VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

# BAKALÁŘSKÁ PRÁCE

Brno, 2021 Vladislav Válek


# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

## ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS


# IMPLEMENTACE AUTO-NEGOCIACE PRO ETHERNETOVÉ ROZHRANÍ O RYCHLOSTECH 25-100 GB/S

AUTONEGOTIATION IMPLEMENTATION FOR 25 - 100 GBPS ETHERNET INTERFACES

**BAKALÁŘSKÁ PRÁCE**
BACHELOR'S THESIS


**AUTOR PRÁCE** — **Vladislav Válek**
AUTHOR

**VEDOUCÍ PRÁCE** — **Ing. Adrián Tomašov**
SUPERVISOR


**BRNO 2021**

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

# Bakalářská práce

bakalářský studijní program **Telekomunikační a informační systémy**

Ústav telekomunikací

***Student:*** Vladislav Válek ***ID:*** 203368
***Ročník:*** 3 ***Akademický rok:*** 2020/21

**NÁZEV TÉMATU:**

## Implementace auto-negociace pro Ethernetové rozhraní o rychlostech 25-100 Gb/s

**POKYNY PRO VYPRACOVÁNÍ:**

Cílem práce je implementovat protokol auto-negociace se síťovou kartou o rychlosti 25/100 Gpbs na FPGA kartě s čipem Xilinx Ultrascale+. Student se v práci seznámí s FPGA čipy Xilinx Ulstrascale+ a protokolem IEEE 802.3 (Ethernet) se zaměřením na auto-negociaci pro rychlost 25/100 Gpbs. Praktickým výstupem práce je návrh a implementace architektury ve VHDL pro negociační vrstvu zmíněných standardů. Správnou funkčnost student ověří pomocí simulace. Následně student implementuje negociační vrstvu na FPGA čipu a ověří funkčnost a kompatibilitu s jinými zařízeními. Na závěr práce, student zhodnotí dosažené výsledky získané srovnáním simulace a testováním na reálné kartě.

**DOPORUČENÁ LITERATURA:**

[1] GARDNER, Steven Lyle; HISCOCK, James Scott; YUEN, Michael. Advanced ethernet auto negotiation. U.S. Patent No 6,580,697, 2003.

[2] SHAFER, Jeffrey; RIXNER, Scott. A reconfigurable and programmable gigabit ethernet network interface card. Network, 2006, 1.2: 3.

***Termín zadání:*** 1.2.2021 ***Termín odevzdání:*** 31.5.2021

***Vedoucí práce:*** Ing. Adrián Tomašov
***Konzultant:*** Ing. Štěpán Friedl

**prof. Ing. Jiří Mišurec, CSc.**
předseda rady studijního programu

## ABSTRAKT

Bakalářská práce se zabývá návrhem auto-negociační komponenty pro síťové karty řízené hradlovými poli. Funkce auto-negociace slouží pro dohodu parametrů síťového provozu mezi dvěma stranami síťového kanálu. Mezi dohodnutelné parametry patří komunikační rychlost, schopnost pozastavení provozu a schopnost provozovat Forward Error Correction (FEC). V úvodu jsou představeny vnitřní bloky architektury UltraScale+ společnosti Xilinx, kde je nejvíce pozornosti věnováno blokům GTY. Dále jsou představeny zásady klauzule 73, standardu IEEE 802.3-2018, jež popisuje mechanismus funkce auto-negociace. Návrh je proveden v jazyce VHDL pro linkovou komunikační rychlost 25 Gb/s a zahrnuje popis postupů a případných změn, které je nutno provést při implementaci zmíněné funkce na hradlových polích využívající vysokorychlostní transceivery. Následně byla funkčnost zapojení ověřena v simulacích, jejichž výsledky jsou poskytnuty rovněž v této práci. Na závěr bylo provedeno testování auto-negociační funkce obsluhované zde vytvořenou komponentou, pročež byla využita síťová karta řízená hradlovým polem Virtex 7 UltraScale+. Při testování bylo využito zavedení sond Integrated Logic Analyzer (ILA) do struktry návrhu. Dosažené výsledky testování, věnující se jak průběhu auto-negociace, tak procesům ve fyzické vrstvě, jsou zde náležitě okomentovány.



## KLÍČOVÁ SLOVA

auto-negociace, Ethernet, FPGA, GTY, PCS, transceiver, UltraScale+, VHDL, Virtex, Xilinx, IEEE 802.3



## ABSTRACT

This bachelor's thesis addresses the design of the auto-negotiation component for network interface cards controlled by FPGAs. Auto-negotiation function allows to advertise the available communication parameters, like the link speed, the transmission pause ability or Forward Error Correction (FEC) ability, by either side of the common link and determine the common abilities, which will be used to establish a connection. In the beginning, the internal parts of Xilinx UltraScale+ FPGA family are introduced with greater emphasis on the description of GTY transceivers. In the next chapter are introduced the mechanisms of auto-negotiation function as described in clause 73 of the IEEE 802.3-2018 standard. The design here is created for Ethernet interfaces running at speed 25 Gbps and is written in VHDL language. The next chapter describes the necessary steps which are required for the implementation on the FPGAs, where high-speed transceivers are in use. Function of the created design was then checked within a simulation and the correspondent results are also provided in this thesis. In the end, the testing of the designed auto-negotiation component took place for which the network card with Vitex 7 UltraScale+ FPGA was used. The testing process includes the use of the Integrated Logic Analyzer (ILA) which was inserted into final design. The achieved results from testing of both, the auto-negotiation process and surrounding physical layer processes, are described here with proper commentary.



## KEYWORDS

auto-negotiation, Ethernet, FPGA, GTY, PCS, transceiver, UltraScale+, VHDL, Virtex, Xilinx, IEEE 802.3

VÁLEK, Vladislav. *Implementace auto-negociace pro Ethernetové rozhraní o rychlostech 25–100 Gb/s*. Brno, 2021, 96 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Adrián Tomašov

Vysázeno pomocí balíčku `thesis` verze 4.00; `http://latex.feec.vutbr.cz`

# PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Implementace auto-negociace pro Ethernetové rozhraní o rychlostech 25–100 Gb/s“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/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

Chci poděkovat svému vedoucímu ve společnosti CESNET, panu Ing. Štěpánu Freidlovi za trvalou asistenci při vývoji a četné faktické poznámky k bakalářské práci. Rovněž chci poděkovat svému vedoucímu na Ústavu komunikací, panu Ing. Adriánu Tomašovi za cenné připomínky ke struktuře bakalářské práce.

# Obsah

# Seznam obrázků

# Seznam tabulek

# Úvod

Jak míří vývoj technologií kupředu, systémy používané v těch minulých mají tendenci být potřebně vylepšeny a zařazeny jako nové. Případem jedné takové je auto-negociace. Tato funkce je již dlouhou dobu k dispozici na běžných síťových kartách používaných v domácnostech a v sítích malého rozsahu, tj. rychlostech do 1Gb/s. Slouží k vzájemné dohodě parametrů mezi dvěma stranami přenosového kanálu. V základu umožňuje dohodu rychlosti linky a schopnosti *Forward Error Correction* (FEC). Tímto velmi přispívá k funkci kompatibility s ostatními zařízeními. Samotná funkce pracuje na fyzické vrstvě jako nejnižší v hierarchii jejích podvrstev.

Tato práce se zabývá návrhem auto-negociační funkce na síťových kartách s hradlovými poli. Vývoj probíhá v rámci výzkumu Akcelerovaných síťových technologií při projektu Liberouter. Místní testovací stroje obsahují karty s hradlovými poli výrobců Xilinx (bude předvedeno v rámci práce) a Intel, na kterých probíhá vývoj síťového hardware, pro zvláště optické sítě, pracující na rychlostech 10–400 Gb/s. V rámci potřebného teoretického základu, pro pochopení funkce auto-negociace a principu hradlových polí, je práce členěna do několika kapitol.

V první kapitole bude krátce představena historie společnosti Xilinx, její začátky a kroky, které vedly k vybudování největšího výrobce hradlových polí vůbec. Ve stručnosti jsou zde zmíněny názvy řad hradlových polí, které tvořily technologickou linii firmy, společně s občasnými ekonomickými údaji.

Ve druhé kapitole bude kompletně představena struktura hradlových polí společnosti Xilinx, konkrétně řady UltraScale+. Ta, jakožto nejvýkonnější z nabízených čipů firmy, představují vhodnou volbu na vývoj zařízení pro zpracování síťového provozu na vysokých rychlostech. Budou představeny základní funkční bloky doplněné o některé složitější. Zvláštní pozornost bude potom věnována strukturám pro vysokorychlostní sériovou komunikaci.

Ve třetí kapitole bude rozebrán standard IEEE 802.3, zejména klauzule 73, která se specializuje na auto-negociaci pro síťové rychlosti 25–100 Gb/s. Veškeré náležitosti standardu jsou zde detailně rozebrány a okomentovány.

Ve čtvrté kapitole bude rozebrán konkrétní návrh auto-negociační komponenty v jazyce VHDL. Důraz je kladen zvláště na, co možná, nejúspornější funkci celého zapojení. Celek je složen z vysílače, přijímače, jakožto rozhraní se zbytkem fyzické vrstvy, a komponenty arbitráže, která slouží k řízení celého procesu. Navíc jsou zde rozebrány změny, které si implementace v hradlových polích vyžaduje.

V páté kapitole bude následně provedena simulace významných částí celého procesu dohody linkových parametrů. Simulace je provedena s pomocí simulačního prostředí Modelsim, odkud jsou poskytnuty ukázky funkcí důležitých mechanismů auto-

negociace. Pozornost je věnována zvláště splnění funkcionality popsané ve standardu 802.3.

Na závěr je komponenta zapojena do struktury fyzické vrstvy a její funkcionalita otestována ve spolupráci s okolními součástmi. Pro sledování vnitřních signálů jsou využity sondy Integrated Logic Analyzer zapojené ve dvou testovacích komponentách, jejichž skupiny sledovaných signálů umožňují zachytit několik scénářů. Většina scénářů představuje výskyt neobvyklých událostí během standardního průběhu auto-negociačního procesu a jejich popisu je věnována značná část dané kapitoly. Dosažené výsledky testování jsou poté srovnány s výsledky pozorovanými v simulacích.

# 1 Historie společnosti Xilinx

Zákaznická pozornost byla v první polovině osmdesátých let soustředěna převážně na výrobu specializovaných obvodů s předem danou funkcí, nazvané *Application Specific Integration Circuit* (ASIC). Tyto obvody, zpočátku s velmi obecnou funkcí, byly vyráběny ve velmi početných sériích s nízkou cenou za kus, které svým výrobcům přinášely značné profity. Nedlouho poté však byly přijímány stále četnější požadavky zákazníků po vyšší specializaci těchto obvodů, což vedlo ke zmenšení objemů výroby jednotlivých čipů a tedy ke snížení profitu z jejich prodejů. Druhý problém spočíval v možném výskytu chyby při vývoji, který často vedl k zahození všech doposud vyrobených součástí a nárůstu čekací doby, pro opravu návrhu a výrobu nových čipů, na straně zákazníka.

V roce 1984 byl vytvořen koncept integrovaného obvodu, který by bylo lze vyrobit jako „čistou pásku“ a jehož funkcionalita by byla dodatečně doprogramována. Nápad byl vytvořen Rossem Freemanem, konstruktérem čipů ve společnosti Zilog, která se v té době zabývala návrhem integrovaných obvodů a *solid-state* zařízení. Zavedením tohoto konceptu lze zároveň docílit významného snížení pravděpodobnosti vzniku chyb při vývoji zařízení využívajících logické obvody.

Pro tehdejší ředitele společnosti Exxon, jehož byl Zilog dceřinou společností, však nebyl nápad dostatečně přesvědčivý, aby do trhu investovali (jeho tehdejší hodnota byla odhadována na 100 milionů dolarů). Freeman, vědom si významnosti svého nápadu, se tedy rozhodl opustit Zilog a, společně s Bernardem Vonderschmittem, započal práci na návrhu prvního hradlového pole (Field Programmable Gate Array, FPGA)[1]. Koncept byl zpočátku velmi složitý, protože vyžadoval velké množství tranzistorů, které byly v té době považovány za vzácné (na základě slov jednoho ze zaměstnanců Xilinx)[2]. Oba následně založili v roce 1984 společnost Xilinx a o rok později započal prodej prvních výrobků firmy, hradlových polí rodiny XC2000 s vnitřní strukturou *Logic Cell Array*[3]. Tímto byl odstartován pohyb trhu s programovatelnými logickými obvody a prodej FPGA XC2000 přinesl konstantní nárůst na 8 let dopředu. V roce 1987 byla do nabídky Xilinx přidána nová generace hradlových polí s počtem hradel čítajícím 9000, čímž mohly programovatelné obvody soupeřit s většinou pokročilých neprogramovatelných obvodů té doby. Konkurenční nevýhodou však stále zůstávala vysoká cena hradlových polí.

Na konci osmdesátých let uzavřel Xilinx partnerství se společností Monolithic Memories Inc. (MMI), kde pozbyl svá patentová práva výměnou za finanční podporu plynoucí z partnerství, což pomohlo firmě udržet svůj postup ve výzkumu a vývoji. Nedlouho poté byla však MMI odkoupena společností Advanced Micro Devices (AMD), což přineslo do rozvoje Xilinx komplikace zejména z toho důvodu, že AMD bylo jedním z jeho hlavních soupeřů. V roce 1989 přesvědčil Vonderschmitt

AMD, aby koupilo 20 % společnosti, která si tímto udržela financování. Zároveň, ve stejném roce, vstoupil Xilinx na burzu.

Na počátku devadesátých let ovládal Xilinx pouze 65 % trhu s programovatelnými obvody a v polovině devadesátých let byl vliv společnosti již stoprocentní. V roce 1993 činil obrat společnosti více jak 250 milionů dolarů. Ostatní společnosti však dominantní pozici rychle nabouraly výrobou svých vlastních konkurenčních produktů. Mezi nimi například Actel s podílem trhu 18% nebo Altera Corporation, která poskytovala výrobky podobné hradlovým polím. Na konci devadesátých let se prodeje Xilinx rozšířily přes celé Spojené státy a 30 % bylo směřováno za hranice. Polovina zisku plynula z nové série hradlových polí XC3000. Zákaznickou základnu tvořilo více jak 3500 společností, mezi nimi například Apple, IBM, Sun Microsystems, Hewlet-Packard Co., Fujitsu nebo Northern Telecom. Společnost se zanedlouho přestěhovala do své nové továrny v San Jose v Kalifornii. V polovině devadesátých let byla rovněž představena řada čipů XC5000, jejichž úkolem bylo uspokojit poptávku trhu po levných hradlových polích, které mohly konkurovat tehdejším čipům ASIC. Společnost dále představila hradlová pole řady XC3100L a XC4000L sloužící pro implementaci v aplikacích vyžadujících nízkou spotřebu energie, po nichž byla poptávka na konci devadesátých let na vzestupu.

V roce 1995 narostly obraty společnosti na 550 milionů dolarů. Ve firmě bylo toho času zaměstnáno přes 1000 zaměstnanců pracujících na pobočkách v Severní Americe, Evropě a Asii. V roce 1996 Bernard Vonderschmitt sestoupil z pozice výkonného ředitele firmy, avšak zůstal členem jejího představenstva až do své smrti v roce 2004. V roce 1998 představil Xilinx novou řadu hradlových polí zvanou Virtex, která dodnes zastává pozici nejvýkonnějších čipů značky. Xilinx a Altera (v tomto období mírně větší než Xilinx) vlastnily přes 30 % trhu s programovatelnými obvody.

V roce 2000 uzavřela společnost s obratem přes jednu miliardu dolarů. I přes velký obrat byly však příjmy firmy rapidně sníženy vzhledem ke zpomalení trhu, čímž zůstalo společnosti mnoho neprodaného zboží. Stejného obratu dosáhla rovněž v roce 2002, ovšem se ztrátami 133 milionů dolarů. Společnost však očekávala nový růst trhu, kde došlo k vytvoření nového odvětví, které zahrnovalo využití programovatelných logických obvodů. Výroba nových elektronických zařízení spatřovala výhodu v prototypování součástek pomocí hradlových polí před jejich výrobou jakožto ASIC. Pro mobilní zařízení začal Xilinx poskytovat čipy řady CoolRunner, jejichž nízká energetická náročnost a malé rozměry je činily vhodné pro tyto aplikace. Další snížení ceny programovatelných obvodů vedlo k jejich výrobě ve vysokých objemech. Konstrukce hradlových polí přišla rovněž vhodná při paralelním zpracování dat, kde často zvyšovaly výkon systémů jinak poháněných standardními procesory. Výroba čipů Virtex převzal v roce 2002 IBM a po vypršení dohody v roce 2004 byla výroba přenesena do společnosti Toshiba Corporation. V roce 2006 činil podíl Xilinx na

trhu s programovatelnými obvody 50 % s obratem 1,7 miliardy dolarů[1].

Největším konkurentem Xilinx zůstává do dnešních dnů společnost Altera, která se v roce 2015 stala součásti firmy Intel[4]. Společnost dnes zaměstnává přes 3200 zaměstnanců a v současnosti nabízí modely ve všech cenových kategoriích a výkonových třídách, základní řady tvoří čipy Spartan, Artix, Kintex, Virtex a Zynq (viz dále). Kromě FPGA dnes společnost vyrábí i obvody *Adaptive Computation Acceleration Platform* (ACAP) spojující programovatelnost hradlových polí a výhod strojového učení poskytujícího adaptabilní funkci celého zařízení. V nabídce je rovněž možno nalézt i hotová řešení pro využití v datacentrech nebo v telekomunikacích[5].

# 2 Architektura UltraScale+

Současný vývoj vnáší do světa FPGA požadavek, aby se spojila výhoda rychlosti ASIC s možností změnit konfiguraci čipu, jako mají hradlová pole. Dnešní FPGA již dosahují pracovních frekvencí až 500MHz srovnatelných s ASIC. Jejich návrh je tedy možno provádět na hradlových polích a tím zásadně zredukovat pravděpodobnost vzniku chyb, jež by prodražovaly konečnou výrobu[6]. Firma Xilinx proto navrhla architekturu UltraScale+, aby na toto reagovala.

Tato architektura s nejmenší 16nm technologií byla navržena, aby splnila nejnáročnější požadavky na výpočetní výkon a to zejména vysokou propustnost. Dosahuje toho zejména díky většímu počtu vnitřních součástí oproti nižším řadám (například UltraScale nebo Spartan) a přidáním některých speciálních bloků, příkladem paměti s vysokou propustností nebo zvýšené množství bloků pro sériovou komunikaci. Hlavní uplatnění tímto nachází třeba v hardwarové akceleraci, 5G sítích, radiofrekvenčních aplikacích, vysokorychlostní komunikaci pomocí kabelových spojení a v neposlední řadě i různých typech testování a měření[7].

## 2.1 Vnitřní struktura

Stěžejí součástí hradlového pole je vnitřní propojovací síť, která na jedné straně zajišťuje vzájemné propojování jednotlivých bloků, na straně druhé potom napojení na externí piny umístěné na pouzdře čipu. Potíže však činí to, že technologické možnosti vnitřních propojů rostou rychleji, než ty používané k vyrábění součástkových pouzder. Nabízí se možnost vytvářet tedy hradlová pole do maximálního počtu vývodů, který technologie dovolí (nejvyužívanější pouzdro v této kategorii je typu Ball Grid Array). Vyššího výkonu by následně mohlo být dosaženo zapojením většího počtu navzájem propojených čipů na desce plošného spoje. Toto ale přináší řadu nevýhod.

Prvním problémem je potenciální vysoká spotřeba výstupních pinů při použití velmi širokých paralelních sběrnic (např. 1024 bitů v systémech SoC), což dosti limituje připojení ostatních komponent, jako externích sběrnic nebo optických modulů s velkým počtem propojů, uvážíme-li ještě piny pro připojení referenčních hodinových signálů, napájecí piny, zemní piny nebo třeba konfigurační rozhraní.

Druhý problém tvoří přidané propoje s nezanedbatelnou indukčností, která způsobuje zpomalení signálu, čímž dochází k omezení maximální využitelné frekvence shora. Ke zlepšení nedojde ani při použití metody časového multiplexu, čímž by se sice dal zdánlivě navýšit počet I/O pinů, ale zpoždění by tímto ještě narostlo.

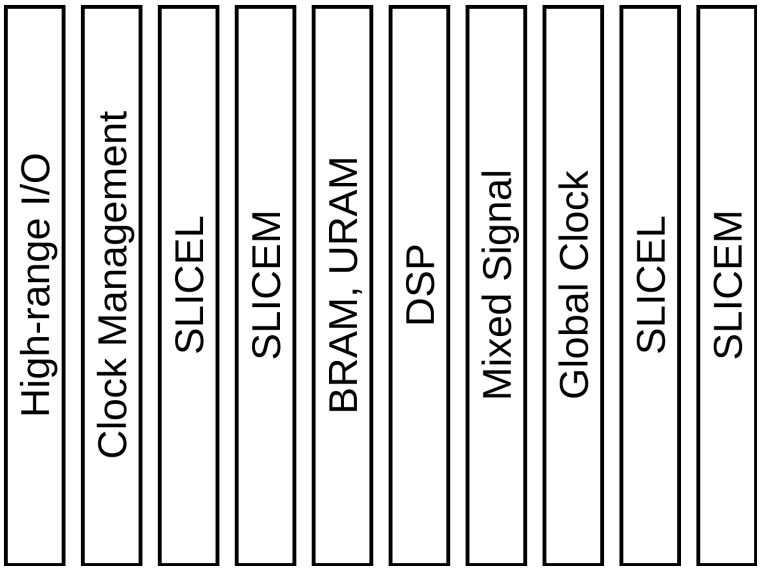
Třetí problém představuje nutnost napájet více zařízení současně společně s budiči sběrnic vedenými mezi nimi.

Poslední problém souvisí s přenosem užitečné informace. Při případné aplikaci v telekomunikacích je nutné zajistit potřebnou integritu dat, aby nedocházelo k jejich ztrátě nebo, aby se při cestě mezi čipy nepoškodila (vlivem rušení, špatné synchronizace).

Řešení problémů s propoji si proto vyžádalo vytvoření nové technologie spojování více hradlových polí dohromady[8].

### 2.1.1 Stacked Silicon Interconnect

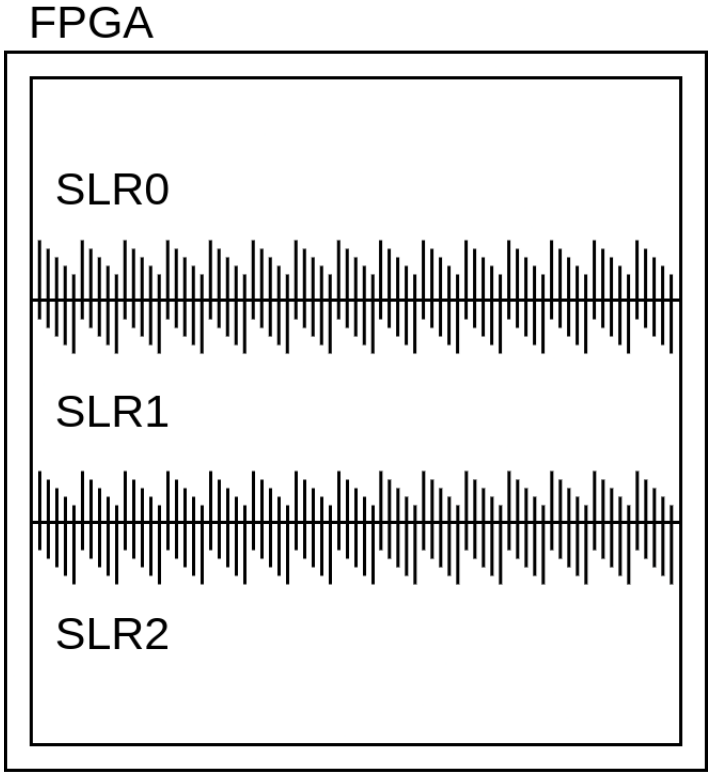
Jedná se o patentovanou technologii firmy Xilinx. Na čipu je umístěno více de facto „vnitřních FPGA“. Jsou pečlivě navržena pro co největší univerzálnost a zároveň velkou efektivitu při vložené konfiguraci. Mezi různými typy hradlových polí si však lze vybrat potřebnou kombinaci, jenž se více hodí pro určitou zákaznickou aplikaci. Architektura takového návrhu se v terminologii Xilinx označuje jako *Advanced Silicon Modular Block* (ASMBL)[9], vnitřní hradlová pole se potom označují jako *Super-logic regions* (SLRs). Každý z nich již obsahuje základní funkční celky typické pro FPGA, jako konfigurační logické bloky, digitální signálové procesory nebo paměti. Příklady různých uspořádání SLR podle ASMBL lze nalézt na obrázcích 2.1 a 2.5.




