se zabývá simulací navrhované jednotky PCS. V závěru jsou shrnuty dosažené výsledky a přínos této práce.

1 Obvody FPGA

Obvody FPGA (Field Programmable Gate Array) jsou v překladu označovány jako programovatelná hradlová pole. Tyto obvody patří do skupiny programovatelných logických zařízení (PLD - Programmable Logic Device). Jedná se tedy o polovodičovou součástku, která je založena na matici programovatelných logických bloků připojených přes programovatelná propojení viz Obr. 1.1. To umožňuje uživateli využívat FPGA k implementaci digitálních obvodů pro širokou škálu aplikací. Tato vlastnost je charakteristickým rysem FPGA, kterým se odlišuje od obvodů ASIC (Aplication Specific Integrated Circuit), které jsou vyráběny na zakázku pro specifické aplikace. [3, 4, 5]



Obr. 1.1: Zjednodušená struktura obvodu FPGA [4]

Z historického hlediska první FPGA s označením XC2064 bylo vyvinuto firmou Xilinx v roce 1984. V současné době se o pomyslné prvenství mezi výrobci FPGA přetahuje právě Xilinx a Intel, který v roce 2015 koupil firmu Altera, v té době druhého nejvýznamnějšího výrobce FPGA ihned za firmou Xilinx. Jelikož Xilinx a Intel patří mezi největší hráče na trhu s FPGA, bude v této kapitole věnována pozornost především těmto výrobcům. [6, 7]

Pro vývoj aplikací s FPGA resp. návrh vlastního digitálního obvodu je zapotřebí použití nástrojů pro syntézu a implementaci. Nástroj pro syntézu převádí textový popis obvodu popsaný pomocí některého z HDL jazyků (např. VHDL, Verilog

nebo ABEL), na netlist reprezentující zapojení elementárních číslicových obvodů (hradel), včetně optimalizace (minimalizace logických funkcí). Výstupem syntézy je tzv. RTL (Register Transfer Logic) schéma. Nástroj pro implementaci zajistí převedení netlistu vytvořeného syntezátorem na netlist využívající prostředky daného FPGA. Taktéž zajistí jejich optimální rozmístění a propojení v FPGA. Výrobci FPGA již většinou nabízejí tyto dva nástroje v jednom balíčku, jako např. firma Xilinx umožňuje uživatelům využívat nástrojů ISE WebPACK nebo Vivado pro novější FPGA. Firma Intel nabízí nástroj Quartus Prime, kde lze taktéž provádět syntézu i implementaci. [4]

V současné době jsou obvody FPGA široce využívány ať už např. v letectví a obraně (FPGA odolné vůči radioaktivnímu záření), v automobilovém průmyslu (pro asistenční systémy řidiče), průmyslové výrobě (průmyslové sledování a automatizace systémů), lékařství (monitorovací a diagnostické zařízení), datová centra (pro servery s velkou šířkou pásma a s malou latencí) a mnoho další odvětvích. [5]

1.1 Architektura FPGA

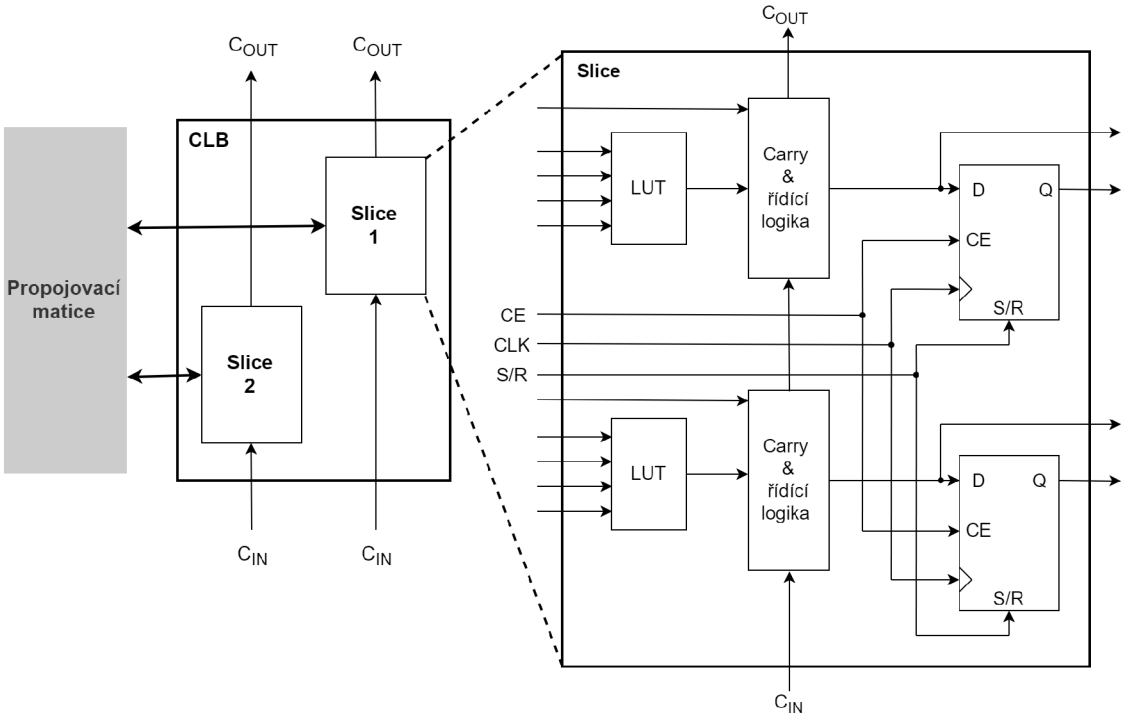
Jelikož existuje několik výrobců FPGA výsledná architektura se od každého výrobce liší, ovšem všichni sdílejí stejnou koncepci základní architektury. Mezi základní stavební prvky FPGA patří: **logické bloky** (LUT a registry), **propojení** a **vstupní/výstupní bloky** viz Obr. 1.1. Dnes už je samozřejmostí, že tyto bloky jsou rozšířeny o tzv. **specializované bloky**, jako jsou např. násobičky, paměti, bloky pro úpravu hodinových signálů, transceivery, procesory a další. [8, 9]

1.1.1 Logické bloky

Základním stavebním prvkem FPGA od společnosti Xilinx je CLB (Configuratable Logic Block). Tyto bloky umožňují realizaci jednoduchých kombinačních i sekvenčních logických funkcí. Propojení mezi jednotlivými CLB je zajištěno pomocí programovatelného propojení. CLB se obvykle skládají ze dvou a více menších bloků označovaných jako Slice viz Obr. 1.2. Slice dále obsahuje **LUT** (Look-Up Table), **rychlou logiku přenosu** (carry chain) a **registry**. [3, 4, 10]

LUT nebo-li vyhledávací tabulky jsou v podstatě konfigurovatelné paměti typu RAM. Při běžném použití tyto paměti obsahují pevně dané konfigurační data. Ve starších architekturách byly využívány LUT se čtyřmi vstupy a jedním výstupem. Dnes už v architekturách FPGA nalezneme LUT, které obsahují až osm vstupů a až dva výstupy. V případě, že budeme uvažovat LUT se čtyřmi vstupy, můžeme zde implementovat libovolnou logickou operaci se čtyřmi vstupními proměnnými.

Pokud chceme implementovat logickou funkci s více vstupními proměnnými, je logická funkce implementována kaskádovitě do více LUT. LUT mohou být využity nejen pro generování kombinačních logických funkcí, ale i jako synchronní paměť typu RAM (16x1 bit v případě čtyř vstupé LUT) nebo taktéž jako posuvný registr (klopný obvod). [4, 8, 10]



Obr. 1.2: Zjednodušena struktura CLB (vlevo) a Slice (vpravo)

Rychlá logika přenosu umožňuje vytvoření různých aritmetických obvodů, jako jsou např. sčítačky, násobičky. Běžné vstupy Slice jsou propojeny na pomalou globální propojovací matici, zatímco vstup a výstup přenosu C_{IN} a C_{OUT} jsou propojeny přímo se sousedními CLB, čímž umožňují rychlé šíření signálu. [4, 8, 10]

Registry umožňují realizaci sekvenčních logických funkcí. Při konfiguraci řezu je obvykle možné nastavit vlastnosti jednotlivých registrů, jako např. clock-enable, polarita a typ set/reset vstupu a další. [10]

Základní stavební prvkem FPGA od společnosti Intel, který nalezneme v její výrobní dokumentaci se nazývá ALM (Adaptive Logic Modules). ALM je v podstatě ekvivalentem Slice. Tedy základ struktury ALM tvoří stejně jako u Slice: LUT, rychlá logika přenosu a registry. Více ALM je sdruženo do bloku označovaného jako LAB (Logic Array Blocks). LAB je v tom případě ekvivalentem CLB. [9]

Základ struktury FPGA od jednotlivých výrobců je tedy stejný, ovšem mohou se lišit použitým označením jednotlivých prvků. Zásadní rozdíl mezi jednotlivými architekturami ovšem spočívá např. v počtu vstupů LUT, počtu Slice v CLB, počtu ALM v LAB, možností ALM fungovat v různých módech (normální mód, rozšířené LUT mód a aritmetický mód) a další. Nelze tedy jednoznačně porovnávat jednotlivé FPGA z hlediska počtů jednotlivých zdrojů. [9, 17]

1.1.2 Propojení

Propojení má při implementaci zásadní vliv na rychlost a kvalitu číslicového obvodu. Proto se k propojení jednotlivých bloků FPGA používají následující typy programovatelných propojovacích matic a spojů: **globální**, **lokální** a **speciální**. [10]

Globální propojovací prostředky umožňují vzájemné propojení libovolných bloků FPGA. Toto propojení bývá nejčastěji realizováno jako několik vrstev horizontálních a vertikálních vodičů různé délky. Takto vytvořená programovatelná matice umožňuje připojení vstupů a výstupů bloků k jednotlivým propojovacím vodičům. Výhodou tohoto propojení je, že poskytuje největší volnost propojení mezi jednotlivými bloky. Naopak nevýhodou je, že globální spoj je většinou dlouhý vodič s velkým množstvím programovatelných spínačů, což se může projevit menší rychlostí šíření signálu. [10]

Lokální propojovací prostředky slouží především k propojení mezi sousedními Slice resp. ALM. Tyto propojení mohou také sloužit jako rychlé spoje pro propojení mezi sousedními CLB resp. LAB. Jelikož jsou tyto spoje mnohem kratší a realizovány méně spínači než globální spoje, tak mají mnohem menší zpozdění signálů. [10]

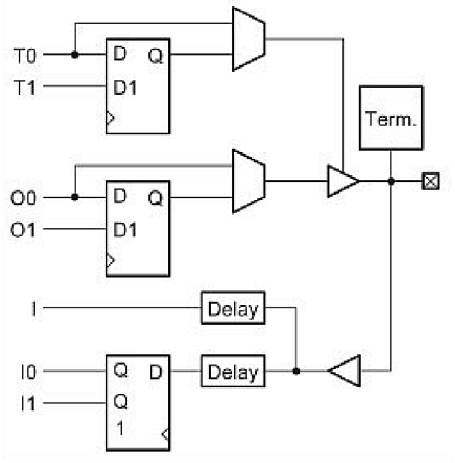
Speciální propojovací prostředky jsou určeny především k šíření hodinových signálů a dalších globálních signálů jako je např. reset. Tyto spoje jsou optimalizovány tak, aby způsobovaly co nejmenší zpoždění procházejícího signálu. Speciální propojení je výhradně určeno jen pro konkrétní typy signálů, nelze je použít pro běžné signály. Toto propojení bývá často realizováno jako globální a lokální podobně jako u běžných signálových spojů. [10]

1.1.3 Vstupní/výstupní bloky

Vstupní/výstupní bloky umožňují spojení mezi FPGA a vnějším prostředím. Tedy základní funkcí těchto bloků je odesílat a přijímat data. Vstupní/výstupní bloky obsahují: vstupní a výstupní registry, zpožďovací linky, obvody impedančního přizpůsobení a ochranné obvody viz Obr. 1.3. Ochranné obvody chrání FPGA před poškozením z vnějšího prostředí např. v důsledku ESD nebo nesprávného použití při zapojení. [9, 10]

Vstupní a výstupní registry jsou dnes u většiny FPGA realizovány pomocí dvou registrů pro podporu DDR (Double Data Rate), kdy je ovšem možné tyto registry nakonfigurovat na běžný režim DDR (Serial Data Rate). Zpožďovací linky umožňují realizaci časového posunu vstupního signálu (korigovat vzájemný fázový posun signálu vůči hodinám). Bloky impedančního přizpůsobení slouží k podpoře více logických i diferenčních standardů např. LVCMOS, LVTTL, HSTL a další. [9, 10]

Firma Xilinx označuje vstupní/výstupní bloky jako IOB (I/O blok), zatímco firma Intel označuje tyto bloky jako IOE (I/O element). V označení těchto bloků se firmy liší, ovšem základní funkce těchto bloků je stejná u obou výrobců. [9]



Obr. 1.3: Zjednodušená struktura vstupního/výstupního bloku [10]

1.1.4 Specializované bloky

Jak již bylo výše zmíněno FPGA se začali rozšiřovat o tzv. specializované bloky. Mezi první tyto bloky patřily paměti, které jsou v FPGA umístěny ve sloupci (namísto sloupce logických bloků). Tyto paměti lze zkonfigurovat do různých paměťových funkcí jako např. ROM, RAM nebo FIFO. Velikost paměťového bloku se u dnešních FPGA pohybuje až ve stovkách megabitů. Jednotlivé bloky lze spojovat dohromady jak do šířky, tak do hloubky, čímž lze vytvářet paměti s organizací vyhovující individuálním požadavkům. [8, 10]

Prvním analogovým blokem, který se v FPGA objevil, byly fázové závěsy (PLL). Tyto bloky umožňují úpravu hodinového signálu jako např. vytvořit z jednoho vstupního kmitočtu několik kmitočtů s různým fázovým posunem. Jelikož se jedná o analogový obvod mají fázové závěsy vlastní napájecí napětí. [8, 10]

Bloky pro aritmetické operace jsou nejčastěji reprezentovány celočíselnými násobičkami (které zároveň obsahují i sčítačku pro realizaci funkce násobení se součtem),

případně složitějšími DSP bloky, které umožňují realizaci i náročnějších matematických operacích v jednom hodinovém cyklu. Bloky násobiček lze taktéž využít při realizaci filtrů např. FIR, IIR, FFT a dalších. [8, 10]

Důležitým blokem pro realizaci velmi rychlých sériových rozhraní jsou sériové transceivery. Sériové transceivery obvykle obsahují diferenciální budič a přijímač, serializer a deserializer, kodér a dekodér, extraktor hodinového signálu a další pomocné obvody. Tyto bloky umožňují komunikovat s okolím rychlostí v jednotek až desítek gigabitů za sekundu. Interní rozhraní je pak v FPGA realizováno synchronními paralelními daty s mnohem nižší hodinovou frekvencí. Sériové transceivery se uplatní u komunikací jako je např. Ethernet, PCI-E, XAUI a další. [10]

1.2 Architektury pro implementaci 400G Ethernetu

Základní požadavek pro implementaci 400Gb/s Ethernetu s variantou fyzické vrstvy 400GBASE-LR8 je, aby FPGA obsahovalo 58 Gb/s PAM4 transceivery. Z tohoto pohledu společnost Xilinx nabízí FPGA s označením Virtex UltraScale+ zatímco společnost Intel nabízí FPGA s označením Stratix 10 TX. Základní parametry a architektury těchto FPGA jsou popsány níže v této kapitole. [2, 11, 12]

1.2.1 Xilinx Virtex UltraScale+

Firma Xilinx nabízí dvě možné varianty FPGA s architekturou Virtex UltraScale+, které obsahují 58 Gb/s transceivery a to FPGA označením VU27P a VU29P. Porovnání těchto dvou FPGA z hlediska základních zdrojů je v Tab. 1.1. [11, 13]

Tab. 1.1: Porovnání FPGA Xilinx Virtex UltraScale+ z hlediska zdrojů [13]

	VU27P	**VU29P**
CLB Flip-Flop	2 592 000	3 456 000
CLB LUT	1 296 000	1 728 000
Max. distribuovaná RAM (Mb)	36,2	48,3
Bloky blokové RAM	2 016	2 688
Bloky RAM (Mb)	70,9	94,5
UltraRAM bloky	960	1280
UltraRAM (Mb)	270	360
Max. HP I/O	520	676
DSP Slice	9 216	12 288
58Gb/s transceivery	32	48

Tyto FPGA jsou vytvořené 16nm FinFET technologií. Architektura Virtex UltraScale+ nabízí největší šířku pásma transceiverů (58Gb/s transceivery), nejvíce DSP bloků a největší hustotu paměti na čipu viz Tab. 1.1. Maximální frekvence hodinového signálu, na které jsou schopny tyto FPGA pracovat, se pohybuje kolem 900 MHz. [11, 13, 14, 15]

