VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV MIKROELEKTRONIKY

DEPARTMENT OF MICROELECTRONICS

FPGA IP JÁDRO PRO PŘÍJEM DAT Z OBRAZOVÝCH SENZORŮ SONY IMX

FPGA IP CORE FOR SONY IMX SENSOR INTERFACE

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Milan Musil

AUTHOR

VEDOUCÍ PRÁCE — Ing. Michal Kubíček, Ph.D.

SUPERVISOR

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Mikroelektronika**

Ústav mikroelektroniky

Student: Bc. Milan Musil ***ID:*** 189980

Ročník: 2 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

FPGA IP jádro pro příjem dat z obrazových senzorů Sony IMX

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s rozhraním SLVS-EC, které využívá společnost Sony u nových průmyslových snímačů IMX k vysokorychlostnímu přenosu obrazových dat. Dále se seznamte s architekturou Xilinx Zynq UltraScale+ a proveďte obecný návrh IP jádra pro příjem obrazových dat rozhraním SLVS-EC s ohledem na zvolenou architektruru. Výstupem IP jádra budou surová obrazová data streamovaná pomocí AXI4-Stream protokolu.

Navrhněte IP jádro s využitím programovacího jazyka VHDL. Vytvořte testbench pro dílčí části IP jádra i pro jádro jako celek a uskutečněte potřebné simulace. Proveďte implementaci a pomocí funkčního senzoru a Demo Boardu pořiďte reálné snímky.

DOPORUČENÁ LITERATURA:

[1] MAXFIELD, Clive. The design warrior's guide to FPGAs: devices, tools, and flows. Boston: Newnes/Elsevier, c2004, ISBN 0-7506-7604-3.

[2] LINDH Lennart, KLEVIN Tommy. Advanced HW/SW Embedded System for Designers, 2018: FPGA - System On Chip. Independently published, 2018. ISBN: 978-1731115812

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 24.5.2022

Vedoucí práce: Ing. Michal Kubíček, Ph.D.
Konzultant: Ing. Tomáš Matějka

doc. Ing. Lukáš Fujcik, Ph.D.
předseda rady studijního programu

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Abstrakt

Práce se zabývá implementací rozhraní SLVS-EC do FPGA. Toto rozhraní je využíváno u nových obrazových senzorů firmy SONY pro vysokorychlostní přenos dat. V úvodu práce se nachází popis rozhraní, jeho možné konfigurace a porovnání s doposud využívaným rozhraním sub LVDS. Následuje výběr vhodného MPSoC s architekturou Zynq Ultrascale+ s ohledem na jeho hardwarové prostředky pro příjem vysokorychlostního signálu. Hlavní část práce se zabývá návrhem přijímače pro rozhraní SLVS-EC a dekódováním přijatých dat. Surová obrazová data jsou následně ukládána do externí RAM paměti. V závěru práce je popsána zvolená metoda testování dílčích částí i celkového designu.

Klíčová slova

SLVS-EC, sub LVDS, AXI, SONY, IMX, obrazový senzor, MPSoC, FPGA

Abstract

The diploma thesis deals with the implementation of the SLVS-EC interface into the FPGA. This interface is used in new SONY's image sensors for high-speed data transmission. The introduction contains a description of the interface, its possible configurations and comparison with the sub LVDS interface used so far. This is followed by the selection of a suitable MPSoC with the Zynq Ultrascale+ architecture with focus to its hardware resources for receiving a high/speed signal. The main part of this thesis deals with the design of receiver for the SLVS-EC interface and decoding of the data. The raw image data is then stored in external RAM. At the end of this thesis is described the chosen methods of testing partial parts and overall design.

Keywords

SLVS-EC, sub LVDS, AXI, SONY, IMX, image sensor, MPSoC, FPGA

Bibliografická citace

MUSIL, Milan. *FPGA IP jádro pro příjem dat z obrazových senzorů Sony IMX* [online]. Brno, 2022 [cit. 2022-05-24].
Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/142429. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav mikroelektroniky. Vedoucí práce Michal Kubíček.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	**Bc. Milan Musil**
VUT ID studenta:	**189980**
Typ práce:	**Diplomová práce**
Akademický rok:	**2021/22**
Téma závěrečné práce:	**FPGA IP jádro pro příjem dat z obrazových senzorů Sony IMX**

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné 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 závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

.

V Brně dne: 24. května 2022 ----------

podpis autora

Poděkování

Srdečně tímto děkuji mému odbornému vedoucímu diplomové práce a nadřízenému Ing. Tomáši Matějkovi za trpělivost a přátelský přístup nejen při řešení této diplomové práce. Dále bych chtěl poděkovat kolegům z PSI za cenné rady v oblasti návrhu embedded systémů.

Můj vděk patří také vedoucímu Ing. Michalu Kubíčkovi, Ph.D. za zprostředkování a konzultace v průběhu řešení této diplomové práce.

Na závěr bych chtěl poděkovat rodině za podporu. Především pak mé mamince za trpělivost a držení mě při životě po čas celého studia.

V Brně dne: 24. května 2022 ----------

podpis autora

Obsah

SEZNAM OBRÁZKŮ

Úvod

Správné pochopení přírody kolem nás je důležité abychom se k ní mohli chovat šetrně a lépe využili potenciál, který nabízí. S jejím pochopením nám pomáhá vědní obor biologie, který jako všechny ostatní vědní obory využívá moderní technologie k rychlejšímu, efektivnějšímu a hlubšímu zkoumání zájmové oblasti. Vývojem nových technologií pro biologii se zabývá i společnost Photon Systems Instruments (PSI). Ta se po dobu více než 25 let specializuje na vývoj high-end přístrojů pro pokročilé měření a zobrazování optických procesů v rostlinách a řasách. Mezi tyto procesy patří například fluorescence jak na buněčné úrovni, tak u celých rostlin. Během svého působení se firma prosadila na světové úrovni a její přístroje se používají na mnoha významných univerzitách ale i v předních světových organizacích mezi které patří i NASA, která využívá přístroje PSI na mezinárodní vesmírné stanici.

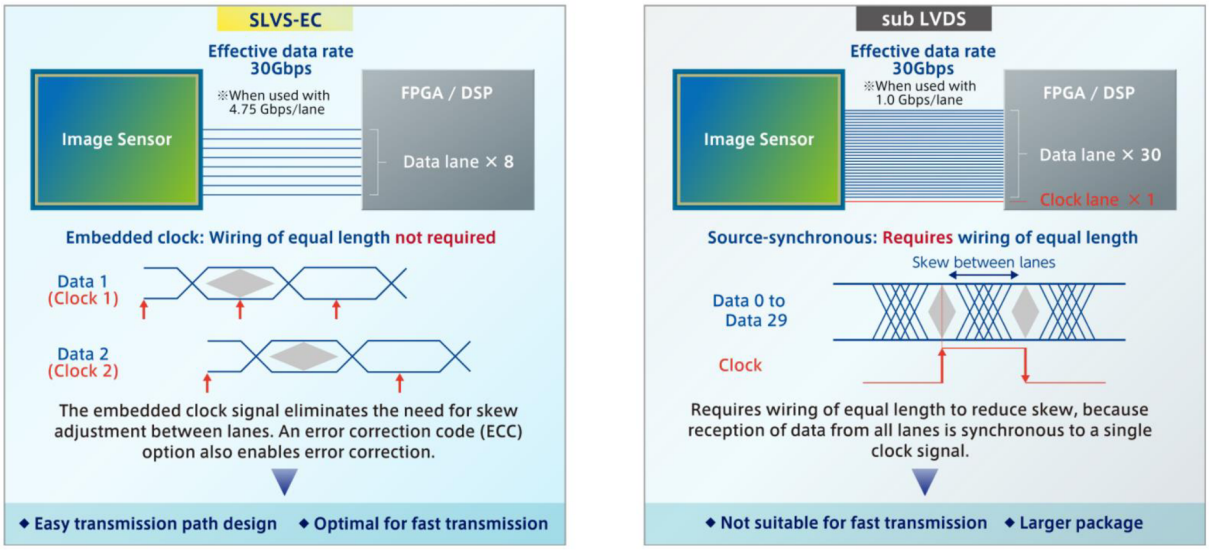
Aby si firma udržela náskok před konkurencí, je nutné neustále vyvíjet a inovovat své přístroje a postupy.

Cílem této práce je návrh a otestování IP jádra pro příjem dat skrze SLVS-EC rozhraní. Toto rozhraní využívají nové obrazové senzory IMX od firmy SONY pro vysokorychlostní přenos obrazových dat. Úlohou IP jádra je příjem obrazových dat ze senzoru, jejich dekódování a následné ukládání do externí DDR4 RAM paměti. Obrazová data musejí být do paměti ukládána ve formátu, který vyžaduje software pro jejich další zpracování. IP jádro je dimenzováno na maximální možnou propustnost dat a bitovou hloubku, kterou zvolená architektura a standard SLVS-EC umožňuje. Ne všechny Obrazové senzory, využívající toto rozhraní, ovšem umožňují využití všech kanálů pro přenos dat a disponují maximální bitovou hloubkou podporovanou SLVS-EC standardem. Z tohoto důvodu je požadována možnost měnit počet kanálů a bitovou hloubku dle všech variant, které standard SLVS-EC definuje.