Obr. 2.1: Příklad sloupcové struktury podle ASMBL

Význam architektury potom spočívá ve speciálním řešení, jak jsou menší hradlová pole propojována. Samotné SLR jsou umístěny na křemíkové destičce (interposer), skrze kterou jsou provedeny mikropropoje, kterými jsou spojeny. Jejich použití je velmi výhodné, jelikož spoje jsou velmi krátké, nevnáší do systému rušení a navíc

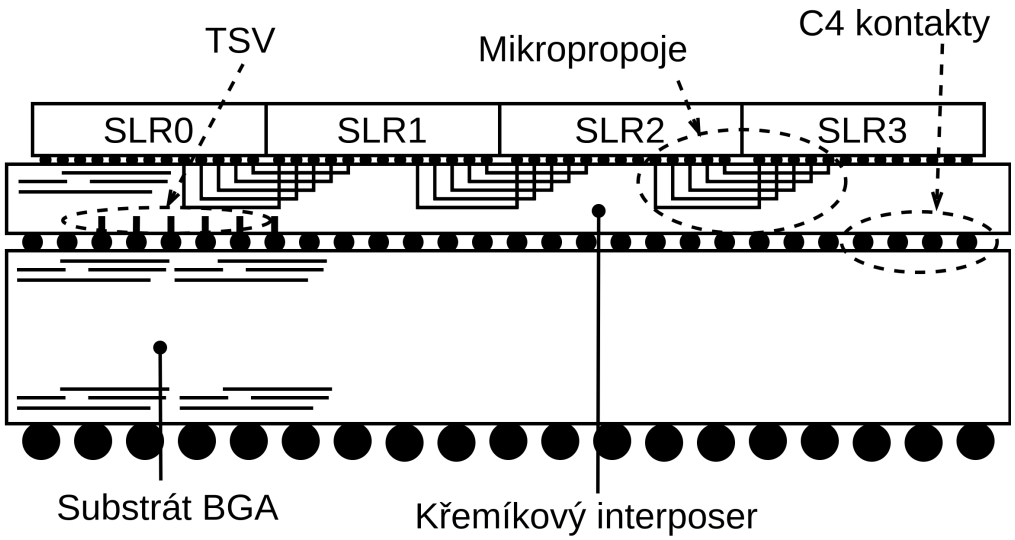
mají minimální zpoždění signálu, což umožňuje přenosové rychlosti až v řádech Tb/s. Propoje tvoří souvislou řadu na rozhraní regionů, schéma tohoto konceptu je znázorněno na obrázku 2.2.




Obr. 2.2: Ukázka mikropropojů mezi SLR (pohled shora)

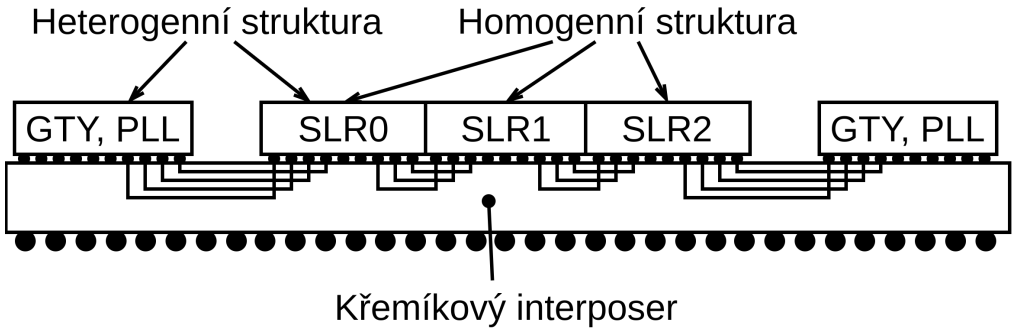
Do křemíkové destičky jsou navíc zavedeny prokovy, označované jako *Through Silicon Via* (TSV), jenž jsou vyvedeny v C4 kontakty, pomocí kterých je křemíková destička připájena již na samotný substrát tvořící BGA pouzdro. Jeho úkolem je propojit kontakty křemíkové destičky s kontakty na spodní straně pouzdra, které již slouží k osazování desek plošných spojů pomocí standardní technologie. Boční pohled na sestavený celek je ukázán na obrázku 2.3.

Prvotní návrhy spočívaly ve vytváření homogenních struktur, kdy jednotlivé bloky SLR společně s některými dalšími byly na vnitřní destičce uloženy těsně k sobě. To však vytváří problém, kdy při použití některých rušivých součástí (například bloků pro vysokorychlostní sériovou komunikaci) proniká rušení i do citlivých součástí čipu. Moderní výroba hradlových polí a i čipů obecně předpokládá vytváření heterogenních struktur, kdy určité části jsou účelově umístěny (oddáleny) oproti hlavním součástem obvodu, aby se tak zamezilo přenášení rušení na vysokých kmitočtech. Rovněž je možno umístit na společný substrát čipy vyráběné pomocí




Obr. 2.3: Schematické znázornění struktury SII

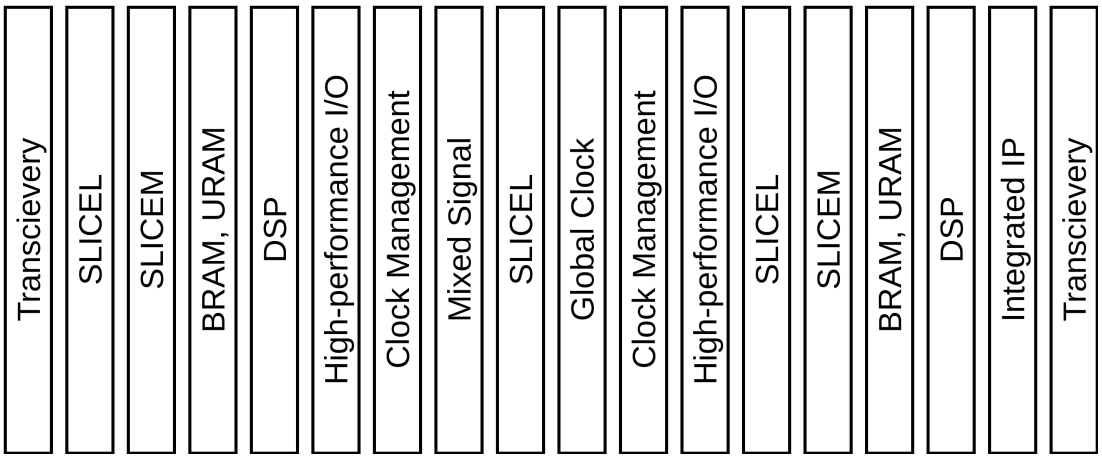
různých technologií. Tato možnost je druhou výhodnou technologie SII[8]. Zjednodušený případ heterogenní struktury je znázorněn na obrázku 2.4.




Obr. 2.4: Schéma heterogenního uspořádání čipu

## 2.2 Funkční bloky

Nyní se podíváme na základní funkční bloky, které tvoří strukturu FPGA a ke kterým je možno přistupovat skrze návrhové nástroje ať už přímo nebo nepřímo. Skupiny funkčních bloků jsou v každém SLR uspořádány do sloupců, kdy každý sloupec je horizontálně dělen na hodinové regiony[7]. Tato struktura je znázorněna na obrázku 2.5.




Obr. 2.5: Příklad sloupcové struktury SLR

### 2.2.1 Konfigurovatelný logický blok

Základní součástí čipů UltraScale+ a hradlových polí vůbec jsou *Configurable Logic Blocks* (CLBs), které jsou na čipu obsažené v největším počtu. Centrum každého bloku je v řadě UltraScale+ tvořeno osmi LUT (Look-up-Table) a šestnácti D klopnými obvody (synchronní), jež lze překonfigurovat na latch (asynchronní). LUT funguje jako generátor logické funkce, či paměť ROM, která se naprogramuje podle požadavků návrháře při konfiguraci. Lze ji zapojit jako šestivstupou s jedním výstupem nebo jako dvě pětivstupé s dvěma separátními výstupy a společnými vstupy. Výstup každé LUT může být opatřen právě jedním klopným obvodem/latchem, kterýmžto vstupům lze rovněž směřovat volitelně i přímé vstupy do CLB. K tomu je v bloku navíc obsažena CARRY logika pro přenos bitu při aritmetických operacích a multiplexor, což umožňuje vytvářet složitější funkce mezi LUT navzájem. Tato sada logických obvodů je v hradlovém poli reprezentována jako SLICEL. Kromě nich existují ještě bloky typu SLICEM. V těch lze kromě funkce LUT zvolit buď funkci RAM nebo posuvného registru. V řadě UltraScale je v CLB obsažen právě jeden blok typu SLICEL nebo SLICEM. Oproti ostatním řadám hradlových polí Xilinx zde také dochází k vylepšení práce s CLB přidáním dalších kontrolních signálů a zavedením hustší propojovací sítě mezi jednotlivými konfigurovatelnými bloky, což umožňuje vyšší flexibilitu při optimalizaci návrhu zapojení[10].

### 2.2.2 Vnitřní paměti

Základní typ pamětí vyskytujících se v hradlových polích jsou *Block RAM* (BRAM). Jedná se o synchronní dvouportové paměti jednotkové velikosti 36kb. Blok disponuje bohatými možnostmi konfigurace. První velkou je *True dual-port* (TDP), kdy

čtecí/zápisový port disponuje šířkou až 32 bitů s volitelnou (až čtyřbitovou) paritou, pro některé šířky umožňující použití Hammingova kódu. Porty jsou na sobě zcela nezávislé, mají oddělené hodiny, adresové vstupy, datové vstupy/výstupy, ale sdílejí společně uložená data. Druhým režimem je *Simple dual-port* (SDP), kdy jsou porty spojeny a celková šířka uložených dat může mít až dvojnásobnou velikost oproti módu TDP. Je tedy možno v bloku ukládat data o šířce až 72 bitů s volitelnou osmibitovou paritou. Dále lze specifikovat chování při současném čtení a zápisu do paměti. Vstupy jsou opatřené registry a volitelně lze opatřit registrem i výstup, což zvyšuje možnou pracovní frekvenci, ale za cenu nárůstu zpoždění. Paměť lze rozdělit i na dvě oddělené o velikosti 18kb. Zároveň lze spojovat více bloků do sebe k vytvoření větších paměťových polí. BRAM lze také nakonfigurovat jako frontu *First-in/first-out* (FIFO) a to jak pro synchronní, tak asynchronní operaci. K dispozici je i vestavěný kontrolér řízení a v případě asynchronní varianty je zajištěna i interní synchronizace mezi hodinovým signálem řídícím zápis a hodinovým signálem řídícím čtení.

Druhý typ paměti je specifický pro řadu UltraScale+, podle níž dostala název *Ultra RAM* (URAM). Jedná se o synchronní dvouportové paměti s vysokou propustností. Na rozdíl od blokových RAM mají však oba porty společné hodiny a datová šířka je pevně nastavena na 72 bitů (64 data + 8 bitů volitelná parita). Dalším rozdílem je zvýšený počet registrů, které mohou být předřazeny vstupům nebo zařazeny výstupům (je možno takto zapojit až čtyři registry pro zabezpečení funkce na vysokých kmitočtech). Tyto paměti disponují vyšší kapacitou oproti předchozím, jednotkové bloky mají velikost 288kb a lze je kaskádně spojovat až do velikosti 100Mb. URAM jsou v každém hodinovém regionu uspořádány do typické sloupcové struktury, kdy každá má svůj dodatečný propojovací systém umožňující spojovat paměti bez nutnosti využití běžných propojovacích cest. Takto lze propojovat i sloupce v regionech umístěných pod sebou i sloupce mezi sebou umístěné napříč regiony[11].

Třetí a poslední typ jsou paměti *High Bandwidth Memory* (HBM). Čipy obsahující tyto bloky jsou vrcholovými modely řady Virtex (viz dále). Jelikož se jedná o paměti typu DRAM, mají oproti statickým pamětem větší kapacitu a menší nároky na spotřebu. Paměti lze nalézt ve variantách 4 nebo 8GB, v počtu jedna až dvě na jednom čipu. Zde je nejvíce výhodné propojení přímo na křemíkové destičce pomocí SII, čímž mají spoje i velmi malou indukčnost. Na čipu se nachází i speciální AXI rozhraní s mikrokontrolérem pro komunikaci s paměťmi. Jeho účel je zejména ve zvyšování efektivity paměťové sběrnice přeuspořádáváním čtecích a zápisových požadavků[7].

### 2.2.3 Komunikační rozhraní

Hradlová pole UltraScale+ disponují celou řadou komunikačních rozhraní a sběrnic. Kromě předešle zmíněných interních pamětí jsou FPGA vybavena také bloky pro komunikaci s externími paměťmi. Příkladem mohou být DDR3/4 paměti, QDRII+ nebo paměti RLDRAM3.

Nejvýznamnějším rozhraním je PCI Express. Na čipu lze nalézt dva bloky, které zaručují kompatibilitu jak s verzí 3, tak i 4. Podporovaný počet kanálů v případě první je 16 při rychlosti 8GT/s. Obvody zajišťující takovou komunikaci nesou v logice hradlového pole název PCIE4. Ve verzi 4 je potom podporováno 8 kanálů při rychlosti 16GT/s. Komponenta pro takové rozhraní nese jméno PCIE4C. Konfiguraci bloků PCI Express lze provést přednostně před konfigurací zbytku zařízení, vzhledem k požadavku nutného sestavení spojení do 100 ms, dle standardu *PCI Express Base Specification*[12].

Další rozhraní není komunikačním rozhraním v pravém slova smyslu. Již z podstaty hradlových polí totiž plyne, že je třeba umožnit nějakým způsobem nahrát konfiguraci pro jeho korektní činnost. Posledním z externích rozhraní, které si rozebereme je konfigurační.

Samotný konfigurační soubor je v terminologii FPGA označován jako *bitstream* a za běhu umístěn ve specializovaném úložišti typu *CMOS Configuration Latch* (CCL). Jedná se o volatilní paměti, které jsou však velmi spolehlivé v udržení informace za provozu. Konfigurační soubor je při konfiguraci nahrán do tohoto úložiště, odkud je naprogramováno vnitřní zapojení hradlového pole. Nevýhodou je, že v případě výpadku energie je konfigurace ztracena a je třeba ji nahrát znovu. Řešením je doplnit na desce plošných spojů hradlové pole řídícím obvodem s nevolatilní pamětí, které při obnovení napájení provede automatickou konfiguraci přes jedno z rozhraní zmíněných dále. Konfigurační soubor umožňuje také inicializovat obsah vnitřních paměťových obvodů, které jsou zahrnuté ve výsledném zapojení (BRAM nebo URAM). V případech, kdy není možné přerušit činnost celého hradlového pole pro jeho konfiguraci, je rovněž možno provést konfiguraci pouze specifických regionů, přičemž zbytek čipu je ponechán v činnosti.

Zařízení UltraScale+ poskytují několik typů konfiguračních rozhraní. Mezi nimi třeba SPI s proměnnou šířkou datové sběrnice, BPI, SelectMAP nebo JTAG. Vnitřní logika hradlového pole sama detekuje její typ a používanou šířku dat. Dalšími poskytovanými možnostmi programování je zvolit zařízení jako master, které používá pro řízení sběrnice vlastní hodinový signál, nebo slave, kdy je pro řízení sběrnice použit hodinový signál konfigurátoru. Zařízení lze rovněž nakonfigurovat skrze rozhraní PCI Express. Pro tyto účely je v hradlovém poli k dispozici rozhraní *Media Configuration Access Point* (MCAP), jehož účelem je propojit konfigurační logiku

s bloky zajišťující přenos přes PCI Express.

Volitelně lze obsah konfiguračního souboru zašifrovat algoritmem AEG-GCM a jeho nahrání do logického obvodu ještě podmínit nutností autentizace. Tato funkce umožňuje nasazení obvodů s hradlovými poli v komerčních aplikacích, či v oblastech, kde je potřeba zamezit zneužití vlivem například změny konfigurace ve prospěch špatných záměrů třetí strany[13].

Zařízení ještě disponuje bloky pro vysokorychlostní sériovou komunikaci nazvané GTY (rozebrány dále) nebo GTM. Ty mohou být poté kombinovány v různých zapojeních, například *100G Ethernet subsystem*, nazvaný také CMAC, nebo *10/25G Ethernet Subsystem*, poskytovaných Xilinxem skrze licenci.

### 2.2.4 DSP a Systémový monitor

Nezanedbatelnou součástí hradlových polí jsou dnes bezesporu bloky pro zpracování digitálního signálu, neboli *Digital signal processing* (DSP). V základu fungují jako násobičky 27×18 bitů doplněné akumulátorem o šířce 48 bitů. Volitelně lze násobičku přemostit a oba vstupy DSP bloku směřovat do speciální aritmetické jednotky *Single Instruction Multiple Data* (SIMD) nebo do logické jednotky tvořící funkci mezi těmito vstupy (příkladem XOR o šířce 96 bitů). SIMD lze nakonfigurovat jako dvojitou sčítačku, odčítačku nebo akumulátor. Stabilita na vysokých kmitočtech je zaručena dostatečným množstvím registrů pro tvoření pipeline. Podobně jako paměti URAM, lze i DSP bloky spojovat do kaskád v rámci hodinových regionů a regionů umístěných pod sebou. V každém sloupci lze nalézt 24 DSP jader. Variabilita využití DSP bloků dnes sahá již daleko za hranice prostého zpracování signálů. Zmínkou třeba šifrování, aplikace jako posouvačů sběrnic, generátorů paměťových adres nebo širokosběrnicových multiplexerů[14].

Předposlední blok, který zmíníme je Systémový monitor, jenž slouží pro sledování stavu čipu během provozu. Tvoří jej 17-ti kanálový ADC převodník s množstvím sond rozmístěných po struktuře FPGA. K monitorovaným veličinám patří externí napětí, teplota čipu a hodnoty napájecích napětí. S blokem lze komunikovat skrze rozhraní JTAG, $I^2C$, DRP (viz dále) nebo Power Management Bus (PMB, pouze v zařízeních UltraScale+)[7].

### 2.2.5 GTY transceivery

Přehled funkčních bloků uzavíráme pro nás nejdůležitější komponentou a to bloky pro vysokorychlostní sériovou komunikaci, jejichž nejnovější verze nese název GTY transceivery. Principiálně se jedná o serializér/deserializér, který má diferenciální vývody na sériové straně. Transceivery jsou organizovány ve skupinách po čtyřech (nazvány *Quad*) s vlastními generátory hodinových signálů. Vysokých pracovních

frekvencí je dosaženo pomocí obvodů *Phase-locked Loop* (PLL), jenž lze nalézt ve dvou typech[15].

Prvním je tzv. *Kanálová PLL* (CPLL) typu *Ring*. Vyznačuje se velkým rozsahem výstupních kmitočtů a malou obsazenou plochou čipu, avšak za cenu vyššího šumu (jenž roste s frekvencí) oproti smyčkám *Quad PLL* (QPLL). Tyto smyčky typu *LC* vykazují minimální hodnotu šumu na výstupu a vyšší dosažitelnou frekvenci za cenu většího množství zabrané plochy a menšího nastavitelného rozsahu výstupní frekvence[16]. V každé čtveřici transceiverů lze využít až čtyři smyčky typu CPLL nebo až dvě typu QPLL. Referenční hodiny pro smyčky lze vést buď z obecného generátoru referenčních hodin nebo z rozvodů dvojice severních, resp. jižních cest nataženými přes dvě čtveřice nahoru, resp. dolů. Tímto je možno využít jedny referenční hodiny pro až pět čtveřic GTY.

Pro odstranění šumu způsobeného mezisymbolovými přeslechy je v GTY nejvíce využívána funkce *Decision Feedback Equalization* (DFE). Pro určení vlastností přenosového média je využíváno vzorků prodloužení sestupné hrany datového signálu (tzv. *post-kurzory*), což přináší výhodu nižší latence při přenosu oproti použití speciálních testovacích sekvencí. Zkreslení přenášeného signálu je způsobeno impedanční charakteristikou přenosového kanálu, většinou vlivem materiálu laděného jako dolní propust. Odečtením post-kurzorů a signálu na výstupu vzniká signál s částečně eliminovaným zkreslením. Ekvalizace DFE zesiluje tímto signál, aniž by zesilovala šum. Vhodné použití nachází rovněž při přenosu po médiích s nehomogenní impedanční charakteristikou, avšak není pomocí ní možno odstranit zkreslení náběžné hrany, tzv. *pre-kurzor*. Pro tento účel je nutné celý obvod ekvalizace doplnit o další prvky. Měření post-kurzorů je v GTY transceiverech prováděno automaticky, což poskytuje výhodu při změnách parametrů linky, jako přerušení nebo rozpojení kabelu, či vlivem jeho stáří. Druhým režimem GTY transceiverů je tzv. *Low-power mode* (LPM). Jelikož na krátkých spojích je použití ekvalizace takřka zbytečné (obecně na spojích s malou pravděpodobností zanesení chyb do datového přenosu), existuje tento režim k jejímu výraznému omezení. Lze tímto tedy zredukovat množství spotřeby energie na těchto spojích až o 30%. Na vysílací straně je zase snaha co nejvíce eliminovat vstupní ztráty, funkce, která toto zajišťuje se zde nazývá *de-emphasis*.

Zapojením PLL a ekvalizací přenosového kanálu mohou transceivery v řadě UltraScale+ dosahovat bitové rychlosti až 32,75 Gb/s. Zvyšování rychlosti ale limituje použití referenčních hodin z jiných čtveřic. Do 16,375 Gb/s lze využívat hodiny standardně do druhé Quad nahoru/dolů, mezi 16,375 Gb/s a 28,21 Gb/s pouze do jedné Quad nahoru/dolů a nad tyto rychlosti již není povoleno vůbec.

Na přijímací straně dochází ještě k tzv. *regeneraci hodinového signálu*. Jeho funkce spočívá v syntéze hodin, které jsou odvozeny na základě rychlosti přijímaných dat. Tímto vzniká druhá hodinová doména oproti vysílacím hodinám řízenými

smyčkami fázového závěsu umístěnými na čipu. Vznik takovéto situace potom přináší požadavky pro řešení přechodů mezi těmito doménami, na něž bude přihlédnuto v kapitole zabývající se návrhem.

Nastavené parametry GTY transceiverů lze, rovněž kvůli implementaci jako „pevného bloku“, měnit za běhu. Pro tento účel je pro každý kanál k dispozici rozhraní *Dynamic Reconfiguration Port* (DRP). Každý registr v obvodu lze adresovat přes sběrnici o šířce 10 bitů. Sběrnice je řízena vlastním hodinovým signálem a platnost čtecích/zápisových požadavků je podmíněna signálem DRPEN nastaveným do log. 1. Rozhraní poskytuje oddělený čtecí a zápisový port, každý široký 16 bitů. Pro rozlišení požadavků pro čtení a zápis je využit signál DRPWE, kdy při hodnotě log. 1 je požadován zápis, při opačné čtení. Připravenost rozhraní pro plnění požadavků je ze strany GTY signalizována pomocí signálu DRPRDY. Pro aplikaci nového obsahu DRP registrů je nakonec nutno restartovat celý kanál GTY. Většinu DRP registrů lze nastavit již při programování hradlového pole a jejich parametry měnit dle potřeby za provozu. Většina má specifickou funkci, mnoho z nich má ale nejasnou nebo nedokumentovanou a jsou pravděpodobně záležitostí *Intelektuálního vlastnictví* (Intellectual Property) společnosti Xilinx.

Technologie transceiverů nachází dnes využití zejména v zařízeních využívající komunikaci přes vysokorychlostní linky. Možnosti uplatnění shledáme třeba v komunikaci mezi jednotlivými čipy (chip-to-chip), mezi kartami v serverové skříni nebo modulárních systémech (tzv. backplane). Možné využití nalézá i ve známých rozhraních, jako SATA (až do verze 3.1), Display Port (pouze vysílač do verze 1.2a) nebo USB 3.0. Některé modely FPGA nabízejí možnost nakonfigurovat GTY transceivery jako PCI Express rozhraní podporující standard až do verze 4.0[15].

## 2.3 Modely řady UltraScale+

Firma Xilinx již od počátku navrhuje svá zařízení pro různé oblasti trhu a jinak tomu není ani u FPGA. Zde jsou představeny tři různé modely nabízené v řadě UltraScale+.

Nejméně výkonným modelem hradlového pole je čip Kintex. Jeho výroba byla podmíněna požadavkem vytvořit čip s výhodami nejrychlejší nabízené řady s důrazem na co nejnižší cenu. Disponuje tedy všemi výhodami řady jako obsahem pamětí Ultra RAM nebo vylepšenými CLB.

Nejvýkonnější modely mezi hradlovými poli u firmy Xilinx spočívají v čipech Virtex. Jedná se o zařízení stavěná na nejvyšší zatížení ve vysokorychlostním zpracování dat a na nejvyšší propustnost. Obsahují nejvyšší počet vnitřních bloků ze všech nabízených FPGA. Mezi nimi nejvyšší množství konfigurovatelných bloků přesahující 8 milionů, celková paměť UltraRAM o velikosti až 360Mb a u některých modelů

i paměť HBM. Množství sériových transceiverů se od řady UltraScale zásadně neliší, jsou však stavěny na vyšší rychlosti.