FPGA založené na architektuře UltraScale+ obsahují v CLB jeden Slice. Slice se skládá z osmi šesti-vstupých LUT a šestnácti výstupních registrů (dva registry na jednu LUT). Všechny výstupní registry mohou být nakonfigurovány jako aktivní na hranu hodinového signálu (Flip-Flop) nebo aktivní na úroveň hodinového signálu (Latch). [14]

V této architektuře jsou dva typy Slice: SliceL a SliceM. SliceL (L jako Logic) jsou především pro podporu logických funkcí. SliceM (M jako Memory) mohou být nakonfigurovány jako LUT, 64-bitová distribuovaná paměť RAM nebo jako 32-bitový posuvný registr. [14]

Šesti-vstupé LUT ve Slice mohou být nakonfigurovány do různých režimů. Základní režim představuje LUT se šesti vstupy a jedním výstupem, kde lze realizovat různé Booleovské funkce s šesti vstupy. Dalším možným režimem šesti-vstupé LUT je realizace jako dvě pěti-vstupé LUT, ovšem dvě Booleovské funkce musí mít společně sdílených pět vstupů. Dále lze šesti-vstupé LUT realizovat jako dvě libovolné Booleovské funkce s třemi nebo méně vstupy. [14]

1.2.2 Intel Stratix 10 TX

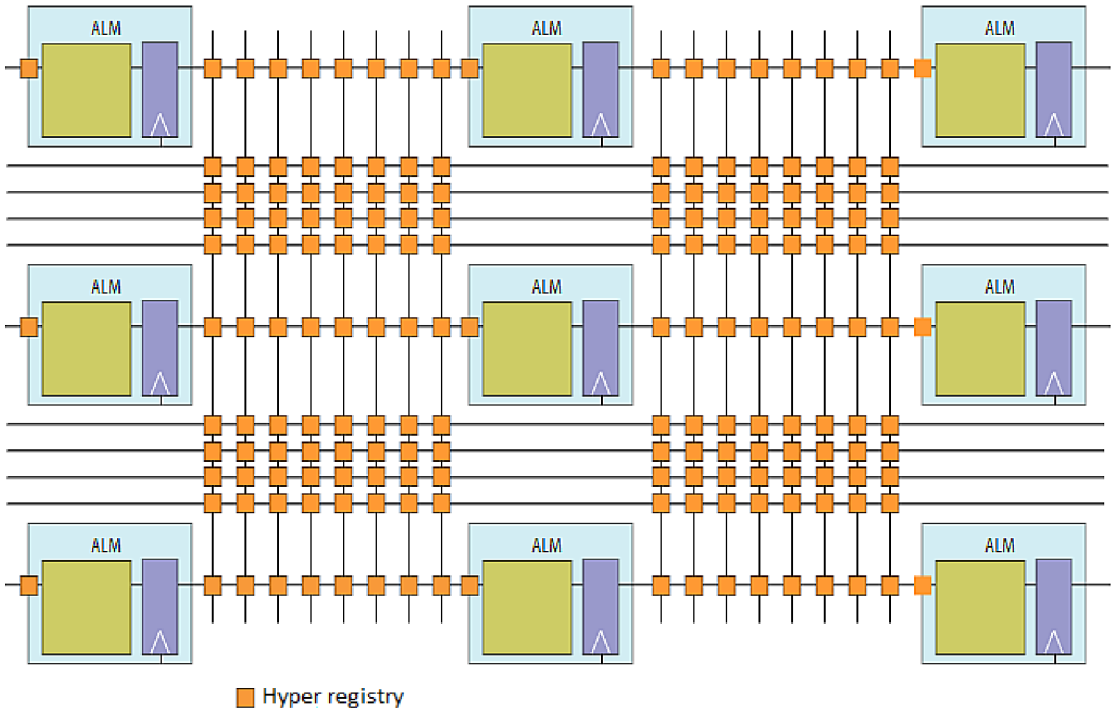
Výrobce Intel nabízí hned čtyři možné varianty FPGA s architekturou Stratix 10 TX, které jsou vhodné pro implementaci 400Gb/s Ethernetu. Jedná se o FPGA s označením TX 1650, TX 2100, TX 2500 a TX 2800, jejichž srovnání z hlediska základních zdrojů v FPGA je v Tab. 1.2. [12, 16]

Tab. 1.2: Porovnání FPGA Intel Stratix 10 TX z hlediska zdrojů [16]

	TX 1650	**TX 2100**	**TX 2500**	**TX 2800**
ALM	569 200	702 720	821 150	933 120
ALM registry	2 276 800	2 810 880	3 284 600	3 732 480
M20K paměťových bloky	6 162	6 847	9 963	11 721
M20K velikost paměti (Mb)	120	134	195	229
MLAB velikost paměti (Mb)	9	11	13	15
DSP bloky	3 326	3 960	5 011	5 760
Max. I/O piny	440 - 544	440 - 544	296 - 544	296 - 544
58Gb/s transceivery	12 - 36	12 - 36	12 - 60	12 - 60

Tyto FPGA jsou založena na 14nm Tri-Gate technologii a obsahují architekturu Intel Hyperflex. Tato architektura obsahuje kromě již tradičních uživatelských registrů v ALM tzv. Hyper-registry, které jsou všude po celém propojení a na vstupech všech funkčních bloků viz Obr. 1.4. Tyto registry umožňují při použití pipeline realizaci velmi rychlých digitálních obvodů. Z hlediska maximální frekvence hodinového signálu jsou tyto FPGA schopny pracovat kolem 1 GHz. [12]

V architektuře Stratix 10 jeden LAB obsahuje deset ALM. Až jedna čtvrtina LAB lze použít jako paměťový prvek označovaný jako MLAB. ALM obsahuje osmy-vstupou LUT a čtyři výstupní registry (čtyři registry na jednu LUT). ALM může pracovat ve třech různých módech: normální mód, rozšířené LUT mód a aritmetický mód. [12, 17]



Obr. 1.4: HyperFlex architektura [12]

Normální mód umožňuje implementovat do ALM jednu Booleovskou funkci až se šesti vstupy nebo dvě Booleovské funkce např. dvě rozdílné čtyř-vstupé funkce, dvě pěti-vstupé funkce, kdy dva vstupy jsou společné nebo nezávislou pěti-vstupou a tří-vstupou funkci. Mód rozšířené LUT umožňuje implementaci až osmi-vstupé Booleovské funkce. Aritmetický mód ALM využívá osmy-vstupou LUT jako dvě čtyř-vstupé LUT. Tyto LUT s úplnými sčítačkami umožňují možnost povolení hodinového signálu, povolení čítání, synchronního řízení čítání a odčítání. [17]

2 Ethernet a spojitost s modelem ISO/OSI

Ethernet se začal rozvíjet v 70. letech 20. století firmami Xerox, Intel a DEC (Digital Equipment Corporation). V roce 1983 byl standardizován pod označením IEEE 802.3, jehož charakteristickým rysem byla přístupová metoda CSMA/CD (Carrier Sense Multiple Access/Collision Detection). [1]

Tato přístupová metoda je založena na následujícím principu. Pokud chce některá stanice vysílat, musí si nejprve zkontrolovat, obsazenost kanálu (zjistit, zda již nevysílá jiná stanice). Jestliže již vysílá jiná stanice (kanál je obsazen), dále je kontrolována obsazenost kanálu a stanice, která chce vysílat čeká. Jakmile je kanál volný (nevysílá žádná jiná stanice), dojde k vysílání dat danou stanicí. Ovšem může zde dojít k situaci, kdy v ten samý okamžik, kdy je kanál volný, začne vysílat i jiná stanice. Proto stanice, která vysílá, kontroluje zda-li signály šířící se vedením odpovídají tomu, co sama vysílá. Pokud tomu tak není, daná stanice přeruší vysílání dat a dojde k detekci kolize. Při detekci kolize je danou stanicí vyslán do lokální sítě signál, který informuje zbylé stanice o kolizi a vysílaná data jsou zahozena. Obě stanice se následně odmlčí na náhodný časový interval a jakmile je kanál volný, znovu začnou vysílat ta samá data. [18, 19]

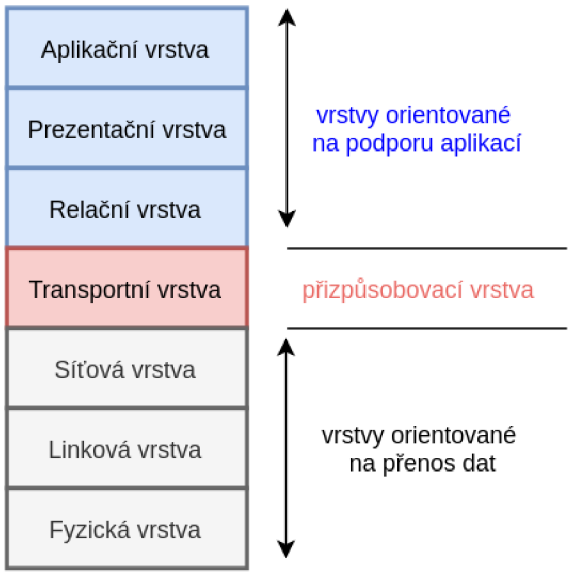
Ethernet s touto přístupovou metodou již nebyl vhodný pro vyšší rychlosti datového přenosu. Vzhledem k tomu pro rychlost od 1 Gb/s vznikl plně duplexní Ethernet. Tento typ Ethernetu využívá dvoubodových spojů, proto zde mohou vysílat oba uzly současně a proti sobě. Dále zde není použita žádná přístupová metoda, díky čemuž mohlo docházet k zvyšování rychlosti i vzdálenosti dosahu. Změny se dočkalo i přenosové médium, kdy se přešlo ze sdíleného média (koaxiálního kabelu) na dvojici médií nezávislých pro každý směr (např. kroucenou dvojlinku, optický kabel). Při použití přenosového média optického kabelu bylo možné Ethernet použít na vzdálenostech až stovek kilometrů, čímž se stal Ethernet použitelný nejen v lokálních sítích (LAN), ale i v metropolitních sítích (MAN) a rozlehlých sítích (WAN). [20]

U následující verze 10 Gb/s Ethernetu byla používána, jako přenosové médium stále kroucená dvoulinka a optický kabel. U současné používané nejvyšší verze Ethernetu 100 Gb/s je jako přenosové médium používán twinax a optický kabel. V současnosti se vyvíjí firmware pro plnohodnotný 200 Gb/s a 400 Gb/s Ethernet. [20, 21]

2.1 Model ISO/OSI

Na začátku 80. let 20. století vypracovala Mezinárodní organizace pro standardizaci (ISO) model nazvaný OSI (Open Systems Interconnection) s cílem strukturovat a standardizovat svět datové komunikace a sítí. Model OSI se skládá ze sedmi vrstev viz Obr. 2.1. Tyto vrstvy vzájemně spolupracují, přičemž každá plní určitou roli

v síťovém komunikačním procesu. Princip spočívá v tom, že vyšší vrstva převezme úkol od podřízené vrstvy, zpracuje jej a předá nadřízené vrstvě. Tři nejnižší vrstvy (síťová, linková, fyzická) modelu OSI viz Obr. 2.1 se zabývají přenosem dat (přenášená data nijak nezpracovávají a ani neinterpretují), zatímco tři nejvyšší vrstvy (aplikační, prezentační, relační) data nějakým způsobem zpracovávají nebo alespoň interpretují, tak aby vyšli vstříc potřebám jednotlivých aplikací. Prostřední vrstva (transportní) má fungovat jako jakési přizpůsobení. Detailnější popis jednotlivých vrstev je uveden níže. [22, 23, 18]



Obr. 2.1: Model OSI

Aplikační vrstva je určitou aplikací (např. oknem v programu) zpřístupňující uživatelům síťové služby. Tedy zajišťuje interakci mezi aplikačním programem a sítí. Protokoly fungující na této vrstvě zajišťují např. přístup k souborům nebo vzdálený přístup k tiskárnám. [18, 19]

Prezentační vrstva má na starosti např. kompresi dat (snížení velikosti dat za účelem jejich rychlejšího přenosu), konverzi dat (zakódování dat, takže data nebudou dostupná neautorizovaným osobám), překlad protokolu (konverze dat z jednoho protokolu do druhého, což poté umožňuje přenos mezi různými platformami a operačními systémy). [18, 19]

Relační vrstva navazuje a po skončení přenosu ukončuje spojení mezi dvěma aplikacemi. Dále zajišťuje bezproblémový přenos z jedné aplikace do druhé a popřípadě hlášení chyb. [18, 19]

Transportní vrstva přizpůsobuje možnosti a způsob fungování trojice nižších vrstev, tomu co požadují tři vyšší vrstvy (např. tři nejnižší vrstvy mohou fungo-

vat nespolehlivě, nestarají se o nápravu eventuálních chyb při přenosu, zatímco vyšší vrstvy požadují spolehlivý přenos). Dále tato vrstva zajišťuje komunikaci mezi dvěma koncovými uzly (rozlišuje různé příjemce a odesílatele v rámci každého jednotlivého uzlu). K jednotlivým paketům je přidán číselný identifikátor příslušného přechodového uzlu čímž je specifikován odesílatel a příjemce. [23, 18, 19]

Síťová vrstva odpovídá za přenos datových paketů přes mezilehlé uzly ke konečnému příjemci. Tedy zajišťuje tzv. směrování, kdy hledá volbu trasy při spojení tak, aby nedocházelo k přetížení sítě. Mezi zařízení, která pracují na síťové vrstvě patří routery a switche. [23, 18, 19]

Linková vrstva zajišťuje přenos linkových rámců výhradně mezi přímými sousedy v síti.Tedy z pohledu nižší vrstvy sestavuje jednotlivé bity do tzv. linkových rámců (frames). Musí rozpoznat začátek i konec každého jednotlivého rámce. Dalšími funkcemi linkové vrstvy mohou být např. řízení toku dat (zabraňuje tomu, aby odesílatel zahlcoval příjemce), zajišťování spolehlivého přenosu (kontrola, zda při přenosu nedošlo k nějaké chybě). [23, 18, 19]

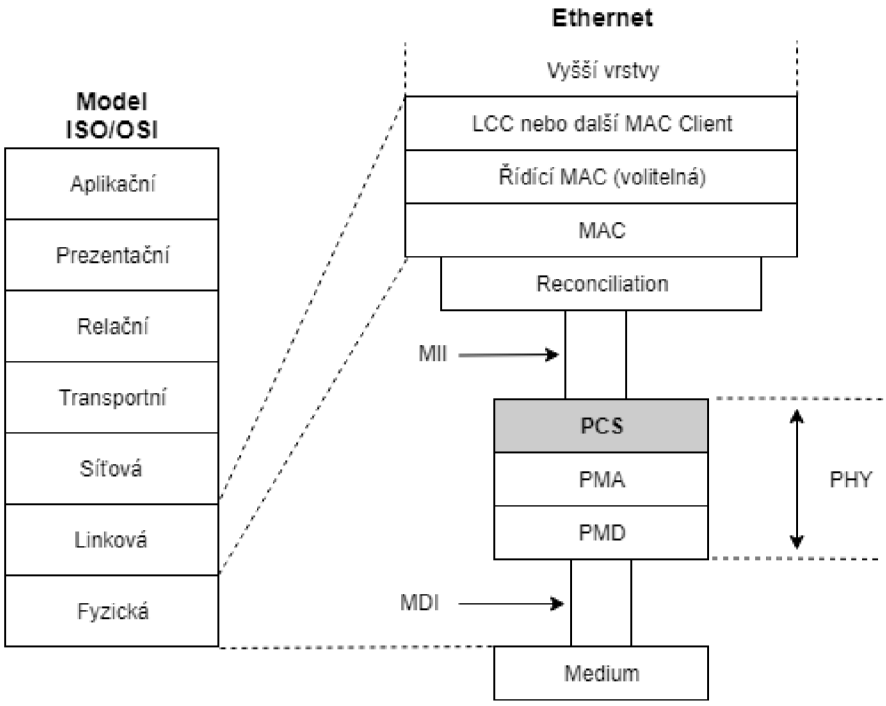
Fyzická vrstva přenáší jednotlivé bity, které tvoří hlavičky protokolů a přenášená data. Musí tedy nabízet služby pro příjem nebo odeslání bitu. Proto, aby se tak mohlo stát musí se fyzická vrstva zabývat tím, jak jsou jednotlivé bity znázorněny na přenášeném médiu (např. jak jsou kódovány a modulovány, jak je řešeno časování a synchronizace, jaké jsou používány konektory a rozhraní a další). Vzhledem k tomu se pak rozlišuje např. synchronní a asynchronní přenos, přenos sériový a paralelní a další. [23, 18, 19]