Práce je rozdělena do tří částí. V první části jsou uvedeny veřejné informace o rozhraní SLVS-EC, jeho možných konfiguracích a výhody oproti staršímu sub LVDS. V této části jsou také zmíněné některé principy, které SLVS-EC standard využívá. Druhá část se věnuje výběru a popisu konkrétního FPGA, respektive MPSoC. Následně je tato část zaměřuje na návrh IP jádra a popis jeho jednotlivých částí. V závěru této části jsou j ještě popsány komponenty připojené k AXI sběrnici, mezi které patří například kontrolní a stavové registry nutné pro správnou konfiguraci přijímače a dekodéru obrazových dat. Poslední část práce se zaměřuje na proces testování celého designu, ale i na testování dílčích částí. Jsou zde také popsány použité metodiky a nástroje pro simulaci.

1. SLVS-EC

SLVS-EC neboli Scalable Low Voltage Signalling – Embedded Clock je standart rozhraní vyvinutý firmou Sony pro moderní rychlé obrazovkové senzory s vysokým rozlišením. Stejně jako jiné typy komunikace, používá pro přenos dat diferenciální páry. Narozdíl však například od sub LVDS, který je využit v současných přístrojích firmy PSI, není hodinový signál přenášen pomocí samostatného kanálu, ale jak již název napovídá, je na straně přijímače regenerován z kanálů datových. Díky tomu nemusí probíhat přenos dat skrze jednotlivé kanály synchronně a není tedy nutné zarovnávat fyzické cesty na stejnou délku. Dále není nutné zohledňovat časové posuny jednotlivých receiverů a další chyby, které se stávají relevantnějšími při zvyšování počtu synchronních kanálů. Datový přenos může tedy dosahovat vyšších rychlostí a to až 5 Gbps skrze jeden kanál [4]. Kromě vyšších přenosových rychlostí využívá SLVS-EC také několik principů k detekci a korekci chyb. Porovnání se sub LVDS, použitým v minulé generaci přístrojů a SLVS-EC je zobrazeno na Obr. 1.1.

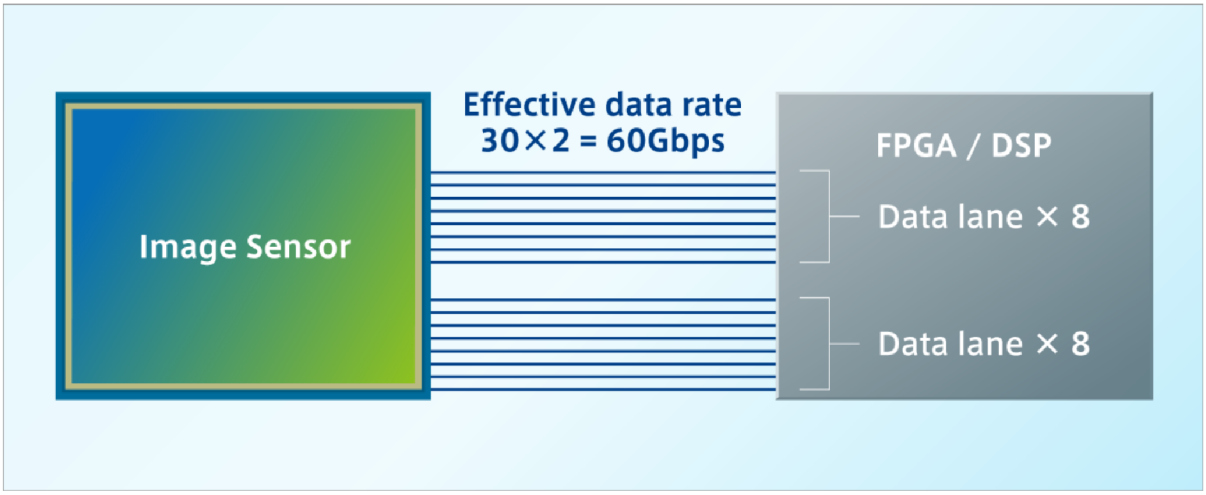


Obr. 1.1 srovnání SLVS-EC a sub LVDS [4]

SLVS-EC je interface pouze pro jednosměrnou komunikaci, která probíhá typicky směrem z obrazového senzoru do FPGA/DSP. Pro konfiguraci senzorů je tedy nutné využít samostatnou komunikaci. Typicky jsou tyto senzory nastavovány pomocí I2C nebo SPI rozhraní.

1.1 Topologie

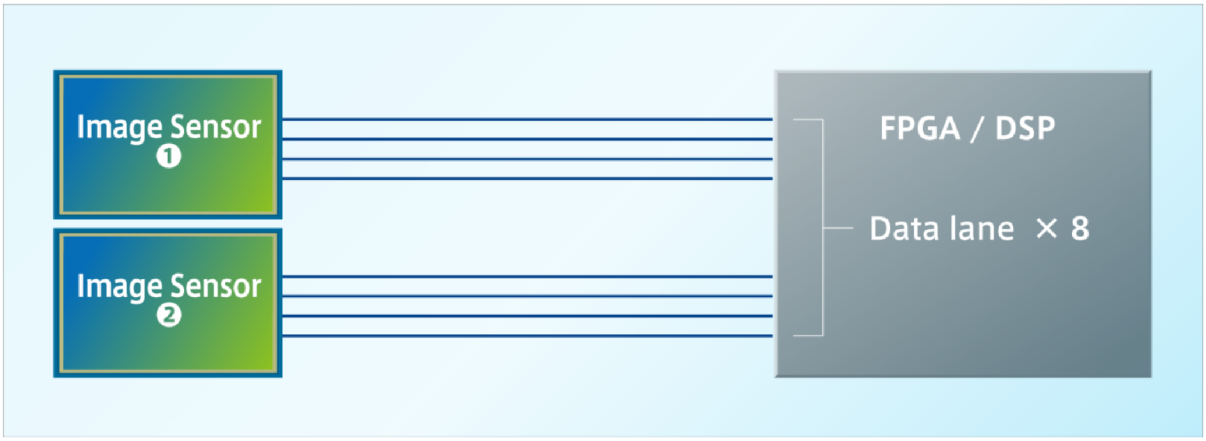
SLVS-EC nabízí mnoho variant pro zapojení jednoho či více senzorů. Pomocí jednoho rozhraní je možné přenášet data po až osmi kanálech. Pokud není z důvodu rychlosti nutné využít všechny kanály, může komunikace probíhat jen po menším počtu kanálu. Není tedy zapotřebí zapojovat všech 8. Pokud by ovšem rychlost komunikace po celé šířce, kterou SLVS-EC umožňuje byla nedostatečná, je možné zapojit jeden senzor přes více rozhraní, jak je zobrazeno na Obr. 1.2.



Obr. 1.2 Zapojení jednoho senzoru přes více rozhraní [4]

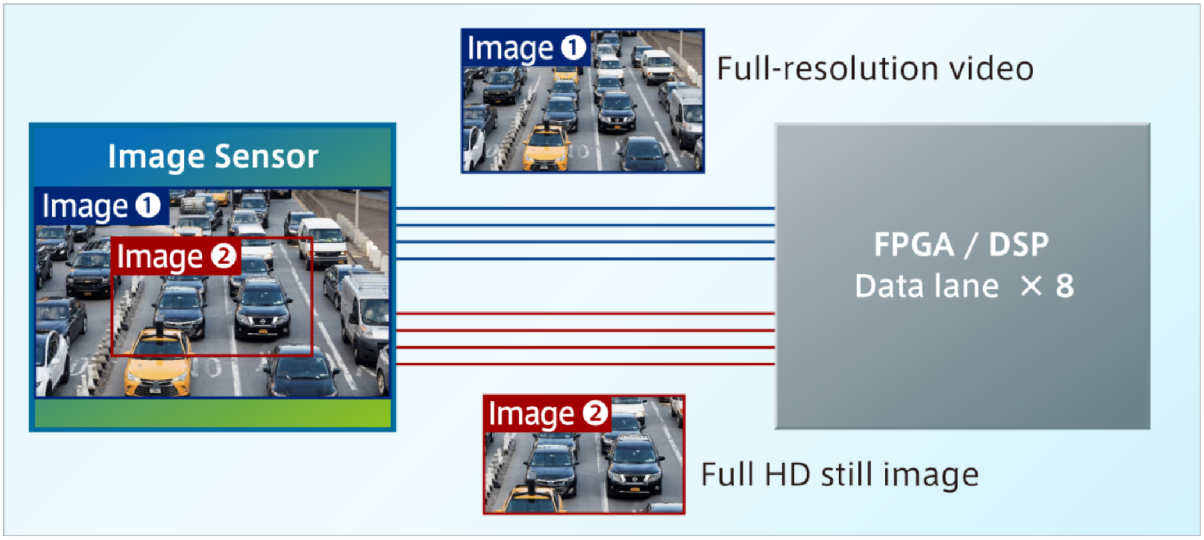
Jaký bude formát přenášených dat závisí na konfiguraci senzoru. Pomocí jednoho rozhraní mohou být přenášeny například jen liché pixely a pomocí druhého sudé.

Pokud je třeba přijímat data z více obrazových senzorů současně při čemž součet potřebných kanálů není větší než 8. Což je maximální počet kanálů v rámci jednoho rozhraní. Je možné tyto senzory zapojit na straně FPGA/DSP do jednoho rozhraní. jak je zobrazeno na Obr. 1.3.



Obr. 1.3 Zapojení více senzorů přes jedno rozhraní [4]

Další možností je přenášení více typů obrazových dat z jednoho senzoru. Tato možnost je označena jako multi-stream a je dostupná pouze u některých obrazových senzorů z nabídky SONY. Ukázka takového přenosu je na Obr. 1.4. Kromě snímků s různým snímaným úhlem a rozlišením, je možné přijímat snímky například s rozdílnou dobou expozice a v post processingu vytvořit snímek s vysokým dynamický rozsahem (HDR).