Speciální místo v portfoliu řady potom tvoří modely Zynq. Zařízení tohoto typu kombinují logiku hradlového pole a procesoru Arm Cortex. Použití CPU jako „pevného bloku“ nese výhodu v homogenním chování při výpočetních operacích. Řada se dělí ještě na dvě, čímž lze lépe zvolit požadované zařízení pro určitou aplikaci. První tvoří systémy MPSoC (Multiple Processors System-on-Chip), které využívají pro výpočty více procesorových jader. Dále mohou být doplněny o grafické jádro nebo video-kodek. Druhá zahrnuje systémy RFSoC (Radio Frequency System-on-Chip), která je upravena pro aplikaci v radiofrekvenční technice. Obsaženy jsou navíc převodníky RF-ADC a RF-DAC pro převádění dat z analogových na digitální a zpět. Posledním doplňkem je blok korekčního kódování SD-FEC (Soft Decision Forward Error Correction)[7].

### 2.3.1 Zde používaný model

Námi používaný model je Virtex UltraScale+ 7.generace, označení *xcvu7p-flvb2104-2-i*. Disponuje dvěma oblastmi SLR o šířce 6 a výšce 5 hodinových regionů. Obsaženo je celkem 76 GTY transceiverů, 788000 LUT a 1576000 D klopných obvodů. Dále je možnost využití celkem 50,6Mb pamětí BlockRAM nebo 180Mb pamětí UltraRAM. Pro vysokorychlostní výpočty je k dispozici až 4560 DSP bloků. Ze sběrnic je přítomny 4 bloky PCIE4 kompatibilní s PCI Express generace 3. Zařízení dále obsahuje 4 „pevné bloky“ pro Ethernetové rozhraní o rychlosti 100Gb/s s funkcí korekčního kódování. Nejsou zde obsaženy žádné paměti HBM, ani kompatibilní bloky s PCI Express generace 4. Celé FPGA je stavěné na rozsah teplot v průmyslovém prostředí (zde −40°C až +100°C) a umístěné v BGA pouzdře s 2014 piny, z nichž 702 je koncipováno jako *high-performance*.

# 3 Standard 802.3

Standard 802.3 pro Ethernet popisuje návrh sítí a síťových zařízení v lokálním (LAN), resp. metropolitním rozsahu (MAN). Definuje společnou podvrstvu *Media Access Control* (MAC), *Management Information Base* (MIB) a *Media Independent Interface* (MII) mezi fyzickou a ethernetovou vrstvou modelu *International Standardization Organization/Open System Interconnection* (ISO/OSI). Standard rozlišuje dva základní koncepty provozu na polo-duplexní a plně-duplexní. Zároveň definuje strukturu fyzické vrstvy (zvané též PHY) pro některé typy přenosových médií[17].

## 3.1 Struktura fyzické vrstvy

Fyzická vrstva zajišťuje přístup ke sdílenému médiu, potřebné kódování signálu, případně další úpravy pro průchod signálu přenosovým kanálem. Rovněž zabezpečuje přechod mezi, na médiu nezávislém, rozhraním MAC vrstvy (Media Independent Interface) a rozhraním připojujícím přenosový kanál, jenž je na médiu závislé (Media Dependent Interface)[18]. Pro rychlosti nad 100Mb/s se fyzická vrstva dále dělí na tři podvrstvy:

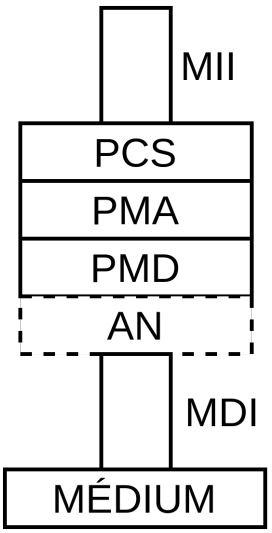
***Physical Coding Sublayer* (PCS)** zajišťuje kódování, resp. dekódování dat přicházejících z MAC, resp. do ní a v našem případě využívá kódování 64b/66b. Další funkcí je zarovnání bloků přijatých dat a odstranění jejich hlaviček. Rovněž zabezpečuje přechod mezi dvěma hodinovými doménami, na jedné straně MAC a na druhé z vrstvy PMA[18].

***Physical Medium Attachement* (PMA)** zajišťuje rozhraní pro připojení k přenosovým modulům (např. QSFP, QSFP+) a na FPGA je reprezentována GTY transceivery. Principiálně funguje jako serializér/deserializér s paralelní sběrnicí o šířce 64 bitů a dvěma bity reprezentující jednoduchou hlavičku[15].

***Physical Medium Dependent* (PMD)** je nejspodnější vrstvou fyzické vrstvy, která je zastoupena v neprogramovatelném hardware. V našem případě se jedná o výměnné moduly pro připojení různých typů kabelů. Z podstaty vyplývá, že tato vrstva má po svém „vyrobení“ nulovou možnost přizpůsobení/rekonfigurace[19].

Mimo tyto hlavní vrstvy lze pod PMD volitelně nalézt i podvrstvu zajišťující autonegociační funkci. Ta obsluhuje základní logiku pro dohodu parametrů síťového provozu mez dvěma síťovými rozhraními. Schéma fyzické vrstvy je možné shlédnout na obrázku 3.1 ([20]).

Vzhledem k absenci rekonfigurace PMD, v podobě jak je reprezentována na hradlovém poli, a jednoduchosti výměnných modulů, je nemožné implementovat




Obr. 3.1: Struktura fyzické vrstvy podle 802.3

auto-negociační funkci na úplném dně síťového modelu. Z tohoto důvodu bylo přistoupeno k mírným změnám oproti standardu a blok auto-negociace byl přesunut mezi vrstvy PMA a PCS. Jedná se o nejnižší umístění v hierarchii FPGA, které dovoluje propojování s ostatními komponentami a rekonfiguraci.

## 3.2 Auto-negociace

Funkce auto-negociace je definována v sekci 5, klauzule 73, standardu 802.3[1]. Klauzule je zaměřena na popis funkce pro komunikační rychlosti 25–100 Gb/s. Přestože je její implementace povinná (v případě médií vyrobených z mědi) pro spoje mezi čipy přes *Printed circuit board* (PCB) (tzv. backplane) nebo DAC kabel, její použití je volitelné. Funkci auto-negociace lze demonstrovat na situaci, kdy každá strana odešle vlastnosti linek, které je schopna provozovat. V základu lze přenášet informaci o podporovaných linkových rychlostech a schopnosti FEC. Pokud jsou některé vlastnosti kompatibilní, na jejich základě potom sestaví spojení. Charakteristickým rysem je, že celý proces se děje automaticky a případná shoda/neshoda je oznámena vyšším vrstvám[20].

### 3.2.1 Base Page

Základní komunikační jednotkou je stránka mající 48 bitů s pevně daným významem a kterou lze nalézt ve dvou variantách. První je tzv. „základní stránka“, neboli *Base*

[1]Standard je pravidelně obnovován, zde popsané vlastnosti jsou záležitostí revize 2018

*Page* (BP), která je využívána při prvotní dohodě společných parametrů. Pakliže je potřeba přenosu dodatečných informací během auto-negociace, lze využít ještě tzv. *Next Page* (NP). Těch lze přenášet více za sebou s odlišným druhem zprávy. Standard ještě definuje dva druhy pro stránky typu Next Page. Pro jejich příjem je vyžadována úprava logiky pro zpracování základních stránek. Na tuto funkci se podíváme podrobně dále. Nyní bude podrobně probrána struktura stránky Base Page. Bity jsou zde rozděleny specificky, jako je vypsáno v tabulce 3.1.

| Bity | Název | Popis |
|---|---|---|
| 4:0 | Selector Field | Identifikace standardu, pro 802.3 nastaveno na „00001“. |
| 9:5 | *Echoed Nonce* (EN) | Zkopírovaná specifická skupina bitů přijatých od protistrany. |
| 12:10 | Funkce Pause | Schopnost vrstvy MAC dočasně pozastavit provoz. Bit 10 indikuje symetrickou pauzu (*PAUSE*), 11 asymetrickou pauzu (*ASM_DIR*) a 12 je rezervována. |
| 13 | *Remote Fault* (RF) | Umožňuje přenášet prosté oznámení, zda na straně vysílače byla detekována chyba. Upřesňující informace lze zaslat pomocí funkce Next Page. |
| 14 | *Acknowledge* (ACK) | Indikace, zda daná strana obdržela od protistrany platnou stránku. |
| 15 | Next Page | Signalizuje, že daná strana potřebuje po přenosu Base Page zaslat ještě jednu nebo více stránek Next Page. |
| 20:16 | *Transmitted Nonce* (TN) | Slouží k potvrzení přijetí na protistraně, kdy TN přijatá je zkopírována do EN stránky vysílané. Musí být generována pomocí náhodného nebo pseudonáhodného generátoru. |
| 43:21 | Technology Ability Field | Seznam technologií podporovaných danou stranou. |
| 47:44 | FEC capability | Indikace schopnosti provádět FEC. |

Tab. 3.1: Rozdělení bitů v Base Page

Linková karta je teoreticky schopna propagovat všechny možné technologie v poli Technology Ability Field. V realitě je však takováto situace nemožná, protože vý-

měnné optické/kabelové moduly jsou, co se týče funkce, velmi jednostranné. Lze však propagovat více než jednu technologii. V současnosti je jich ve standardu k nalezení 11 a jsou vypsány v tabulce 3.2. Každý typ má navíc přidělenou svoji prioritu, která se uplatňuje při rozhodování v případě, kdy zařízení disponuje vícero druhy přenosových technologií. V našem případě bude implementace testována na rozhraní 25GBASE-CR.[2] Toto zaměření bude udržováno v celém rozsahu práce na spojení pomocí metalických kabelů a experimentálně i optických. Pokud je v kabelu využito pro danou rychlost více linek, auto-negociace probíhá vždy po lince 0. Řešení návrhu se tímto redukuje na rychlosti 10 Gb/s a 25 Gb/s.

| Číslo bitu v Base Page | Technologie | Priorita |
|---|---|---|
| 21 | 1000BASE-KX | 11 |
| 22 | 10GBASE-KX4 | 10 |
| 23 | 10GBASE-KR | 9 |
| 24 | 40GBASE-KR4 | 6 |
| 25 | 40GBASE-CR4 | 5 |
| 26 | 100GBASE-CR10 | 4 |
| 27 | 100GBASE-KP4 | 3 |
| 28 | 100GBASE-KR4 | 2 |
| 29 | 100GBASE-CR4 | 1 |
| 30 | 25GBASE-KR-S/25GBASE-CR-S | 8 |
| 31 | 25GBASE-KR/25GBASE-CR | 7 |
| 43:32 | Rezervováno pro budoucí použití | ignorováno |

Tab. 3.2: Přidělení bitů a specifických technologií

Posledním dohodnutelným atributem je schopnost daných stran provozovat FEC. Vyhrazené místo v negociační stránce spočívá ve čtyřech posledních bitech. Základní dohodnutelné typy FEC se dělí podle rychlosti jedné linky. V případě rychlosti 25 Gb/s lze využít varianty Reed-Solomon nebo BASE-R. V druhém případě (10 Gb/s) existuje pouze jedna varianta kódování, avšak dva bity účelně zůstaly, aby rozlišily stav žádosti/schopnosti (v prvním případě existuje pouze žádost). Konkrétní přidělení bitů a jejich název je poskytnut v tabulce 3.3. Stejně jako dohoda technologií, tak i korekčního kódování má svou vlastní sadu pravidel pro ustanovení shody, které se nazývají *prioritní rezoluční funkce*. Problematika jejího chování a návrhu bude probrána v části o implementaci[20].

[2]Logika bitů je, v případě standardu, brána zvláště jako pozitivní, tzn. log.1 znamená „povolení“, „ano“, „zapnout“, apod., kdežto log.0 znamená opak. Pokud se tento charakter v textu

| Číslo bitu v Base Page | Název |
|---|---|
| 44 | 25G RS-FEC requested |
| 45 | 25G BASE-R FEC requested |
| 46 | 10G FEC ability |
| 47 | 10G FEC requested |

Tab. 3.3: Přidělení bitů pro schopnost FEC

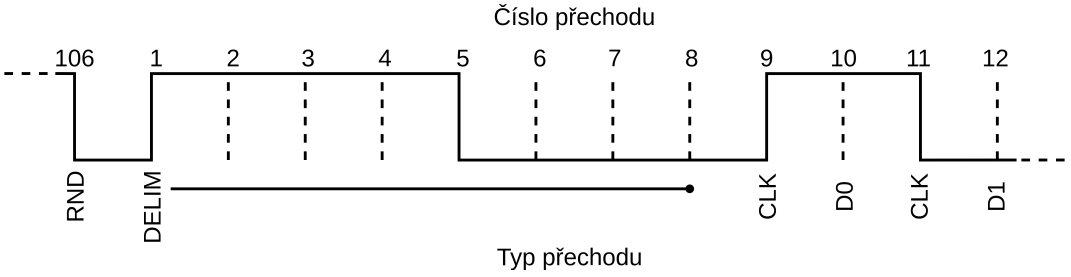
### 3.2.2 Kódování stránky

Při přenosu komunikačním kanálem je pro negociační stránku využito speciálního typu linkového kódování nazvaného *Diferenciální kódování Manchester* (DME). Vyznačuje se nesením jednotlivých bitů v přechodech, namísto v konkrétních hladinách (striktně log.1 nebo 0). Výhoda použití přechodu oproti hladinám spočívá ve snadnějším rozeznání v zašumělém signálu. Kódování neobsahuje stejnosměrnou složku a umožňuje signálu obnovit synchronizaci, protože se mění i při přenášení informace o dlouhodobě neměnné hodnotě[21].

Auto-negociační stránka má v zakódovaném stavu celkem 106 přechodů, z nichž 48 je hodinových a 48 datových. Logická 1 je reprezentována změnou hodnoty oproti předcházející, čili přechod na daném místě vzniká. Logická 0 naopak zapsáním stejné hodnoty, jakou měl předchozí symbol, čili přechod na daném místě nevzniká. Hodinové přechody se střídají s datovými v pravidelných intervalech, kdy na lichých pozicích jsou přítomny hodiny (u nichž přechod musí nastat vždy), na sudých data, která se kódují od LSB. Na počátek zakódované stránky je dále umístěn oddělovač, tzv. *Manchester violation delimiter* (DELIM), sloužící pro zachycení začátku stránky v proudu bitů. Oddělovač je tvořen osmi intervaly s existujícími přechody pouze na 1. a 5. pozici. Jedná se o jediné místo v zakódované stránce, kde existují čtyři intervaly mezi přechody. Maximální přípustné množství ve zbytku stránky je potom 2. Bitový tvar takového oddělovače je buď „11110000“ nebo „00001111“.

V zakódovaném sledu bitů ještě zbývá přidělit funkci poslednímu přechodu. Jelikož jsou stránky přenášeny neustále za sebou bez přestávky, či mezer, vznikal by potom periodický signál s periodou rovné délce stránky. Takováto situace má nepříznivý vliv na celkové spektrum přenášeného signálu. Poslední přechod je tedy vyhrazen pro pseudonáhodný bit, který zajišťuje občasné invertování/neinvertování následující stránky. Hodnotu lze odvodit z generačního polynomu 3.1, na základě kterého je sestrojen *Linear Feedback Shift Register* (LFSR) generátor, jehož stav je inkrementován jednou za odeslanou stránku. Tímto je zajištěna eliminace vzniku

nějak změní, bude na to upozorněno.




Obr. 3.2: Ukázka kódování stránky pomocí Diferenciálního kódování Manchester

periodického signálu[20].

$$P(x) = x^7 + x^3 + 1 \text{ nebo } P(x) = x^7 + x^6 + 1 \tag{3.1}$$

### 3.2.3 Řízení

Důležitým systémem pro funkci auto-negociace je systém managementu. Obsahuje potřebné registry, pomocí nichž lze řídit průběh procesu (vypnutí, restartování, povolení). Rovněž disponuje registrem obsahujícím potřebné stavové bity (dokončení auto-negociace, příjem stránky, stav linky). Dále poskytuje místo pro uložení Base Page přijaté od protistrany a druhé pro odeslání místní stranou. Rovněž umožňuje uložení hodnoty stránek Next Page a dále s nimi operuje. Poslední důležitou informací drženou v systému řízení je, jaké technologie byly mezi linkovými stranami dohodnuty. O adresách registrů a jejich názvech je pojednáno v tabulce 3.4 (každý registr disponuje šestnácti bity pro zápis, čtení nebo obojí). Delší názvy jsou vhodně zkráceny: *Link Partner* (LP) a *Extended Next Page* (XNP)[22].

| Adresa registru | Název | Přístup z pohledu AN komponenty |
|---|---|---|
| 7.0 | AN control | R |
| 7.1 | AN status | WR |
| 7.16, 7.17, 7.18 | AN advertisement | R |
| 7.19, 7.20, 7.21 | AN LP BP ability | R/WR |
| 7.22, 7.23, 7.24 | AN XNP transmit | R |
| 7.25, 7.26, 7.27 | AN LP XNP ability | WR |
| 7.48 | BASE-R copper status | WR |

Tab. 3.4: Management registry pro auto-negociaci

### 3.2.4 Funkce Next Page

Standard rovněž umožňuje zasílání doplňujících zpráv mezi stranami. Struktura stránky Next Page je volnější oproti stránce Base Page. Nalézt ji lze ve dvou formách, z nichž první je tzv. *Zpráva* (Message). Zpráva se dělí na dvě části, přičemž první je vyhrazena pro specifikaci typu zprávy. Druhá část slouží již pro přenos samotné informace. Mezi nimi lze nalézt ještě některé řídící bity. Struktura stránky Next Page ve stylu Zprávy je sepsána v tabulce 3.5. Možné druhy zpráv jsou například informace o počtu následujících stránek (1 nebo 2), které budou obsahovat doplňující informace o technologiích; informace o podstatě chyby indikované v bitu RF stránky Base Page; jedinečný *Organizationally Unique Identifier* (OUI) nebo identifikátor fyzické vrstvy. Pokud je na jedné straně potřeba zasílat Next Page a na druhé nikoliv, zasílá druhá strana tzv. *Nulovou zprávu* (Null Message).

| Bity | Název | Popis |
|---|---|---|
| 10:0 | Message Code Field | Určuje typ zprávy |
| 11 | Toggle | Slouží pro jednoduchou synchronizaci mezi Next Page. |
| 12 | Ack 2 | Potvrzuje protistrana, zda je schopna reagovat na danou informaci. |
| 13 | *Message Page* (MP) | Nastaven do log. 1, pokud se jedná o stránku typu Message, log. 0, pokud se jedná o stránku Unformatted. |
| 14 | ACK | Stejná funkce jako v BP. |
| 15 | NP | log. 1, pokud budou následovat ještě další Next Page. |
| 47:16 | Unformated Code Field | Místo nesoucí informaci dané zprávy. |

Tab. 3.5: Rozdělení bitů v Next Page

Druhou formou je styl stránky nazvaný *Neformátovaný* (Unformatted). Rozdíl od prvního stylu spočívá v odlišné funkci bitů [10:0]. Ty zde slouží stejně, jako druhá polovina, tj. nositel informace dané zprávy. Z principu vyplývá, že daná skupina Neformátovaných stránek musí být vždy v závěsu za jednou Zprávou. Standard definuje i Next Page s rozšířeným polem pro nesení informace, který nazývá Extended Next

Page. Tento formát je uvažován rovněž v rámci klauzule 73 a v návrhu komponenty bude hleděno právě na něj.

Zpracování informace ze stránek Next Page nespadá již do funkce komponenty pro auto-negociaci a musí být ošetřeno ve vrstvě zajišťující řízení. Přestože je teoreticky možné zaslat libovolně velké množství dalších stránek, neměl by jejich přenos příliš zdržet dokončení samotného procesu dohody linkových parametrů. Auto-negociační komponenty totiž blokují provoz do vyšších vrstev do doby, než dojde k výměně informací mezi nimi na obou stranách linky, tj. i v případě, že se zasílají stránky Next Page[20].

# 4 Návrh

Tato část bude věnována samotnému návrhu auto-negociační komponenty. Vzhledem k odlišnému umístění v hierarchii fyzické vrstvy je nutné pozměnit některé aspekty týkající se architektury, než jak jsou specifikovány ve standardu 802.3. V průběhu procesu vzájemné dohody parametrů je TX kanál přepnut na vysílací výstup auto-negociační komponenty a je takto ponechán až do dokončení její činnosti. Následně je TX kanál přepnut zpět na PCS.

Návrh počíná odspodu, kdy jako první bude vytvořena struktura vysílače, jelikož je nejjednodušší. Poté bude probrán návrh komponenty arbitráže, která zajišťuje zpracování dat. Nakonec bude posouzen návrh přijímače, jenž je nejsložitější. Zapojení komponent včetně pojmenování jednotlivých vstupních, výstupních i mezilehlých signálů je poskytnuto na obrázku 4.1. Postup bude probrán na variantě o linkové rychlosti 25Gb/s. Zdrojový kód komponent je sepsán v jazyce VHDL, jelikož je v evropských zemích nejrozšířenější a jeho struktura je velmi dobře čitelná. Firma Xilinx poskytuje fyzickou vrstvu, zahrnutou jako *Ethernet subsystem*, a integrace auto-negociace do ní je potom řešena v jazyku Verilog. Návrhové nástroje jsou dnes již navrženy ke zpracování kombinace obou používaných jazyků pro popis hardware.

## 4.1 Vysílač

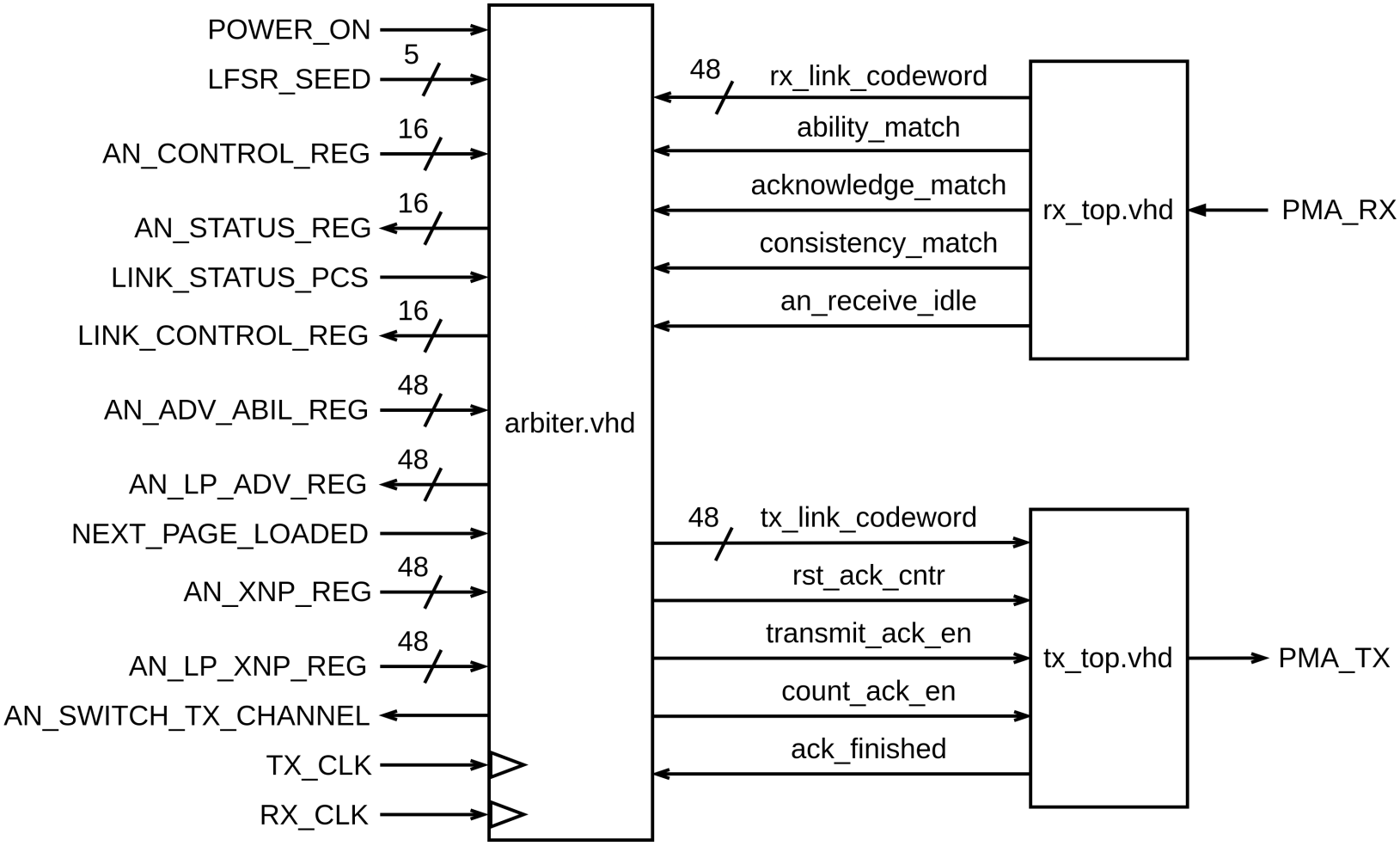
Primární funkcí vysílače je připravit AN stránku podle standardu 802.3 na přenos přes sdílené médium. Sestává ze dvou komponent umístěných v jednom „obalu“ (název `tx_top.vhd`). V něm je vytvořeno propojení obou komponent a zároveň zajišťuje rozhraní s GTY. Navíc obsahuje čítač odeslaných stránek které mají nastavený ACK bit. Funkce tohoto čítače je důležitá pro řízení stavového automatu při arbitráži (viz dále). Schéma vysílače je ukázáno na obrázku 4.2. První komponentou v obalu je kodér Diferenciálního Manchesteru.