2.2 Vrstvy a rozhraní Ethernetu

Z hlediska referenčního modelu ISO/OSI lze Ethernet zařadit do linkové a fyzické vrstvy viz Obr. 2.2. Linková vrstva se u Ethernetu skládá ze dvou vrstev: LCC (Link Layer Control) a MAC (Media Access Control). Fyzická vrstva je složena ze tří podvrstev: PCS (Physical Coding Sublayer), PMA (Physical Medium Attachment) a PMD (Physical Medium Dependent) viz Obr. 2.2. Spojení mezi MAC vrstvu a fyzickou vrstvu zajišťuje rozhraní MII (Media Independent Interface). Právě těmto základním vrstvám a rozhraní Ethernetu bude věnována pozornost níže v této kapitole. [2]

Jak již bylo zmíněno výše z hlediska modelu ISO/OSI do linkové vrstvy patří dvě podvrstvy LCC a MAC. Vyšší **podvrstva LCC** zajišťuje, aby v jedné síti bylo možné používat více protokolů (např. IPv4, IPv6, IPX). Tato vrstva tedy poskytuje mechanismus multiplexování mezi těmito protokoly. Funkcí této podvrstvy může být taktéž řízení toku dat a oprava chyb při přenosu (kontrolní součet CRC). Nižší **podvrstva MAC** zajišťuje adresování a výběr přístupu k médiu (např. CSMA/CD,

Token passing). Další funkcí podvrstvy MAC může být kontrola maximální velikosti paketu nebo kontrola formátu paketu (zda-li obsahuje preambuli, kontrolní součet CRC a další). [21, 24]



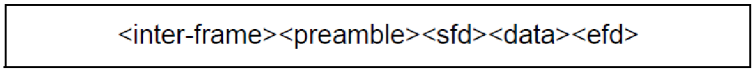
Obr. 2.2: Vztah mezi modelem ISO/OSI a Ethernetem

Rozhraní MII (Media Independent Interface) propojuje fyzickou vrstvu (PHY) s MAC vrstvou viz Obr. 2.2. Toto rozhraní umožňuje použít různé typy fyzické vrstvy pro propojení k různým mediím (např. kroucená dvojlinka, optický kabel), aniž by došlo k novému návrhu nebo nahrazení hardwaru MAC vrstvy. Díky tomu může být použita jakákoliv MAC vrstva s libovolnou fyzickou vrstvou nezávisle na médiu pro přenos signálu. Vybrané varianty rozhraní MII jsou vypsány v Tab. 2.1. [2, 21]

Tab. 2.1: Varianty rozhraní MII [2, 21]

GMII	Gigabit Media Independent Interface
XGMII	10-Gigabit Media Independent Interface
XLGMII	40-Gigabit Media Independent Interface
CGMII	100-Gigabit Media Independent Interface
200GMII	200-Gigabit Media Independent Interface
400GMII	400-Gigabit Media Independent Interface

Pakety jsou přes rozhraní MII přenášeny v datovém toku, který představuje proud posloupnosti bajtů. Každý bajt předává datový nebo řídící znak (oktet). Část datového toku je znázorněn na Obr. 2.3 [21]



Obr. 2.3: Datový tok na MII [2]

Inter-frame je doba, během které se nevyskytují žádné rámce dat, tedy jedná se o mezeru mezi rámci. Inter-frame začíná řídícím znakem Terminate, pokračuje Idle řídícími znaky. ***Preambule*** začíná vysílání rámců z MAC vrstvy, kdy první bajt je řídící znak Start. ***SFD*** označuje začátek MAC rámce. ***Data*** jsou tvořeny datovými znaky. ***EFD*** označuje konec rámce a představuje řídící znak Terminate, který se započítává do mezery mezi rámci. [21]

Fyzickou vrstvu tvoří tři podvrstvy (PCS, PMA, PMD), přičemž každá plní určitou funkci. Nejvyšší a zároveň nejsložitější **podvrstva PCS** fyzické vrstvy má za úkol kódování a dekódování dat, rozrušení periodicity dat a synchronizaci dat. Nižší **podvrstva PMA** provádí multiplexování n fyzických linek do x PCS linek případně opačně. Podle použité verze Ethernetu (100 Gb/s, 200 Gb/s) se může lišit počet PCS linek. Stejně tak se může lišit počet fyzických linek vzhledem k použitému přenosovému médiu (kroucená dvojlinka, optický kabel). Další funkcí této vrstvy může být serializace a deserializace dat. Nejnižší **podvrstva PMD** definuje přímo přenosové médium (optický kabel, kroucenou dvojlinku), tedy převádí kódovaná data na signál pro dané médium nebo obráceně. V případě použití přenosového média optického kabelu se může jednat o optické transceivery. [2, 21]

3 Podvrstvy fyzické vrstvy 400G Ethernetu

Jelikož se tato práce zabývá implementací jednotky PCS pro 400 Gb/s Ethernet, jsou v této kapitole podrobně popsány podvrstvy (PCS, PMA, PMD) fyzické vrstvy pro 400 Gb/s Ethernet, které spolu vzájemně souvisí. Zejména je zde věnována pozornost na podrobnému popisu funkce PCS podvrstvy.

3.1 Podvrstva PCS

Podvrstva PCS (Physical Coding Sublayer) je podvrstvou fyzické vrstvy (PHY) známá pod označením 400GBASE-R PCS viz Obr. 2.2. Tato vrstva obsahuje vysílací (TX - Transmit) a přijímací (RX - Receive) procesy. Podvrstva PCS při komunikaci s rozhraním 400GMII využívá osmy-oktetovou šířku dat (osmkrát osm bitů, tedy 64 bitů) označovaných TXD (Transmit Data) nebo RXD (Receive Data). Zda se jedná o datový nebo řídící znak (oktet) je určeno pomocí řídících signálů označovaných TXC (Transmit Control) nebo RXC (Receive Control). [2]

Podvrstva PCS využívá při komunikaci s podvrstvou PMA šestnáct kódovaných bitových toků (šestnáct linek). Právě šestnáct PCS linek je dáno tím, že se jedná o nejmenší společný násobek všech možných variant počtu fyzických linek (variant fyzické vrstvy) viz Tab. 3.1. Tím je zaručeno, že při multiplexování v podvrstvě PMA nedojde k rozdělení datového toku z jedné PCS linky na více fyzických linek. Specifikace definuje, že na každé PCS lince by měl být datový tok 26,5625 Gb/s, to odpovídá datovému toku na všech PCS linkách 425 Gb/s. Výsledná rychlost je tedy větší než 400 Gb/s vzhledem k tomu, že do datového toku jsou vkládány zarovnávací značky a přidávány paritní bity (vysvětlení viz níže v této kapitole). [2]

3.1.1 Vysílací proces

Základním prvkem vysílacího procesu fyzické kódovací podvrstvy je 64b/66b kódování. Toto kódování podporuje přenos datových a řídících znaků (oktetů). K přenášeným datům (payload) je při tomto kódování přidána synchronizační hlavička následujícím způsobem. [2]

Pokud je všech osm znaků datových (D_0 - D_7) viz Obr. 3.1, 64b/66b kódování přidá k TXD (přenášeným datům) z rozhraní 400GMII synchronizační hlavičku "01", čímž vznikne 66-bitový blok. Jestliže 64-bitový blok TXD z rozhraní 400GMII obsahuje některou z kombinací řídících znaků (oktetů) Idle (C_0 - C_7), Start (S_0), Terminate (T_0 - T_7), Error (C_0 - C_7), Ordered set (O_0) viz Obr. 3.1, je daný řídící znak dále kódován. Následně se provede 64b/66b kódování, kdy se přidá synchronizační hlavička "10", osmi-bitový blok typu pole a payload (kódované řídící znaky,

přenášená data nebo kombinace těchto dvou možností) viz Obr. 3.1. Toto přidání synchronizační hlavičky slouží k určení začátku bloku a synchronizaci dat. Součástí 64b/66b kódování je taktéž stavový automat, který rozhoduje o výstupních datech z 64b/66b kódování vzhledem k posloupnosti jednotlivých bloků (např. povolená posloupnost bloků: řídící blok se znakem Start následovaný datovými bloky a zakončen řídícím blokem se znakem Terminate, dle specifikace). [2, 21]

Input Data	**Sync**	**Block Payload**							
Bit Position: / Data Block Format:	0 1	2							65
$D_0\ D_1\ D_2\ D_3\ D_4\ D_5\ D_6\ D_7$	01	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
Control Block Formats:		**Block Type Field**							
$C_0\ C_1\ C_2\ C_3\ C_4\ C_5\ C_6\ C_7$	10	0x1E	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
$S_0\ D_1\ D_2\ D_3\ D_4\ D_5\ D_6\ D_7$	10	0x78	D_1	D_2	D_3	D_4	D_5	D_6	D_7
$O_0\ D_1\ D_2\ D_3\ Z_4\ Z_5\ Z_6\ Z_7$	10	0x4B	D_1	D_2	D_3	O_0	0x000_0000		
$T_0\ C_1\ C_2\ C_3\ C_4\ C_5\ C_6\ C_7$	10	0x87		C_1	C_2	C_3	C_4	C_5	C_6	C_7
$D_0\ T_1\ C_2\ C_3\ C_4\ C_5\ C_6\ C_7$	10	0x99	D_0		C_2	C_3	C_4	C_5	C_6	C_7
$D_0\ D_1\ T_2\ C_3\ C_4\ C_5\ C_6\ C_7$	10	0xAA	D_0	D_1		C_3	C_4	C_5	C_6	C_7
$D_0\ D_1\ D_2\ T_3\ C_4\ C_5\ C_6\ C_7$	10	0xB4	D_0	D_1	D_2		C_4	C_5	C_6	C_7
$D_0\ D_1\ D_2\ D_3\ T_4\ C_5\ C_6\ C_7$	10	0xCC	D_0	D_1	D_2	D_3		C_5	C_6	C_7
$D_0\ D_1\ D_2\ D_3\ D_4\ T_5\ C_6\ C_7$	10	0xD2	D_0	D_1	D_2	D_3	D_4		C_6	C_7
$D_0\ D_1\ D_2\ D_3\ D_4\ D_5\ T_6\ C_7$	10	0xE1	D_0	D_1	D_2	D_3	D_4	D_5		C_7
$D_0\ D_1\ D_2\ D_3\ D_4\ D_5\ D_6\ T_7$	10	0xFF	D_0	D_1	D_2	D_3	D_4	D_5	D_6	

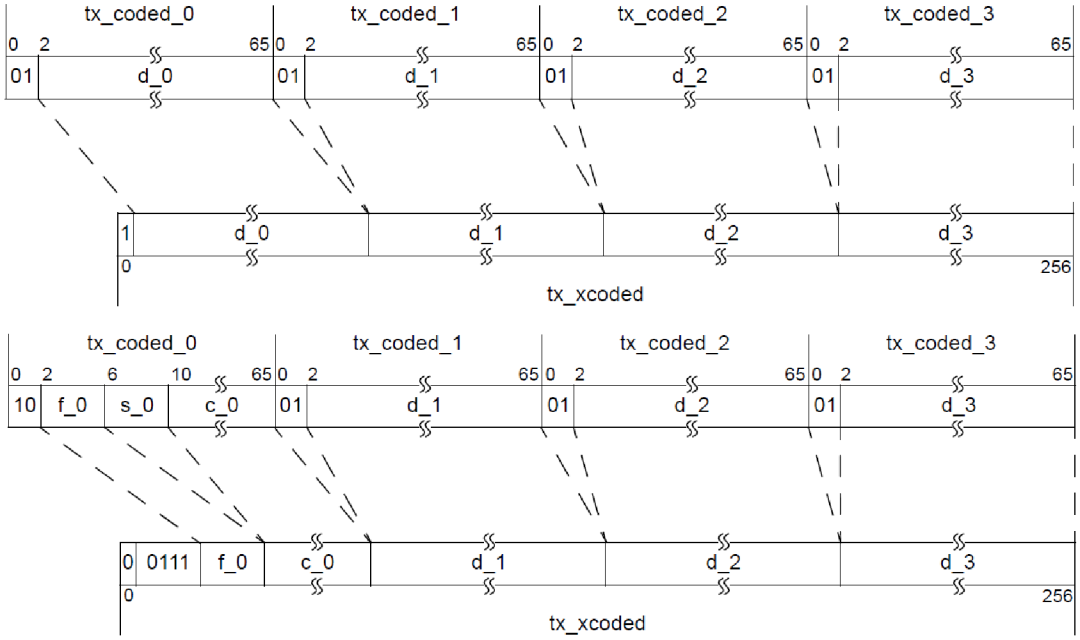
Obr. 3.1: 64b/66b formát bloku [21]

V dalším kroku dochází k mazání Idle sekvencí pro vytvoření prostoru pro vkládané zarovnávací značky. Mazání Idle sekvencí je též dovoleno pro kompenzaci rychlosti mezi hodinovými doménami. Čtyři 66-bitové kódované bloky jsou následně překódovány. [2]

Transcoderem na 257-bitový blok viz Obr. 3.2. Zde dochází k redukci synchronizačních hlaviček čtyř kódovaných 66-bitových bloků, aby vznikl prostor pro kontrolní symboly (paritní bity) zpráv z FEC Encoderu (Forward Error Correction). [2]

Pokud jsou synchronizační hlavičky všech čtyřech 66-bitových bloků "01"(všechny oktety jsou datové), nejnižší bit 257-bitového bloku bude ’1’ a bude následován daty z každého 66-bitového bloku viz Obr. 3.2. V případě, že některý z bloků bude obsahovat synchronizační hlavičku "10"(66-bitový blok obsahuje řídící znak), nejnižší bit 257-bitového bloku bude ’0’, následován čtyřmi bity, které určují typ 66-bitových bloků (’1’ - 66-bitový blok obsahuje pouze datové oktety, ’0’ - 66-bitový blok obsahuje

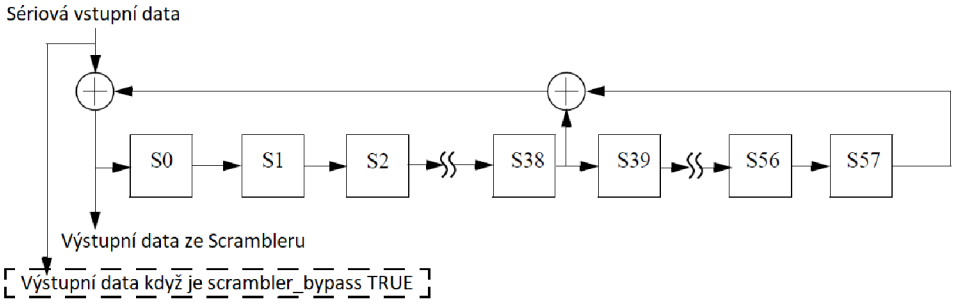
řídící znak). Za těmito čtyřmi bity budou dále data ze čtyřech 66-bitových bloků, kdy u nejnižšího 66-bitového bloku dojde ke smazání druhé části (4-bitů) bloku typu pole viz Obr. 3.2. [2]



Obr. 3.2: Transcodování čtyřech 66-bitových bloků [2]

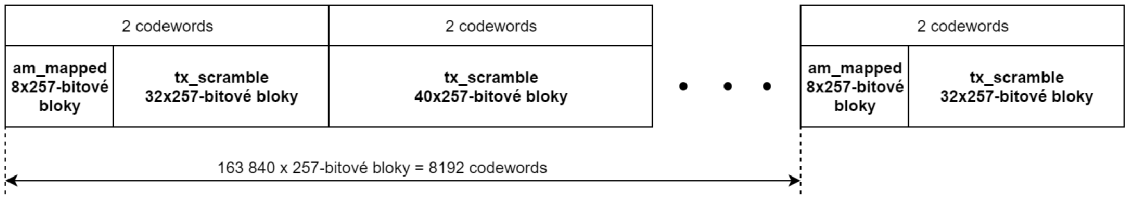
257-bitové bloky z Transcoderu jsou následně ve Scrambleru šifrovány podle daného polynomu viz rovnice 3.1. Toto šifrování, kde se využívá logické funkce XOR viz Obr. 3.3 zajistí, že data ze Scrambleru obsahují dostatečný počet změn úrovní, čímž dojde k rozrušení periodicity. To je důležité pro synchronizaci. [2, 21]

$$G(x) = 1 + x^{39} + x^{58} \tag{3.1}$$



Obr. 3.3: Scrambler [21]

Aby bylo možné seřadit a sesynchronizovat data z více linek, jsou do datového toku vkládány zarovnávací značky (2056-bitové bloky tedy 8x257-bitové bloky). Při vkládání dojde k přerušení datového toku a vložení zarovnávací značky po každém 163832 257-bitovém bloku ze Scrambleru viz Obr. 3.4. Struktura zarovnávacích značek je znázorněna na Obr. 3.5. [2]



Obr. 3.4: Periodické vkládání zarovnávacích značek [2]

Jelikož je v podvrstvě PCS použit FEC Encoder je každý 10280-bitový blok distribuován 10-bitovou symbolovou round robin distribucí. Tato distribuce rozdělí 10280-bitový blok na dvě 514-symbolové zprávy m_A a m_B pro FEC Encoderu. [2]