Obr. 1.4 Přenášení více typů obrazových dat z jednoho senzoru [4]

1.2 Průběh komunikace

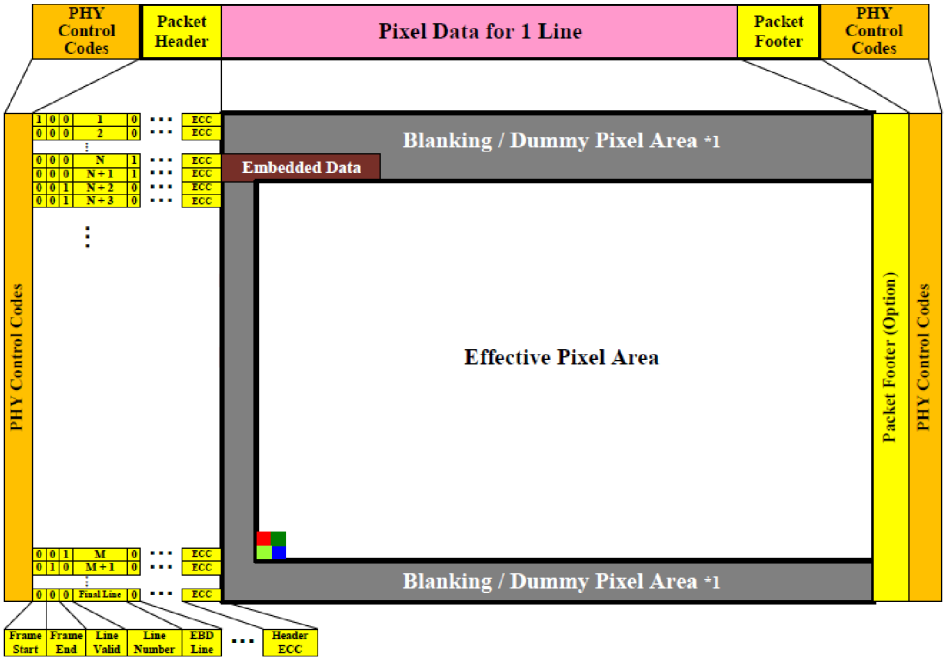
Při zahájení komunikace, její obnově po ukončení standby režimu nebo po změně počtu datových kanálů, přes které jsou přenášena data je nutné inicializovat přijímač pro každý datový kanál. Z tohoto důvodu jsou před začátkem vysílání obrazových dat vysílány kontrolní kódy fyzické vrstvy. Jedná se o sekvence slov z 8b10b kódování. Cílem vysílání těchto kontrolních kódů při zahájení přenosu je správná obnova hodinového signálu a eliminace rozdílného zpoždění mezi jednotlivými kanály. V průběhu komunikace tyto kontrolní kódy označují například začátek a konec paketu.

Ke správné obnově hodinového signálu je vhodné volit taková slova z 8b10b kódování, ve kterých dochází k co možná nejčastější změně logické hodnoty. Vhodným slovem, které je zvoleno i jako defaultní, je slovo D.10.5 [1], jehož tvar je "0101010101". Po synchronizaci hodinového signálu každého kanálu je nutné synchronizovat jednotlivé kanály mezi sebou. K této synchronizaci je opět využito definovaného kontrolního kódu, který je vysílán nejen při zahájení komunikace, ale i v jejím průběhu.

Konkrétní podoby všech kódů fyzické vrstvy a počty jejich opakování jsou definovány ve standartu SLVS-EC. Jedná se ovšem o defaultní hodnoty, které je možné změnit. V tom případě je ovšem nutné nastavit stejnou podobu těchto kódů jak na straně obrazového senzoru, tak i v přijímači.

1.3 Formát přenášených dat

Data z obrazového snímače jsou do DSP/FPGA zasílána po jednotlivých snímcích. A tyto snímky po jednotlivých řádcích neboli paketech. Podoba jednoho snímku je znázorněna na Obr. 1.5.



Obr. 1.5 Formát snímku [1]

Jak je z obrázku patrné, přenášená data neobsahují pouze obrazová data, ale i mnoho redundantních dat. Ty jsou důležité pro správný příjem a vyhodnocení přijatých dat na straně DSP/FPGA.

Začátek každého paketu je označen kontrolními kódy fyzické vrstvy, za kterými následuje hlavička paketu, ve které jsou obsaženy informace pro správné složení snímku. Mezi tyto informace patří například označení začátku a konce snímku, číslo řádku a kontrolní kódy. Dále pak, zda se na konkrétním řádku nachází obrazová data, embedded data nebo prázdná data bez informací.

Embedded data nejsou definována SLVS-EC standardem, ale konkrétními obrazovými snímači. Zde je možné předávat informace o nastavení snímače jako je například velikost a formát dat. Pro konkrétní podobu embedded dat je třeba nahlédnout do dokumentace ke konkrétnímu obrazovému snímači.

Poslední část hlavičky obsahuje výsledek hashovací funkce CRC16 pro detekci chyb. Pro zvýšení pravděpodobnosti správného odhalení a možnosti opravy chyb je celá hlavička včetně výsledku CRC odeslána celkem 3x. To sice snižuje efektivitu přenosu, přináší to ovšem možnost velmi jednoduše opravit náhodné chyby.

Za obrazovými či embedded daty následuje zápatí a kódy fyzické vrstvy pro konec paketu. Generování a vysílání zápatí obrazovým snímačem je volitelné a v případě potřeby je nutné povolit v konfiguračním registru použitého snímače. Obsah zápatí je výsledkem hashovací funkce CRC32 z obrazových či embedded dat. Pro tento výpočet nejsou použita data z hlavičky ani řídící kódy fyzické vrstvy.

1.4 Detekce chyb v přenosu

Během přenosu informace z jednoho místa na druhé může dojít vlivem šumu a interferencí k náhodným chybám. V digitální technice se tyto chyby projevují změnou hodnoty jednoho či více bitů. V určitých oblastech, jako například při přenosu audiovizuálních dat, nemusí být tyto chyby v malém počtu kritické. Jedná-li se ovšem například o řídící signály, může být chyba, byť jen jednoho bitu, kritická. Z tohoto důvodu se k posílaným datům ve většině případech přidávají redundantní informace. Díky těmto informacím je možné detekovat chybu v přenosu a v některých případech, pokud není chyba příliš velká, je možné ji odstranit a zrekonstruovat původní data. Při správném bezchybném přenosu jsou tyto informace nadbytečné, protože neobsahují přenášená data a snižují tak efektivitu přenosu.

Součástí hlavičky je výsledek hashovací funkce CRC16 a celá hlavička se i s tímto výsledkem odešle celkem 3x. Pokud nastane u nějakého bitu během přenosu chyba, nebude mít bit ve všech třech případech stejnou hodnotu. V tom případě se použije hodnota, která byla přijata vícekrát. Pokud budou například pro jeden byt přijaty hodnoty ‘0‘ ‘1‘ ‘0‘, bude použita hodnota ’0’. Pokud budou přijaty hodnoty ‘1’ ‘1’ ‘0’, bude použita hodnota ‘1’. Tento způsob sice dokáže opravit libovolný počet bitů, (pouze pokud je chyba jen jednou ve trojici bitů) ale snižuje efektivitu přenosu pro daný úsek dat na 1/3. Proto jsou při přenosech dat používány jiné metody, které mají sice omezenější možnosti při rekonstrukci dat, nebo slouží pouze k detekci chyby, ale vyžadují přenos výrazně menšího objemu redundantních dat.

1.5 8b10b kódování

Jelikož se ve vysílaných datech může vyskytovat větší množství stejné logické hodnoty v řadě, což může vést k chybné detekci a rekonstrukci hodinového signálu z přijatých dat. U obrazových dat se může jednat například o větší počet logických jedniček při přesaturovaném snímku. Nebo může počet jedné logické hodnoty významně přesahovat počet druhé logické hodnoty a tím zanášet do přenosu nezanedbatelnou stejnosměrnou složku signálu, využívá se 8b10b kódování, kdy se z 8b znaku vytvoří 10b slovo. Přidáním dvou bitů ke každému bajtu sice klesne efektivita přenosu o 20 %, ale dostaneme k dispozici 4x více slov. Díky tomu můžeme většinu slov vyřadit a pro přenos

dat nepoužívat. Kvůli potlačení stejnosměrné složky v signálu se ponechají pouze slova, která mají počet jedniček a nul buď stejný nebo jiný pouze o 2. Dále se pro lepší rekonstrukci hodinového signálu a snížení požadavků na spodní hranici šířky pásma vyřadí slova, která obsahují více než 5 stejných logických hodnot v řadě. Po této redukci je k dispozici stále dostatek použitelných slov. Konkrétně [8]:

- 252 slov se stejným počtem jedniček a nul (5x '1' a 5x '0')
- 210 slov s kladnou disparitou (6x '1' a 4x '0')
- 210 slov se zápornou disparitou (4x '1' a 6x '0')

8b10b kódování je možné rozdělit na dvě části [9]: 5b6b a 3b4b kódování. Původní 8b číslo je tedy rozděleno na dvě části, které se kódují samostatně a následně se spojí za sebe. Například číslo 00011_2 (3_{10}) je v 5b6b kódování změněno na 110001_2. V tomto případě je počet jedniček a nul tzv. disparita výsledného čísla neutrální. To ovšem není možné zajistit pro všechny hodnoty. 8b číslo může nabývat 256 hodnot, ale 10b slov s neutrální disparitou a maximálně pěti stejnými hodnotami v řadě je jen 252. Proto jsou některá 5b čísla prezentovány dvěma 6b slovy. Jedním s kladnou a jedním se zápornou disparitou, jak je vidět v Tab. 1. Stejně je tomu tak i u 3b4b kódování, jak je vidět v Tab. 2. Po přidělení dvou hodnot některým číslům zůstává ještě 12 desetibitových kombinací, které jsou využity pro řídící signály. Tyto signály se označují písmenem K a jsou vypsány v Tab. 3. V SLVS-EC standardu jsou tyto řídící signály využity např. pro synchronizaci a označení začátku a konce řádku.