### 4.1.1 Kodér

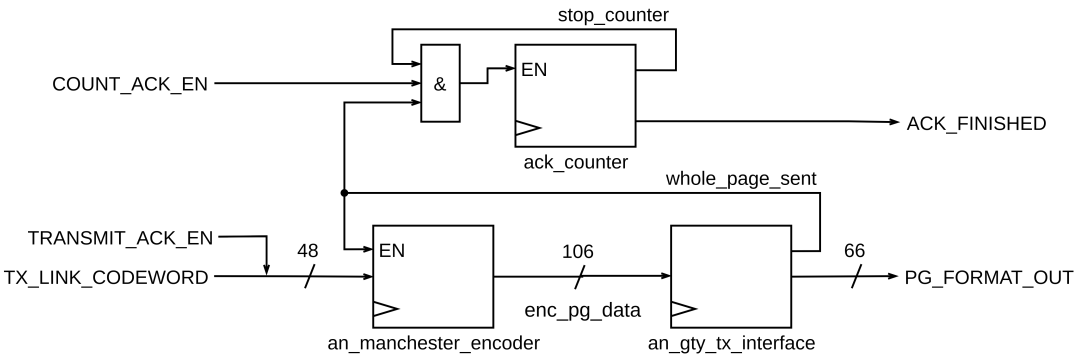
Jeho funkcí je zakódovat 48-mi bitovou AN stránku podle pravidel Diferenciálního kódování Manchester. Pro tento účel byla navržena paralelní struktura podle obrázku 4.3.

Vstup je opatřen registrem (signál `base_page_int`[1]), aby změna na vstupu ne-

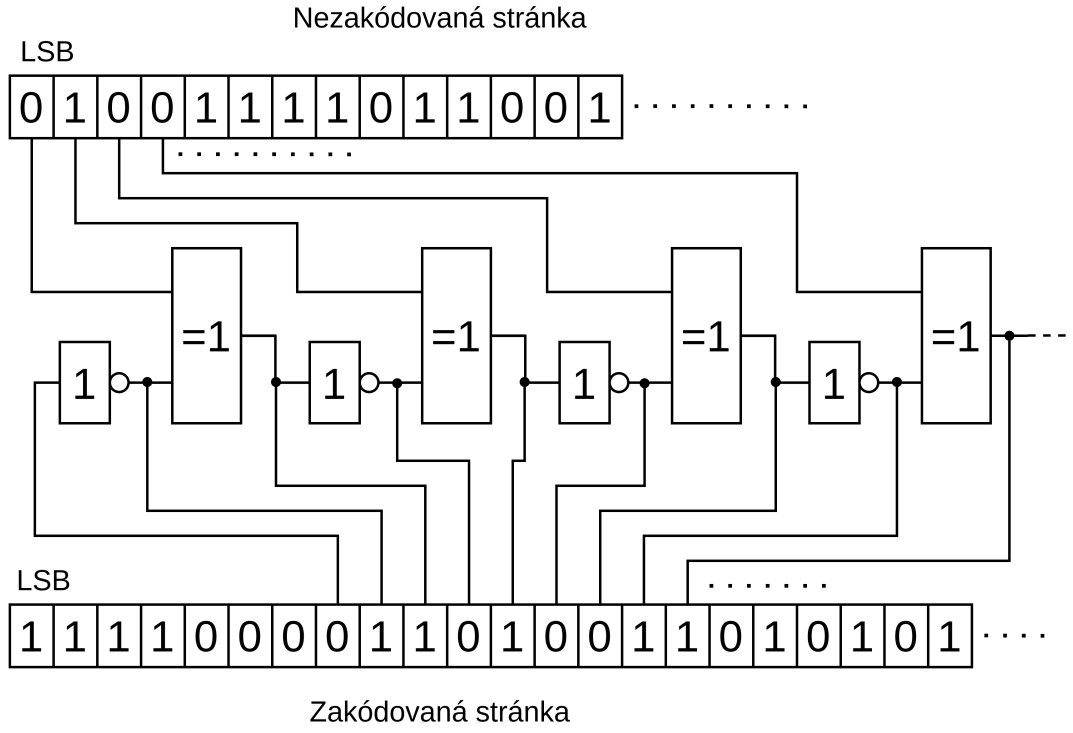
[1]Vnitřní signály komponent jsou psány malými písmeny, vstupy/výstupy komponent a konstanty potom písmeny velkými. Pro oddělení slov v názvech je používáno podtržítko. Tato konvence je udržována ve zdrojovém kódu a bude dodržována i v této práci.




Obr. 4.1: Podrobné schéma zapojení auto-negociační komponenty




Obr. 4.2: Schéma zapojení komponenty vysílače




Obr. 4.3: Schéma komponenty kodéru a příklad kódování stránky s oddělovačem

ovlivnila probíhající kódování, a výstup je opatřen registrem také (signál `output_word`), aby náhlá změna neovlivnila následující komponentu. Dochází také ke zvýšení pracovní frekvence. Registry mají společné povolovací vstupy. Kódování stránky $n$ na vstupu začne vždy, až následující komponenta dokončí zpracování stránky $n+2$. Pro tento účel je do komponenty zařazen RS klopný obvod, který plní funkci jednoduchého handshakingu. Oznámení o dokončení zpracování stránky je přijato signálem `WHOLE_PAGE_SENT`. Ten je následně zachycen v klopném obvodu, vstupní slovo je

posunuto ke kódování, zakódované slovo na výstup a hodnota RS klopného obvodu vynulována (`shifted_to_output`). Vzhledem k jednoduchosti zapojení komponenty je možné zajistit zakódované slovo již v dalším hodinovém taktu po nasunutí ze vstupu.

### 4.1.2 Napojení na GTY

Napojení na GTY je řešeno v samostatné komponentě `an_gty_tx_interface.vhd`, která tvoří druhou z dvojice vysílače. Rychlost přenosu auto-negociačních dat je totiž podle standardu stanovena „pouze“ na 156,25 Mb/s, kdežto nejnižší rychlost transceiveru je 500 Mb/s[15]. Reálně je však přepínání rychlostí transceiveru složitá záležitost vzhledem k následujícím optickým modulům v PMD, které jsou stavěny na rychlosti v řádech desítek Gb/s. Tento fakt je nutné zohlednit při potřebě pomalého přenosu přes GTY.

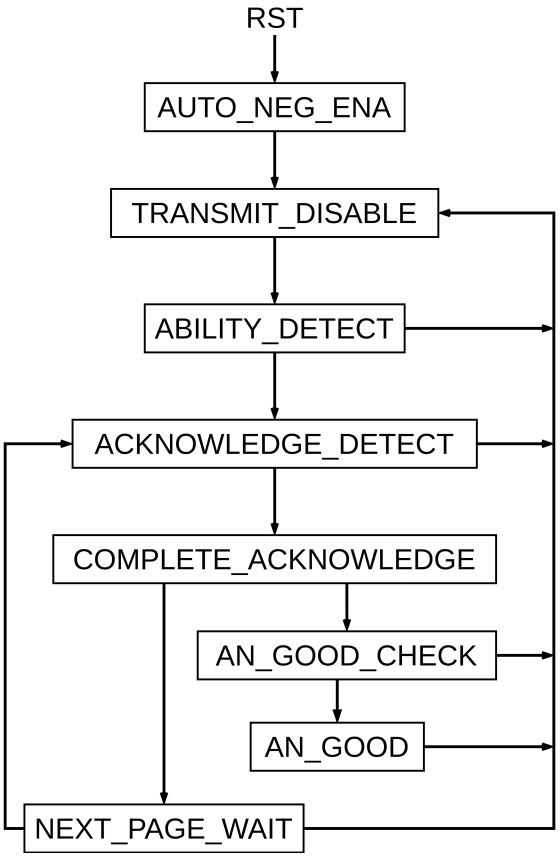
Sériová bitová rychlost pro 25Gb rozhraní je 25,78125 Gb/s, pro 10Gb variantu potom 10,3125 Gb/s. Perioda auto-negociačního signálu je, podle standardu, stanovena na 3,2 ns. V zakódované stránce je tedy nutno naklonovat každý bit tak, aby počet vytvářel stejně dlouhou logickou hodnotu, jaká je stanovena standardem, tj. 63× při rychlosti 10 Gb/s nebo 83× při 25 Gb/s. Tyto hodnoty vycházejí ze vztahů $L = L' \cdot \frac{D+H}{D}$ a $N = T_{AN} \cdot L$, kde $D$ je délka dat, $H$ délka hlavičky, $L'$ linková rychlost v Gb/s udávaná, $L$ linková rychlost v Gb/s skutečná (uvedená výše), $T_{AN}$ perioda AN přenosu v ns a $N$ počet klonů. Vzorce zahrnují i nepřesnost údajů o linkových rychlostech, u jejichž datových šířek není započítána dvoubitová hlavička. Avšak při běžném provozu jsou tyto údaje správné, protože k odstranění hlavičky dochází ve vrstvě PCS, která s jejich pomocí identifikuje začátky bloků dat. Vzhledem k místní implementaci je však třeba ji zohledňovat. Celková šířka datové sběrnice je tedy v našem případě 66 bitů.

Vstupní stránky jsou uloženy ve dvou registrech, kdy jeden ukládá stránku aktuální (`old_page`), druhý následující (`new_page`). Komponenta dále zajišťuje potřebné pseudo-náhodné otáčení následující stránky (viz 3.2.2). Bit z LFSR generátoru je uložen v přídavném registru `old_page_rnd`. Po zpracování aktuální stránky je do registru přesunuta stránka následující („novou“ následující stránkou je ta zakódovaná z výstupu kodéru), což je oznámeno signálem `WHOLE_PAGE_SENT`.

## 4.2 Arbitráž

Komponenta `arbiter.vhd` se stará o řízení dvojice přijímač-vysílač a logiky zajišťující korektní průběh auto-negociačního procesu. Dále zajišťuje přístup k registrům

v komponentě zajišťující management, přičemž umožňuje operaci s nimi. Mechanismus je řízen stavovým automatem o osmi stavech, jehož zjednodušená podoba je poskytnuta na obrázku 4.4 (podrobná forma je poskytnuta v příloze A na straně 87). Výchozím stavem je `AUTO_NEG_ENA`, v kterém se automat nachází po resetu komponenty. Existují tři druhy resetovacích signálů.




Obr. 4.4: Stručné schéma stavového automatu arbitráže

Prvním je `POWER_ON`, který signalizuje, zda zařízení je v režimu úspory energie. Logika tohoto signálu je ve standardu definována s opačnou logikou, než jak je naprogramována v zařízení, tj. log. 1 určuje stav úspory energie, 0 běžný provoz.

Druhý resetovací signál lze nalézt v bitu 15 registru 7.0 a v komponentě nese název `MR_MAIN_RESET`. Jedná se o hlavní reset AN procesu, který zároveň navrací do výchozích hodnot obsah všech registrů.

Třetím signálem je `MR_RESTART_AN`, který se nachází v bitu 9 stejného registru jako předchozí. Tento bit je implementován za základě požadavků standardu, kdy je možno, aby PMA nebo PMD signalizovaly, že nejsou schopny provozovat autonegociaci. V našem případě je jeho funkce shodná s předchozím signálem.

V následující sekci budou popsány stavy automatu při arbitráži spolu s podmínkami přechodů mezi nimi. Při návrhu bylo přistoupeno v určitým změnám, kdy některá přiřazení nebo dokonce stavy byly záměrně vypuštěny.

### 4.2.1 Popis stavů

Pro návrh stavového automatu byl použit model *Arbitration state diagram* zobrazený v klauzuli 73, standardu 802.3.

#### AUTO_NEG_ENA

Jak bylo zmíněno, jedná se o výchozí stav automatu po resetu komponenty. Do následujícího se přepíná povolovacím vstupem `MR_AN_ENABLE`, nacházejícím se v bitu 12, registru 7.0.

#### TRANSMIT_DISABLE

Zde jsou deaktivovány všechny povolené technologie v registru 7.48 (výstup `LINK_CONTROL_REG`. Současně je spuštěn čítač `break_link_timer`, který vymezuje dobu, po kterou by měl ustat provoz po přenosovém kanálu. Standard definuje dobu běhu v rozsahu 60–75 ms. Na konci běhu se automat přepíná do dalšího stavu, čítač se zastavuje.

#### ABILITY_DETECT

Při vstupu dochází ke spuštění vysílače a přečtení Base Page v registrech 7.16–7.18 (vstup `AN_ADV_ABIL_REG`). Zároveň je jednou inkrementován LFSR generátor, jehož hodnota je zapsána do pole TN stránky Base Page. Úkolem tohoto stavu je zjistit, zda protistrana je schopna přenosu AN dat (výstup `MR_LP_AUTONEG_ABLE`, bit 0 registru 7.1). Určuje tak podle toho, zda došlo k přijetí alespoň tří stejných stránek (signál z přijímače označovaný jako `ABILITY_MATCH`). V případě pozitivního nálezu je ještě zkontrolováno, zda nejsou stejná pole TN jak v odesílané stránce, tak v přijaté. Hodnota tohoto pole musí být v rámci jedné relace náhodná. Pakliže jsou hodnoty různé, je proveden přechod do dalšího stavu. Pokud tomu tak není, vrací se automat do stavu `TRANSMIT_DISABLE`.

#### ACKNOWLEDGE_DETECT

Automat nyní čeká na příjem stránek s bitem ACK nastaveným do log. 1. Podmínka příjmu je stejná, jak v případě předchozího stavu, tj. je třeba alespoň tří stejných stránek v řadě za sebou (signál z přijímače `ACKNOWLEDGE_MATCH`). Současně dochází ke zkopírování pole TN ze stránky přijaté do pole EN stránky odesílané.

Pro přechod do dalšího stavu je, mimo jiné, nutná korektní činnost tohoto mechanismu i na protistraně, což je kontrolováno (`ack_nonce_match`)[2]. K tomu existuje navíc signál `CONSISTENCY_MATCH`, pomocí kterého je v přijímači určeno, zda stránky, které způsobily nastavení `ABILITY_MATCH` a `ACKNOWLEDGE_MATCH` jsou totožné, tj. mají stejná všechna pole kromě EN a ACK. Jsou-li všechny předchozí podmínky splněny, je přijatá stránka uložena do registru `AN_LP_ADV_REG`, případně `AN_LP_XNP_REG` podle toho, zda je v průběhu výměna stránek Base Page, či Next Page. Dále je přepnuto do stavu `COMPLETE_ACKNOWLEDGE`, jinak se automat navrací do stavu `TRANSMIT_DISABLE`.

### COMPLETE_ACKNOWLEDGE

Zde již systém přijal validní stránky s nastaveným ACK bitem. Pro zajištění úspěšného přijetí i na protistraně je zasláno ještě alespoň 5 stránek s nastaveným ACK, k čemuž je využit zmíněný čítač nacházející se ve vysílači. Při dokončení běhu čítače (vstup `ACK_FINISHED`) je zkontrolováno, zda jedna ze stran potřebuje zaslat stránku typu Next Page. Určuje tak podle nastaveného bitu NP. Automat se v tomto případě přepíná do stavu `NEXT_PAGE_WAIT`. Pakliže netřeba ani na jedné straně zasílat další stránky, přechází se do stavu `AN_GOOD_CHECK`.

### NEXT_PAGE_WAIT

V tomto stavu automat čeká na příjem stránky typu Next Page. Pokud není potřeba v některém ze zařízení odesílat stránky Next Page, zasílá dané zařízení Nulovou zprávu. Stránka k odeslání musí být, ještě před přechodem do tohoto stavu, nahrána do registrů 7.19–7.21, k oznámení slouží signál `NEXT_PAGE_LOADED` řízený komponentou zajišťující management, která se i stará o správu více Next Page za sebou a rovněž indikuje skrze bit Ack 2, zda je na zprávy schopna reagovat. Registry jsou napojeny vstupem `AN_XNP_REG`, odkud je nahraná stránka přesunuta na výstup `TX_LINK_CODEWORD`. Při přenosu se uplatňuje synchronizační mechanismus zajištěný pomocí bitu Toggle. Jeho hodnota je v odesílaném slově invertována při každém průchodu stavem `COMPLETE_ACKNOWLEDGE`. Stav `NEXT_PAGE_WAIT` slouží pro čekání na nastavení signálu `ABILITY_MATCH` a pokud jsou rozdílné hodnoty Toggle bitu v přijatém a právě odesílaném slově, je přepnuto do stavu `ACKNOWLEDGE_DETECT`. V situaci, kdy náhodou dojde k přerušení činnosti přijímače oznámené signálem `AN_RECEIVE_IDLE` (viz dále), se automat navrací do `TRANSMIT_DISABLE` a celý proces je opakován včetně výměny stránek Base Page.

[2]Nutno podotknout, že v případě výměny Next Page není tento mechanismus použit, jelikož NP neobsahuje pole EN a TN.

### AN_GOOD_CHECK

Při vstupu je aktivována rezoluční funkce, která určí společnou technologii, se kterou by měly strany sestavit spojení. Při volbě je uvážena priorita dle tabulky 3.2. Nalezení shody pokračuje v případě FEC, kde je rezoluční funkce rozdělena podle typu použité fyzické vrstvy. Pro technologie 25GBASE-CR a 25GBASE-CR-S probíhá rezoluční funkce podle tabulky 4.1. Druhý typ se týká technologií mající rychlost jednotlivých linek 10 Gb/s, tj. 40GBASE-CR4 nebo 100GBASE-CR10. Funkce FEC je povolena v případě nastavení bitu 46 do log. 1 na obou stranách a zároveň nastavený bit 47 do log. 1 na alespoň jedné straně přenosového kanálu.

| Technologie | Bity stránky Base Page | | | | Výsledek |
|---|---|---|---|---|---|
| | Lokální zařízení | | Zařízení protistrany | | |
| | RS-FEC | BASE-R | RS-FEC | BASE-R | |
| 25GBASE-CR | 0 | 0 | 0 | 0 | FEC nepovoleno |
| | 1 | X | X | X | povoleno RS-FEC |
| | X | X | 1 | X | |
| | 0 | 1 | 0 | X | povoleno BASE-R |
| | 0 | X | 0 | 1 | |
| 25GBASE-CR-S | X | 0 | X | 0 | FEC nepovoleno |
| | X | 1 | X | X | povoleno BASE-R |
| | X | X | X | 1 | |

Tab. 4.1: Rezoluční funkce při vyřizování schopnosti FEC

V případě společných vlastností nastává shoda, která je signalizována nastavením příslušných bitů v registru 7.48. Komponenta již v tomto stavu signalizuje, že je auto-negociace dokončena (výstup `AN_LINK_GOOD`) a přepíná TX kanál zpět na PCS. Současně je spuštěn čítač `link_fail_inhibit_timer`, který zajišťuje prodlevu, ve které by mělo dojít k sestavení spojení a zahájení užitečného provozu po přenosovém kanálu. Doba běhu čítače je pro většinu technologií definována standardem v intervalu 500–510 ms. Detekování normálního provozu je signalizováno z vrstvy PCS signálem `LINK_STATUS_PCS` a při jeho pozitivní hodnotě je přepnuto do stavu `AN_GOOD`.

V případě, že dojde k přetečení čítače nebo není nalezena shoda při zjišťování společných vlastností, potom se automat navrací do stavu `TRANSMIT_DISABLE` a celý cyklus se znovu opakuje.

#### AN_GOOD

Jedná se o konečný stav automatu arbitráže, kdy je stále signalizováno dokončení auto-negociace výstupem `AN_LINK_GOOD` a k tomu navíc je tato skutečnost zapsána do stavového registru 7.1. Při jakékoliv chybě na lince signalizované pomocí `LINK_STATUS_PCS` je přepnuto do stavu `TRASMIT_DISABLE`, což je jediný případ, jak ze současného stavu vystoupit.

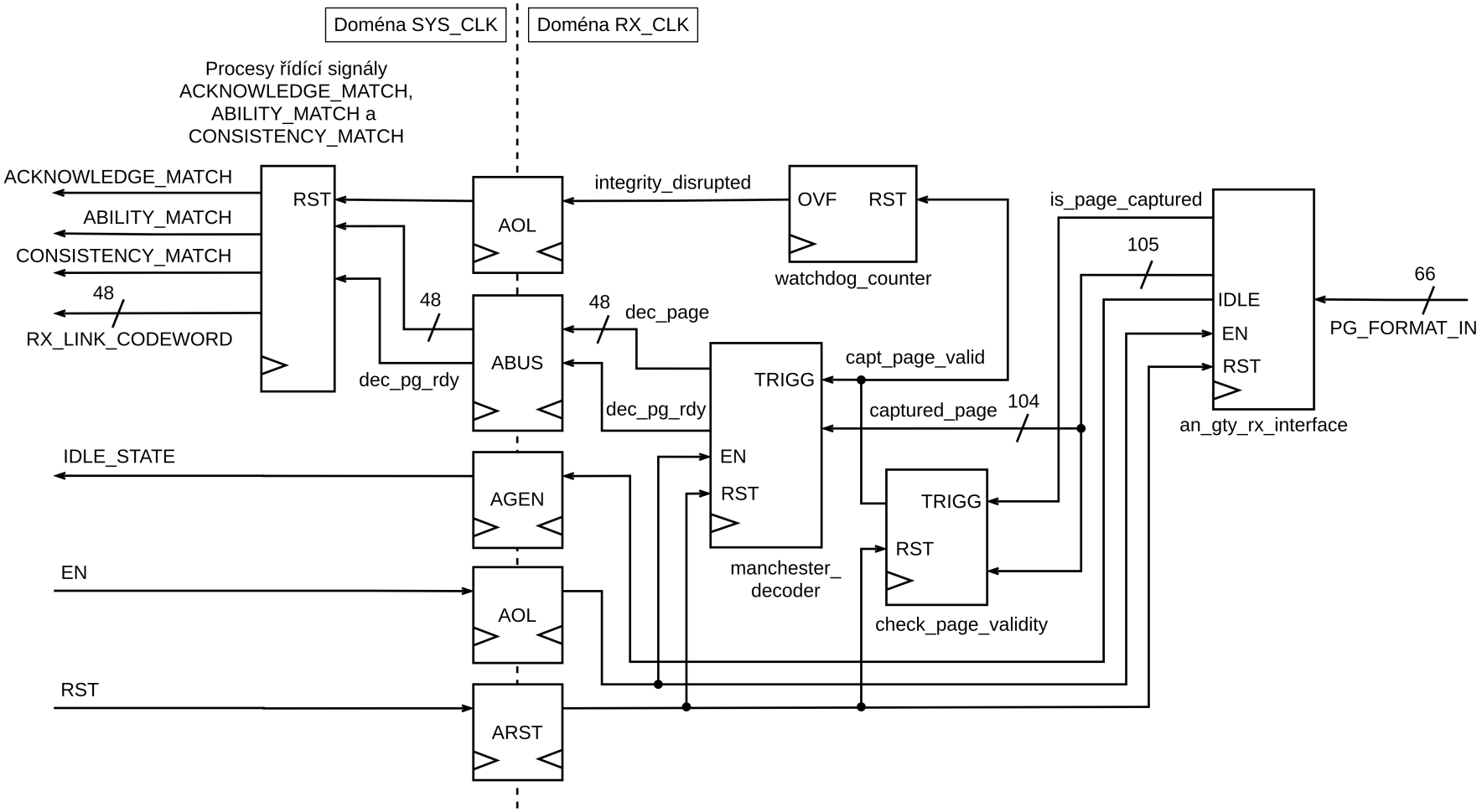
## 4.3 Přijímač

Představuje z celku nejsložitější komponentu. Zahrnuje všechny mechanismy k zajištění a ověření korektnosti přijatých dat. Obsahuje tři velké komponenty doplněné řadou obvodů. První je dekodér Diferenciálního Manchesteru pracující na podobném principu jako kodér. Druhou je napojení na GTY ze strany RX, jejíž princip bude podrobně rozebrán dále. Poslední je komponenta ověřující validitu přijaté stránky před jejím dekódováním, jenž tvoří mezivrstvu mezi dvěma předchozími. Po dekódování dochází ještě ke kontrole správnosti hlavičky v přijatém slově, zda odpovídá některé z povolených kombinací pro Base Page nebo Next Page. Okolní obvody tvoří zvláště logiku pro obsluhu signálů `ACKNOWLEDGE_MATCH` a `ABILITY_MATCH`, kdy pro každý signál je k dispozici jeden registr a čítač modulo 3, jehož přetečení znamená nastavení příslušných signálů. Podmínka přiřazení `CONSISTENCY_MATCH` spočívá potom v obou předchozích signálech nastavených do log.1 a současném totožném obsahu obou registrů, tj. všechna pole stejná kromě ACK a pole EN. Navíc jsou v přijímači přítomny jednotky pro asynchronní přechody. Návrh přijímače je řešen podle obrázku 4.5.