FEC Encoder je tvořen Reed-Solomon Encoderem, který pracuje nad GF(2^{10}), kde je velikost symbolu 10-bitů. Encoder zpracovává k symbolových zpráv pro vytvoření *2t* paritních symbolů, které jsou pak připojeny ke zprávě pro vytvoření kódových slov $n = k+2t$ symbolů. Rovnice 3.2 definuje paritní polynom, jehož koeficienty jsou symboly parity. [2]

$$p(x) = p_{2t-1} \cdot x^{2t-1} + p_{2t-2} \cdot x^{2t-2} + ... + p_1 \cdot x + p_0 \tag{3.2}$$

Pro tento FEC Encoder je Reed-Solomon kód označen RS*(n,k)*. V podvrstvě PCS pro 400BASE-R se využívá FEC Encoder RS(544,514), jehož funkce je založena na rovnici 3.3. Prvek konečného pole α v rovnici 3.3 je definován polynomem podle rovnice 3.4. [2]

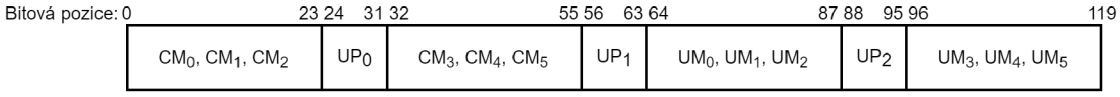
$$g(x) = \prod_{j=0}^{2t-1} (x - \alpha^j) = g_{2t}x^{2t} + g_{2t-1}x^{2t-1} + ... + g_1x + g_0 \tag{3.3}$$

$$\alpha = x^{10} + x^3 + 1 \tag{3.4}$$

Úkolem FEC Encoderu ve vysílacím procesu je tedy generovat paritní bity, aby příjemce mohl provést opravu případných poškozených dat. Posledním krokem vysílacího procesu je 10-bitová symbolová distribuce dvou kódových slov z FEC Encoderu do šestnácti PCS linek. [2]

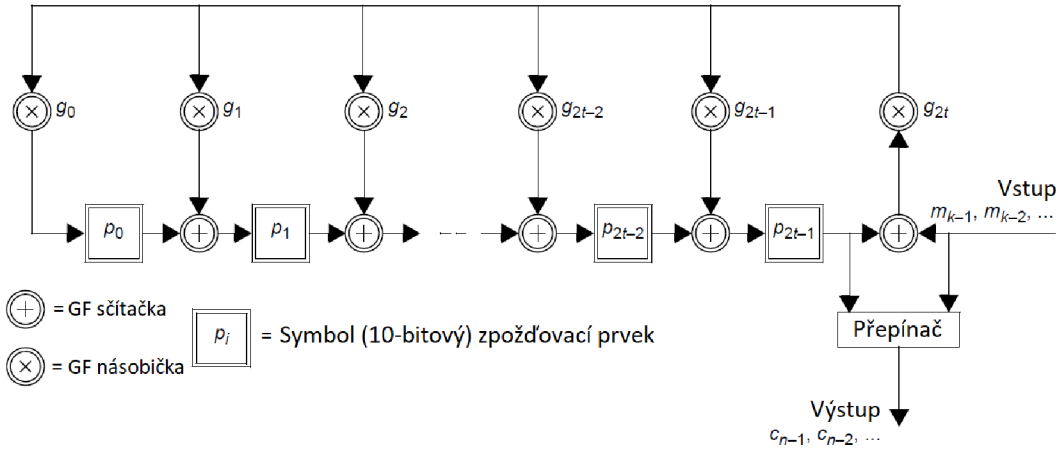
3.1.2 Přijímací proces

Přijímací proces pracuje obdobně jako vysílací jen v opačném pořadí. Nejprve musí být nalezeny v šestnácti PCS linkách zarovnávací značky, dle formátu viz Obr. 3.5 (kde CM_0 až CM_5 je společná část pro všechny značky, UM_0 až UM_5 je jedinečná část a UP_0 až UP_2 je unikátní blok, který je při příjmu ignorován). Ve chvíli, kdy je nalezena zarovnávací značka, dojde k jejímu tzv. uzamčení. Jelikož každý datový tok na šestnácti PCS linkách může být přijímán do podvrstvy PCS s různou časovým posunutím vůči předchozí lince, musí se datové toky na všech PCS linkách sesynchronizovat právě podle zarovnávacích značek. Specifikace pro 400 Gb/s Ethernet uvádí maximální možné zpoždění mezi první a poslední zarovnávací značkou na dané PCS lince 180 ns. [2]



Obr. 3.5: Formát zarovnávací značky [2]

Všechny nalezené a sesynchronizované datové toky z šestnácti PCS linek jsou následně přeskupeny do daných linek. Přeskupení datových toků je provedeno na základě jedinečné části (UM_0 až UM_5) zarovnávacích značek. Tato jedinečná část zarovnávacích značek určuje, na které z šestnácti PCS linek se daný datový tok má nacházet. Poté co jsou datové toky ze všech PCS linek sesynchronizovány a přeskupeny, jsou datové toky z jednotlivých linek rozděleny na dvě kódová slova pro FEC Decoder symbolovou (10-bitovou) distribucí. [2]



Obr. 3.6: FEC Decoder

Reed-Solomon Decoder extrahuje dvě kódová slova, v případě potřeby je opraví a odstraní kontrolní symboly (paritní bity). Oprava kódových slov je realizována na základě paritních bitů, kdy do sčítaček a násobiček vstupují kódová slova a paritní bity viz Obr. 3.6, přičemž je využito rovnice 3.3. Po zpracování Reed-Solomon Decoderem jsou vzniklé dvě 514-symbolové zprávy (m_A a m_B) distribuovány 10-bitovou round robin distribucí. Tedy ze dvou symbolových zpráv odpovídajících 40 blokům z Transcoderu se opět vytvoří přenášený datový tok. [2]

Z vytvořeného datového toku jsou následně mazány zarovnávací značky. Tyto zarovnávací značky jsou pravidelně přijímány z nižší podvrstvy PMA po 163832 257-bitových blocích viz Obr. 3.4. Ke smazání zarovnávacích značek musí dojít, aby bylo možné přijímaný datový tok dešifrovat v Descrambleru. [2]

Descrambler plní opačnou funkci Scrambleru, kdy je zde využit polynom z rovnice 3.1. Dešifrovaný 257-bitový datový blok z Descrambleru je pomocí zpětného Transcoderu kódován na čtyři 66-bitové bloky. Tedy zpětně se vytvoří synchronizační hlavičky všem 64-bitovým blokům. [2]

Vzhledem k tomu, že dochází po každých 163832 257-bitových blocích k smazání osmi 257-bitových bloků, které představují zarovnávací značky, dochází namísto těchto značek ke vkládání Idle sekvencí do datového toku. Tím dojde ke kompenzaci smazaných zarovnávacích značek, jelikož stejná velikost dat obdržených z podvrstvy PMA musí být odeslána do rozhraní 400GMII. Idle sekvence mohou být taktéž vkládány v případě kompenzace rychlosti mezi hodinovými doménami. Následně jsou 66-bitové bloky 66b/64b Decoderem dekódovány, čímž se vytvoří 64-bitový blok přenášených dat (RXD) a 8-bitový blok řídících signálů (RXC). Tyto datové bloky jsou následně odeslány do vyšších vrstev přes rozhraní 400GMII. [2]

3.2 Podvrstva PMA

Podvrstva PMA (Physical Medium Attachment) se nachází mezi podvrstvou PCS a podvrstvou PMD viz Obr. 2.2. Tato podvrstva zajišťuje u 400 Gb/s Ethernetu přizpůsobení datového toku z šestnácti PCS linek na n fyzických linek. [2]

Tab. 3.1: Varianty fyzické vrstvy pro 400 Gb/s Ethernet [2]

Typ fyzické vrstvy	**Počet linek**	**Kódování**	**Typ vlákna**	**Dosah [km]**
400GBASE-SR16	16	NRZ	multi-mode	0,1
400GBASE-DR4	4	PAM4	single-mode	0,5
400GBASE-FR8	8	PAM4	single-mode	2
400GBASE-LR8	8	PAM4	single-mode	10

Například pokud je navrhována verze fyzické vrstvy 400GBASE-LR8, je datový tok z šestnácti PCS linek přizpůsoben na osm fyzických linek viz Tab. 3.1. Přizpůsobení na fyzické linky se provádí pomocí bitového multiplexování. Podvrstva PMA dále může provádět serializaci a deserializaci dat. [2]

3.3 Podvrstva PMD

Podvrstva PMD (Physical Medium Dependent) je nejnižší vrstvou fyzické vrstvy (PHY) viz Obr. 2.2. Tato podvrstva v případě 400 Gb/s Ethernetu plní funkci převodu kódovaných dat (digitálního signálu) na optický signál nebo obráceně. Tedy tato podvrstva je realizována jako optický transceiver viz Obr. 3.7. Typ transceiveru je určen podle použitého typu fyzické vrstvy viz Tab. 3.1, což může ovlivňovat například kódování (NRZ nebo PAM4), dále například zda se jedná o multi-mode (kratší vzdálenosti, levnější) nebo single-mode (delší vzdálenosti, dražší) vlákno. V případě použití fyzické vrstvy 400GBASE-LR8, kde se využívá kódování PAM4, lze použít například optický transceiver QSFP-DD viz Obr. 3.7. [2, 25, 26]

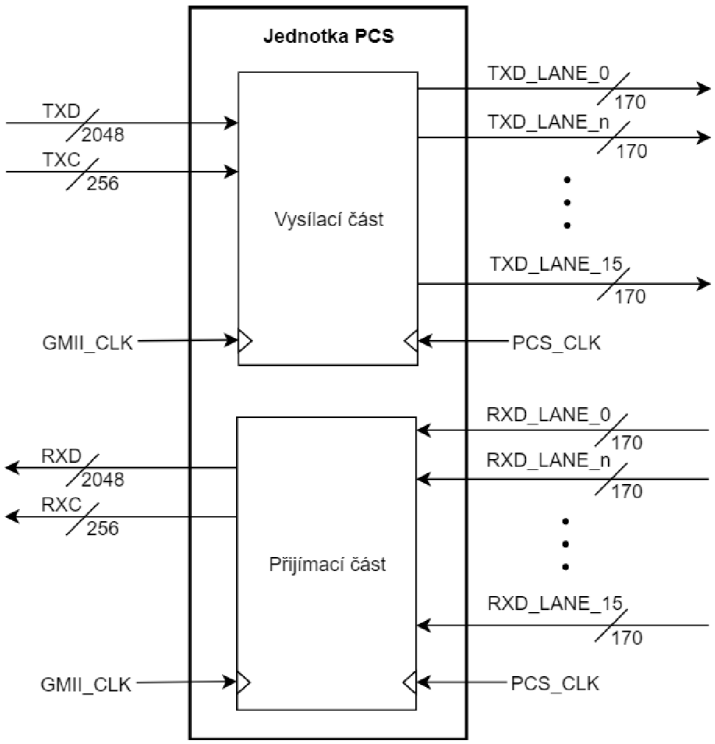
Obr. 3.7: Optický transceiver QSFP-DD [26]

Optický transceiveru je součástka, která umožňuje připojení vnějšího komunikačního vedení (optického kabelu). Základem optického transceiveru je vysílač (Transmitter) a přijímač (Receiver). Optický transceiveru se typicky skládá z optického rozhraní se vstupními konektory, převodníků optického signálu na elektrický signál, vlastní elektroniky pro zpracování signálů, napájecích obvodů, paměti a výstupního konektoru. [27]

4 Návrh jednotky PCS

V této kapitole je podrobně popsán návrh jednotky realizující podvrstvu fyzické vrstvy PCS. Návrh byl vytvořen tak, aby byl zvolen kompromis mezi datovou šířkou a frekvencí hodinového signálu. Velká datová šířka by znamenala malou frekvenci hodinového signálu (jednodušší na návrh z hlediska časování), ale velké využití zdrojů FPGA vzhledem k paralelnímu zpracování dat. Zatímco malá datová šířka by znamenala velkou frekvenci hodinového signálu, která by nebyla možná u současných FPGA. Vzhledem k těmto dvěma faktorům musela být při návrhu zvolena střední cesta, přičemž se vycházelo ze zkušeností při vývoji 100 Gb/s Ethernetu.

Navrhovaná jednotka PCS byla rozdělena do dvou základních částí a to na vysílací část a přijímací část. Na Obr. 4.1 je znázorněna navrhovaná jednotka z hlediska šířky dat a hodinových signálů na rozhraní této jednotky.



Obr. 4.1: Datové rozhraní navrhované PCS jednotky

Při návrhu datového rozhraní viz Obr. 4.1 a frekvencí hodinových signálů bylo vycházeno ze specifikace, která definuje nominální rychlost toku dat na každé z šestnácti PCS linek 26,5625 Gb/s. První návrhy uvažovaly, že celá jednotka PCS poběží na jedné hodinové frekvenci. Nejmenší možná vstupní datová šířka TXD z 400GMII se nabízela 64-bitů vzhledem k 64b/66b kódování. Ovšem při této šířce dat by muselo

FPGA pracovat na frekvenci hodinového signálu v řádech GHz, což je při využití současných FPGA nereálné. [2]

Proto bylo při návrhu datového rozhraní pro jednotku PCS vycházeno ze známé implementace 100 Gb/s Ethernetu, kde byla datová šířka TXD z rozhraní CGMII 512-bitů. První návrh datové šířky byl tedy proveden pro čtyřnásobek vstupní datové šířky, než jaká byla použita u 100 Gb/s Ethernetu, tedy pro vstupní datovou šířku TXD 2048-bitů. Při této datové šířce a úvaze, že celá jednotka PCS poběží na jedné frekvenci hodinového signálu by byla šířka dat na každé z šestnácti PCS linek 136-bitů. Velikost na každé PCS lince je dána tím, že nejprve jsou data zakódována 64b/66b Encoderem (2048b na 2112b), následně jsou překódována Transcoderem (2112b na 2056b), poté provedena symbolová distribuce 10280-bitového bloku (5x2056b) a zpracována FEC Encoderem (10280b na 10880b). 10880-bitový blok podělený pěti takty, které byly zapotřebí pro vytvoření bloku a šestnácti linkami, určí velikost datového toku na jedné PCS lince 136-bitů. Frekvence hodinového signálu by v tomto případě byla 195,3125 MHz viz rovnice 4.1.

$$GMII\_CLK = \frac{Gbps\_na\_linku}{bitu\_na\_linku} = \frac{26,5625 \cdot 10^9}{136} = 195,3125\ MHz \tag{4.1}$$

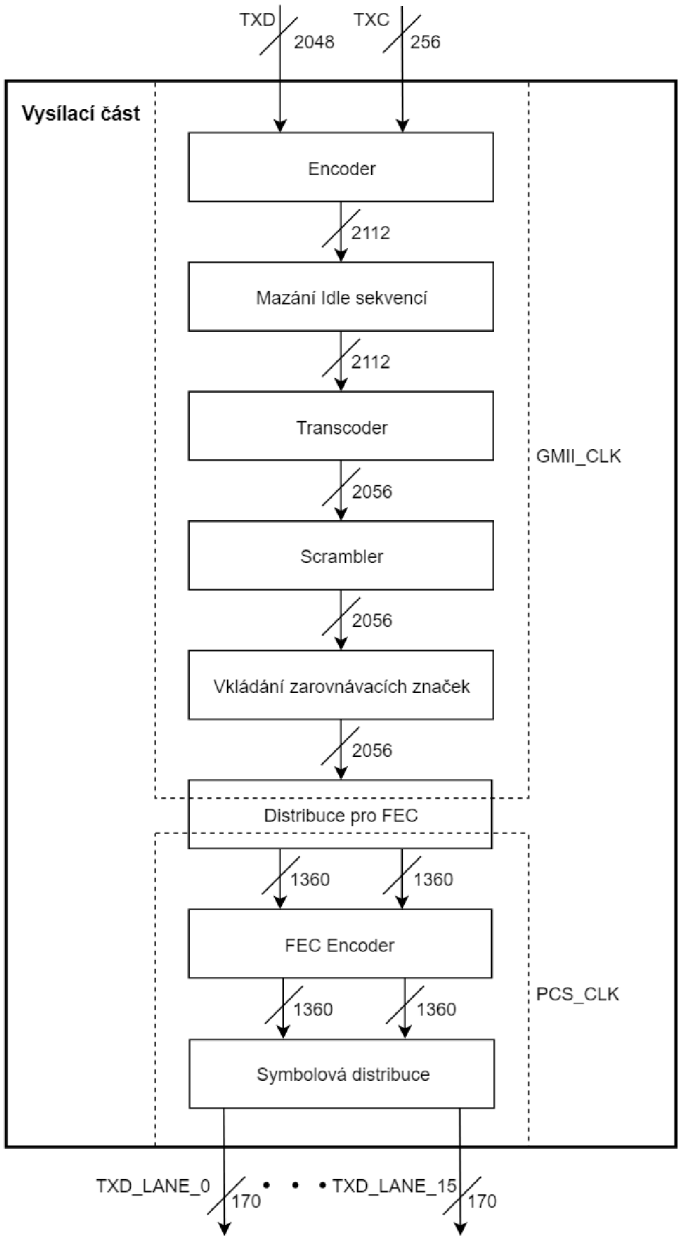
Jelikož podvrstva PCS obsahuje symbolovou 10-bitovou distribuci, která je nezbytná pro FEC Encoder respektive FEC Decoder, není možné vzhledem k navrhované vstupní datové šířce, aby jednotka fungovala na jedné frekvenci hodinového signálu. Důvodem je, že 2056-bitový blok nejde distribuovat po symbolech (deseti bitech). Proto byla provedena optimalizace návrhu, kdy na každé z šestnácti PCS linek bude datový tok 170-bitů (vysvětlení změny výstupní datové šířky viz níže v podkapitole návrh vysílacího procesu). S touto změnou by frekvence hodinového signálu na rozhraní jednotky PCS a PMA byla 156,25 MHz viz rovnice 4.2

$$PCS\_CLK = \frac{Gbps\_na\_linku}{bitu\_na\_linku} = \frac{26,5625 \cdot 10^9}{170} = 156,2500\ MHz \tag{4.2}$$

Tedy navrhovaná PCS jednotka bude pracovat s rozhraním 400GMII s datovou šířkou TXD nebo RXD 2048-bitů při frekvenci hodinového signálu GMII_CLK 195,3125 MHz a na rozhraní s podvrstvou PMA s datovou šířkou na jedné PCS lince 170-bitů (2720-bitů při součtu datových šířek na všech šestnácti linkách) při frekvenci hodinového signálu PCS_CLK 156,25 MHz. Vzhledem ke dvěma frekvencím hodinového signálu je zřejmé, že v navrhované jednotce PCS bude zapotřebí asynchronní vyrovnávací paměť typu FIFO.