Při kódování je aktuální hodnota disparity uložena v proměnné, která může nabývat pouze hodnot -1 a +1. Podle aktuální hodnoty proměnné je u dalšího slova, které nemá neutrální disparitu, volena hodna s kladnou nebo zápornou disparitou. Tím je dosaženo výsledného kódu, který má maximálně o 2 jiný počet jedniček a nul. V Tab. 2 jsou pro D.x.7 kromě slov s kladnou a zápornou disparitou ještě další dvě možnosti. Ty jsou voleny na základě předešlého 6b slova tak, aby nenastala situace, kdy je v řadě za sebou více než 5 bitů stejné hodnoty.

Tab. 1: 5b6b kódování převzato z [9]

vstup		aktuální disparita	
		-1	+1
kód	EDCBA	a b c d e i	
D.00	00000	100111	011000
D.01	00001	011101	100010
D.02	00010	101101	010010
D.03	00011	110001	
D.04	00100	110101	001010
D.05	00101	101001	
D.06	00110	011001	
D.07	00111	111000	000111
D.08	01000	111001	000110
D.09	01001	100101	
D.10	01010	010101	
D.11	01011	110100	
D.12	01100	001101	
D.13	01101	101100	
D.14	01110	011100	
D.15	01111	010111	101000

vstup		aktuální disparita	
		-1	+1
kód	EDCBA	a b c d e i	
D.16	10000	011011	100100
D.17	10001	100011	
D.18	10010	010011	
D.19	10011	110010	
D.20	10100	001011	
D.21	10101	101010	
D.22	10110	011010	
D.23	10111	111010	000101
D.24	11000	110011	001100
D.25	11001	100110	
D.26	11010	010110	
D.27	11011	110110	001001
D.28	11100	001110	
D.29	11101	101110	010001
D.30	11110	011110	100001
D.31	11111	101011	010100

Tab. 2: 3b4b kódování převzato z [9]

vstup		aktuální disparita	
		-1	+1
Kód	HGF	f g h j	
D.x.0	000	1011	0100
D.x.1	001	1001	
D.x.2	010	0101	
D.x.3	011	1100	0011
D.x.4	100	1101	0010
D.x.5	101	1010	
D.x.6	110	0110	
D.x.P7	111	1110	0001
D.x.A7		0111	1000

Tab. 3: 8b10b řídící slova převzato z [9]

vstup		aktuální disparita	
		-1	+1
kód	HGF EDCBA	abcdei fghj	
K.28.0	000 11100	001111 0100	110000 1011
K.28.1	001 11100	001111 1001	110000 0110
K.28.2	010 11100	001111 0101	110000 1010
K.28.3	011 11100	001111 0011	110000 1100
K.28.4	100 11100	001111 0010	110000 1101
K.28.5	101 11100	001111 1010	110000 0101
K.28.6	110 11100	001111 0110	110000 1001
K.28.7	111 11100	001111 1000	110000 0111
K.23.7	111 10111	111010 1000	000101 0111
K.27.7	111 11011	110110 1000	001001 0111
K.29.7	111 11101	101110 1000	010001 0111
K.30.7	111 11110	011110 1000	100001 0111

2. NÁVRH PŘIJÍMAČE SLVS-EC

Hlavním cílem této práce je návrh přijímače pro SLVS-EC rozhraní. Funkce tohoto přijímače je příjem, zpracování a uložení surových obrazových dat do externí RAM paměti. SLVS-EC je vysokorychlostní rozhraní s rychlostí až 5 Gbps/kanál, přičemž může v rámci jednoho rozhraní obsahovat až osm kanálů. Z důvodu této rychlosti je nutné využít hardwarové prostředky zvoleného FPGA pro příjem dat. Přijímač je navržen pro co nejvyšší datový přenos, který zvolené FPGA nebo SLVS-EC standard umožňuje. Je tedy využito maximální frekvence přijímaných dat, kterou SLVS-EC standard umožňuje a maximálního možného počtu kanálů, které je FPGA schopné přijmout, potažmo zpracovat.

Dále se počítá s maximální možnou bitovou hloubkou, kterou SLVS-EC standard podporuje. Tou je formát RAW16, neboli 16 b hloubka jednoho pixelu. Ne všechny obrazové senzory ovšem umožňují využití tohoto formátu a disponují pouze menší bitovou hloubkou. Z tohoto důvodu a z důvodu větší variability je požadováno zpracování všech formátů, které standard podporuje. Těmi jsou RAW8, 10, 12, 14 a 16.

Obrazová data jsou ve standardu SLVS-EC přeskládána tak, aby byl jejich přenos co nejefektivnější. Z důvodu kompatibility se stávajícím softwarem firmy PSI pro zpracování těchto dat, je tedy nutná jejich konverze do příslušného formátu.

Aby bylo možné obrazová data dále zpracovávat operačním systémem, respektive softwarem v operačním systému, jsou tato data ukládána do externí RAM paměti.

2.1 Výběr FPGA

Pro realizaci přijímače SLVS-EC je v této práci využíván MPSoC s architekturou Xilinx Zynq Ultrascale+, který je využíváno v nových přístrojích firmy PSI. Firma Xilinx dělí tyto MPSoC do tří řad označených CG, EG a EV [2]. Nejnižší řada CG je označována jako vstupní bod pro práci s touto architekturou a heterogenní zpracování dat. Řada EG je uzpůsobena pro využití v co nejširším spektru aplikací. Poslední řada EV je zaměřena na zpracování multimediálních dat. K tomuto účelu je v této řadě integrován například video kodek H.264 / H.265.

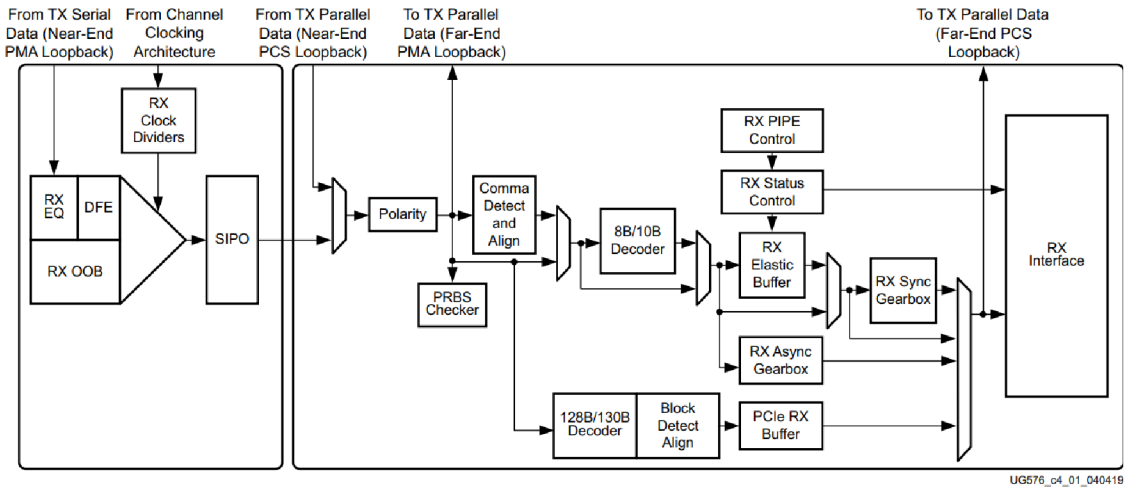
Architektura Xilinx Zynq Ultrascale+ disponuje částí „procesní systém“ (processing system, dále jen PS) a částí s programovatelnou logikou (programmable logic, dále jen PL). PS obsahuje CPU sestavené z jader ARM Cortex A53. Tyto jádra jsou využívané například v některých modelech Raspberry Pi a jedná se o nejrozšířenější procesor v mobilních telefonech [3]. V architektuře Xilinx Zynq Ultrascale+ jsou využita v řadě CG dvě a ve zbylých řadách čtyři jádra ARM Cortex A53. V tomto případě je na jádrech provozován operační systém Linux. Kromě CPU obsahuje PS hardwarově

implementované periferie jako například zde využité DMA, DDR4 kontrolér nebo SPI. PL pak obsahuje standartní FPGA

Pro příjem dat z obrazového senzoru je využito hardwarových vysokorychlostních transceiverů. Ty jsou v této architektuře označovány jako GTH. Dostupné jsou v řadách MPSoC ZU4 a vyšších. Při zahájení této práce na vývoji přijímače pro SLVS-EC rozhraní byl bohužel trh s polovodiči významně poznamenán výrobní krizí. Z MPSoC dostupných na trhnu a ve skladových zásobách firmy PSI byl vybrán model ZU5EV v pouzdře SFVC784. tento konkrétní model v tomto pouzdře obsahuje pouze 4 GTH transceivery, což je nejmenší možný nenulový počet v architektuře Xilinx Zynq Ultrascale+. Výsledný návrh bude tedy vytvářen s ohledem na tuto skutečnost a bude využívat nanejvýše 4 datové kanály pro přenos dat z obrazového senzoru.