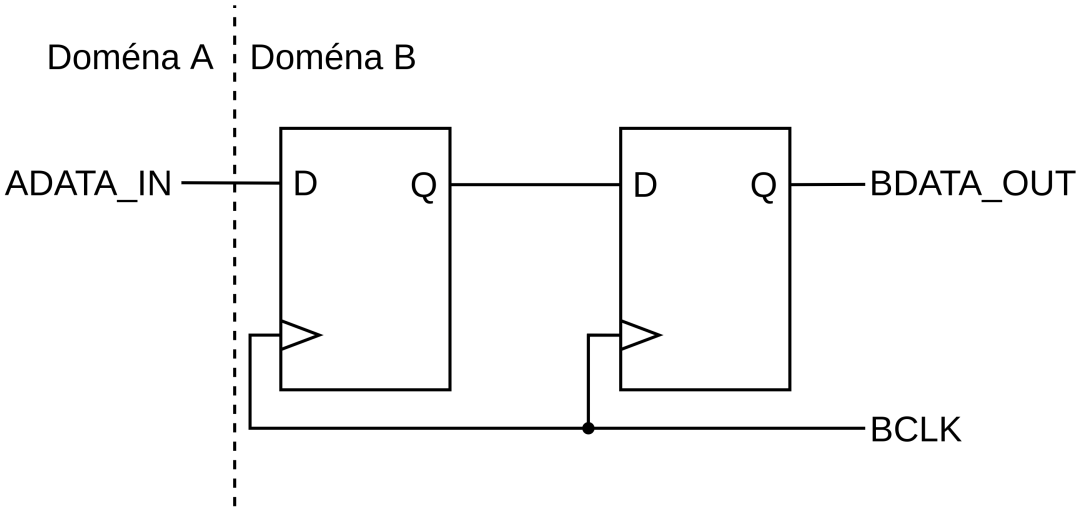
### 4.3.1 Asynchronní přechod

Již zmíněná skutečnost, kdy GTY transceivery vytvářejí druhou hodinovou doménu s periodou úměrné rychlosti přijímaných dat, přináší požadavek pro řešení asynchronních přechodů. Komponenty arbitráže, vysílače a části přijímače jsou řízeny hodinami totožnými s těmi, které používá PCS. Alespoň část přijímače však musí být řízena hodinami regenerovanými z GTY. V návrhu jsou proto použité jednotky popsané v práci [23], které zabraňují vzniku metastabilních stavů při křižování přechodu. Zde jsou konkrétně použity komponenty `ASYNC_OPEN_LOOP` pro synchronizaci pomalých signálů, `ASYNC_GENERAL`, pro případ požadavku obecné synchronizační komponenty, `ASYNC_RESET` pro resety a `ASYNC_BUS_HANDSHAKE`[3] pro vícebitové signály. Ukázka jedné z nich je na obrázku 4.6.

[3]Názvy jsou v celkovém schématu přijímače vhodně zkráceny




Obr. 4.5: Schéma komponenty přijímače. Asynchronní přechod je zde vyznačen přerušovanou čarou



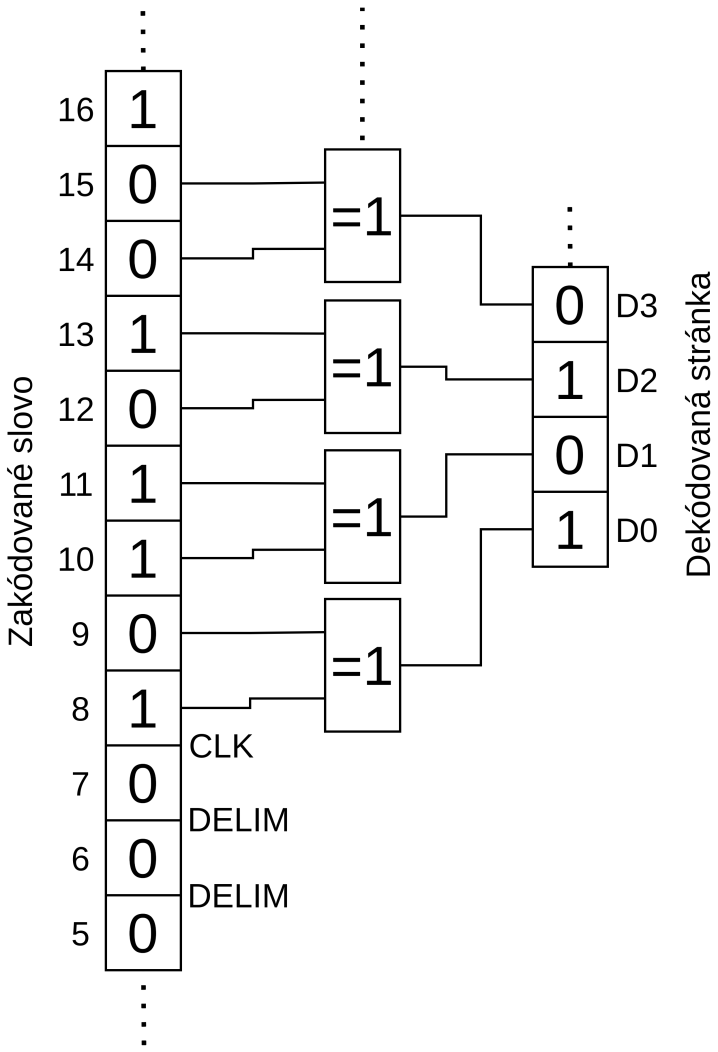

Obr. 4.6: Schéma komponenty ASYNC_OPEN_LOOP

### 4.3.2 Dekodér

Do obvodu je zařazen dekodér Diferenciálního Manchesteru, jehož zapojení je řešeno jednodušeji, než v případě kodéru. Vstupy i výstupy jsou opatřeny registry. Principiální schéma je na obrázku 4.7. Tvoří souvislou řadu hradel XOR napojené vždy na $n$-tém sudém bitu a bitu $n + 1$. Výstupy této řady hradel potom tvoří dekódovanou stránku směřující dále přes asynchronní přechod do registrů `ack_match_reg` a `ab_match_reg`.

### 4.3.3 Kontrola validity stránky

Přijímá zakódovanou stránku z komponenty zajišťující napojení na GTY, ale dále ji nezpracovává. Logický obvod ověřuje validitu stránky tím, že zkoumá, zda existuje přechod na pozici, kde jsou standardně zakódovány hodiny podle pravidel DME. Tento přístup rovněž umožňuje kontrolovat přítomnost více jak tří bitů v řadě v zakódovaném slově a zabrání tak postupu dané stránky do dalších vrstev. Tento problém byl vypozorován během experimentálního sledování simulačních výsledků. Schéma komponenty je stejné jako v případě dekodéru s tím rozdílem, že řada XOR hradel je u vstupního slova napojena druhým vstupem na bit $n - 1$ a u jejich výstupů je ověřováno, zda jsou rovny log.1. Pokud tomu tak je, je zaslán nastavený signál `PG_VALID`. Ten zároveň slouží k resetu watchdog čítače umístěného v komponentě přijímače. Čítač slouží k pomoci rozeznat situaci, kdy pravděpodobně došlo ke špatnému zachycení přechodu mezi bity v přijímaných vektorech. Při přetečení čítače dochází k resetování napojovacího rozhraní, stejně jako registrů `ack_match_reg` a `ab_match_reg`.




Obr. 4.7: Schéma komponenty dekodéru

### 4.3.4 Napojení na GTY

Poslední komponenta (pojmenována `an_gty_rx_interface.vhd`) slouží k extrakci zakódované stránky z proudu bitů. Vstupní sekvence je přijímána s každým hodinovým taktem na sběrnici z GTY, kdy každý bit je 83× naklonován. Je ale třeba počítat s tím, že tento počet nebude vždy pevně daný, protože může občas dojít ke ztrátě nebo naopak přidání klonů. Na základě přijatých dat ze zařízení Spirent SPT-N11U, bylo zjištěno, že počet klonů je proměnlivý v intervalu 80–85. Proměnlivou délku jednoho bitu ve vstupní sekvenci je třeba zvážit při návrhu této komponenty. Princip funkce je popsán na obrázku 4.8.

Na vstup je napojena řada XOR hradel, jejichž vstupy jsou napojeny na každou dvojici sousedních bitů. Tento stupeň slouží k detekci přechodu ve vstupní sekvenci. Podle délky vstupní sběrnice a středního množství klonů je možno určit, že v rámci

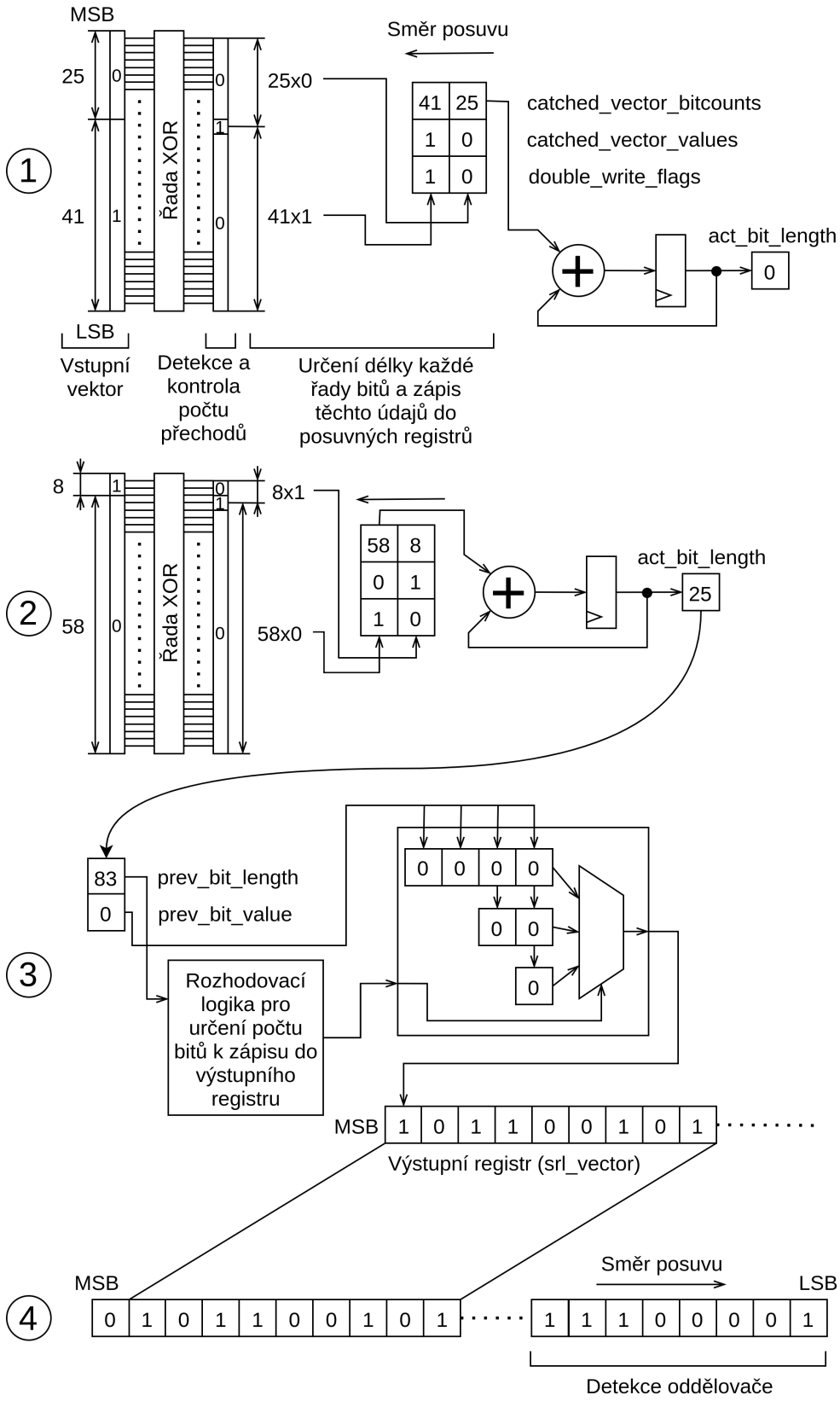
jednoho vstupního slova v jeden časový okamžik lze nalézt buď jeden nebo žádný přechod vyskytující/nevyskytující se v daném slově. Případ, kdy toto není splněno je kontrolován a systém je při jeho výskytu vrácen do výchozího stavu.

Dále jsou zřízeny dva posuvné registry pro záznam ze vstupu. Pokud je ve vstupním slově přítomen přechod, do prvního registru (`catched_vect_bitcounts`) je zaznamenána délka (počet bitů) od bitu LSB k přechodu a od přechodu k MSB v tomto pořadí. Do druhého registru (`catched_vector_values`) jsou potom ukládány hodnoty daných bitů ze spodní a horní poloviny rovněž v tomto pořadí. Pokud není ve vstupním slově přítomen přechod, potom je do každého registru zapsána jen jedna hodnota, tj. délka rovna šířce přijímací sběrnice a hodnota prvního bitu slova. Registry tímto vytváří ze vstupu redukovanou posloupnost dvojic délka a hodnota bitu, se kterými je pracováno současně.

Druhou fází je sčítání délek bitů se společnou hodnotou vyskytující se v řadě za sebou. K tomu slouží právě druhý posuvný registr, kde lze ony řady rozeznat a poté již do sčítačky přesunout patřičné délky z prvního registru. Signál `double_write_flags` slouží pro identifikaci, zda došlo naráz k zápisu dvou hodnot za sebou v předchozích registrech. Zároveň pomáhá odlišit stav, kdy je přechod v přijaté posloupnosti přítomen mezi vektory přijatými ve dvou časových okamžicích po sobě. V opačném případě by totiž došlo k identifikaci přechodu až o hodinový takt později a k současné délce sčítané řady by byla zapsána délka bitu s opačnou hodnotou.

Ze sečtených délek je určen počet bitů, které budou zapsány do koncového posuvného registru (`srl_vector`). K tomu slouží signál `write_vector` určující hodnotu a počet bitů k zapsání. V koncovém registru je vyhledáván oddělovač stránek a při jeho nalezení dojde i k posunutí stránky na výstup. Tento systém umožňuje přijímat i stránky v řadě za sebou, což je standardem vyžadováno. Platnost stránky na výstupu je oznámena nastavením signálu `an_page_captured_int` nastaveným do log.1. Protože je hodnota tohoto signálu dlouhá, je účelně zkrácena v podkomponentě `PULSE_SHORT` na délku jedné hodinové periody.

V komponentě je dále řešen vznik stavu `IDLE_STATE`, který je signalizován stejnojmenným signálem. Podstata řídícího stavového automatu je odvozena od situace, kdy, při dlouhodobě úspěšném provozu, nastane nesplnění podmínky pro počet přechodů ve vstupním slově. V takovém případě je zastaven provoz posuvných registrů pro záznam bitů na vstupu a rovněž registru pro extrakci zakódovaného slova. Po splnění podmínky je spuštěn čítač `idle_counter`, jenž vytváří prodlevu, mimo kterou by mohlo dojít k přijetí předchozího slova, jehož začátek se může nacházet v registru `slr_vector`. Normální stav je obnoven na konci běhu čítače. Signál `IDLE_STATE` plní důležitou roli ve stavech `ACKNOWLEDGE_DETECT` a `NEXT_PAGE_WAIT` komponenty arbiteru.




Obr. 4.8: Princip funkce komponenty poskytující rozhraní s GTY. Každý časový okamžik je zde znázorněn číslem vlevo.

# 5 Simulační výsledky

Zde budou předvedeny výsledky simulací předešle popsaných komponent. K zobrazení je využito programu Modlesim, který poskytuje simulační a verifikační prostředí při návrhu hardware. V případě velkých rozměrů některých obrázků jsou tyto přesunuty do přílohy B.

## 5.1 Proces příjmu stránek

V první demonstraci bude předvedena funkce příjmu stránky. Počátek se nachází v rozhraní s GTY v RX směru, kde bude ukázáno (obrázek B.1) na následujících řadách vektorů (v simulaci jsou vstupem do komponenty značeny jako `RECEIVED_PMA_VECTOR`)[1]:

```
11111111111111111111111111111111111111111111111111111111111111111111
00000000000000000000000000011111111111111111111111111111111111111111
11111111110000000000000000000000000000000000000000000000000000000000
11111111111111111111111111111111111111111111111111111111111111111111
00000000000000000000000000000000000000000000000000000000000000011111
11111111111111111111111111111111111111111111000000000000000000000000
00000000000000000000000000001111111111111111111111111111111111111111
11111111110000000000000000000000000000000000000000000000000000000000
```

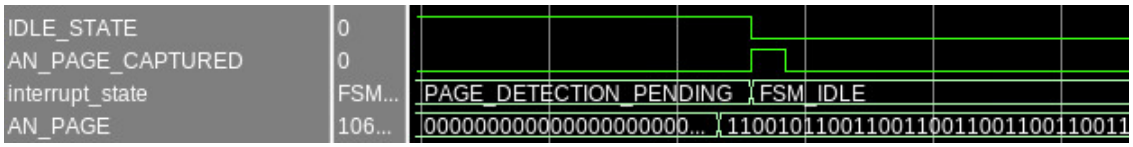
Lze spatřit, že v délkách řad (`cloned_vector_bitcounts`) jsou již z předchozího uloženy hodnoty 4 a 62. Současně je na vstupu první z řady vektorů vypsaných výše. Protože se v předchozím přijatém vektoru nacházel přechod, je rovněž v současném taktu spočítána délka řady bitů, což je následně zapsáno do registru celkových délek bitů (`prev_bit_length`), tj. 163.

Protože v současném vektoru (první z výpisu) není přítomen žádný přechod, je do každého registru zapsána pouze jedna hodnota, tj. počet odpovídající šířce vstupní sběrnice 66 a hodnota log.1. V dalším taktu dochází k přijetí druhého vektoru z výpisu, jenž obsahuje přechod. Do každého registru jsou proto zapsány dvě hodnoty v pořadí od nižší poloviny k vyšší, což zde znamená 39, 27 do prvního a 1, 0 do druhého. Následně je možné v řadě hodnot `catched_vector_values` rozpoznat souvislou hodnotu v řadě za sebou, zde 01 | 11 | 10. Tyto hodnoty představují onu „redukovanou posloupnost bitů“.

[1] Seřazení bitů je vždy zleva od MSB, doprava k LSB. Tato konvence bude v práci dodržována. V případě nutné změny uspořádání řady bitů bude na tuto skutečnost upozorněno.

Jakmile dojde k detekci přechodu, je přičtena (k signálu `act_bit_length`) ještě aktuální délka sledovaného bitu a výsledek v dalším taktu je opět zapsán do `prev_bit_length` (zde 167). Z něj je určen počet bitů k zapsání do výstupního registru `srl_vector` pomocí signálu `write_vector`, kdy hodnota 011 znamená zápis dvou bitů (význam 01) o hodnotě log.1 (hodnota z `prev_bit_value`), za sebou. K povolení zápisu navíc slouží signál `permit_to_write` (není zde zobrazen).

Tento popsaný proces funguje nepřetržitě po celou dobu funkce komponenty nebo dokud nedojde ke vzniku chybového stavu. Pokud dojde k úspěšnému zachycení stránky ve výstupním registru (obrázek B.2) je tato skutečnost na výstupu oznámena signálem `AN_PAGE_CAPTURED`. Současně, v případě přijetí první stránky od restartu komponenty, je vynulován signál `IDLE_STATE`, který slouží pro přenos informace, že integrita příjmu dat byla nějakým způsobem narušena. Hlavní řízení chybových stavů je obstaráno stavovým automatem `interrupt_state`, do jehož výchozí hodnoty (`FSM_IDLE`) je přepnuto během normálního provozu rovněž po přijetí první zakódované stránky (obr. 5.1).




Obr. 5.1: Přepnutí stavového automatu do normálního stavu při přijetí první stránky

### 5.1.1 Dekódování

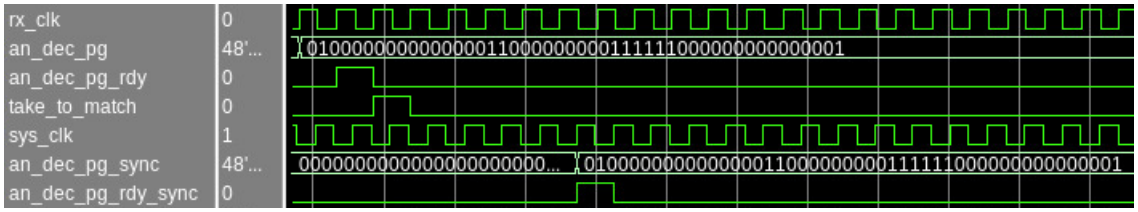
Po přijetí stránky nastává kontrola validity stránky a její dekódování (obr. B.3). Výstupním potvrzovacím signálem z GTY rozhraní je spuštěna kontrola. Při jejím úspěšném dokončení je nastaven signál `an_capt_page_valid`, čímž je zahájeno dekódování. Proces je velmi rychlý a dekodér hned v následujícím taktu vystaví na sběrnici dekódovanou stránku. Konec procesu je oznámen nastavením signálu `an_dec_pg_rdy`. Před asynchronním přechodem probíhá druhá kontrola, zda je počátek stránky složen z jedné z kombinací Selector Field nebo Message Code Field[2]. Shoda je signalizována nastavením `take_to_match`.

### 5.1.2 Asynchronní přechod

Podstatou asynchronního přechodu je synchronizovat signály dvou hodinových domén na náběžnou hranu hodinového signálu na výstupu. Na obrázku 5.2 je pří-

[2]Z každé kontroly plynou i možné chybové stavy, které mohou nastat (viz dále)

klad funkce komponenty `ASYNC_BUS_HANDSHAKE` při přenosu dekódované stránky. Na první pohled lze spatřit, že přechod způsobuje zpoždění signálu, s čímž je třeba počítat. Signálem `take_to_match` je spuštěn mechanismus, který končí vystavením `an_dec_pg_sync` na straně hodinové domény `sys_clk`. Přechod navíc ohlásí tuto skutečnost signálem `an_dec_pg_rdy_sync`, kterým jsou aktivovány následující registry pro hledání shodných stránek.




Obr. 5.2: Demonstrace funkce asynchronního přechodu

### 5.1.3 Hledání shodných stránek

Hledání tří shodných stránek v řadě za sebou probíhá již za asynchronním přechodem ve dvou registrech, jejichž činnost je ukázána na obrázku B.4. Každý registr řídí odpovídající výstup komponenty přijímače, tj. `ability_match` a `acknowledge_match`[3]. Registry jsou doplněny čítači `abl_match_counter` a `ack_match_counter`, jejichž hodnota je inkrementována pokud je přijatý stránka shodná s aktuálním obsahem daného registru (povolovacím signálem čítače je `an_dec_pg_rdy_sync`). Na obrázku je vidět, že hodnota `ability_match` je již dlouho nastavena a přetečení čítače nemá vliv na její hodnotu, protože přijaté stránky jsou stále shodné s obsahem registru. Chování signálu `acknowledge_match` je shodné s předchozím s tím rozdílem, že k funkci jeho odpovídajícího čítače/registru je nutné nastavení bitu ACK a nenulová hodnota Echoed Nonce Field v přijaté stránce. Při každé nalezené shodě je hodnota registru přesunuta na výstup `rx_link_codeword`. Nakonec logika přijímače zkontroluje shodný obsah obou registrů a nastaví výstupní signál `consistency_match`.

### 5.1.4 Vznik chybových stavů

Na začátku příjmu stránky může dojít ke vzniku celkem tří chybových stavů. První z nich je nesplnění podmínky pro počet přechodů ve vstupním vektoru. Simulace reakce komponenty je ukázána na obrázku B.5. Stavový automat je přepnut do stavu

[3]Přestože se jedná o výstupy, jsou zde napsány malými písmeny, protože se v sestavené auto-negociační komponentě jedná o vnitřní signály.

TOO_MUCH_TRANSITION_DETECTED, v němž je, po vynulování signálu too_much_transitions, spuštěn čítač idle_counter. Při konci jeho běhu přechází automat do stavu PAGE_DETECTION_PENDING, kdy je vyčkáváno na zachycení stránky ve výstupním registru a v případě úspěchu je automat znovu přepnut do normálního stavu. Po celou dobu, od detekce nesprávného počtu přechodů, až po zachycení stránky je aktivní signál IDLE_STATE.

Druhý chybový stav lze nalézt v případě, pokud přijatá stránka neprojde kontrolou validity, tj. obsahuje uvnitř sebe více bitů v řadě za sebou, než je povoleno, nebo neobsahuje přechod na místě, kde mají být zakódovány hodiny podle pravidel DME (obr B.8). V takovém případě dochází po určité době k přetečení watchdog čítače, které je signalizováno pomocí integrity_disrupted, což způsobí reset výstupu rozhraní s GTY a registrů pro shody. Délka cyklu čítače je nastavena s dostatečnou prodlevou pro pokrytí krátkodobých výpadků příjmu.

Posledním chybovým stavem je, pokud dekódovaná stránka nezačíná jednou z povolených kombinací pole Selector field nebo Message Code field. V takovém případě je stránka zahozena za dekodérem a dále se již nedostane. Na obrázku B.7 je vidět zastavení činnosti signálu take_to_match, který funguje jako povolovací vstup asynchronního přechodu. Reakce registrů shody a odpovídajících čítačů je nulová. Zakázaná hodnota prvních pěti bitů je přijata v signálu an_dec_pg[4].