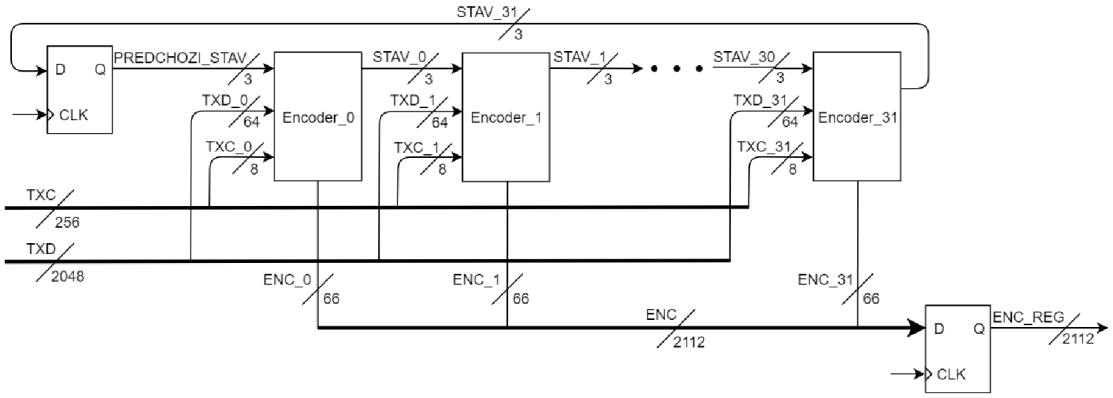
4.1 Vysílací část

Blokové schéma navrhované vysílací strany s datovými šířkami mezi jednotlivými bloky a hodinovými doménami je znázorněno na Obr. 4.2. Vysílací část bude pracovat se dvěma hodinovými doménami: GMII_CLK s frekvencí 195,3125 MHz a PCS_CLK s frekvencí 156,25 MHz. Navrhovaná vysílací strana se skládá z osmi základní bloků, přičemž každý plní určitou funkci. Funkce jednotlivých bloků je popsána níže v této kapitole. [2]



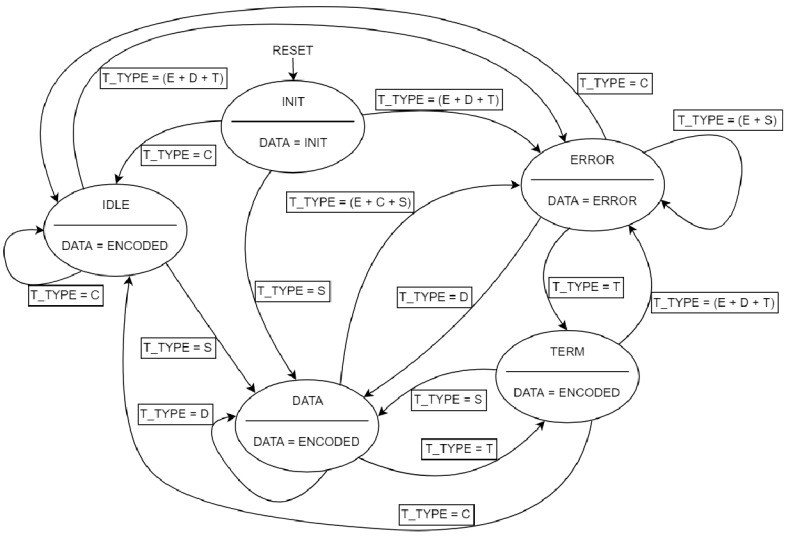
Obr. 4.2: Blokové schéma vysílací strany

První blok ve vysílacím procesu Encoder byl navržen tak, že bude pracovat na frekvenci hodinového signálu GMII_CLK. Podle návrhu datových šířek jednotky PCS bude do tohoto bloku vstupovat datový blok TXD s šířkou 2048-bitů a blok řídících signálů TXC s šířkou 256-bitů. Výstupem budou zakódované data s šířkou 2112-bitů. Jelikož v tomto bloku bude prováděno 64b/66b kódování, bude zde dvaatřiceti těchto Encoderů. Při prvotním návrhu bylo tedy uvažováno paralelní zapojení všech dvaatřiceti 64b/66b Encoderů viz Obr. 4.3.



Obr. 4.3: První návrh Encoderu

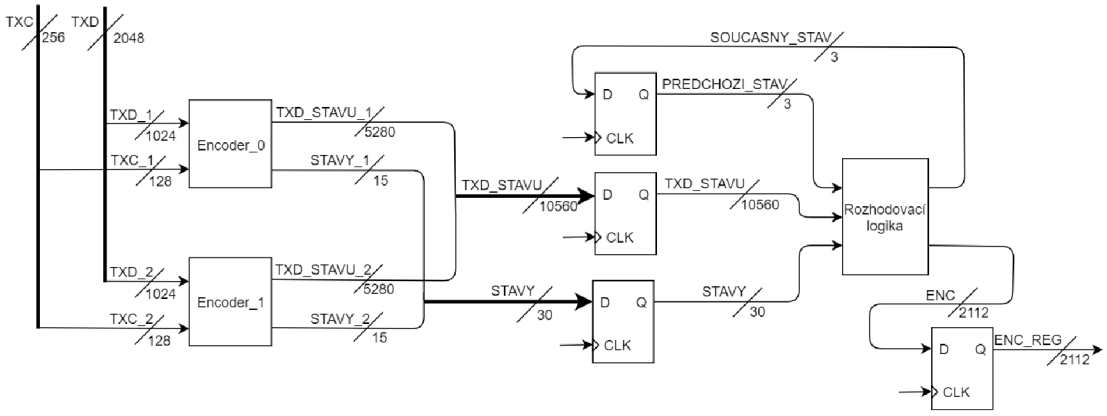
Jelikož všechny 64b/66b Encodery jsou stavové, tedy následující blok závisí na stavu bloku předchozího viz Obr. 4.4, byl první návrh bloku Encoder v praxi nepoužitelný vzhledem k nesplnění podmínek časování (viz Implementace jednotky PCS).



Obr. 4.4: Stavový automat 64b/66b Encoderu na vysílací straně

Z tohoto důvodu byl proveden druhý návrh bloku Encoder, kterým byly vyřešeny problémy s časováním (viz Implementace jednotky PCS). Tento návrh bude provádět zakódování dvou 1024-bitových bloků TXD. Tedy při zakódování tohoto bloku bude zapojeno šestnáct 64b/66b Encoderů stejným způsobem jako je na Obr. 4.3.

Princip druhého návrhu spočívá v tom, že v prvním hodinovém taktu bude provedeno zakódování dvou 1024-bitových bloků TXD pro všechny možnosti vstupních stavů (pět možných stavů) s určením posledního (výsledného) stavu. Výsledky budou následně uloženy do registrů viz Obr. 4.5. V druhém taktu bude pomocí rozhodovací logiky na základě předchozího stavu vybrán z první pětice 1024-bitových bloků z registru právě jeden a určen poslední stav tohoto bloku. Na základě tohoto stavu bude vybrán z druhé pětice 1024-bitových bloků z registru právě jeden a určen současný stav. V tomto taktu bude taktéž prováděno zakódování další dvou 1024-bitových bloků TXD, čímž bude docíleno souvislého datového toku.

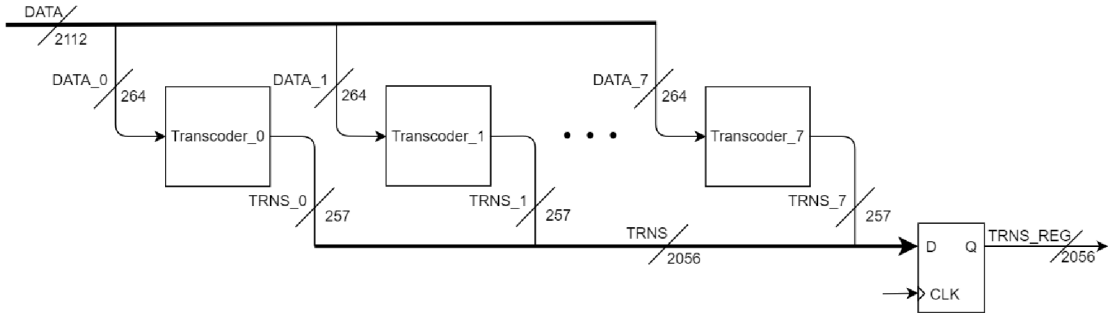


Obr. 4.5: Optimalizovaný návrh Encoderu

Zakódovaná data budou následně zpracována v bloku, kde dochází k mazání Idle sekvencí. Zde bude vytvářen prostor v datovém toku pro později vkládané zarovnávací značky. Základem tohoto bloku bude vyrovnávací synchronní paměť FIFO. Součástí tohoto bloku bude taktéž kombinační logika, která bude mít úkol hledat Idle sekvence (66-bitové bloky obsahující osm Idle řídících znaků), hlídat zaplněnost FIFO paměti a v případně smazání Idle sekvence posouvat data v paměti FIFO. Synchronní FIFO paměť bude pracovat s frekvencí hodinového signálu GMII_CLK se vstupní datovou šířkou z Encoderu 2112-bitů a se stejnou výstupní datovou šířkou.

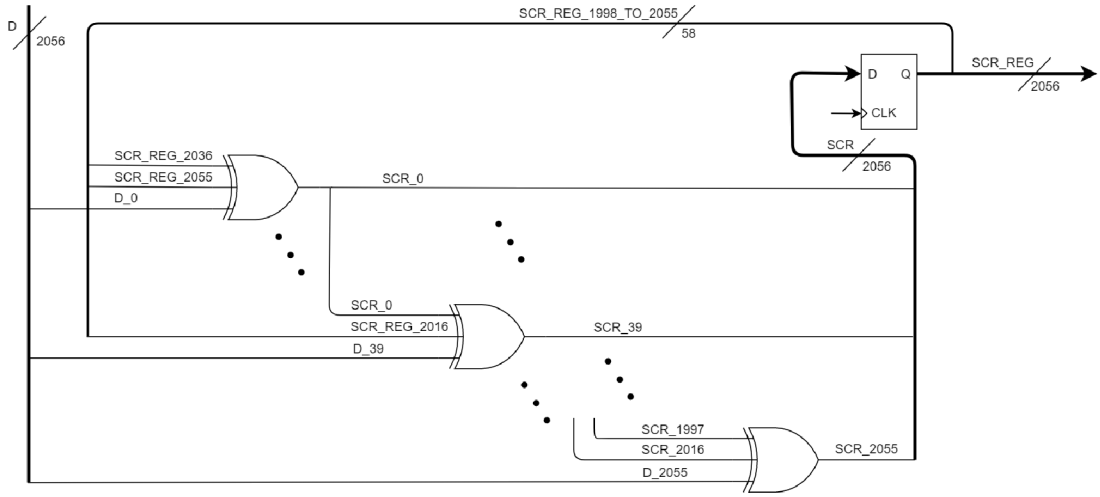
Další blok ve vysílacím procesu byl navrhnut Transcoder. Tento blok bude pracovat s frekvencí hodinového signálu GMII_CLK. Vstupní datová šířka zde bude 2112-bitů. Tento blok bude provádět redukci hlaviček čtyř 66-bitových bloků (odpovídajícím 264-bitům) na 257-bitový blok. Jelikož vstupní datová šířka sebou nese

dvaatřicet 66-bitovým bloků, tak tento blok bude realizován jako osm paralelně zapojených Transcoderů viz Obr. 4.6. Výstupní datová šířka tohoto bloku pak bude 2056-bitů. Jelikož zde nejsou jednotlivé bloky na sobě závislé, nebylo zapotřebí dělat optimalizaci jako v případě Encoderu.



Obr. 4.6: Návrh Transcoderu

U výstupních dat z Transcoderu (2056-bitů) bude rozrušována periodicita tohoto signálu pomocí Scrambleru, který bude pracovat s frekvencí hodinového signálu GMII_CLK. Tento blok bude fungovat na principu samo-synchronizačního polynomu viz rovnice 3.1. První návrh byl realizován zapojením XOR logických členů, kdy devětatřicáté výstupní data již závisela na předchozím výsledku XOR logické funkce. S dalšími vyššími výstupními daty rostlo zpoždění přes logické členy viz Obr. 4.7. Vzhledem k tomu se tento návrh Scrambleru stal při implementaci nepřijatelným z důvodů velké délky kombinační cesty (viz Implementace jednotky PCS). Proto byla provedena optimalizace pro splnění podmínek časování.



Obr. 4.7: První návrh Scrambleru

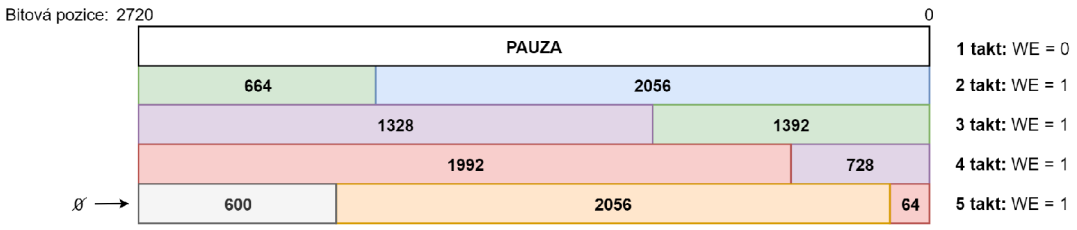
Návrh optimalizovaného Scrambleru vychází z maximálního využití šesti-vstupé LUT, které obsahuje architektura Virtex UltraScale+. Tento návrh se zakládá na vytvoření vektoru, který obsahuje sloučená vstupní data, posledních 58-bitů předchozích dat ze Scrambleru a současné data ze Scrambleru. Tento 4170-bitový vektor je zpracován pomocí XOR logických členů. Prvních devětatřicet výstupních bitů dat bude získáno obdobně jako v předchozím návrhu pomocí tří-vstupé logické funkce XOR. Další výstupní data již budou získána z pěti-vstupé logické funkce XOR. Tedy v případě získání čtyřicátého bitu výstupních dat budou vstupovat do logického členu stejné 3-bity jako u prvního členu plus další dva bity dle daného polynomu. Tím dojde ke snížení délky kombinační cesty.

Následující blok v návrhu bude zajišťovat vkládání zarovnávacích značek do datového toku. Každý 20480 hodinový takt bude přerušen datový tok a vloženy zarovnávací značky (2056-bitů). 2056-bitový blok se bude skládat ze zarovnávacích značek pro všech šestnáct PCS linek (1920-bitů), 133-bitů z PRBS9 (Pseudo Random Binary Sequencer) a 3-bitového statusu pro FEC Encoder. Komponenta vkládající zarovnávací značky bude pracovat na frekvenci hodinového signálu GMII_CLK. Součástí bloku zajišťujícího vkládání zarovnávacích značek bude tedy mapování zarovnávacích značek, PRBS9, který je dán polynomem viz rovnice 4.3 a čítač.

PRBS9 (Pseudo Random Binary Sequencer) bude realizován podle prvního návrhu Scrambleru viz Obr. 4.7, kde nebude problém se splněním podmínek časování (dlouhé kombinační cestě) vzhledem k malé vstupní datové šířce (133-bitů).