2.2 Popis transceiveru a jeho nastavení

Pro konverzi přijatého vysokorychlostního elektrického signálu na logické hodnoty využívané v FPGA je v architektuře Xilinx Zynq Ultrascale+ využíváno GTH transceiveru. GTH transceivery je možné konfigurovat jak pro vysílání, tak i pro příjem signálu. Cílem této práce je pouze příjem dat z obrazového senzoru, využije se tedy jen část pro příjem dat. Blokové schéma RX části transceiveru je zobrazeno na Obr. 2.1.



Obr. 2.1 Blokové schéma RX části GTH transceiveru [6]

Pro správnou a zjednodušenou konfiguraci poskytuje firma Xilinx UltraScale FPGAs Transceivers Wizard. Ve zvolené architektuře je možné pomocí tohoto průvodce konfigurovat pouze GTH transceivery [5]. Průvodce je vybaven také velkou sadou přednastavených profilů pro přenos dat. Pro přenos obecných dat poskytuje přednastavené profily například pro Gigabit Ethernet nebo SATA. Pro přenos obrazových dat například HDMI nebo DisplayPort až do rychlosti 8,1 Gb/s. Pro účely této práce

nevyhovuje ale ani jeden z nabízených profilů. Pro konfiguraci je tedy vybrána možnost „Start from scratch“ a všechny parametry jsou nataveny manuálně.

V základním nastavení jsou vyplněny frekvence přijímaných dat a hodinového signálu, vybrané možnosti dekódování, bitová šířka výstupních dat a parametry fyzické sběrnice dle dokumentace standartu SLVS-EC. Frekvence vstupních dat je zvolena nejvyšší, kterou SLVS-EC standart nabízí, tedy 5 Gbps [4]. Pro výpočet potřebné frekvence hodinového signálu je potřeba vést v úvahu bitovou šířku výstupních dat a dekódování. Bitová šířka výstupních dat je v tomto případě zvolena 32 bitů a dekódování z 8b10b se provádí přímo v transceiveru. Výpočet potřebné frekvence hodinového signálu je potom následující:

$$f_{CLK} = \frac{f_{vstupní\ data} \cdot dekódování}{šířka\ výstupních\ dat}$$

$$f_{CLK} = \frac{5 \cdot 10^9 \cdot \frac{8}{10}}{32} = 125\ MHz$$

Pro správné zpracování dat za transceivere je tedy nutné, aby hodinová frekvence následného designu byla nejméně 125 MHz. Proto byla zvolena frekvence 150 MHz, která má téměř 17 % rezervu.

Transceiver nabízí možnost využít hardwarově implementovaného dekodéru. Při zvolení možnosti 8b10b dekódování poskytne na svém výstupu dekódovaná data zarovnaná do bajtů a další 4 výstupní kontrolní vektory obsahující informace z dekodéru [6]. První kontrolní vektor informuje, zda je dekódované slovo datové, nebo zda se jedná o řídící signály. Takzvané K charaktery. Druhý vektor předává informaci o chybě disparity. Viz kapitola 1.5 8b10b kódování. Následující vektor předává informaci, která slova byla dekódována jako takzvaný comma charakter. Jako comma charakter se většinou využívá slovo K.28.5 a slouží například pro označení začátku sekvence kódů. V případě dekódování comma charakteru je nastaven i příslušný bit ve vektoru označujícím K charaktery. Poslední vektor informuje, zda byly přijatá data správně dekódována. Pokud jsou přijata data, která neodpovídají 8b10b kódování, je příslušný bit v tomto vektoru nastaven na ‘1’. Na datové sběrnici je zobrazeno spodních 8 bitů přijatého desetibitového slova a na prvním a druhém kontrolním vektoru je zobrazen 9. a 10. přijatý bit. Tím je zajištěno, že se data neztratí a je možné jejich dekódování v jiné části systému.

Bitová šířka výstupních dat je volena s ohledem na frekvenci hodinového signálu. Při malé bitové šířce musí mít hodinový signál vyšší frekvenci, aby bylo možné zajistit dostatečnou propustnost dat. Při frekvenci 150 MHz je s určitou rezervou minimální možná bitová šířka dat 32 bitů. Při tomto výběru jsou také zohledněny sekvence vyskytující se v SLVS-EC standartu. Ty mají délku právě 4 bajty, tedy 32 bitů [1].

Významným kritériem pro volbu bitové šířky dat je tedy shoda s délkou vyhledávaných sekvencí a možnosti jejich zarovnání na celou šířku sběrnice. GTH transceiver umožňuje zarovnávat data při detekci comma charakteru o kterém bylo psáno výše. V Tab. 4 jsou možné varianty zarovnání výstupních dat pomoc comma symbolu. Z tabulky je patrné, že čtyř bajtové sekvence s jistotou nepřesáhnou do dalšího hodinového taktu jen ve dvou případech. A to při šířce dat 32 nebo 64 bitů a ALIGN_COMMA_WORD = 4, tedy zarovnání na 4 bajty. Při volbě šířky dat 32 bitů se bude vyhledávaná sekvence a tím i začátek obrazových či jiných dat vyskytovat na předem známém a vždy stejném místě. Tato volba usnadní tedy další zpracování dat a návrh návazných bloků. Při volbě větší šířky by sice návazný design mohl využívat nižší hodinové frekvence, ale bylo by pro jeho realizaci potřeba větší množství hardwarových prostředků z FPGA.

Tab. 4 Zarovnání comma charakteru [6]

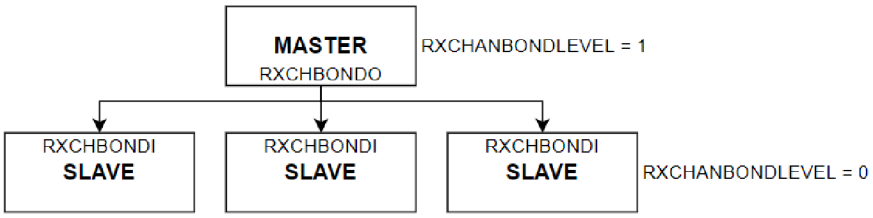
RX_DATA_WIDTH	RX_INT_DATAWIDTH	ALIGN_COMMA_WORD	Possible RX Alignments (Gray = Comma Can Appear on Byte)
16/20 (2-byte)	0 (2-byte)	1	Byte1 Byte0
16/20 (2-byte)	0 (2-byte)	2	Byte1 Byte0
16/20 (2-byte)	0 (2-byte)	4	Invalid Configuration
32/40 (4-byte)	0 (2-byte)	1	Byte3 Byte2 Byte1 Byte0
32/40 (4-byte)	0 (2-byte)	2	Byte3 Byte2 Byte1 Byte0
32/40 (4-byte)	0 (2-byte)	4	Invalid Configuration
32/40 (4-byte)	1 (4-byte)	1	Byte3 Byte2 Byte1 Byte0
32/40 (4-byte)	1 (4-byte)	2	Byte3 Byte2 Byte1 Byte0
32/40 (4-byte)	1 (4-byte)	4	Byte3 Byte2 Byte1 Byte0
64/80 (8-byte)	1 (4-byte)	1	Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0
64/80 (8-byte)	1 (4-byte)	2	Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0
64/80 (8-byte)	1 (4-byte)	4	Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0

Ve volitelných funkcích umožňuje průvodce zapnutí a nastavení detekce a zarovnání comma charakteru. Zde je potřeba kromě jeho tvaru jak s kladnou, tak zápornou paritou zvolit i na kolik bajtů mají být výstupní data zarovnána.

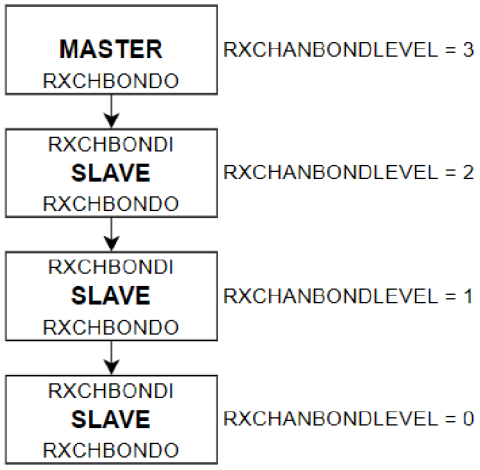
Další volitelnou funkcí je takzvaný channel bonding. Tato funkce umožňuje detekovat předem známé sekvence sloužící k synchronizaci jednotlivých kanálů mezi

sebou. Kromě délky a tvaru sekvence je zde nutné správně zvolit maximální posun jednotlivých kanálů. Ten je závislý na frekvenci opakování synchronizačních sekvencí a pro správné fungování by tato hodnota neměla přesahovat polovinu periody mezi opakováním. Perioda i počet opakování má sice ve standardu SLVS-EC definovanou výchozí hodnotu, tu je ale možné v nastavení obrazového senzoru změnit. Navýšení hodnot těchto parametrů je vhodné zvážit, pokud jsou data přicházející do DSP/FPGA značně rozsynchronizována.