## 5.2 Odesílání stránek

Počátek procesu odeslání stránky začíná na vstupu tx_link_codeword, kde je komponentou arbitráže poskytnuta nezakódovaná stránka. Vysílač je, vzhledem ke své jednoduchosti, schopen zpracovat stránku jakékoliv formy, neprovádí žádnou kontrolu. Na obrázku B.6 je znázorněn výstup enc_pg_data kodéru, který má povolovací vstup whole_page_sent. Změna vstupní stránky je provedena pomocí signálu transmit_ack_en, který standardně slouží k zapsání hodnoty ACK bitu do odesílané stránky[5].

Poslední zmiňovaný signál spolu s count_ack_en slouží ke spuštění čítače stránek s nastaveným ACK bitem. Čítač se na konci běhu zastavuje a lze jej resetovat pomocí rst_ack. Jeho funkci ukazuje obrázek 5.3.

V rozhraní s GTY je každý bit naklonován podle linkové rychlosti (viz 4.1.2), naformátován do vektorů o šířce paralelní sběrnice a každý jednotlivě odesílán s periodou hodinového signálu. Ukázka několika naformátovaných vektorů je poskytnuta

[4]Hodnota „11111“ začínající od LSB je rezervovaná kombinace pole Message Code Field. V takovém případě je standardem předepsáno, že musí být stránka ignorována.

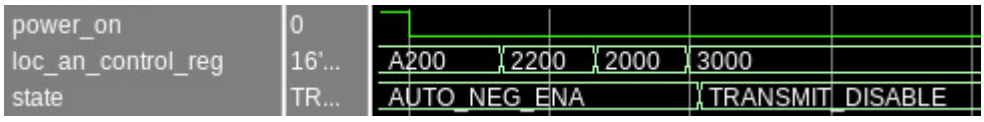
[5]Signál base_page uvedený v obrázku je ve skutečnosti vstupní tx_link_codeword s napojeným bitem 14 na vstup transmit_ack_en.


Obr. 5.3: Funkce čítače ACK stránek

v příloze C. Z výpisu je patrný začátek stránky v podobě delimiteru s dlouhodobě konstantní hodnotou. Následuje střídání hodinových přechodů a bitů zakódované stránky.

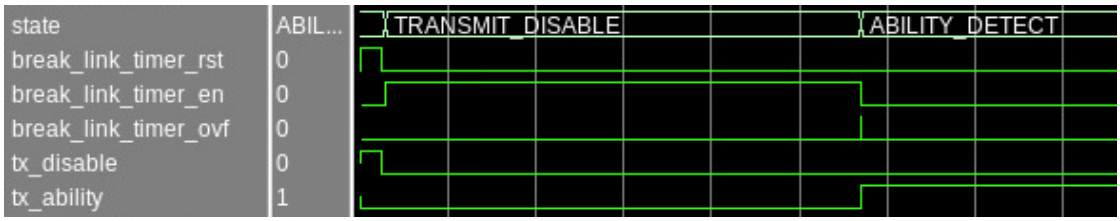
## 5.3 Arbitráž

Stavový automat arbitráže začíná svou činnost po deaktivaci všech resetů a povolením skrze bit 12 v registru `an_control_reg`[6] (pro přehlednost je jeho hodnota uvedena v šestnáctkové soustavě), jak je ukázáno na obrázku 5.4.




Obr. 5.4: Spuštění stavového automatu arbitráže

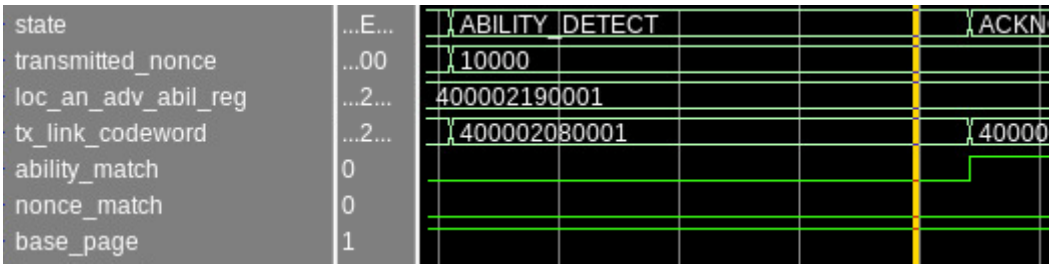
Následně dochází k přepnutí do stavu `TRANSMIT_DISABLE` (obr. 5.10), kde je vysílač resetován (`TX_DISABLE`) a deaktivován (`TX_ABILITY`). Po dobu trvání stavu je spuštěn čítač `break_link_timer`, při jehož přetečení je přepnuto do následujícího stavu.




Obr. 5.5: Chování stavu `TRANSMIT_DISABLE`

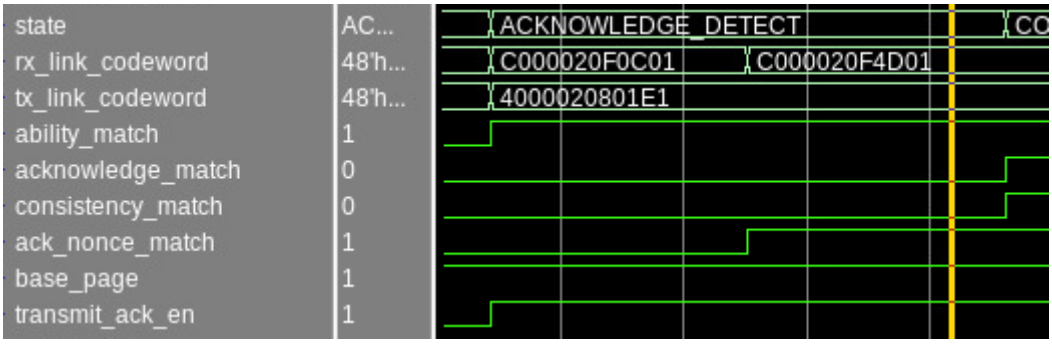
[6] V obrázku jsou uvedeny názvy některých signálů se začátkem `loc_`, resp. `part_`, což upřesňuje, zda se jedná o lokální, resp. protilehlou stranu přenosového kanálu. V rámci zkoumání funkce komponenty bylo přistoupeno k simulaci celého komunikačního systému, tj. s oběma linkovými stranami.

Na obrázku 5.6 je vidět, že při přepnutí do stavu `ABILITY_MATCH` je hodnota LFSR registru (`trasmitted_nonce`) zapsána do výstupu `tx_link_codeword`[7], jehož ostatní hodnoty byly staženy z registru `an_adv_abil_reg` (HEX). Pokud jsou rozdílné hodnoty Transmitted nonce v přijaté stránce (příjem signalizován `ability_match`), tj. `nonce_match` v log.0, je přepnuto do následujícího stavu.




Obr. 5.6: Činnost ve stavu `ABILITY_DETECT`

Přepnutím do stavu `ACKNOWLEDGE_MATCH` (obr. 5.7) je již ověřeno, že zařízení protistrany podporuje funkci auto-negociace. Lze vidět zkopírování pole Transmitted nonce přijaté, do pole Echoed Nonce stránky vysílané (změna signálu `tx_link_codeword` oproti obrázku 5.6). Tento systém „ozvěn“ musí pracovat korektně a splnění správných forem obou polí je signalizováno pomocí `ack_nonce_match`. Do vysílače je vyslán povel k odesílání stránek s nastaveným ACK bitem (`transmit_ack_en`). Po zaregistrování nastavených vstupů `acknowledge_match` a `consistency_match` je povoleno přepnutí do dalšího stavu.

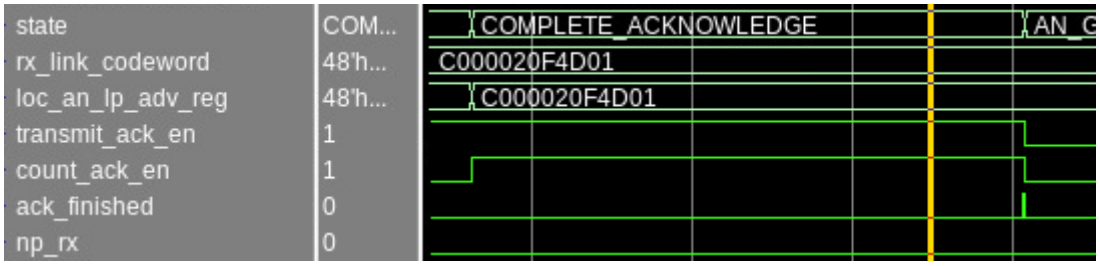



Obr. 5.7: Zápis signálů ve stavu `ACKNOWLEDGE_DETECT`

Při přepnutí do `COMPLETE_ACNKOWLEDGE` (obr. 5.8) poskytuje zařízení protistraně potřebný čas pro přijetí dostatečného množství ACK stránek, čehož je dosaženo povolením čítače ve vysílači (`count_ack_en`) a při jeho přetečení (`ack_finished`) je umožněno stavovému automatu posun dále. Možností je ještě příjem stránek Next

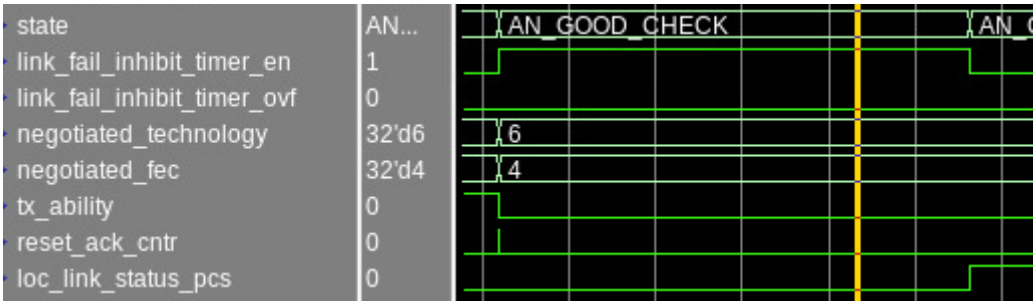
[7]Pro lepší přehlednost zapsáno hexadecimálně, dále jen HEX. Jinak jsou ostatní hodnoty v poskytnutých obrázcích zobrazeny v binárním formátu.

Page, což v poskytnutém obrázku znázorněno není (`np_rx` v log.0 znamená, že jej protistrana nežádá).




Obr. 5.8: Ukázka simulace stavu `COMPLETE_ACKNOWLEDGE`

V předposledním stavu (`AN_GOOD_CHECK`, obr. 5.9) dochází ke spuštění prioritní rezoluční funkce, kdy jsou dohodnuty společné technologie (vyjádřeny jako indexy `negotiated_technology` a `negotiated_fec` v registru 7.48). Činnost vysílače je zde již zastavena a přenosový kanál přepnut do normálního provozu. Současně je spuštěn čítač poskytující prodlevu pro zahájení užitečného provozu po přenosovém kanálu, což je signalizováno z vrstvy PCS (`link_status_pcs`) a, v případě pozitivní indikované hodnoty, přepnuto do dalšího stavu. Pakliže dojde k přetečení čítače nebo nahlášení nekompatibilní dohodnuté technologie, jedná se o chybový stav a celý automat se resetuje (viz dále).

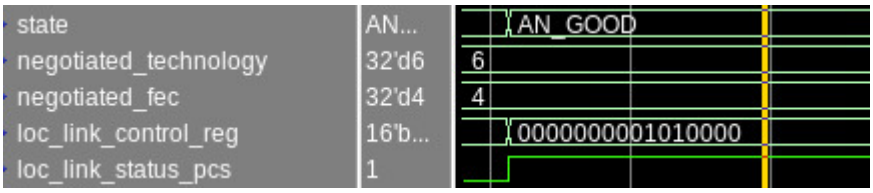



Obr. 5.9: Chování stavu `AN_GOOD_CHECK`

V případě úspěchu končí stavový automat činnost ve stavu `AN_GOOD` (obr. 5.10). Zde jsou zapsány konkrétní bity do registru 7.48 (`link_control_reg`) určené předešle zmíněnými registry. V případě vzniku některé chyby, která způsobí vynulování signálu `link_status_pcs`, je stavový automat resetován, což je jediný případ, jak z posledního stavu vystoupit.

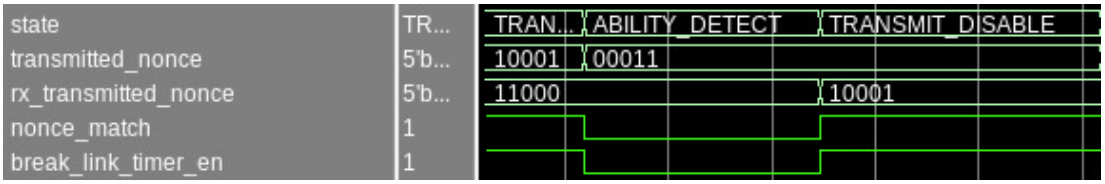
### 5.3.1 Vznik chybových stavů

První možný chybový stav po spuštění stavového automatu dokazuje důležitost, aby úvodní hodnota LFSR generátoru byla určena z náhodného, či pseudonáhod-



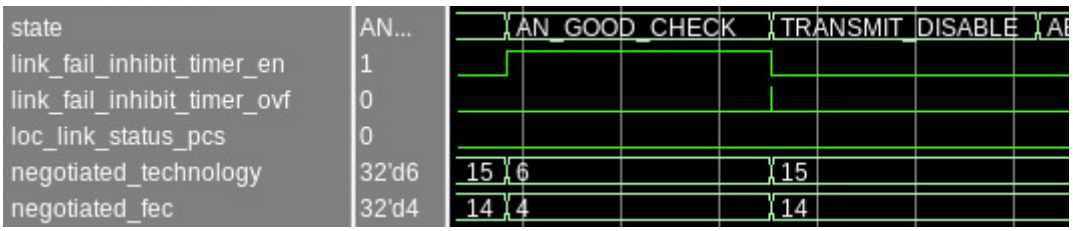

Obr. 5.10: Chování stavu `AN_GOOD`

ného generátoru čísel. Obrázek 5.11 demonstruje vznik takovéto situace, kdy je chyba nahlášena signálem `nonce_match`, pokud jsou rovna pole Transmitted nonce v odesílané a přijaté stránce. V takovém případě je automat přepnut opět do stavu `TRANSMIT_DISABLE` a celý proces se opakuje.




Obr. 5.11: Možný chybový stav při `ABILITY_DETECT`

Druhý možný chybový stav (obr. 5.12) nastává v případě, že není zahájen užitečný provoz na přenosové lince. V takové situaci dochází k přetečení čítače `link_fail_inhibit_timer`, všechny dohodnuté parametry jsou odstraněny, přijaté stránky ignorovány a automat se přepíná do stavu `TRANSMIT_DISABLE`.




Obr. 5.12: Vznik chybového stavu při nenahození provozu

# 6 Implementace a testování

Podle dřívějšího popisu se komponenta pro auto-negociaci nachází mezi vrstvami PCS a PMA. Při zakomponování mechanismu do struktury fyzické vrstvy bylo nutno přistoupit k řadě změn. První spočívala v odstranění kontroly hlaviček při příjmu stránek. Koncept hlaviček, jakožto specifických začátků přenášených jednotek, je přítomen pouze u stránek Base Page a stránek Next Page typu Message. Při zavedení kontroly hlaviček vzniká problém, kdy určitá počáteční kombinace bitů, přicházející v *neformátovaných* Next Page, může být vyhodnocena jako „neplatná hlavička“ a zahozena.

Druhá změna spočívala v odstranění povolovacích vstupů konfiguračních registrů. Implementace ve společnosti CESNET nevyžaduje jejich použití, skutečnost zápisu do registru je možné zjistit například periodickým sledováním změny zapamatované hodnoty.

Klonováním bitů auto-negociačních dat jsou vytvářeny velmi dlouhé sekvence bitů v porovnání s běžně odesílanými daty. Pro správnou funkci mechanismů v GTY transceiverech, jako regenerace hodinového signálu nebo ekvalizace DFE, jsou ovšem nutné velmi časté změny hodnot v přijímaném signálu. Z tohoto důvodu je třeba změnit konfiguraci transceiverů při každém průběhu auto-negociace, mimo stavy `AN_GOOD_CHECK`, resp. `AN_GOOD`, kdy je vyčkáváno na zahájení provozu na lince, resp. kdy je proces vzájemné dohody parametrů již ukončen. Pro tento účel je transceiver přepnut signálem `rxlpmen_in_0` do režimu LPM a zároveň pozastavena automatická regulace zesílení nízkých frekvencí diferenciálního přijímače na vstupu (signál `rxlpmlfhold_in_0`). Vstupem `rxcdrhold_in_0` je zároveň pozastavena funkce regenerace hodinového signálu. Na základě zkoumání jádra *25G Ethernet Subsystem* společnosti Xilinx bylo dále zjištěno, že je nutné rovněž změnit konfiguraci GTY transceiverů pomocí rozhraní DRP. Ovládání je, v tomto případě, zajištěno speciální komponentou, která je převzata ze stejného jádra včetně měněného obsahu v konfiguračních registrech, jejichž význam je společností Xilinx utajen. Transceivery je na závěr potřeba resetovat pomocí vstupu `RXPMARESET`, což je řešeno automaticky v komponentě zajišťující změnu konfigurace pomocí DRP[15, 24].

## 6.1 Testovací komponenty

Jelikož fyzická vrstva nedisponuje žádným rozhraním pro sledování vnitřních signálů, došlo k vytvoření speciálních komponent pro testování. První se nachází uvnitř komponenty pro auto-negociaci (název `debug_core.vhd`) a slouží pro sledování signálů podílejících se na řízení procesů při vzájemné dohodě parametrů. Druhá komponenta se nachází mezi vrstvami PCS a PMA a slouží ke sledování signálů mezi nimi,

případně signálů uvnitř PCS (název `an_pcspma_debug_core.vhd`). Skupina sledovaných signálů je vybrána tak, aby bylo možno co nejlépe zachytit obraz některého z pozorovaných scénářů v průběhu auto-negociace.

Každá testovací komponenta využívá sondu *Integrated Logic Analyzer* (ILA), která je poskytovaná společností Xilinx jako proprietární IP Core. Sonda umožňuje být aplikována na kterékoliv místo v hierarchii FPGA, které umožňuje konfiguraci. Generování je možno provést ve vývojovém prostředí Vivado, kde je specifikováno množství vstupů, jejich bitová šířka, maximální množství ukládaných jednotek pro každý vstup (nezávisle na bitové šířce jednotlivých vstupů), počet komparátorů pro nastavení podmínek spuštění sondy a případně speciální vstupy/výstupy umožňující propojovat a spouštět sondy navzájem mezi sebou. Sond je možno vytvářet na hradlovém poli libovolné množství, ovšem s rostoucí bitovou šířkou a počtem vstupů roste rovněž spotřeba zdrojů, která může v krajním případě vést až k nesplnění podmínek časování obvodu. Počet sond a jejich umístění je proto nutné volit s rozvahou. Sondy není nutné propojovat s vyššími strukturami hradlového pole na externí rozhraní, které by umožnilo jejich sledování pomocí testovacích aplikací v připojeném počítači. Sondy jsou spojeny automaticky, algoritmy vývojového prostředí, do komponenty `DEBUG_HUB`, ke které je připojena aplikace Hardware Manager pro sledování z počítače. Nevýhodou je řízení sond pouze jedním hodinovým signálem, kdy při připojení signálů z více hodinových domén, je nutné jejich přečasování.

Zachycení patřičných signálů pomocí ILA sond je možno provést pomocí několika způsobů. Nejjednodušším je spouštět každou sondu ručně, případně lze vyslat toto spuštění (Trigger) v sérii přes více sond. Druhým způsobem je spuštění sondy automaticky v reakci na nějakou událost způsobenou signálem. Spouštěč lze nastavit pro reakci na hladinu nebo náběžnou, sestupnou, či libovolnou hranu. Pokročilejší formou automatického spuštění je sestavení stavového automatu. Tímto způsobem je možno vytvořit více čekacích stavů pro sledování složitějších událostí. Pro řízení přechodů mezi stavy jsou k dispozici větvení pomoci „if“ konstrukcí, v jejichž podmínkách může být, kromě signálů sondy, využito až čtyřech nastavitelných příznaků nebo až čtyřech čítačů o šířce 32 bitů. Zápis specifických hodnot do registrů v sondách může být dále podmíněn (funkce *Capture Control*)[25]. Počet a použití čítačů nebo větvení je ale omezen, proto bylo přistoupeno k zavedení uživatelských testovacích obvodů do návrhu zařízení a jejich výstupy jsou poté sledovány skrze ILA sondy. Obvody navržené pomocí jazyků pro popis hardware mohou tímto fungovat pro přesné zachycení daných situací.

## 6.2 Signály sledované během auto-negociace

Stavový automat arbitráže obsahuje množství chybových větví, které značí vznik neobvyklé situace během procesu auto-negociace. Vzhledem k obtížnému zachycení všech možných událostí najednou během testování bylo nutno přistoupit k doplnění současného designu o další komponenty. Tímto je testování v reálném světě odlišeno od simulace navrhnutého obvodu, kde je možné zachytit všechny možné události během jediného měření. V případě reálného testování by tato schopnost znamenala enormní spotřebu zdrojů. Přestože není možné v mnoha případech určit přesnou okolnost vzniku chyby, během průběhu auto-negociace, je každý z testovacích prvků, popsaných v následujících odstavcích, určen ke zvýšení pravděpodobnosti úspěšného určení dané okolnosti.

Při každém spuštění sondy je zachycen stav arbitráže, což je nutné při libovolném přerušení obvyklého běhu automatu. Pro orientaci byl vytvořen čítač `arb_state_time_counter`, který poskytuje relativní měřítko pro určení doby strávené v jednotlivých stavech v čase spuštění sondy. Dále je připraven čítač `arb_fsm_reset_count`, jehož hodnota je inkrementována při každém přerušení běžného průběhu stavového automatu. V našem případě je totiž velikost „paměti“ pro data v ILA sondě nastavena na 2048, což stačí jen pro zachycení nejnutnějších událostí během jednoho spuštění sondy, proto je použití čítačů pro měření doby, či množství událostí vhodné.

Jelikož druhá strana komunikačního kanálu je tvořena proprietárním zařízením Dell Z9100-ON, zbývá zkoumat funkci tohoto zařízení pomocí přijímaných dat. Jelikož pole TN tvoří neměnnou kombinaci během jednoho procesu výměny stránek, lze na základě počtu jeho změn určit, kolikrát je protistrana resetována. Pro tento účel je vytvořen čítač `rx_linkcw_tr_nonce_change_counter`, který počítá změnu pole TN přijatých stránek Base Page. Za změnu hodnoty pole není považován přechod z nulové nebo na nulovou hodnotu. Pokud je počet změn pole Transmitted Nonce roven počtu přerušení stavového automatu, odpovídá tato skutečnost jeho normální funkci. K testování byl rovněž zaveden čítač změn pole TN mezi dekódovanými stránkami (pojmenován `rx_dec_pg_tr_nonce_change_cnt`), tj. těmi, které ještě neprošly hledáním třech shodných stránek na výstupu přijímače.

Dále jsou zavedeny dva čítače celkového počtu přijatých/odeslaných stránek (pojmenované `rx_page_counter` a `tx_page_counter`). V případě příliš výrazného rozdílu v jejich hodnotách je možné předpokládat chybnou funkci na jedné nebo druhé straně přenosového kanálu. Hodnota čítače odeslaných stránek je ještě navíc uložena do registru při každém resetu stavového automatu, čítač je vynulován.

Pro přesnější určení počtu specifických stránek jsou přítomny speciální čítače, jejichž hodnota je inkrementována při každém přijmu stránky, která způsobila nastavení `ABILITY_MATCH`, případně `ACKNOWLEDGE_MATCH`. Čítače umožňují vést počty

jak pro stránky Base Page, tak i Next Page. Navíc jsou přítomny ještě ve variantách počítající od počátku měření a počítající přijaté stránky v době mezi dvěma přerušeními chodu stavového automatu, tj. od stavu `TRANSMIT_DISABLE` až do následujícího vzniku chybového stavu. Použití těchto specifických čítačů umožňuje určit, zda nejsou stránky určité skupiny posílány v nadměrně nižším, či vyšším množství. Při přijmu, mezi dvěma po sobě přijatými stránkami Next Page, probíhá navíc počítáni změn bitu Toggle, aby byla ověřena korektní funkce tohoto mechanismu i během testování.

Ve stavu `AN_GOOD_CHECK` je nakonec použit čítač doby, po kterou je vyčkáváno na zahájení užitečného provozu po přenosovém kanálu, jenž je potvrzen signálem `LINK_STATUS_PCS`. Většinu čítačů je možné resetovat pomocí bitu 9 v registru 7.0.