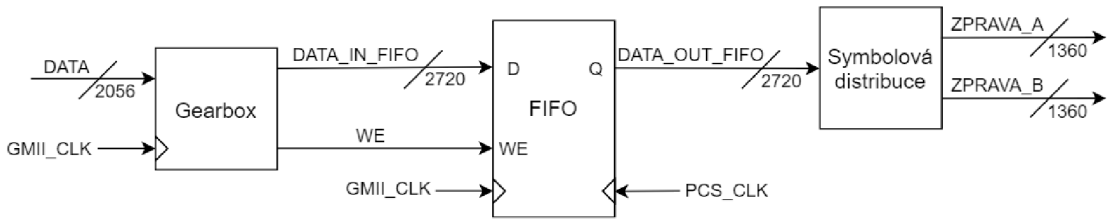
$$P(x) = x^9 + x^5 + 1 \tag{4.3}$$

Před samotným zpracováním FEC Encoderem bude zapotřebí data rozdistribuovat symbolovou (10-bitovou) distribucí. Jelikož výstupní datová šířka z bloku vkládajícího zarovnávací značky bude 2056-bitů, což není násobek desíti, byla distribuce před FEC navrhnutu s logikou pro transformování 2056-bitové vstupní datové šířky na 2720-bitovou šířku výstupních dat viz Obr 4.8. Transformace na právě tuto datovou šířku byla navrhnuta, jelikož optimálně vychází samotná transformace a tato datová šířka je vhodná pro další zpracování dat FEC Encoderem.



Obr. 4.8: Gearbox pro transformaci 2056b/2720b

Blok realizující symbolovou distribuci před FEC Encoderem bude tvořit Gearbox pro transformaci datové šířky, asynchronní FIFO paměť pro přechod mezi dvěma hodinovými doménami a samotná symbolová distribuce viz Obr 4.9.



Obr. 4.9: Blokové schéma distribuce před FEC Encoderem

Gearbox bude pracovat s frekvencí hodinového signálu GMII_CLK a bude mít za úkol transformaci vstupní 2056-bitové datové šířky na výstupní 2720-bitovou datovou šířku viz Obr 4.8. Součástí Gearboxu bude paměťový prvek (registr) pro ukládání dat, čítač a logika pro posouvání dat v registru. Důležitým výstupním signálem Gearboxu bude WE (Write Enable), který bude dávat po každém pátém hodinovém taktu informaci FIFO paměti, že nemá docházet k zápisu do této paměti, protože nebude v registru Gearboxu celý 2720-bitový blok.

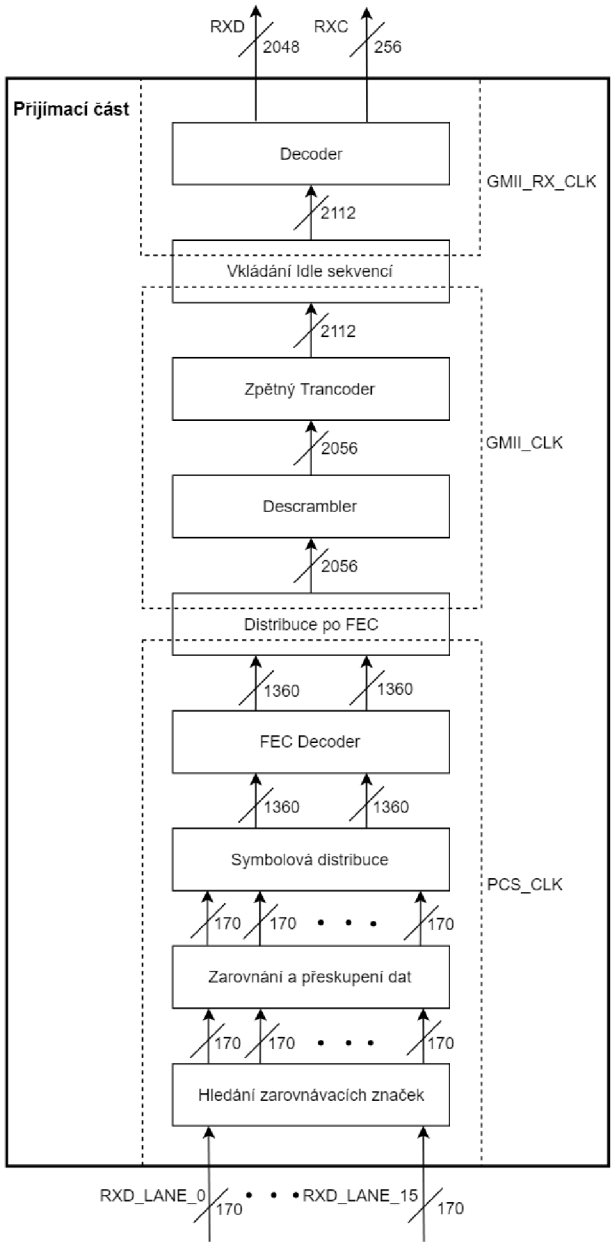
Jelikož zde dochází k transformaci datové šířky a cílem bude zajistit plynulý datový tok, bylo zde použito asynchronní FIFO paměti. Tato paměť bude zapisovat vstupní data (2720-bitů) s frekvencí hodinového signálu GMII_CLK a vyčítat výstupní data (2720-bitů) z této paměti s frekvencí hodinového signálu PCS_CLK. Tyto dvě hodinové domény budou vůči sobě v přesném poměru 1:0,8 což ve výsledku zajistí plynulý datový tok. Výstupní data budou následně distribuována symbolovou (10-bitovou) distribucí na dvě zprávy pro FEC Encoder.

V FEC Encoderu budou zpracovávány dvě 1360-bitové zprávy. Tento blok bude pracovat na frekvenci hodinového signálu PCS_CLK, kdy výstupem budou dvě 1360-bitové zprávy slova. Jelikož hlavním úkolem FEC Encoderu bude vytvářet paritní bity podle rovnice 3.2, bude možné tento blok realizovat jako síť XOR logických funkcí s paměťovým prvkem. Paměťový prvek bude mít za úkol po každém zpracování části zprávy uložit paritní bity a po zpracování celé zprávy (čtyř hodinových taktů) nahradit paritní bity vložené v Gearboxu za získané paritní bity v FEC Encoderu.

Na závěr budou data distribuována symbolovou (10-bitovou) distribucí do šestnácti PCS linek. Návrh předpokládá implementaci do programovatelného propojení (přepínačů), tedy bez využití logiky.

4.2 Přijímací část

Na Obr. 4.10 je zobrazeno blokové schéma navrhované přijímací strany jednotky PCS včetně datových šířek mezi jednotlivými bloky a navrhovanými hodinovými doménami. Na rozdíl od vysílací strany, kde byly použity dvě hodinové domény, zde budou použity tři hodinové domény: PCS_CLK s frekvencí 156,25 MHz, GMII_CLK s frekvencí 195,3125 MHz a GMII_RX_CLK s frekvencí 195,3125 MHz ±100 ppm viz Obr. 4.10. [2]



Obr. 4.10: Blokové schéma přijímací strany

Frekvence hodinového signálu GMII_RX_CLK byla do přijímací části přidána, jelikož hodinový signál PCS_CLK je odvozen od přijímaných dat (hodinové frekvence odesílatele), zatímco hodinový signál GMII_RX_CLK je z vnitřního oscilátoru (příjemce). Vzhledem k tomuto faktu specifikace připouští maximální toleranci každého oscilátoru ±100 ppm. Zároveň může docházet k vzájemnému fázovému posunutí obou hodinových signálů. [2]

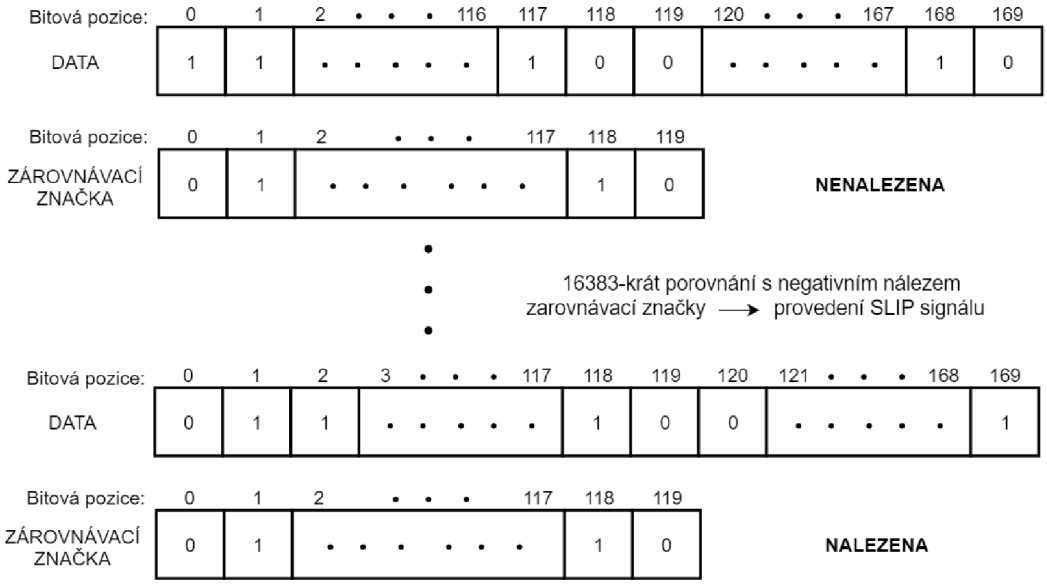
Při návrhu se nabízela možnost vložit toleranci ±100 ppm do hodinového signálu GMII_CLK. Tato volba by ale měla za následek nekorektní chování obvodu, jelikož vzhledem k návrhu je zapotřebí dodržet přesný poměr 1:0,8 mezi hodinovými doménami GMII_CLK a PCS_CLK. Tedy frekvence hodinového signálu GMII_CLK bude taktéž odvozena od přijímaných dat. [2]

Přijímací proces plní ve výsledku reverzní funkci vysílacího procesu, tedy použití jednotlivých bloků bude velmi podobné. Přijímací část se skládá z devíti základních bloků viz Obr. 4.10, jejichž funkce je popsána níže v této kapitole.

První blok v přijímacím procesu byl navrhnut tak, že bude v šestnácti PCS linkách hledat zarovnávací značky. Jelikož přijímací strana dělá reverzní funkce vysílací strany, byla navrhnuta datová šířka na každé PCS lince 170-bitů. Tento blok bude pracovat s frekvencí hodinového signálu PCS_CLK. Prvotní návrh byl proveden tak, že při prvním taktu byl 170-bitový blok uložen do nejnižších bitů 340-bitového registru. V dalším taktu byl následující 170-bitový blok uložen do nejnižších bitů registru a předchozí 170-bitový blok byl posunut do nejvyšších bitů registru. Následně byla provedena kontrola všech 170 možných pozic, kde by se mohl formát zarovnávací značky (viz Obr. 3.5) vyskytovat. Tento návrh byl optimalizován vzhledem k velkému počtu využití zdrojů FPGA, kdy by byla pro každou PCS linku potřeba paralelně 170-krát logika pro porovnávání dat.

Jelikož samotná specifikace předpokládá, že před nahozením linky pracuje Ethernet v tzv. testovacím režimu, kdy je tvořen datový tok Idle sekvencemi se zarovnávacími značkami, byla provedena optimalizace návrhu. Optimalizace spočívá v principu, že na každé PCS lince bude pouze jedna logika pro porovnávání datového toku se zarovnávacími značkami. Tedy 16384 hodinových taktů bude bude v datovém toku hledána zarovnávací značka na jedné pozici. Pokud nedojde k jejímu nalezení dojde k aktivaci signálu SLIP na jehož základě se provede posunutí datového toku na lince o jeden bit viz Obr. 4.11. Poté znovu dojde po dobu 16384 hodinových taktů k porovnávání datového toku se zarovnávacími značkami. Jelikož vysílací strana obsahuje FEC Decoder, bude dle specifikace validní zarovnávací značka obsahovat devět a více stejných niblů (4-bitů) z dvanácti z jedinečné části (UM_0 až UM_5) zarovnávací značky viz Obr. 3.5. Část zarovnávací značky unikátní blok (UP_0 až UP_2) bude při příjmu ignorována. [2]

Pokud dojde k nalezení validní značky spustí se čítač. Jakmile další nalezená zarovnávací značka (po 16384 hodinových taktech) bude odpovídat první nalezené validní značce, přiřadí se k této zarovnávací značce příznak LOCK. Výstupem tohoto bloku bude taktéž informace, o kterou z šestnácti PCS linek se jedná. Hledání zarovnávacích značek bude tedy obsahovat logiku pro porovnávání datového toku se zarovnávacími značkami, logiku pro určení čísla linky a čítač, který bude pracovat na frekvenci hodinového signálu PCS_CLK.



Obr. 4.11: Funkce SLIP signálu

Z tohoto optimalizovaného návrhu hledání zarovnávacích značek byl proveden výpočet pro maximální dobu než dojde k nalezení všech značek viz rovnice 4.4. Tato doba bude závislá od frekvence hodinového signálu PCS_CLK (t_h = 6,4 ns), intervalu mezi výskytem zarovnávací značky na dané PCS lince (t_{intrv} = 16384), šířce přijímaných dat na jedné PCS lince (bit = 170-bitů) a použitém transceiveru na FPGA (takt), který provádí posouvání dat v daném datovém toku na základě signálu SLIP.

$$t_{nalezeni} = t_h \cdot t_{intrv} \cdot (bit + takt) = 6,4 \cdot 10^{-9} \cdot 16384 \cdot (170 + 32) = 21,18\,ms \quad (4.4)$$

Funkcí transceiveru je posouvat data v datovém toku na základě signálu SLIP. Zpracování signálu SLIP trvá transceiveru určitý časový interval a až poté dojde k vykonání posunutí datového toku o jeden bit. Časový interval zpracování se liší

od použitého transceiveru. Z tohoto důvodu lze uvažovat tuto hodnotu jako proměnou (ze zkušenosti při návrhu 100 Gb/s Ethernetu se tato hodnota bude pohybovat okolo dvaatřiceti hodinových taktů).

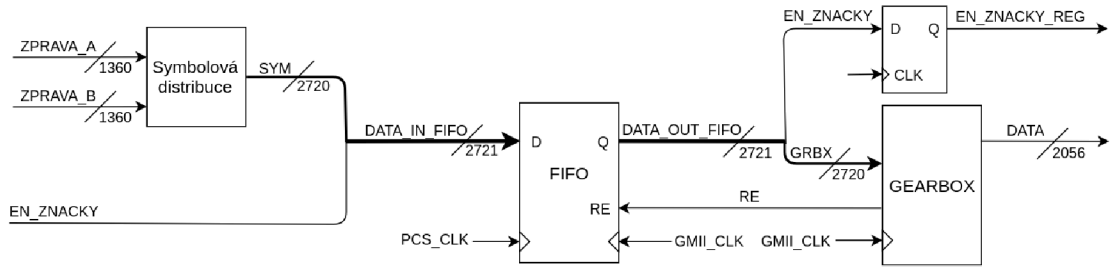
Další blok v navrhované přijímací straně zarovnání a přeskupení dat bude pracovat se stejnými datovými šířkami (170-bitů na linku) a na stejné frekvenci hodinového signálu PCS_CLK jako blok pro hledání zarovnávacích značek. Úkolem této komponenty bude, jakmile dojde aktivaci příznaku LOCK na všech PCS linkách, zarovnat tyto linky a přeskupit datové toky na dané PCS linky.

Zarovnání všech linek bylo navrženo tak, že v okamžiku, kdy dojde k aktivaci příznaku LOCK všech zarovnávacích značek na všech linkách, začne s příchodem první značky ukládání daného toku do posuvného registru. Jelikož specifikací je dáno maximální zpoždění mezi první a poslední zarovnávací značkou na linkách 180 ns, tak tento posuvný registr pro jednu linku bude mít datovou šířku devětadvacetkrát 170-bitů. S ukládáním do posuvného registru budou spuštěny čítače pro všechny linky. S příchodem dat se zarovnávací značkou dojde k zastavení čítače pro danou linku, čímž bude určen index (pozice) pro vyčítání z registru. Jakmile dojde k uložení všech datových toků se zarovnávacími značkami do registru (všechny čítače budou zastaveny), tak v daný okamžik budou vyčítána data z pozice registru daných jednotlivými čítači pro danou PCS linku. V případě většího zpoždění než 180 ns mezi zarovnávacími značkami na linkách, dojde k restartování procesu hledání zarovnávacích značek. [2]

Před samotným vysláním dat do dalšího bloku musí dojít k přeskupení datových toků podle zarovnávacích značek na dané PCS linky. K tomu bude využito výstupních bitů z bloku hledání zarovnávacích značek, které nesou informaci o daném číslu linky, kde by se měl daný datový tok nacházet. Přeskupení bylo navrhnuto jako multiplexovaní výstupního toku podle čísla linky. Zarovnané a přeskupené datové toky na dané PCS linky, budou následně distribuována 10-bitovou symbolovou distribucí na dvě 1360-bitové zprávy. Z hlediska implementace symbolová distribuce nebude využívat žádnou logiku, ale bude využito programovatelného propojení k této realizaci.