Kromě detekování správné sekvence je také nutné správně propojit synchronizované kanály. V každém designu musí být jeden kanál nastaven jako master a ostatní, které se budou synchronizovat podle něj, jako slave [6]. Při synchronizaci poté předává kanál označený jako master informace pomocí výstupního portu RXCHBONDO slave kanálům. Slave kanály tyto informace přijímají pomocí vstupního portu RXCHBONDI. Je možné napojit všechny slave kanály paralelně na jeden master kanál, jak je vidět na Obr. 2.2. Při této konfiguraci je ovšem obtížné dodržet časové constraints. Z toho důvodu umožňuje konfigurace kanálů zapojení do takzvaného Daisy Chain, kdy jsou slave kanály zapojeny do série, jak je vidět na Obr. 2.3. Tyto dvě konfigurace je také možné libovolně kombinovat. Vždy je ovšem nutné dodržet pouze jeden master kanál a správně nastavit RXCHANBONDLEVEL jak je vidět na Obr. 2.2 a Obr. 2.3.



Obr. 2.2 Paralelní zapojení kanálů při channel bondingu

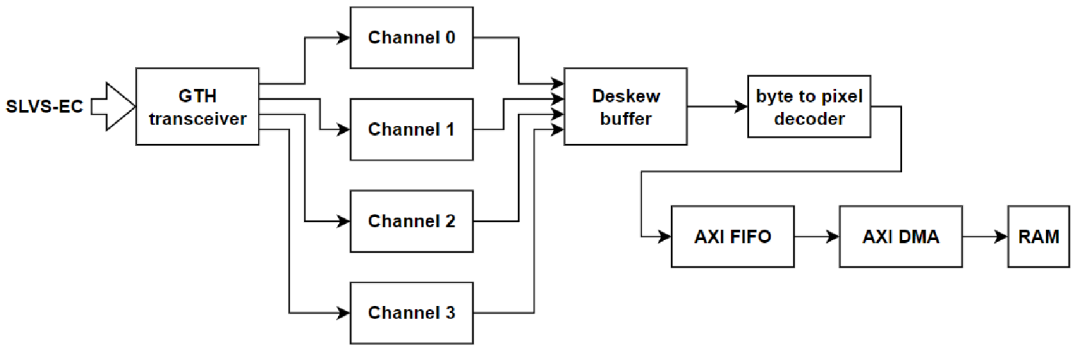


Obr. 2.3 zapojení kanálů při channel bondingu do Daisy Chain [6]

Následuje nastavení obnovy hodinového signálu z příchozích dat. I zde je nutné nastavit vyhledávanou sekvenci, která je k tomu účelu využívána. Jak již bylo uvedeno výše, standard SLVS-EC využívá k označení začátku této sekvence comma symbol a následuje trojice slov, ve kterých se periodicky mění logická hodnota co nejčastěji. Tedy slovo D.10.5 z 8b10b kódování, jehož hodnota je 0101010101_2.

2.3 Hierarchie navrženého designu

Data přicházející z obrazového senzoru do MPSoC jsou nejdříve převedeny na logické hodnoty používané v FPGA, následně deserializována a synchronizována napříč jednotlivými kanály. Tyto operace jsou provedeny hardwarovém GTH transceiveru, jak již bylo popsáno v kapitole 2.2. Odtud jsou nejen data, ale i kontrolní vektory přivedeny do bloků zajišťující zpracování informací obsažených v přenosu a extrakci obrazových dat. Tyto bloky jsou v Obr. 2.4 označeny jako Channel 0 až Channel 3. Odtud jsou už jen obrazová data přivedena do Deskew bufferu, který zajistí dodatečnou synchronizaci jednotlivých kanálů. Následuje byte to pixel decoder, který zajistí dešifrování obrazových dat. Výstupem z byte to pixel decoderu jsou již surová nezašifrovaná obrazová data. Následuje FIFO zajišťující správné propojení s DMA (direct memory access), které se stará o streamování obrazových dat do externí RAM paměti

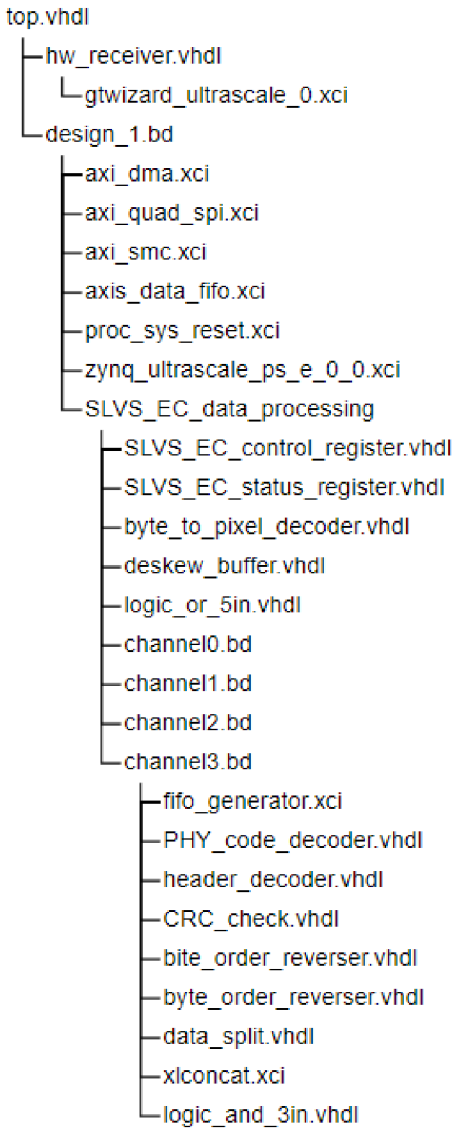


Obr. 2.4 Zjednodušené blokové schéma celého návrhu

Na Obr. 2.5 je patrná zjednodušená struktura celého návrhu. Z této struktury jsou vypuštěny například obvody NOT a názvy souborů jsou mnohdy psány ve zkrácených podobách pro lepší čitelnost a orientaci.

Vrchní soubor top.vhdl v sobě propojuje 2 části designu – hw_receiver.vhdl a design_1.bd. Hw_receiver.vhdl je zde z důvodu zjednodušení a zpřehlednění nadřazeného top.vhdl. Jsou zde popsány resetovací algoritmy pro GTH transceiver, který obsahuje přes deset portů resetovacích signálů. Dále rozvedení hodinového signálu, pro který má GTH transceiver také přes deset portů. V neposlední řadě také propojení

jednotlivých kanálů k zajištění správného channel bondingu, který byl popsán výše. Na závěr také rozdělení datového a kontrolních vektorů do jednotlivých kanálů pro zpřehlednění. Celkový počet 751 bitů potřebných pro správný provoz GTH receiveru v této konkrétní konfiguraci je v top.vhdl zredukován pouze na 232, z čehož je 128 pro přenášená data a 192 pro kontrolní signály.



Obr. 2.5 Zjednodušená struktura designu

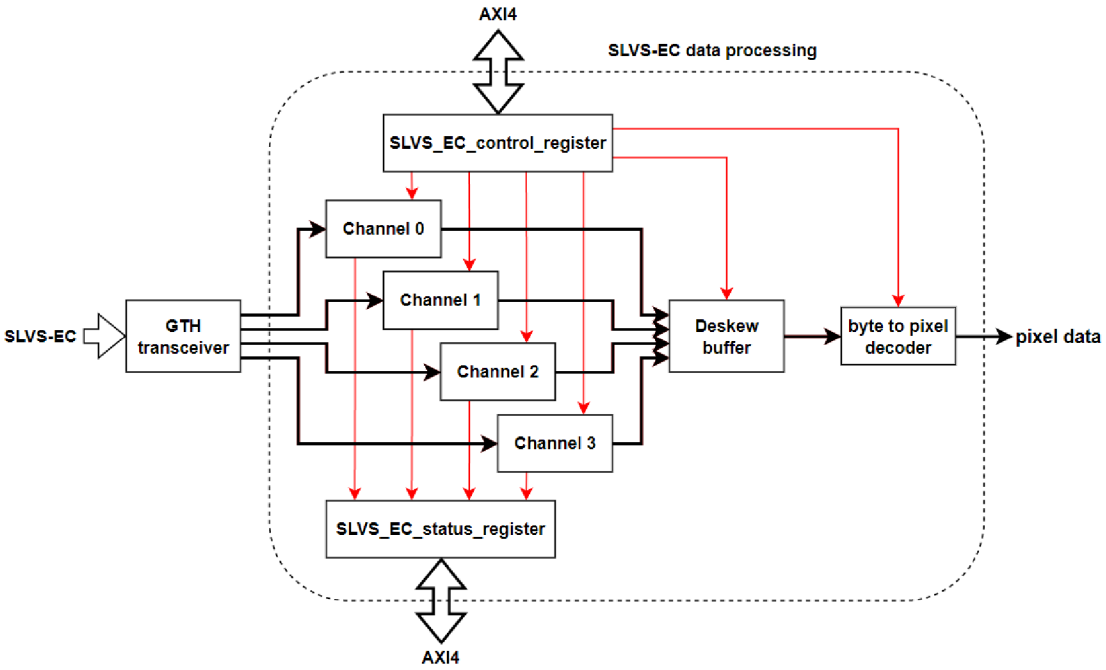
Druhou komponentou v top.vhdl je design_1.bd. Jedná se formát, který používá program Vivado od společnosti Xilinx pro blokové diagramy. Ty významně ulehčují práci a zvyšují přehlednost návrhu. Součástí tohoto blokového diagramu jsou všechny zbylé komponenty využívané v této práci. Můžeme zde naleznout IP jádro Zynq Ultrascale+, které je zde využito pro propojení designu vytvořeného v FPGA části

MPSoC s částí procesorovou. Z tohoto IP jádra je zde využit přivedený hodinový signál s frekvencí 150 MHz a resetovací signál. Dále také propojení s datovými sběrnicemi uvnitř procesorové části. Toto propojení nám umožňuje například nastavovat kontrolní registr pro SLVS-EC přímo z operačního systému Linux nebo vyčítat hodnoty ze stavového registru. Dále nastavovat DMA IP jádro nebo komunikovat s obrazovým senzorem přes SPI. V neposlední řadě je zde pomocí IP jádra Zynq Ultrascale+ propojeno DMA IP jádro s DDR4 řadičem a následně s externí RAM pamětí.