Pro kompletní testování procesu vzájemné výměny parametrů uvnitř komponenty zajišťující auto-negociaci je na závěr vytvořen zachytávač některých signálů během výskytu chybového stavu. Při kterémkoliv vybočení z běžného průběhu stavového automatu, které je oznámeno signálem `ARB_DBG_ABNORMALITY_DETECTED` z komponenty `arbiter.vhd`, dojde k zachycení signálů do přednastavených bufferů, z nichž je cyklicky čteno na rozhraní ILA sondy. Zachycením některých důležitých signálů je možné určit podstatu vzniku chybového stavu, přestože signály od té doby již několikrát změnily svou hodnotu. Druhý typ bufferu je určen pro přijaté stránky s nastaveným ACK bitem, tj. ve stavu `ACKNOWLEDGE_DETECT`. Tyto stránky jsou zachycovány, vzhledem ke svému významu, jako komunikační jednotky používané na konci procesu vzájemné dohody parametrů. Při testování nebyla zjištěna potřeba vytvářet buffer stránek přijatých ve stavu `ABILITY_DETECT`, či těsně po dekódování, protože správná funkce přijmu byla zjištěna z posouzení již přijatých ACK stránek a hodnot výše zmíněných čítačů. Zavedením záchytných bufferů je možno obejít omezené množství hodnot pro uložení do ILA sondy.

## 6.3 Physical Coding Sublayer

Před popisem dalších testovacích scénářů, vznikajících mimo proces auto-negociace, je třeba provést ještě popis vrstvy PCS. Její funkce je, pro rozhraní 25 Gb/s, specifikována v klauzuli 107, standardu 802.3. Většina principů je však převzata z klauzule 49 definující funkcionalitu PCS vrstvy pro rozhraní o rychlostech 10 Gb/s. Vrstva tvoří přechod mezi GTY transceivery, jejichž uzavřený systém umožňuje vytvořit ze sériových dat data paralelní, a MAC vrstvou, která zavádí pojem adres a rámců, coby komplexní komunikační jednotky. PCS jako první v hierarchii fyzické vrstvy zavádí pojem hlavičky, která je v ní zpracována.

Data jsou z vyšších vrstev poskytována pomocí dvou sběrnic, které jsou součástí MII. První je tvořena osmi oktety bitů, jejichž význam je specifikován pomocí

druhé sběrnice o šířce 8 bitů. Bity zde tvoří masku, kde nastavení do log. 1 značí oktet obsahující kontrolní sekvence a log. 0 oktet obsahující datové sekvence. Mezi základními kontrolními sekvencemi lze nalézt uvození *Start* a zakončení *Terminate*, které jsou nutné pro ohraničení datových sekvencí, dále *Idle* zasílané v situaci, kdy daná strana nemá data k odeslání, a *Error*, které jsou nastaveny v případě chyby v běžném průběhu stavového automatu přijímací nebo vysílací části PCS. Příkladem chybového stavu je opomenutí sekvence Start, či Terminate při začátku/konci datových sekvencí.

Na vstupu PCS je obsah obou sběrnic vložen do kodéru, kde skupina oktetů obsahující pouze datové sekvence není překódována vůbec a výstupní blok o šířce 64 bitů je doplněn hlavičkou 01 od LSB. Pokud data na MII obsahují alespoň jednu kontrolní sekvenci, jsou zakódována pomocí speciální pravidel kódování 64b/66b. Kontrolní blok je na začátku specifikován polem *Block Type Field* a kontrolní sekvence, s výjimkou Start a Terminate, překódovány na délku 7 bitů. Takto zakódovaný blok je nakonec doplněn hlavičkou 10. Dále jsou odesílaná data bez hlavičky převedena přes skrambler na pseudonáhodnou posloupnost sloužící pro lepší rozprostření spektra přenášeného signálu a bezchybnou činnost následující DFE. V posledním kroku jsou takto „rozmíchaná“ data směřována ke GTY transceiverům[18].

Na přijímací straně dochází k dekódování podle převrácených principů popsaných v předchozím odstavci. Navíc je zde zavedena kontrola zarovnání bloků na začátek sběrnice. Z principu GTY transceiverů totiž plyne, že vlivem zkreslení v přenosovém kanálu, či jeho vlastnostem (například délce), může dojít ke znásobení nebo zmenšení počtu bitů na vstupu GTY transceiverů v RX směru, což může vést k situaci, kdy hlavičky přicházejících bloků nebudou, na sběrnici mezi PCS a GTY, zarovnány na bit 0. Následné posunutí začátku bloku může nepříjemně ovlivnit integritu dat ve vyšších vrstvách, zejména tam, kde je zarovnání datových jednotek podle hlaviček kritické. V kontrole PCS vrstvy je proto, při sestavování spojení, sledována pozice hlavičky a v případě jejího posunutí, je signalizováno GTY transceiverům posunutí přijímaných bloků o jeden bit (vstup transceiveru `RXGEARBOXSLIP`). Komponenta transceiverů, jež zajišťuje tuto funkci, nese název *Gearbox*. Proces posouvání je opakován až doby zarovnání blokových hlaviček na začátek přijímací sběrnice, jež je oznámeno signálem `rx_block_lock_0`[15, 18].

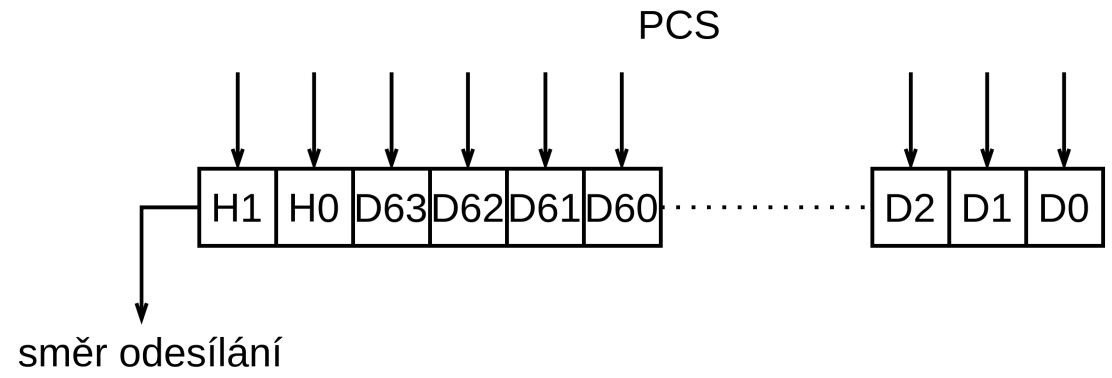
Následně je započata kontrola přicházejících hlaviček. Podle pravidel kódování je totiž možné nalézt hlavičky ve formátu pouze 10 nebo 01, hodnoty 11 a 00 jsou brány jako nepovolené[18]. V případě nalezení velkého množství nepovolených hlaviček (zjištěno pomocí čítače), je nastaven signál `hi_ber_0` do log. 1. Nízký počet nepovolených hlaviček a korektní zarovnání bloků přicházejících dat je znakem úspěšného zahájení síťového provozu po síťovém kanálu, což je očekáváno při ukončování procesu auto-negociace. V případě nalezení velkého množství nepovolených hlavi-

ček probíhá opětovné hledání pozice hlavičky, včetně jejího případného zarovnávání s okrajem sběrnice.

Vrstva PCS rovněž řeší vznik dvou hodinových domén podobně jako komponenta pro auto-negociaci. K vyrovnání rozdílu je použita fronta FIFO asynchronního typu v přijímacím i vysílacím směru[24].

## 6.4 Signály sledované mimo auto-negociaci

Následující odstavce popisují sledované chování komponent PCS a GTY transceiverů během čekání na zahájení užitečného provozu (stavy `AN_GOOD_CHECK` a `AN_GOOD`). Jelikož je k přenosu auto-negociačních dat použita i sběrnice pro hlavičky, docházelo by k falešnému počítání nepovolených hlaviček a zbytečnému zarovnávání přijímaných dat. Proto je deaktivována funkce jejich posuvu v komponentě Gearbox. Odesílání i příjem dat je zde navíc otočen, kdy první jsou odeslány/přijaty bity 1 a 0 hlavičky následované datovými bity 63 až 0. Princip je ukázán na obrázku 6.1 (analogicky platí i pro data v přijímacím směru)[15]. Bitové převrácení sběrnic je řešeno na vstupu/výstupu komponenty pro auto-negociaci.




Obr. 6.1: Princip převrácení odesílaných dat v komponentě Gearbox

Vzhledem k tomu, že signál `pcs_link_status` je funkcí signálů `rx_block_lock_0` a `hi_ber_0`, je, v sondě zachycující procesy mezi PCS a GTY, jejich chování sledováno odděleně. K tomuto postupu bylo přistoupeno z důvodu nekauzality obou signálů, kdy zarovnání přicházejících bloků k okraji sběrnice neznamená vždy příjem i dostatečného množství povolených hlaviček.

Při resetování transceiverů je nutné počkat určitou dobu na opětovné nastartování všech jeho vnitřních systémů. Tato skutečnost je následně oznámena nastavením výstupu `RXPMARESETDONE` do log. 1, kdy transceivery vykazují již normální funkci.

Resetovací vstupy samotných transceiverů jsou realizovány jako asynchronní, ale jejich řízení je obstaráno pomocí stavového automatu umístěného v „obalové“ komponentě společně s GTY. Komponenta rovněž umožňuje reset všech bloků transceiveru pomocí zvláštního vstupu nebo reset jeho specifických částí, jak je provedeno v tomto případě. Resetovací vstupy vyvedené z obalu transceiveru jsou tedy, vlivem zavedení stavového automatu, realizovány jako synchronní[15]. Do ILA sondy jsou zavedeny signály dokončení resetů jak z přijímacího, tak i odesílacího směru, společně s danými resetovacími vstupy. Jelikož je možné provést reset GTY i z vyšších vrstev, je třeba sledovat rovněž funkci těchto vstupů. Navíc je přiveden signál `drp_done`, který značí dokončení změny konfigurace DRP registrů a slouží zároveň jako reset funkce transceiverů[24]. Nastane-li situace, že některý z procesů auto-negociace začíná během probíhajícího resetu, je nutné tuto situaci vyřešit buď vyčkáváním na jeho dokončení, či zvážit, zda případné zahazování vysílaných dat netvoří problémy.

V případě, že při zahajování provozu dojde k přijmu nevhodných kontrolních sekvencí, či přijetí v nesprávném pořadí, je doplňkově do sondy připojen signál `stat_rx_bad_code_0` z PCS, jenž je nastaven do log. 1 s každým vstoupením přijímacího stavového automatu do stavu ERROR. Dále je sledováno chování výstupu `rx_valid_ctrl_code_0`, který je odvozen z předchozího. Nastavení do log. 1 značí kontinuální příjem kontrolních sekvencí ve správném pořadí během určité doby[24]. Pro zkoumání přijímaných 64/66b sekvencí byl přiveden i výstup deskrambleru, kde je možné zachytit třeba začátky, či konce datových bloků nebo sekvence Idle. Nakonec je přiveden výstup podkomponenty Gearbox `RXHEADERVALID`, kde log. 1 značí platnou hodnotu hlavičky na výstupu `RXHEADER`[15]. Výběr těchto signálů byl zvolen zejména pro utvoření širšího obrazu o procesech probíhajících mezi vrstvou PCS a GTY transceivery.

## 6.5 Výsledky testování

Při zavedení komponenty pro řízení transakcí přes DRP rozhraní do návrhu společnosti CESNET bylo vypozorováno neustálé resetování GTY transceiverů během čekací doby ve stavu `AN_GOOD_CHECK`, které zpozdilo jejich spuštění až do doby konce čekání, kdy byl již stavový automat restován. Tento problém byl vyřešen přidáním podmínky, která umožňovala komponentě provést konfiguraci a reset transceiverů pouze jednou za změnu stavu arbitráže.

Původní návrh nebyl zcela soustředěn na zasílání stránek typu Next Page, avšak obě proprietární zařízení na protistraně vyžadovala jejich přenos. Proto byl systém jejich přenosu více ověřen v simulacích a následně i testován. U obou proprietárních zařízení tvořil obsah přicházejících stránek identifikátor OUI s rozsahem přes dvě zaslané stránky, jednu typu Message a druhou Unformatted.

Jakmile bylo dosaženo korektního přenosu stránek Base Page i Next Page, byla pozornost soustředěna na chování signálů mezi PCS a GTY transceiverem. Právě zahájení užitečného provozu po síťovém kanálu, spočívající v detekci dostatečného množství povolených hlaviček a zarovnání bloků přicházejících dat s bitem 0 na přijímací sběrnici, je poslední podmínkou k úspěšnému dokončení procesu auto-negociace. Z reálného testování však bylo dosáhnuto smíšených výsledků. Nastavení signálu `LINK_STATUS_PCS` do log. 1 sice dokázalo udržet chvilkovou stabilitu, avšak po chvíli byla jeho hodnota opět v log. 0. Mezi pozorovanými scénáři byl výskyt jak nepravidelného střídání logických hodnot tohoto signálu, tak i jeho celková neaktivita. V poslední řadě byly sledovány sekvence přicházející z deskrambleru, zda neobsahují některé ze specifikovaných kontrolních sekvencí. Pakliže by neprobíhala žádná komunikace, bylo by možno zachytit v přijatých bitech IDLE sekvence, zasílané mj. pro udržení stability vnitřních systémů GTY transceiverů připravených k přenosu dat. Tato situace však nenastala a dosavadní přijaté sekvence mají pseudonáhodný charakter. V současném stavu není možné zjistit příčinu nezahájení užitečného provozu po síťovém kanálu.

Z pozorování přicházejících pseudonáhodných sekvencí byly vytvořeny dvě domněnky, které by mohly stát za jejich vznikem. První tvrdí, že protistrana provádí test a přizpůsobování vlastností pomocí funkce *Link Training*, která je definována v kapitole 72.6.10, standardu 802.3. V tomto případě je nutné implementovat danou funkci na obou stranách přenosového kanálu současně s mechanismem ladění GTY transceiverů pro přizpůsobení jeho parametrů vlastnostem přenosového média. Druhá domněnka tvrdí, že protistrana provádí test přenosového média pomocí pseudonáhodných, případně PRBS9 nebo PRBS31 (Pseudorandom Binary Sequence o délce 9 nebo 31 bitů), sekvencí, pro jejichž generování je využito skrambleru. Na přijímací straně potom dochází k ověřování platnosti daných testovacích sekvencí na výstupu deskrambleru. Tento mechanismus je definován v klauzuli 49, standardu 802.3, jako funkce vrstvy PCS. Protože současný návrh této vrstvy u společnosti CESNET neobsahuje ani generátor, ani tester pseudonáhodných sekvencí, byla by nutná jeho modifikace.

K vyvrácení, či potvrzení předchozích domněnek bylo závěrečné pozorování soustředěno na pokusy s využitím jiných komponent, než která byla zde vytvořená. První zahrnoval využití neupraveného IP Core 25G Ethernet Subsystem. Společností Xilinx byla zadarmo poskytnuta časově omezená licence, během níž je možné otestovat funkci daného systému jak v simulacích, tak při testování na reálném zařízení, kdy je však jeho činnost po uplynutí určité doby automaticky zastavena. Pozornost byla věnována spuštění funkce auto-negociace a její následné chování během simulací při spojení dvou IP Core, jakožto dvou stran přenosového kanálu. Výsledky simulace však nepotvrdily zasílání pseudonáhodných sekvencí. Zároveň, ačkoliv funkce Link

Training je v poskytnutém IP Core implementována, není nutné pomocí ní provádět ladění vlastností GTY transceiverů, čili není nutná pro bezchybný průběh auto-negociačního procesu. Při simulacích však bylo zjištěno odlišné chování komponenty pro rekonfiguraci pomocí DRP, jejíž činností nebylo prováděno neustálé resetování GTY transceiverů během stavu `AN_GOOD_CHECK` a její funkce byla tímto posouzena jako správná.

Reálné testování komponenty se specifikovaným IP Core však provedeno nebylo. Vzhledem k jí poskytnutým vstupům není možné sestavit dostatečně podrobné množství signálů pro sledování chování komponenty sondami ILA. Jednou z možností řešení tohoto problému je vepsat instance sond přímo do zdrojového kódu tohoto Ethernetového jádra, jež umožňují snímat chování signálů v těsné blízkosti auto-negociační komponenty, případně komponenty reprezentující vrstvu PCS (samotné zdrojové kódy těchto komponent jsou poskytnuty v zašifrované formě). Následně je možné, tímto způsobem, otestovat funkcionalitu dvou proprietárních zařízení s poskytnutým rozhraním, pro sledování fyzické vrstvy v jednom z nich. Nevýhodou tohoto postupu je možné smazání vepsaného kódu při vygenerování nových zdrojových souborů.

V poslední řadě byla, po dlouhé době, vyzkoušena komunikace se serverem Spirent. Starší verze testovacího prostředí umožňovala provozovat funkci auto-negociace i na linkách tvořených optickými kabely. Nová verze však zakazuje použití funkce auto-negociace na těchto typech médií a povoluje pouze při použití metalického DAC kabelu. Během komunikace bylo zaznamenáno několik chyb již při výměně stránek Base Page. Povaha tohoto problému nebyla přesně určena zejména z důvodu předchozí funkčnosti výměny parametrů při komunikaci se switchem Dell. Poslední pokus zahrnoval implementaci stejného designu od společnosti CESNET na dvě stejné, navzájem propojené karty. Přestože sestavení spojení nečinilo problémy, nastaly opět komplikace při procesu vzájemné dohody parametrů. Druhý problém činilo zajištění jedinečné kombinace soli pro LFSR registru, z něhož je generována hodnota pole TN. V opačném případě dochází k častému resetování stavového automatu ve stavu `ABILITY_DETECT` kvůli detekci totožných hodnot těchto polí v přijatých a odeslaných stránkách. Testováním na dvou programovatelných síťových kartách lze však ověřit i funkčnost zde vytvořeného návrhu s protistranou tvořenou IP Core od Xilinx nebo dvou návrhů, kde každý je tvořen IP Core. Výhodou této varianty je možnost sledovat chování signálů na obou stranách přenosového kanálu.

# Závěr

Tato práce se věnovala implementaci funkce auto-negociace podle standardu 802.3 na hradlových polích. V úvodu byl představen kompletní přehled hradlových polí řady UltraScale+ společnosti Xilinx. Důraz zde byl zvláště kladen na popis GTY transceiverů, které tvoří nejvýznamnější součást těchto čipů. Jejich přizpůsobení pro vysoké pracovní frekvence je dosaženo pomocí vylepšených PLL, funkci ekvalizace a hlavně díky technologii *Stacked Silicon Interconnect* (SII) umožňující spojování více čipů pomocí mikropropojů umístěných na mezisubstrátu připevněném na substrát pouzdra *Ball Grid Array* (BGA).

Ve třetí kapitole byl poskytnut potřebný teoretický úvod pro pochopení funkce auto-negociace. Představen byl standard 802.3-2018, konkrétně klauzule 73, která se zabývá danou funkcí pro provoz na linkách o rychlostech 25–100 Gb/s. Detailnímu popisu byl podroben mechanismus a sestavení stránek jako komunikačních jednotek pro auto-negociaci. Další části klauzule jsou zde detailně okomentovány, případně přeuspořádány pro lepší začlenění do kontextu vývoje.

Následující kapitoly se zaobírají již samotným návrhem, jenž byl zpracován v jazyce VHDL. Zde docházelo k občasné úpravě popsaných mechanismů, aby bylo umožněno jejich začlenění do struktury hradlových polí. Velké změny se týkaly zejména přesunutí auto-negociační komponenty v hierarchii fyzické vrstvy a z toho plynoucí odlišná funkce komponent vysílače a přijímače. Výsledná složitost vysílače plyne z nutnosti naklonovat bity zakódovaného slova podle rychlosti vysílacího rozhraní a následně dělit v každém časovém okamžiku tyto sekvence na řady o délce rovné šířce vysílací sběrnice. Složitost komponenty přijímače vyplývá z potřeby zachytit, na prvním místě, sekvence naklonovaných bitů v blocích přicházejících dat a nalézt přechody mezi nimi. Následně jsou vypočteny délky těchto sekvencí na jejichž základě je potom zapsán správný počet bitů do výstupního registru pro zachycení přijatého slova, které je následně dekódováno. Drobné úpravy se dále týkaly i stavového automatu arbitráže. Jelikož při dokončení auto-negociace probíhá volba parametrů rozhraní na obou stranách, jsou zde dále popsány rezoluční funkce pro dohodu linkové rychlosti a schopnosti provádět FEC.

Po navržení komponenty bylo přistoupeno k simulaci všech popsaných mechanismů, aby demonstrovaly korektní funkcionalitu popsanou v kapitolách 3 a 4. Simulační výstupy byly ořezány od nadbytečností a v jednoduchosti předvedeny, případně doplněny potřebným komentářem. Simulace byla nutným prekurzorem k ověření funkce před samotným testováním, kde znamenala velkou úsporu času vzhledem k jinak velmi dlouhým dobám potřebným pro výpočty při spojování a umisťování implementovaného návrhu.

V poslední kapitole byla navržená komponenta podrobena testování na reálné

síťové kartě s hradlovým polem. Pro zkoumání vnitřních signálů hradlového pole s implementovaným designem, byly vytvořeny dvě testovací komponenty, z nichž každá využívá sondu ILA poskytovanou společností Xilinx k těmto účelům. Při úvodním testování byla zjištěna chyba v přijmu i vysílání auto-negociačních dat, způsobená konfigurací transceiverů pro funkci na velmi vysokých rychlostech. Vzhledem k účelnému zpomalení auto-negociačních dat bylo tedy GTY transceivery třeba překonfigurovat pomocí rozhraní DRP.

Proces auto-negociace dále doprovázely komplikace spočívající v neustálém ustávání přenosu při přijmu stránek Next Page. Po dlouhém bádání bylo zjištěno, že dané zahazování přicházejících stránek bylo způsobeno příliš selektivní kontrolou hlaviček stránek. Přenos byl, v tomto případě, realizován s protistranou reprezentovanou switchem Dell Z9100-ON.

Následně byla vysvětlena funkce PCS vrstvy, jejíž vnitřní signály slouží k oznámení úspěšného zahájení užitečného provozu po síťovém kanálu. Právě tento krok je poslední podmínkou k úspěšnému dokončení procesu auto-negociace. Z jejich testování však bylo dosáhnuto smíšených výsledků. Nastavení signálu `LINK_STATUS_PCS` do log. 1 sice dokázalo udržet chvilkovou stabilitu, ovšem výjimkou nebyly ani situace kmitající hodnoty tohoto signálu, či jeho úplné neaktivity. Přijímané sekvence na výstupu deskrambleru měly pseudonáhodný charakter, kolem jejichž původu vyvstaly dvě domněnky. První tvrdí, že protistrana provádí test přenosového kanálu pomocí funkce Link Training, druhá tvrdí, že testování provádí komponenta PCS na protistraně pomocí pseudonáhodných, či PRBS sekvencí.

Vzhledem k vytvoření předchozích domněnek spočívalo závěrečné pozorování v pokusech s využitím jiných komponent, kde jedna například zahrnovala použití neupraveného IP Core 25G Ethernet Subsystem. Funkce auto-negociace zde byla úspěšně vyzkoušena v simulacích v zapojení dvou IP Core proti sobě. V tomto případě byly vyvráceny předchozí dvě domněnky. Při provozu auto-negociace neprobíhá zasílání pseudonáhodných sekvencí, ani není nutné spuštění funkce Link Training. Reálné testování komponenty se specifikovaným IP Core však provedeno nebylo.

Přes dosažené výsledky nebylo možné stabilně zprovoznit funkci auto-negociace ani v jednom z testovaných případů. Všechny návrhy k pokusům zde otestovány nebyly, avšak jsou námětem k dalšímu zkoumání. Způsob testování pomocí samostatných komponent, implementovaných v designu, může být ale využit i při zkoumání jiných mechanismů v programovatelném hardware.