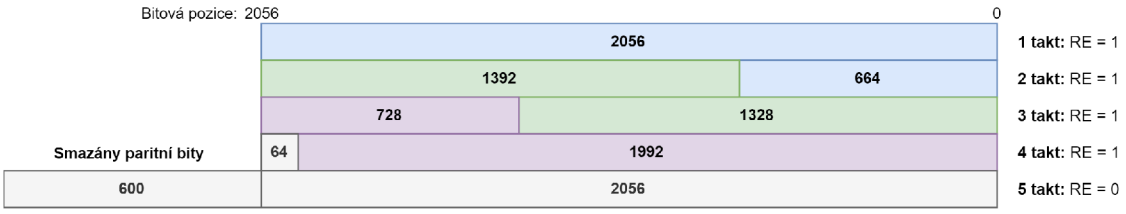
Tyto dvě 1360-bitové kódové slova jsou zpracovávána FEC Decoderem. Tento blok bude pracovat na frekvenci hodinového signálu PCS_CLK. Funkcí FEC Decoderu bude extrahovat tyto dvě zprávy a v případě nalezení chyby na základě paritních bitů opravit 5140-bitové slovo. FEC Decoder bude tvořen sčítačkami a násobičkami viz Obr. 3.6, paměťovým prvkem pro ukládání celého 5440-bitového slova (jelikož k opravě může dojít až po získání celého slova) a logikou, která bude mít za úkol opravu daných zpráv. Tento blok bude vnášet největší zpoždění signálu na přijímací straně vzhledem k své složitosti.

Do následujícího bloku distribuce po FEC Decoderu budou vstupovat dvě zprávy o datové šířce dvakrát 1360-bitů. Tyto zprávy budou nejprve distribuována symbolovou (10-bitovou) distribucí. Jelikož zde bude použit Gearbox, kde bude docházet k transformaci vstupní datové šířky 2720-bitů na výstupní datovou šířku 2056-bitů s cílem docílit plynulého datového toku, bude zde použita asynchronní paměť FIFO viz Obr. 4.12. Vstupní data do FIFO paměti budou zapisována s frekvencí hodinového signálu PCS_CLK a datovou šířku 2721-bitů. Výstupní data budou z této paměti vyčítána s frekvencí GMII_CLK se stejnou datovou šířkou. Datová šířka 2721-bitů je dána tím, že musí být přenášena i informace o tom, že tyto data obsahují zarovnávací značky, které musí být později smazány.



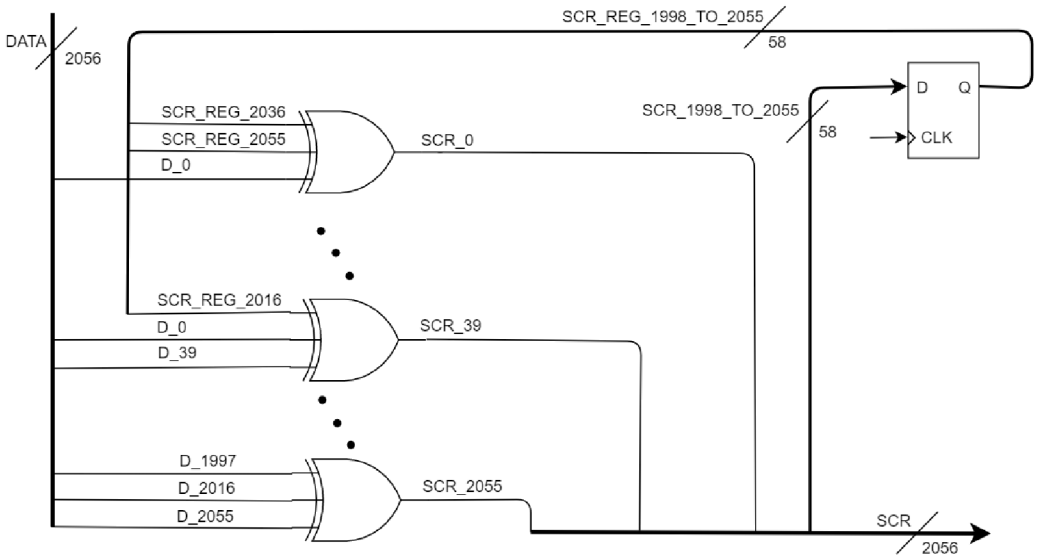
Obr. 4.12: Blokové schéma distribuce po FEC Decoderu

Úkolem Gearboxu je transformovat vstupní datovou šířku 2720-bitů na výstupní datovou šířku 2056-bitů. Jelikož tento blok bude pracovat na jedné hodinové frekvenci signálu GMII_CLK, docházelo by zde neustále k přetékání vnitřní paměti Gearboxu. Tento problém byl vyřešen tak, že každý pátý hodinový takt viz Obr. 4.13 bude docházet k deaktivaci signálu RE (Read Enable). Tento signál zajistí, že nedojde k vyčítání dat z FIFO paměti. Vzhledem k přesnému poměru frekvencí hodinových signálů PCS_CLK a GMII_CLK (0,8:1) nebude docházet k hromadění dat v FIFO paměti. Gearbox bude rovněž každý pátý hodinový takt mazat paritní bity z datového toku viz Obr. 4.13.



Obr. 4.13: Gearbox pro transformaci 2720b/2056b

Přetransformovaná data budou dále rozšifrována Descramblerem, který bude pracovat na frekvenci hodinového signálu GMII_CLK. Descrambler bude plnit reverzní funkci Scrambleru podle rovnice 3.1. Z hlediska návrhu se bude jednat o síť zapojených logických členů XOR viz Obr. 4.14. Výstupem tedy budou rozšifrovaná výstupní data s šířkou 2056-bitů.



Obr. 4.14: Návrh Descrambleru

Pomocí zpětného Transcoderu budou výstupní data z Descrambleru překódovaná. Tedy úkolem zpětného Transcoderu bude vytvářet synchronizační hlavičky k čtyřem 64-bitovým bloků. Vzhledem k tomu bude výstupní datová šířka z Transcoderu 2112-bitů. Tento blok bude pracovat se stejnou frekvencí hodinového signálu GMII_CLK jako Descrambler. Samotný návrh bude obdobný jako na vysílací straně viz Obr. 4.6, kdy bude paralelně zapojeno osm těchto zpětných Transcoderů.

Datový tok ze zpětného Transcoderu bude následně zpracován v bloku, kde bude docházet k mazání Idle sekvencí. Tento blok bude pracovat se dvěma hodinovými doménami GMII_CLK 195,3125 MHz GMII_RX_CLK 195,3125 MHz ±100 ppm a jejichž použití bylo vysvětleno již výše v přijímací části. Vzhledem k tomu zde bude použita asynchronní FIFO paměť. Do tohoto bloku bude vstupovat také signál, který bude signalizovat zarovnávací značky v datovém toku, čímž dojde k deaktivaci signálu WE (Write Enable) FIFO paměti. Deaktivace tohoto signálu způsobí, že nedojde k zapsání datového toku se zarovnávacími značkami, čímž dojde ke smazání zarovnávacích značek a nahrazení Idle sekvencemi. Tento blok bude tedy tvořen FIFO pamětí, kdy budou data zapisována s frekvencí hodinového signálu GMII_CLK a z paměti vyčítána s frekvencí hodinového signálu GMII_RX_CLK se stejnou

datovou šířkou 2112-bitů. Dále zde bude logika, která bude hlídat zaplněnost FIFO paměti a vkládání Idle sekvencí.

Posledním navrhovaným blokem v přijímací části bude Decoder. Vstupní datová šířka Decoderu bude 2112-bitů, což představuje dvaatřicet 66b/64b Decoderů. Jelikož stejně jako u Encoderu, všechny 66b/64b Decodery jsou stavové, kdy přechod mezi jednotlivými stavy vypadá obdobně jako u Encoderu viz Obr. 4.4, bylo při prvotním návrhu vycházeno již s optimalizované verze Encoderu na Obr. 4.5. Vzhledem k tomu že stavový automat je o trochu složitější než u vysílacího procesu, byla zde provedena druhá optimalizace Decoderu. Princip zde zůstal stejný jako u návrhu Encoderu. Pro splnění podmínek časování, zde byla provedena optimalizace, kdy na začátku procesu bylo provedeno zakódování čtyř datových bloků s šířkou 528-bitů. Výsledky jednotlivých Decoderů opět byly uloženy do registrů. V druhém hodinovém taktu budou pomocí rozhodovací logiky vybrána správná výstupná data. Decoder bude tedy pracovat na frekvenci hodinového signálu GMII_RX_CLK, kdy výstupem budou data RXD s šířkou 2048-bitů a řídící signály RXC s šířkou 256-bitů.

5 Implementace jednotky PCS

V této kapitole je popsán výběr FPGA pro implementaci jednotlivých bloků navržené jednotky PCS. Dále jsou zde výsledky syntézy implementovaných komponent z hlediska využití zdrojů FPGA a časových parametrů (zpoždění na nejdelší kombinační cestě). Tyto výsledky jsou zde jak pro prvotní implementace, tak i pro optimalizované implementace. V této části není uvažována implementace na vysílací straně FEC Encoderu ani na přijímací straně FEC Decoderu vzhledem k náročnosti implementace těchto bloků. Navrhovaná jednotka PCS tedy nebude schopna opravovat případné chyby v datovém toku.

Navržená jednotka PCS byla implementována do FPGA společnosti Xilinx s architekturou UltraScale+ s označením VU27P. Pro implementaci jednotky bylo zvoleno FPGA od společnosti Xilinx, jelikož při implementaci této podvrstvy se vycházelo z již známé implementace pro 100 Gb/s Ethernet. Dalším důvodem této volby bylo, že společnost Cesnet již má na tuto platformu napsané a optimalizované FIFO paměti, které byly v této jednotce taktéž použity. Samotný VHDL kód byl psán tak, aby byl co možná nejsnazší přechod z FPGA od Xilinxu na FPGA od Intelu. Tedy v případě implementace jednotky FPGA do platformy od Intelu bude potřeba nahradit FIFO paměti pro danou platformu a případně odladit zbývající bloky. Jelikož bylo použito FPGA od společnosti Xilinx, byl využit pro syntézu nástroj Vivado 2019. Syntézy byly prováděny na překladovém stroji s operačním systémem Linux, osmy jádrovým procesorem Intel Xeon s frekvencí jader 3,60 GHz, operační pamětí 32 GB a pevným diskem 500 GB.

Implementovaná jednotka se skládá ze dvou samostatných bloků pro vysílací část (tx_path_400g.vhd) a pro přijímací část (rx_path_400g.vhd) viz přiložené CD. Rozhraní entity pro vysílací část obsahuje vstupy pro hodinové domény, reset signály, bypass signál (umožňuje vynechání funkce daného bloku), vstupní data GMII_TXD (2048-bitů) a řídící signály GMII_TXC (256-bitů) z rozhraní 400GMII. Výstupem vysílací části je vektor PCS_OUT (2720-bitů) obsahující datový tok na šestnácti PCS linkách (viz tx_path_400g.vhd na přiloženém CD). Rozhraní entity přijímací části tvoří vstupy pro hodinové domény, reset signály, bypass signál, vektor PCS_LANES (2720-bitů) obsahující datový tok na šestnácti PCS linkách, signál SLIP_DONE z transceiveru o provedení posunutí dat o 1-bit. Výstupem je signál SLIP, který informuje transceiver o vykonání posunutí datového toku, data GMII_RXD (2048-bitů) a řídící signály GMII_TXC (256-bitů), které jsou odesílány do rozhraní 400GMII (viz rx_path_400g.vhd na přiloženém CD). Architektury těchto dvou entit jsou tvořeny zapojením dílčích komponent uvedených v Tab. 5.4 a Tab. 5.5.

Při implementaci navrhovaných bloků Encoderu, Scrambleru a Decoderu byly problémy s nesplněním podmínek podmínek časování. Z tohoto důvodu byly provedeny optimalizace u těchto daných komponent. Výsledky syntézy z hlediska využití zdrojů v FPGA a nejdelší kombinační cesty byly zpracovány do tabulek níže.

První blok ve vysílací části, u kterého bylo zapotřebí provést optimalizaci byl Encoder. Zde byl problém s dlouhou kombinační cestou viz Obr. 4.3, která měla zpoždění 8,810 ns. Vzhledem k tomu, že Encoder pracoval na frekvenci 195,3125 MHz (5,12 ns), byl zde proveden optimalizovaný návrh pro zkrácení délky kombinační cesty viz Obr. 4.5. Tento návrh již splňoval podmínky časování (zpoždění kombinační cesty 4,543) ovšem na úkor velkého nárůstu využití zdrojů FPGA viz. Tab. 5.1

Tab. 5.1: Výsledky syntézy implementací Encoderu

Iplementace	**Spotřeba LUT**	**Spotřeba FF**	**největší zpoždění [ns]**
1.	12005	4323	8,810
2.	17764	14907	4,543

Problém s časováním na vysílací straně byl řešen taktéž u Scrambleru, který pracuje na frekvenci hodinového signálu 195,3125 MHz (5,12 ns) jako Encoder. Zde byla dlouhá kombinační cesta způsobena zapojením tří-vstupých XOR logických funkcí viz Obr. 4.7. Maximální zpoždění průchodu kombinační části bylo přes dvaadvacet LUT, kdy výsledné zpoždění signálu bylo 8,810 ns. Proto byl proveden optimalizovaný návrh, kdy byl zkrácený průchod kombinační části na patnáct LUT. Jelikož optimalizace byla provedena za účelem maximálního využití šesti-vstupé LUT, které obsahují architektury UltraScale+, nedošlo zde ani k velkému nárůstu využití zdrojů FPGA viz. Tab. 5.2. Větší nárůst klopných obvodů aktivních na aktivní hranu hodinového signálu (FF), byl způsoben přidáním výstupního datového registru do druhé implementace.

Tab. 5.2: Výsledky syntézy implementací Scrambleru

Iplementace	**Spotřeba LUT**	**Spotřeba FF**	**největší zpoždění [ns]**
1.	5348	58	5,722
2.	5519	2114	3,937

Problém se splněním podmínek časování byl taktéž na přijímací straně u Decoderu, který pracuje na frekvenci hodinového signálu 195,3125 MHz (5,12 ns). Jelikož principiálně plní Decoder opačnou funkci Encoderu byla první implementace provedena podle optimalizovaného návrhu Encoderu viz Obr. 4.5. Tato implementace

ovšem u Decoderu nesplnila podmínky časování vzhledem k dlouhé kombinační cestě při předávání stavů jednotlivých 66b/64b Encoderů. Proto byla provedena druhá optimalizace Decoderu, která je popsána v části návrhu jednotky PCS. Druhou verzí implementace se výrazně zlepšili podmínky časování, kdy nebyl ani moc velký nárůst využití zdrojů v FPGA viz. Tab. 5.3.

Tab. 5.3: Výsledky syntézy implementací Decoderu

Iplementace	**Spotřeba LUT**	**Spotřeba FF**	**největší zpoždění [ns]**
1.	17801	15830	5,185
2.	17863	16327	2.845

Před spuštěním syntézy top komponenty vysílací nebo přijímací části, vždy byly provedeny syntézy jednotlivých navrhovaných bloků. Výsledky syntézy vysílací strany z hlediska využití zdrojů v FPGA jednotlivých komponent je v Tab. 5.4. Tedy na vysílací straně celkové využití zdrojů vybraného FPGA činí 43943 LUT (3,39 %), 34595 klopných registrů FF (1,33 %) a 67,50 blokových pamětí RAM (3,35 %). Doba trvání syntézy na daném překladovém stroji byla jednu hodinu a dvaatřicet minut.

Tab. 5.4: Výsledky syntézy vysílací strany

Komponenta	**Spotřeba LUT**	**Spotřeba FF**	**Spotřeba BRAM**
tx_block_encode_400g	17764	14907	0
pcs_tx_fifo	13731	8470	29,50
tx_block_transcode	1680	2056	0
scrambler58_ethernet	5519	2114	0
tx_am_insertion	195	2215	0
tx_distribution	5054	2120	38,00
tx_symbol_distr	0	0	0

V Tab. 5.4 jsou výsledky syntézy přijímací strany z hlediska využití zdrojů FPGA. Doba trvání syntézy na překladovém stroji přijímací straně byla sedmačtyřicet minut. Celkové využití zdrojů přijímací strany činí 87806 LUT (6,78 %), 41774 klopných obvodů FF (1,61%), 67,5 blokových RAM (3,35 %) a 78880 (6,70 %) LUTRAM (10,33%). V přijímací straně je již více bloků s větším využití zdrojů např. Decoder, pcs_rx_fifo nebo rx_deskew_400g, jehož implementace využívá LUTRAM k vytvoření posuvného registru. Symbolová distribuce zde opět nevyužívá logiky a programovatelného propojení.