Nejdůležitější komponentou z pohledu zpracování dat v design_1.bd je ovšem podskupina SLVS_EC_data_processing. V této podskupině se nacházejí všechny komponenty pro zpracování a dekódování přijatých dat a také registry pro správné nastavení dekódovacích bloků.

2.4 Zpracování přijatých dat

Deserializovaná data a kontrolní vektory z GTH transceiveru jsou přivedeny do podskupiny SLVS_EC_data_processing v design_1.bd. Zjednodušené blokové schéma této podskupiny je na Obr. 2.6. Zde je každý kanál nejdříve zpracováván samostatně v blocích channel 0 až channel 3 a následně dojde k jejich spojení a synchronizaci v bloku Deskew buffer. Z Deskew bufferu jsou data přivedena do byte to pixel dekodéru. V byte to pixel dekodéru jsou data dekódována a jeho výstupem jsou surová obrazová data v podobě, ve které je vyžaduje software tyto data zpracovávající. Kromě zmíněných bloků se v podskupině SLVS-EC data processing nachází také kontrolní a stavový registr. Tyto registry jsou připojeny k AXI4 sběrnici a je možné jejich namapování a následný zápis nebo čtení přímo z operačního systému Linux.

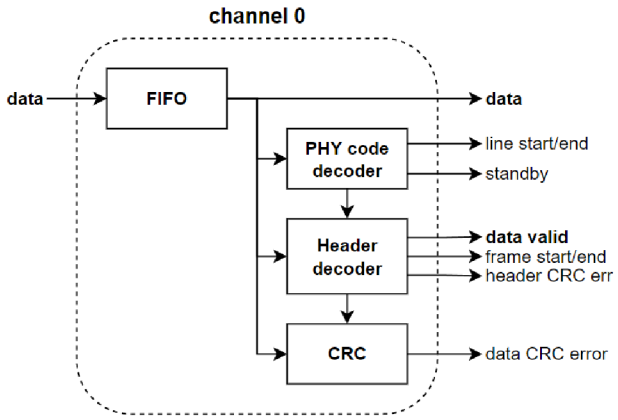


Obr. 2.6 Blokové schéma SLVS-EC data processingu

Bloky channel 0 až channel 3 jsou totožné a jejich zjednodušená vnitřní struktura je zobrazena na Obr. 2.7. Na vstupu jsou data s kontrolními vektory spojena do jednoho celku a přivedena do vstupního FIFO bufferu. Před tímto bufferem jsou data přenášena s hodinovým signálem regenerovaným z příchozích dat. Za tímto bufferem jsou data již na stejné hodinové doméně jako zbytek designu. Za vstupním FIFO bufferem jsou data a kontrolní vektory opět rozděleny. Data jsou vyvedena přímo a výstup bloku channel bez ohledu na to, zda se jedná o obrazová nebo jiná data. Data včetně kontrolních vektorů jsou přivedena do PHY code decoderu, kde jsou dekódovány kódy pro řízení fyzické vrstvy. Tyto kódy slouží pro označení začátku a konce paketu nebo například pro informaci o přechodu konkrétního kanálu do idle stavu.

Data ze vstupního FIFO bufferu a informace o začátku a konci paketu jsou předávána do header decoderu, kde je detekován začátek a konec snímku, zda se jedná o obrazová, embedded či prázdná data a další. Header decoder má zpoždění přesně jeden hodinový cyklus po přijmutí celé hlavičky. Díky tomuto zpoždění mohou být data ze vstupního FIFO bufferu přivedena přímo na výstup bloku channel. Pokud je v hlavičce uvedeno, že se v aktuálním paketu vyskytují obrazová data, následují tato data přímo za hlavičkou. Z tohoto důvodu je zpoždění právě jednoho hodinového taktu ideální a informace o validních obrazových datech se na výstupu z header decoderu objeví ve stejný čas jako obrazová data na výstupu FIFO bufferu.

Poslední komponenta v bloku channel je označena jako CRC a jejím cílem je ověření kontrolního součtu přenášených dat. Jelikož se tento kontrolní součet vypočítává pouze z obrazových dat, je nutné znát jejich začátek a konec. Z toho důvodu jsou do CRC přivedeny i informace z Header decoderu.



Obr. 2.7 Blokové schéma bloku channel 0 až channel 3

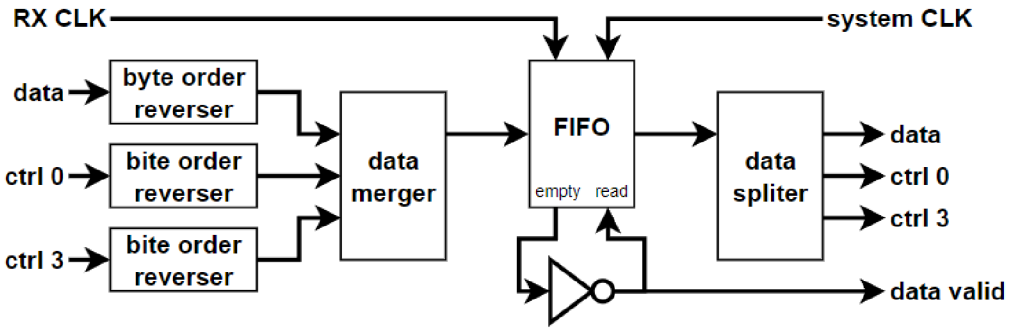
2.4.1 Vstupní FIFO

Hlavním úkolem vstupního bufferu je oddělení hodinových domén před ním a za ním. K tomuto účelu se využívají buffery typu FIFO, tedy First In First Out. Data na vstupu jsou do bufferu nahrávána s hodinovým signálem regenerovaným z dat přicházejících z obrazového senzoru. Tento hodinový signál je společně s daty výstupem GTH transceiveru a na Obr. 2.8 je tento hodinový signál označen jako RX CLK. Zatím co data na výstupu bufferu jsou již synchronizována s hodinovým signálem používaným ve zbytku designu. Tento hodinový signál je na Obr. 2.8 označen jako system CLK.

Na Obr. 2.8 je zakresleno blokové schéma vstupního FIFO bufferu a jeho nejbližšího okolí. Z levé části vstupují kromě regenerovaného hodinového signálu také data a dva ze čtyř kontrolních vektorů. Ve vektoru ctrl0 jsou obsaženy informace, zda a na jakých pozicích byly v současných datech obsaženy K charaktery z 8b10b kódování. A ve vektoru crtl3, zda byla současná data správně dekódována z platných 10 b slov z 8b10b kódování. Zbylé vektory nejsou v designu nijak využívány, takže není nutné jejich přivedení.

Vstupní data přicházejí z GTH receiveru v opačném pořadí bajtů, než ve kterém jsou definovány sekvence v SLVS-EC standardu používané pro kódy fyzické vrstvy. Z toho důvodu je nejdříve u dat otočeno pořadí bajtů v byte order reverser a u kontrolních vektorů otočeno pořadí bitů v bite order reverser. V kontrolních vektorech náleží každý jeden bit právě jednomu bytu ze souběžných dat. Z tohoto důvodu je u kontrolních vektorů otočeno pořadí bitů, nikoliv bytů.

Po správném seřazení jsou data i kontrolní vektory spojeny do jednoho a nahrány do FIFO bufferu. Na výstupu bufferu jsou data i kontrolní vektory opět rozloženy. Pro realizaci FIFO bufferu, který umí přijímat a vysílat data na různých hodinových doménách je zde využito IP jádro od firmy Xilinx. Toto IP jádro má pro vstupní data pouze jeden port. Stejně tak pro výstupní data. Z tohoto důvodu je zde přistoupeno ke sloučení a následnému rozdělení dat s kontrolními vektory.



Obr. 2.8 Blokové schéma propojení vstupního FIFO bufferu

Jelikož v bufferu nejsou stále obsažena data, která by bylo možné číst, je potřeba určit, kdy se mají data z bufferu vyčítat. K tomuto účelu je zde využit výstupní signál empty, který informuje o vyprázdnění bufferu. Pokud tento signál není v log 0, Jsou v bufferu obsažena data a je možné je vyčítat. Jak je patrné z Obr. 2.8 je tento signál negován a přiveden zpět do bufferu na vstupní port povolující vyčítání dat z bufferu. Tedy port read. Zároveň je tento již znegovaný signál použit pro označení validních dat dále v designu.

Bloky pro otáčení pořadí bytů a bitů jsou stejně jako bloky spojující a rozdělující data a kontrolní vektory pouze kombinační logika. Proto není třeba jejich připojení k hodinovému signálu a zároveň nezpůsobují zpoždění, které by bylo potřeba řešit umělým zpožďováním signálu data valid.