# Literatura


[1] REFERENCE FOR BUSINESS. Xilinx, Inc.: Company Profile, Information, Business Description, History, Background Information on Xilinx, Inc. *Reference for Business* [online]. USA: Reference for Business, ©2021 [cit. 06. 04. 2021]. Dostupné z: <https://www.referenceforbusiness.com/history2/89/Xilinx-Inc.html>

[2] COMPANIESHISTORY.COM. Xilinx, Inc. history, profile and history video. *CompaniesHistory.com* [online]. Rumunsko: Dektel Solutions, ©2020 [cit. 10. 10. 2020]. Dostupné z: <https://www.companieshistory.com/xilinx/>

[3] XILINX, INC. XC2000 Logic Cell Array Families. XILINX, INC. *Xilinx* [online]. USA: Xilinx, ©2020 [cit. 28. 05. 2021]. Dostupné z: <https://media.digikey.com/pdf/Data%20Sheets/Xilinx%20PDFs/XC2000%20Families.pdf>

[4] DUN & BRADSTREET. Altera Corporation. *Dun & Bradstreet*, Inc. [online]. USA: Dun & Bradstreet, ©2021 [cit. 06. 04. 2021]. Dostupné z: <https://www.dnb.com/business-directory/company-profiles.altera_corporation.127902f44135c2e404e501d990d89a43.html>

[5] XILINX, INC. *Xilinx* [online]. USA: Xilinx, ©2020 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/>

[6] XILINX, INC. What is an FPGA? *Xilinx* [online]. USA: Xilinx, ©2020 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/products/silicon-devices/fpga/what-is-an-fpga.html>

[7] XILINX, INC. DS890: UltraScale Architecture and Product Data Sheet: Overview. *Xilinx* [online]. USA: Xilinx, ©2020 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/data_sheets/ds890-ultrascale-overview.pdf>

[8] XILINX, INC. WP380: Xilinx Stacked Silicon Interconnect Technology Delivers Breaktrhrough FPGA Capacity, Bandwidth, and Power Efficiency. *Xilinx* [online]. USA: Xilinx, ©2020, 11. 12. 2012 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/white_papers/wp380_Stacked_Silicon_Interconnect_Technology.pdf>

[9] XILINX, INC. UG474: 7 Series FPGAs Configurable Logic Block: User Guide. *Xilinx* [online]. USA: Xilinx, ©2020, 27. 9. 2016 [cit. 10. 10. 2020]. Dostupné z:

<https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf>

[10] XILINX, INC. UG574: UltraScale Architecture Configurable Logic Block: User Guide. *Xilinx* [online]. USA: Xilinx, ©2020, 28. 2. 2017 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/user_guides/ug574-ultrascale-clb.pdf>

[11] XILINX, INC. UG573: UltraScale Architecture Memory Resources: User Guide. *Xilinx* [online]. USA: Xilinx, ©2020, 18. 8. 2020 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/user_guides/ug573-ultrascale-memory-resources.pdf>

[12] XILINX, INC. WP464: PCI Express for UltraScale Architecture-Based Devices. *Xilinx* [online]. USA: Xilinx, ©2020, 30. 6. 2015 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/white_papers/wp464-PCIe-ultrascale.pdf>

[13] XILINX, INC. UG570: UltraScale Architecture Configuration: User Guide. *Xilinx* [online]. USA: Xilinx, ©2020, 28. 7. 2020 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/user_guides/ug570-ultrascale-configuration.pdf>

[14] XILINX, INC. UG579: UltraScale Architecture DSP Slice: User Guide. *Xilinx* [online]. USA: Xilinx, ©2020, 22. 9. 2020 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/user_guides/ug579-ultrascale-dsp.pdf>

[15] XILINX, INC. UG578: UltraScale Architecture GTY Transceivers: User Guide. *Xilinx* [online]. USA: Xilinx, ©2020, 20. 9. 2017 [cit. 10. 10. 2020]. Dostupné z: <https://www.xilinx.com/support/documentation/user_guides/ug578-ultrascale-gty-transceivers.pdf>

[16] HAMEL, John Starr, NORRIS, Ryan, ed. LC Tank Voltage Controlled Oscillator Tutorial. *PLD Guru* [online]. University of Waterloo, Ontario, Kanada: John Starr Hamel, ©2005 [cit. 31. 10. 2020]. Dostupné z: <http://pld.guru/_hdl/2/-asic.uwaterloo.ca/files/vcotut.pdf>

[17] IEEE COMPUTER SOCIETY. *IEEE Standard for Ethernet* [online]. 2018. Std 802.3-2018 (Revision of IEEE Std 802.3-2015). USA: IEEE, 2018, 31.8.2018 [cit. 01. 11. 2020]. ISBN 978-1-5044-5090-4. Dostupné z: <https://doi.org/10.1109/IEEESTD.2018.8457469>

[18] IEEE COMPUTER SOCIETY. 107. Physical Coding Sublayer (PCS) for 64B/66B, type 25GBASE-R. IEEE COMPUTER SOCIETY. *IEEE Standard for Ethernet* [online]. 2018. Std 802.3-2018 (Revision of IEEE Std 802.3-2015). USA: IEEE, 2018, 31.8.2018, s. 4781–4785 [cit. 01. 11. 2020]. ISBN 978-1-5044-5090-4. Dostupné z: <https://doi.org/10.1109/IEEESTD.2018.8457469>

[19] IEEE COMPUTER SOCIETY. 110. Physical Medium Dependent (PMD) sublayer and baseband medium, type 25GBASE-CR and 25GBASE-CR-S. IEEE COMPUTER SOCIETY. *IEEE Standard for Ethernet* [online]. 2018. Std 802.3-2018 (Revision of IEEE Std 802.3-2015). USA: IEEE, 2018, 31.8.2018, s. 4826–4827 [cit. 01. 11. 2020]. ISBN 978-1-5044-5090-4. Dostupné z: <https://doi.org/10.1109/IEEESTD.2018.8457469>

[20] IEEE COMPUTER SOCIETY. 73. Auto-Negotiation for backplane and copper cable assembly. IEEE COMPUTER SOCIETY. *IEEE Standard for Ethernet* [online]. 2018. Std 802.3-2018 (Revision of IEEE Std 802.3-2015). USA: IEEE, 2018, 31.8.2018, s. 3156–3182 [cit. 01. 11. 2020]. ISBN 978-1-5044-5090-4. Dostupné z: <https://doi.org/10.1109/IEEESTD.2018.8457469>

[21] IEEE COMPUTER SOCIETY. Physical Layer. IEEE COMPUTER SOCIETY. *IEEE Standards for Local Area Networks: Token Ring Access Method and Physical Layer Specifications* [online]. 1989. Std. 802.5-1989. USA: IEEE, 1998, s. 65–66 [cit. 04. 11. 2020]. ISBN 1-55937-012-2. Dostupné z: <https://doi.org/10.1109/IEEESTD.1989.108547>

[22] IEEE COMPUTER SOCIETY. 45. Management Data Input/Output (MDIO) Interface: 45.2.7 Auto-Negotiation registers. IEEE COMPUTER SOCIETY. *IEEE Standard for Ethernet* [online]. 2018. Std 802.3-2018 (Revision of IEEE Std 802.3-2015). USA: IEEE, 2018, 31.8.2018, s. 2111–2126 [cit. 01. 11. 2020]. ISBN 978-1-5044-5090-4. Dostupné z: <https://doi.org/10.1109/IEEESTD.2018.8457469>

[23] CABAL, Jakub. Jednotky pro asynchronní přechody v obvodech FPGA. Brno, 2015. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. Vedoucí práce Ing. Marek Bohrn.

[24] XILINX, INC. PG210: 10G/25G High Speed Ethernet Subsystem v3.3: Product Guide. XILINX, INC. *Xilinx* [online]. USA: Xilinx, ©2020, 16.12.2020 [cit. 28. 05. 2021]. Dostupné z: <https://www.xilinx.com/support/documentation/ip_documentation/xxv_ethernet/v3_3/pg210-25g-ethernet.pdf>

[25] XILINX, INC. Vivado Design Suite User Guide: UG908: Programming and Debugging. XILINX, INC. *Xilinx* [online]. USA: Xilinx, ©2020, 3. 7. 2020 [cit. 28. 5. 2021]. Dostupné z: <https://www.xilinx.com/support/documentation/sw_manuals/xilinx2020_1/ug908-vivado-programming-debugging.pdf>

# Seznam symbolů, veličin a zkratek

**ACAP** Adaptive Computation Acceleration Platform

**ACK** Acknowledge

**ASIC** Application Specific Integration Circuit

**ASMBL** Advanced Silicon Modular Block

**BGA** Ball Grid Array

**BP** Base Page

**BRAM** Block RAM

**CCL** CMOS Configuration Latch

**CLB** Configurable Logic Block

**CTLE** Continuous Time Linear Equalization

**DFE** Decision Feedback Equalization

**DME** Differential Manchester encoding

**DRP** Dynamic Reconfiguration Port

**DSP** Digital signal processing

**EN** Echoed Nonce

**FEC** Forward Error Correction

**FIFO** First-in/first-out

**FPGA** Field Programmable Gate Array

**HBM** High Bandwidth Memory

**ILA** Integrated Logic Analyzer

**IP** Intellectual Property

**ISO/OSI** International Standardization Organization/Open System Interconnection

**LFSR** Linear Feedback Shift Register

**LP** Link Partner

**LPM** Low-power mode

**LSB** Least Significant Bit

**MAC** Media Access Control

**MCAP** Media Configuration Access Point

**MDI** Media Dependent Interface

**MIB** Management Information Base

**MII** Media Independent Interface

**MP** Message Page

**MPSoC** Multiple Processors System-on-Chip

**MSB** Most Significant Bit

**NP** Next Page

**OUI** Organizationally Unique Identifier

**PCB** Printed circuit board

**PCS** Physical Coding Sublayer

**PLL** Phase-locked Loop

**PMA** Physical Medium Attachement

**PMD** Physical Medium Dependent

**PRBS** Pseudorandom Binary Sequence

**RF** Remote Fault

**RFSoC** Radio Frequency System-on-Chip

**SDP** Simple dual-port

**SII** Stacked Silicon Interconnect

**SIMD** Single Instruction Multiple Data

**SLR** Super-logic region

**SoC** System-on-Chip

**TDP** True dual-port

**TN** Transmitted Nonce

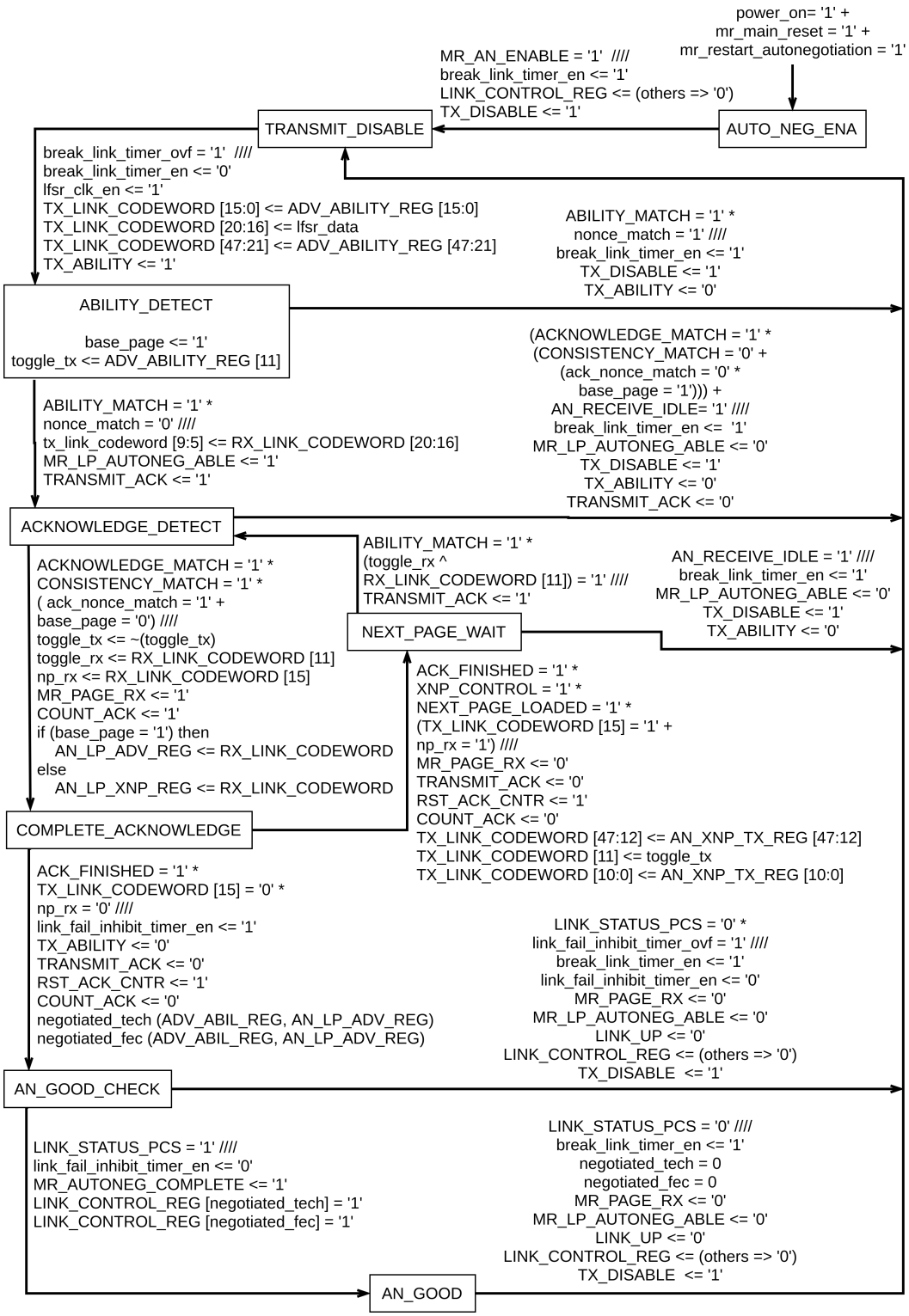
**TSV** Through Silicon Via

**URAM** Ultra RAM

**XNP** Extended Next Page

# Seznam příloh

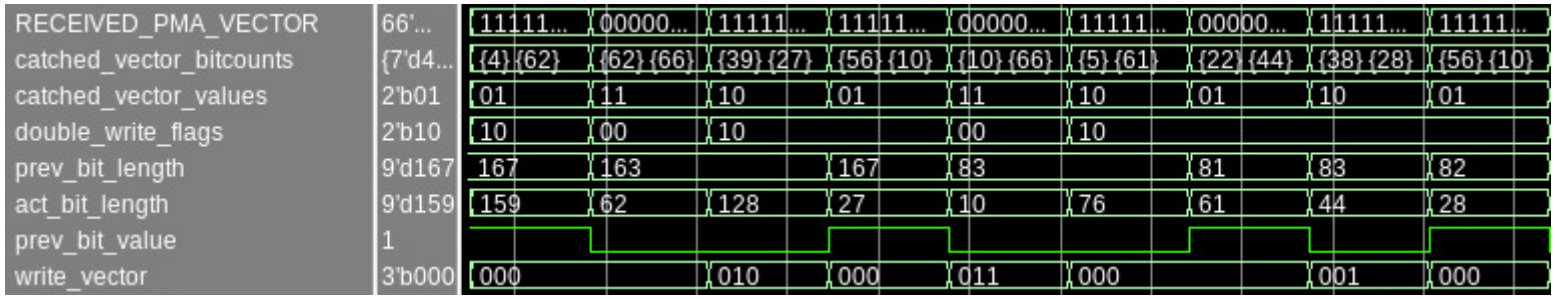
# A Podrobný stavový automat arbitráže



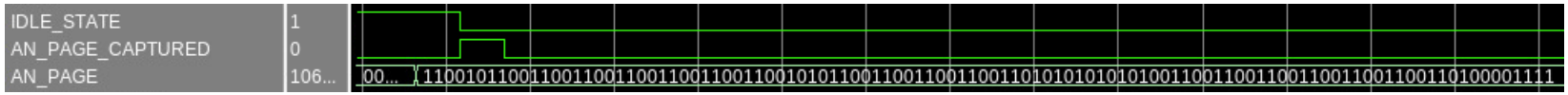

Obr. A.1: Podrobný stavový automat komponenty arbiter.vhd

# B Doplňující obrázky simulace

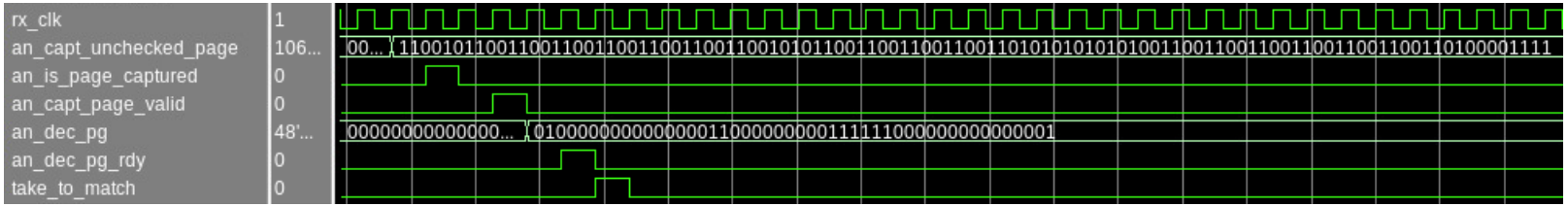
Zde byly přesunuty některé obrázky týkající se simulace, které by jinak narušovaly tok textu.



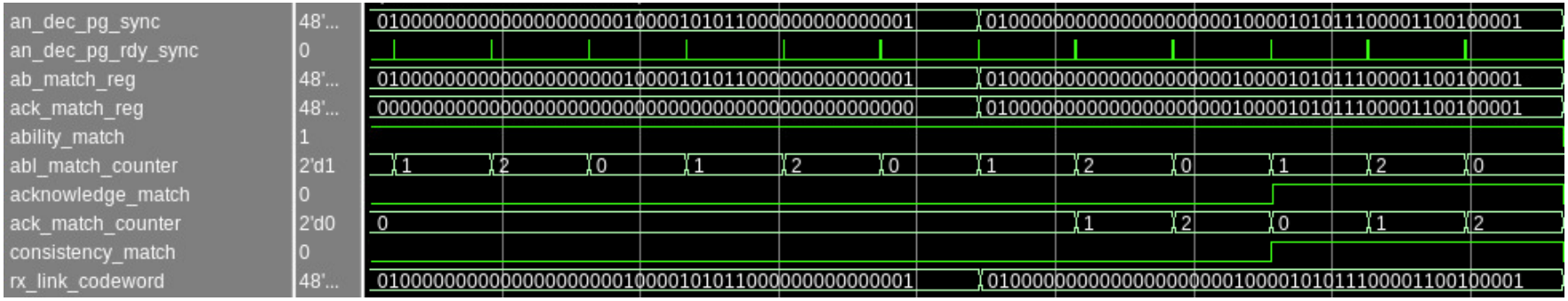

Obr. B.1: Ukázka příjmu řady bitů na rozhraní s GTY




Obr. B.2: Zachycení stránky na vstupu

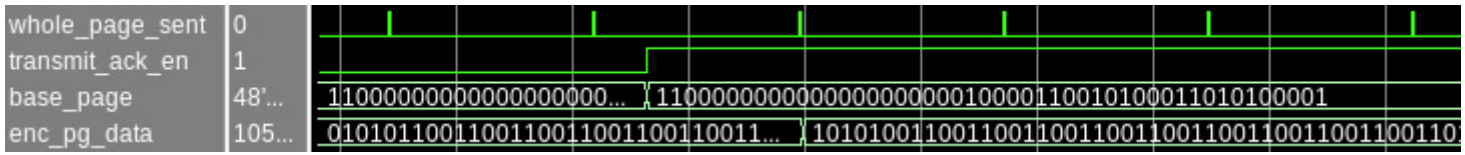



Obr. B.3: Ukázka řízení během zpracování zakódované stránky

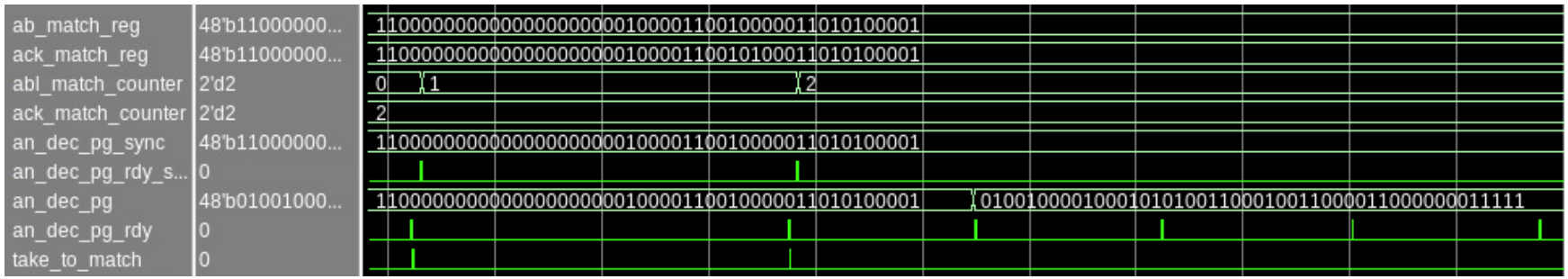



Obr. B.4: Simulace registrů hledajících shodné stránky


Obr. B.5: Vznik chybového stavu při nesplnění podmínky pro počet přechodů

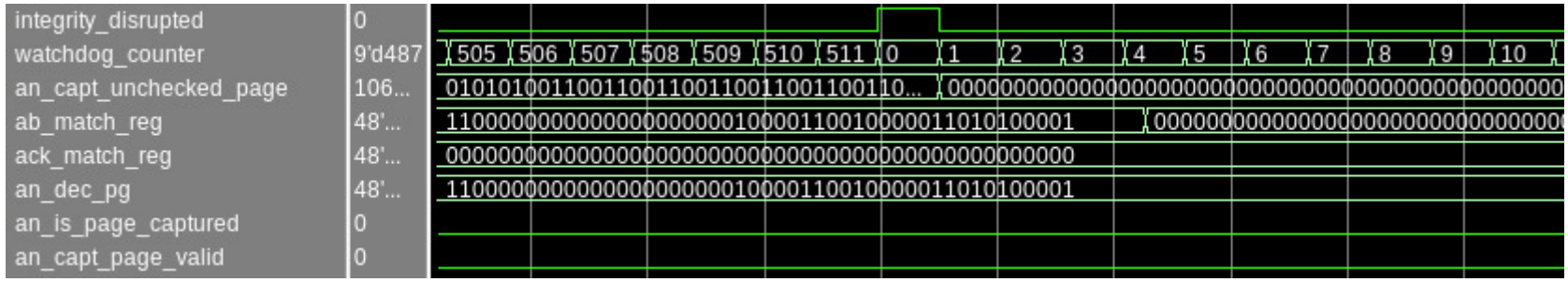



Obr. B.6: Změna vstupní stránky komponenty vysílače




Obr. B.7: Chování přijímače v případě příjmu stránky se špatnou „hlavičkou“






Obr. B.8: Ukázka resetu způsobeného přetečením watchdog čítače

# C Ukázka formátovaných PMA vektorů

```
11111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111
00000000000000000000000000000000000000000000000000000000000000011111
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
11111111111111111111111111111111111111111111111111111111111100000
11111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111
00000000000000000000000000000000000000000000000000000000000011111
11111111111111111111111111111111111111111111000000000000000000000
00000000000000000000000001111111111111111111111111111111111111111111
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
11111111111111111111111111111111111111111111111111111111111100000
11111111111111111111111111111111111111111111111111111111111111111111
00000000000000000000000001111111111111111111111111111111111111111111
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
11111111111111111111111111111111111111111111111111111111111100000
11111111111111111111111111111111111111111111111111111111111111111111
```

# D Obsah přiloženého archivu

Níže je vypsána adresářová struktura přílohy, ve které jsou poskytnuty zdrojové soubory.

```
/an
├── comp/
│   ├── arbiter/
│   │   ├── arbiter.vhd
│   │   └── Modules.tcl
│   ├── receiver/
│   │   ├── an_gty_rx_interface/
│   │   │   ├── an_gty_rx_interface.vhd
│   │   │   └── Modules.tcl
│   │   ├── an_manchester_decoder/
│   │   │   ├── an_manchester_decoder.vhd
│   │   │   └── Modules.tcl
│   │   ├── check_page_validity/
│   │   │   ├── check_page_validity.vhd
│   │   │   └── Modules.tcl
│   │   ├── sim/
│   │   ├── synth/
│   │   ├── Modules.tcl
│   │   └── rx_top.vhd
│   └── transmitter/
│       ├── an_gty_tx_interface/
│       │   ├── an_gty_tx_interface.vhd
│       │   └── Modules.tcl
│       ├── an_manchester_encoder/
│       │   ├── an_manchester_encoder.vhd
│       │   └── Modules.tcl
│       ├── sim/
│       ├── synth/
│       ├── Modules.tcl
│       └── tx_top.vhd
├── misc/
│   ├── mgmt_reg_constants_pkg.vhd .. balík konstant pro arbiter a rezoluční funkce
│   ├── debug_core.vhd
│   ├── an_pcspma_debug_core.vhd
│   ├── Modules.tcl
│   └── create_ilas.tcl ..... skript pro vytvoření zdrojových souborů ILA sond
├── sim/
│   ├── test_rx_tx/.....simulační zdroje RX a TX kanálu bez arbitráže
│   └── two_device/..... simulační zdroje komunikačního systému o dvou zařízeních
├── synth/
├── an_top.vhd ..... obalující komponenta obstarávající celý proces
└── Modules.tcl
```

Soubory `Modules.tcl` slouží k zařazení dané komponenty v hierarchickém systému při syntéze/simulaci. Zde poskytnutá hierarchie adresářů odpovídá hierarchii auto-negociačních podkomponent. Každá komponenta je umístěna v samostatné složce a má své vlastní simulační a syntézní zdroje. Skript Vivado.tcl je vytvořen pouze u komponent obsahujících dvě hodinové domény. Simulační zdroje některých komponent na přijímací straně mohou obsahovat navíc soubory s testovacími vektory. Struktura patřičných adresářů je následující:

```
/ ..... složka každé komponenty
├── sim ..... simulační zdrojové soubory
│   ├── signals.fdo ..... výpis signálů pro zahrnutí do simulace v Modelsim
│   ├── sim.fdo ..... simulační makro
│   ├── sim_sig.fdo ..... simulační makro
│   └── testbench.fdo ..... samotný testovací soubor komponenty
└── synth ..... zdrojové soubory pro syntézu
    ├── Makefile
    └── Vivado.tcl ..... skript pri zavedení dvou hodinových signálů
```