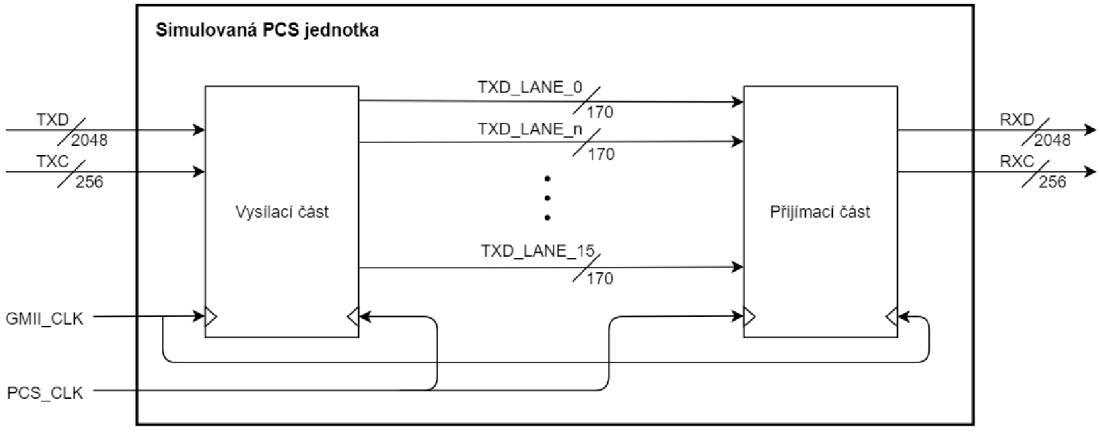
Výsledná jednotka PCS (vysílací i přijímací část) bude tedy využívat 131749 LUT (10,17 %), 76363 klopných obvodů FF (2,95 %), 135 blokovou paměť RAM(6,70 %) a 78880 LUTRAM (10,33 %) ze zvoleného FPGA.

Tab. 5.5: Výsledky syntézy přijímací strany

Komponenta	**Spotřeba LUT**	**Spotřeba FF**	**Spotřeba BRAM**	**Spotřeba LUTRAM**
rx_block_decode_400g	17863	16327	0	0
pcs_rx_fifo	27690	12739	29,50	0
rx_block_transcode	1056	2112	0	0
descrambler_gen	286	58	0	0
rx_distribution	14933	4118	38,00	0
rx_symbol_distr	0	0	0	0
rx_deskew_400g	3923	6288	0	78880
rx_amps_lock_400g	22055	132	0	0

6 Simulace jednotky PCS

Pro ověření funkčnosti navržené a implementované jednotky PCS byla provedena simulace. K simulaci byl použit program ModelSim SE-64 10.6c. Jelikož v jednotce PCS dochází ke kódování, překódování, šifrování a distribuci, bylo pro testování této jednotky zvoleno zapojení viz Obr. 6.1. Princip tohoto zapojení spočívá v tom, že pokud pošleme validní data na vstup simulované jednotky PCS na výstupu by měla být vstupní data. Při simulaci byly použity hodinové domény PCS_CLK s frekvencí 156,25 MHz a GMII_CLK 195,3125 MHz. Byl uvažován ideální stav, kdy GMII_RX_CLK odpovídá GMII_CLK.



Obr. 6.1: Blokové schéma zapojení při simulaci PCS

Pro simulaci nejprve musela být vygenerována data, která simulovala rámce z rozhraní 400GMII. Tyto data obsahují prvních 84000 hodinových taktů Idle sekvence, což odpovídá příchodu dvěma datovým toků se zarovnávacími značkami. Za tuto dobu dojde na přijímací straně k otevření linek. Otevření nebo-li nahození linek je provedeno jakmile jsou přijaty dvě stejné zarovnávací značky na jedné PCS lince.

V další části simulace již mohou být odesílána validní data z 400GMII. Validní data obsahují Start řídící znak, následovaný datovými oktety a znakem terminate. Za těmito daty je taktéž povinná dvanácti bajtová mezera, která obsahuje Idle znaky. Tyto data jsou ve vysílací části nejprve kódována, překódována, šifrována a rozdistribuována. Výstup vysílací částí je tak vstupem přijímací části. V přijímací části je provedeno zpětné rozdistribuování, šifrování, překódování a dekódování. Výsledkem jsou na výstupu simulované jednotky PCS vstupní validní data.

7 Závěr

Cílem diplomové práce bylo navrhnout, implementovat a následně simulovat správnou funkčnost implementované jednotky PCS označovanou jako 400GBASE-R dle standardu IEEE 802.3bs-2017. Využití této jednotky PCS se uplatní ve fyzické vrstvě varianty 400GBASE-LR8, která využívá 58 Gb/s PAM4 transceivery, v sítích Ethernet pro přenosovou rychlost 400 Gb/s.

V první části byla nastudována problematika základů architektury FPGA, kterou je důležité znát pro následnou implementaci digitálního obvodu. V této části byly taktéž určeny vhodné FPGA právě pro implementaci fyzické vrstvy ve verzi 400GBASE-LR8 (jejíž součástí je PCS) a nastudována jejich základní architektura a počet základních zdrojů v FPGA. V další části byl nastudována Ethernet, kdy byla věnována pozornost především fyzické vrstvě pro 400 Gb/s Ethernet (jejíž součásti je podvrstva PCS).

Dále byl proveden návrh jednotky PCS. Navrhovaná jednotka PCS byla rozdělena na vysílací a přijímací část. Při návrhu byla zvolena šířka vstupních dat TXD 2048-bitů. Vzhledem k této šířce dat byla určena základní hodinová frekvence 195,3125 MHz. Výstupní datová šířka byla zvolena 170-bitů na každé z šestnácti PCS linek (2720-bitů) s výstupní frekvencí hodinového signálu 156,25 MHz. Tím byly splněny podmínky pro 400 Gb/s Ethernet dané specifikací. V této části práce byly taktéž popsány optimalizované návrhy jednotlivých bloků v případě nesplnění podmínek časování, které byly způsobeny dlouho kombinační cestou.

V následující části práce byla popsána volba FPGA pro implementaci navrhované jednotky PCS. Navrhovaná jednotka PCS byla následně implementována do zvoleného cílového FPGA. V této části byly získány výsledky syntézy jednotky PCS. Využití zdrojů v FPGA od firmy Xilinx s architekturou UltraScale+ s označením VU27P činí 131739 LUT (10,17 %), 76363 klopných obvodů FF (2,95 %), 135 blokovou paměť RAM (6,70 %) a 78880 LUTRAM (10,33 %). Toto využití zdrojů je přijatelné vzhledem k očekávanému využitím zdrojů 30 %. Přesto by bylo možné ještě provést optimalizaci např. u Encoderu a Decoderu, kdy po optimalizaci, která byla provedena na základě nesplnění časování, vzrostl počet využitých zdrojů téměř dvojnásobně. Na závěr byla provedena simulace navrhované jednotky PCS, kdy byla ověřena správná funkčnost implementované jednotky PCS.

V rámci další spolupráce se společností Cesnet bude provedena optimalizace pro přechod na platformu od Intelu, která bude využita na akcelerační kartu pro 400 Gb/s Ethernet. Dále bude provedena verifikace implementované jednotky a testování v hardware. Na přelomu roku je pak plánováno představení akcelerační karty pro plnohodnotný 400 Gb/s Ethernet.

Literatura

[1] PETERKA, Jiří. *Báječný svět počítačových sítí: Část XX.: Příběh Ethernetu. EArchiv.cz: Archiv článků a přednášek Jiřího Peterky* [online]. 2015, [cit. 10. 11. 2018]. Dostupné z: `http://www.earchiv.cz/b06/b1200001.php3`

[2] *IEEE Standard for Ethernet: Amendment 10: Media Access Control Parameters, Physical Layers, and Management Parameters for 200 Gb/s and 400 Gb/s Operation*. New York: The Institute of Electrical and Electronics Engineers, 2017. ISBN 978-1-5044-4451-4.

[3] DANĚK, PH.D., Ing. Martin. *Programovatelná hradlová pole - FPGA. AUTOMA: časopis pro automatizační techniku* [online]. 2006, 2006(2), [cit. 10. 11. 2018]. Dostupné z: `http://automa.cz/cz/casopis-clanky/programovatelna-hradlova-pole-fpga-2006_02_30930_672/`

[4] PECH, Ing. Jan. *Neboj se FPGA. Vyvoj.hw.cz: profesionální elektronika* [online]. Praha, 2002, 6. Únor 2002, [cit. 10. 11. 2018]. Dostupné z: `https://vyvoj.hw.cz/teorie-a-praxe/dokumentace/nebojte-se-fpga.html`

[5] *What is an FPGA? Field Programmable Gate Array. Xilinx* [online]. San José (Kalifornie), c2018, [cit. 10. 11. 2018]. Dostupné z: `https://www.xilinx.com/products/silicon-devices/fpga/what-is-an-fpga.html`

[6] TRIMBERGER, Stephen M. *Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology. Proceedings of the IEEE* [online]. 2015, 103(3), 318-331, [cit. 10. 11. 2018]. DOI: 10.1109/JPROC.2015.2392104. ISSN 0018-9219. Dostupné z: `http://ieeexplore.ieee.org/document/7086413/`

[7] *Intel Completes Acquisition of Altera. Intel Newsroom* [online]. Santa Clara (Kalifornie), 2015, 28. Prosinec 2015, [cit. 10. 11. 2018]. Dostupné z: `https://newsroom.intel.com/news-releases/intel-completes-acquisition-of-altera/`

[8] PINKER, Jiří a Martin POUPA. *Číslicové systémy a jazyk VHDL*. 1. Praha: BEN - technická literatura, 2006. ISBN 80-730-0198-5.

[9] SMITH, Gina R. *FPGAs 101: everything you need to know to get started*. 1. Boston: Newnes, c2010. ISBN 978-1-85617-706-1.

[10] PECH, Ing. Jan. *Programovatelné logické obvody – část 1. DPS* [online]. 2014, 2014(3), [cit. 10. 11. 2018]. Dostupné z: `https://www.dps-az.cz/vyvoj/id:2404/programovatelne-logicke-obvody-cast-1`

[11] *Virtex UltraScale+. Xilinx* [online]. San José (Kalifornie), c2018, [cit. 10. 11. 2018]. Dostupné z: `https://www.xilinx.com/products/silicon-devices/fpga/virtex-ultrascale-plus.html`

[12] *Intel Stratix 10 TX Advance Information Brief. Intel* [online]. Santa Clara (Kalifornie), 2018, [cit. 10. 11. 2018]. Dostupné z: `https://www.intel.com/content/www/us/en/programmable/documentation/jzw1474049428757.html`

[13] *UltraScale Architecture Configurable Logic Block: User Guide*. 1.5. San José (Kalifornie): Xilinx, 2017.

[14] *UltraScale Architecture and Product Data Sheet: Overview*. 3.5. San José (Kalifornie): Xilinx, 2018.

[15] *Virtex UltraScale+ FPGA Data Sheet: DC and AC Switching Characteristics*. 1.1. San José (Kalifornie): Xilinx, 2019.

[16] *Intel Stratix 10: INTEL STRATIX 10 TX PRODUCT TABLE*. 1. Santa Clara (Kalifornie): Intel, 2018.

[17] *Intel Stratix 10 Logic Array Blocks and Adaptive Logic Modules User Guide. Intel* [online]. Santa Clara (Kalifornie), 2018, [cit. 10. 11. 2018]. Dostupné z: `https://www.intel.com/content/www/us/en/programmable/documentation/wtw1441782332101.html`

[18] HORÁK, Jaroslav a Milan KERŠLÁGER. *Počítačové sítě pro začínající správce*. 3., aktualiz. vyd. Brno: Computer Press, 2006. Bestseller (Computer Press) ISBN 80-251-0892-9.

[19] SHINDER, Debra Littlejohn. *Počítačové sítě: nepostradatelná prírucka k pochopení sítové teorie, implementace a vnitrních funkcí [sic]*. 1. Praha: SoftPress, c2003. Cisco systems. ISBN 80-864-9755-0.

[20] PETERKA, Jiří. *Báječný svět počítačových sítí: Část XXII.: Gigabitový a ještě rychlejší Ethernet. EArchiv.cz: Archiv článků a přednášek Jiřího Peterky* [online]. 2015, [cit. 10. 11. 2018]. Dostupné z: `http://www.earchiv.cz/b07/b0200002.php3`

[21] *IEEE Standard for Ethernet*. New York: The Institute of Electrical and Electronics Engineers, 2015. ISBN ISBN 978-1-5044-0079-4.

[22] ZEZULKA, František a Ondřej HYNČICA. *Průmyslový Ethernet II: Referenční model ISO/OSI. AUTOMA: časopis pro automatizační techniku* [online]. 2007, 2007(3), [cit. 10. 11. 2018]. Dostupné z: http://automa.cz/cz/casopis-clanky/prumyslovy-ethernet-ii-referencni-model-iso/osi-2007_03_34209_3890/

[23] PETERKA, Jiří. *Báječný svět počítačových sítí, část III. - Síťové architektury. EArchiv.cz: Archiv článků a přednášek Jiřího Peterky* [online]. 2015, [cit. 10. 11. 2018]. Dostupné z: http://www.earchiv.cz/b05/b0500001.php3

[24] PETERKA, Jiří. *Co je čím ... v počítačových sítích: Linková vrstva - III. EArchiv.cz: Archiv článků a přednášek Jiřího Peterky* [online]. 2015, [cit. 10. 11. 2018]. Dostupné z: https://www.earchiv.cz/a92/a220c110.php3

[25] *Optico: Rozdíl mezi multimode a Single Mode vlákno (Patch Cord) Identifikace jediného režimu a multi-mode Fiber. Optico* [online]. [cit. 10. 11. 2018]. Dostupné z: http://cz.fttxsolution.com/news/the-difference-between-multimode-and-single-mo-9277587.html

[26] *FINISAR: Optical Transceivers 400GBASE-FR8 QSFP-DD Optical Transceiver. FINISAR* [online]. Sunnyvale (Kalifornie), 2019, [cit. 10. 11. 2018]. Dostupné z: https://www.finisar.com/optical-transceivers/ftcd1333e1pcl

[27] *Optické Transceivery a síťové technologie. RLC* [online]. Praha, [cit. 10. 11. 2018]. Dostupné z: Dostupnéz:https://rlc.cz/wp-content/uploads/2017/09/Transceivery_technologie_v3.pdf

Seznam symbolů, veličin a zkratek

FPGA	Field Programmable Gate Array
PLD	Programmable Logic Device
ASIC	Aplication Specific Integrated Circuit
HDL	Hardware Description Language
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
RTL	Register Transfer Logic
LUT	Look-Up Table
CLB	Configuratable Logic Block
RAM	Random Access Memory
ALM	Adaptive Logic Modules
LAB	Logic Array Blocks
ESD	ElectroStatic Discharge
DDR	Double Data Rate
DDR	Serial Data Rate
LVCMOS	Low Voltage Complementary Metal Oxide Semiconductor
LVTTL	Low Voltage Transistor-Transistor Logic
HSTL	High Speed Transceiver Logic
IOB	Input Output Block
IOE	Input Output Element
ROM	Read Only Memory
FIFO	First In First Out
DSP	Digital Signal Processing
PLL	Phase Locked Loop
PCI-E	Peripheral Component Interconnect - Express
XAUI	Xtended Attachment Unit Interface
FIR	Finite Impulse Response
IIR	Infinite Impulse Response
FFT	Fast Fourier Transform
FinFET	Fin Field Effect Transistor
HP	High Performance
MLAB	Memory Logic Array Blocks
ISO	International Organization for Standardization
OSI	Open Systems Interconnection
IEEE	Institute of Electrical and Electronics Engineers
CSMA	Carrier Sense Multiple Access
CD	Collision Detection

DEC	Digital Equipment Corporation
LAN	Local Area Network
MAN	Metropolitan Area Network
WAN	Wide Area Network
PCS	Physical Coding Sublayer
MII	Media Independent Interface
LCC	Link Layer Control
MAC	Media Access Control
CRC	Cyklický redundantní součet
PMA	Physical Medium Attachment
FEC	Forward Error Correction
PHY	Physical Layer Device
PMD	Physical Medium Dependent
PRBS	Pseudo Random Binary Sequencer
NRZ	Not Return To Zero
PAM	Pulse-Amplitude Modulation

A Obsah přiloženého CD

V této kapitole je popsán obsah přiloženého CD. Obsah CD tvoří především zdrojové kódy v jazyce VHDL implementovaných bloků jednotky PCS. Součástí CD jsou i externí zdrojové kódy a skripty firmy Cesnet, které byly použity v práci. V kořenové adresáři je taktéž elektronická verze diplomové práce.