2.4.2 PHY code decoder

Dekodér kódů fyzické vrstvy kontroluje, zda se v přijatých datech označují sekvence definované SLVS-EC standardem. Tyto sekvence využívají 10 b slov z 8b10b kódování. Jelikož je využit 8b10b dekodér umístěný v GTH transceiveru, pracuje se zde s již dekódovanými 8 b daty. Z těchto dekódovaných dat není možné zjistit, zda byly dekódovány z datových nebo kontrolních slov definovaných v 8b10b kódování. Je tedy nutné tuto informaci předávat samostatně. K tomuto účelu slouží kontrolní vektor ctrl0. Jelikož se během přenosu mohou vyskytnou chyby, a ne všechny kombinace 10 b slov jsou v 8b10b kódování využity. Je možné, že budou přijata data, která nelze pomocí základní tabulky dekódovat. Pokud k tomuto případu dojde, je nastaven příslušný bit v kontrolním vektoru ctrl3. Všechny kontrolní vektory mají pouze osminovou bitovou šířku oproti datům. Každý jednotlivý bit v kontrolním vektoru totiž náleží právě jednomu bajtu v datech.

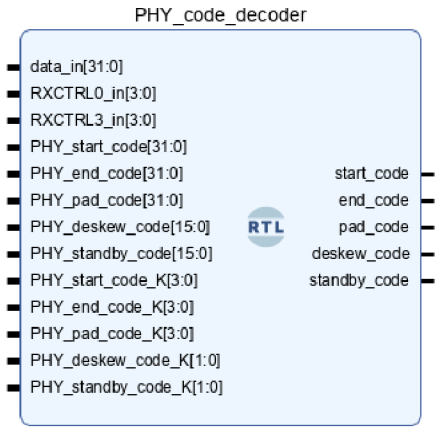
SLVS-EC standard má definované hodnoty pro zde hledané sekvence. Jedná se nicméně pouze o výchozí hodnoty, které je možné měnit. Pokud ke změně dojde, je ovšem nutné, aby byly nastaveny stejné hodnoty jak na straně vysílače, tak na straně přijímače. I když jejich změna tedy není nezbytná a celý design by mohl pracovat pouze s výchozími hodnotami. Případně jinými konstantními hodnotami nastavenými před syntézou. Je zde implementována možnost snadno měnit podobu těchto sekvencí i v již implementovaném designu. Nejen pro tuto funkcionalitu je zde implementován kontrolní registr, do kterého je možný přístup přímo z nadřízeného operačního systému. Viz kapitola 2.5.1

Dekodér kódů fyzické vrstvy tedy porovnává přijatá data s daty uloženými v kontrolním registru. Neslouží ovšem k detekci všech sekvencí definovaných v SLVS-EC standardu. Například sekvence pro správnou regeneraci a synchronizaci hodinového signálu není totiž nikde v designu využita. Tato sekvence je potřebná pouze v GTH transceiveru. V tomto designu je zapotřebí detekovat začátek a konec paketu. K tomuto účelu slouží signály start_code a end_code. Jelikož není možné datový přenos dle SLVS-EC standardu pozastavit, je nutné neustále vysílat data. Může ovšem nastat situace, kdy obrazový senzor nemá zrovna žádná data, která by mohl odvysílat. Pokud tato situace

nastane během vysílání paketu, je nutné odvysílat speciálně označená data, aby nedošlo k jejich zamíchání mezi data obrazová. V tomto případě se využívá k vyplnění prostoje PAD kód. Jeho detekování je indikováno signálem pad_code.

Pokud přechází obrazový senzor nebo alespoň některé kanály do standby stavu. Je obrazovým senzorem opět odvysílána sekvence, která o této skutečnosti předá informaci přijímači. Standby stav je využit i při změně konfigurace komunikace. Detekování této sekvence je indikováno signálem standby_code, který je přiveden do stavového registru. Ze stavového registru je poté možné zjistit, které kanály se nacházejí ve standby módu. Aby bylo možné ve stavovém registru změnit stav kanálu zpět na aktivní, je přiveden signál deskew_code. Ten signalizuje příjem sekvence pro synchronizaci kanálů. Tuto sekvenci vysílá obrazový senzor jak při začátku komunikace, tak i během ní.

Dekodér kódů fyzické vrstvy je složen pouze z kombinační logiky, a tudíž nezpůsobuje zpoždění vzhledem k hodinovému signálu.

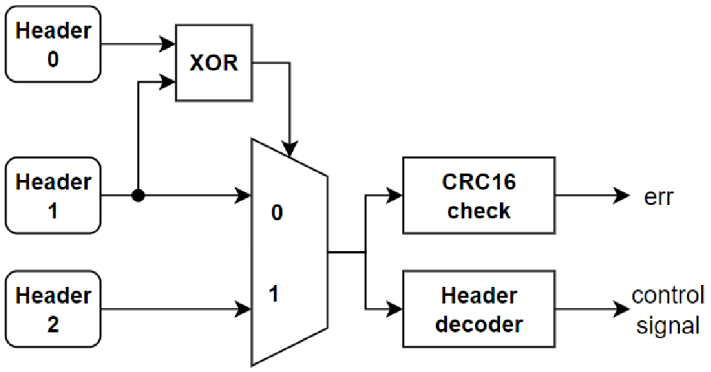


Obr. 2.9 Dekodér kódů fyzické vrstvy

2.4.3 Header decoder

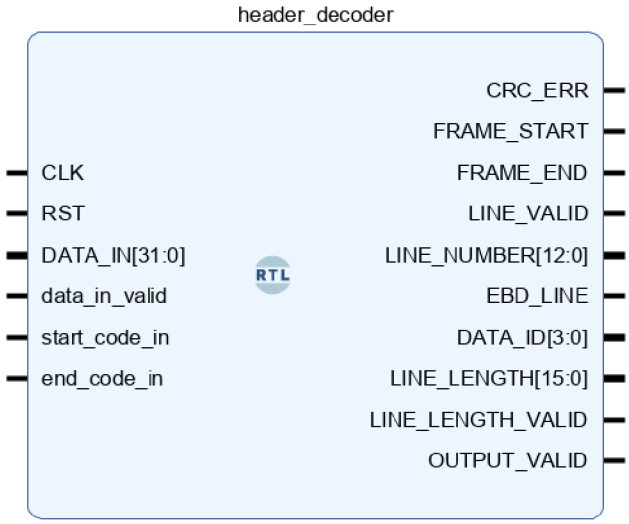
Účelem header dekodéru je přijmout hlavičku paketu, zkontrolovat, zda nedošlo při jejím přenosu k chybám a dekódovat v ní obsažená data. Z hlediska obrazových dat jsou zde obsaženy informace o začátku a konci snímku. Dále také zda jsou v aktuálním paketu obsažena obrazová data. Kromě informací čistě o obrazových datech jsou v hlavičce uvedeny také informace, zda jsou v paketu přenášena takzvaná embedded data. Obsah embedded dat není v SLVS-EC standardu nikterak definován a záleží pouze na konkrétním obrazovém senzoru a jeho nastavení. Dále je v hlavičce obsaženo číslo řádku neboli paketu v rámci jednoho vysílaného snímku. Toto číslování obsahuje ovšem i řádky na kterých nejsou obsažena obrazová data. Pokud by bylo číslo řádku někde využíváno, je potřeba tuto skutečnost mít na paměti. Poslední částí hlavičky je kontrolní součet CRC16. Na závěr je celá hlavička odvysílána ještě dvakrát.

Header dekodér po přijetí informace o začátku paketu z dekodéru fyzické vrstvy začne přijímat hlavičku paketu, která je přijata celkem třikrát. Pokud dojde během přenosu dat k chybě, projeví se tato chyba změnou hodnoty bitu. Konkrétní bit tedy nebude mít ve všech třech hlavičkách stejnou hodnotu. V tom případě se využije hodnota, která se vyskytuje častěji. Pokud budou například pro jeden byt přijaty hodnoty ‘0‘ ‘1‘ ‘0‘, bude použita hodnota ’0’. Pokud budou přijaty hodnoty ‘1’ ‘1’ ‘0’, bude použita hodnota ‘1’. Pro vyhodnocení správné hodnoty je nejdříve nad jednotlivými bity z první a druhé hlavičky provedena funkce XOR. Pokud se hodnota konkrétního bitu v první a druhé hlavičce neshoduje, je využita hodnota ze třetí hlavičky. Jelikož hodnota bitu může nabývat pouze dvou hodnot a nachází se v první a druhé hlavičce rozdílné hodnoty, je hodnota ve třetí hlavičce hodnotou většinovou. Pokud se v první a druhé hlavičce nachází stejná hodnota, není třeba třetí hlavičku vůbec uvažovat a je využita hodnota z druhé hlavičky. V tomto případě je bez ohledu na hodnotu ve třetí hlavičce totiž hodnota první hlavičky vždy většinová. Grafické znázornění této operace je zakresleno na Obr. 2.10.



Obr. 2.10 kontrola a korekce dat hlavičky

Po kontrole a redukci tří hlaviček do jedné je hlavička podrobena kontrolnímu cyklickému redundantnímu součtu CRC16. Polynom tohoto součtu stejně jako počáteční hodnota je definována ve standardu SLVS-EC. Pokud při kontrole tohoto součtu vyjde nevyhovující výsledek, jsou data z hlavičky zahozena a zbytek paketu není označen za validní. Při nevyhovujícím výsledku je navíc tato informace vyvedena skrze CRC_ERR port. Signál z tohoto portu je doveden do stavového registru, kde je pro pozdější diagnostiku vytvořen čítač těchto chyb. Jelikož jsou obrazová data rozložena do všech aktuálně využívaných kanálů způsobem, který neumožňuje vynechání informací z byť jen jediného kanálu, jsou v případě nevyhovujícího kontrolního součtu v jakémkoliv aktuálně využívaném kanálu zahozena data i z ostatních kanálů. V tomto případě dojde tedy ke ztrátě jednoho řádku obrazu.



Obr. 2.11 Dekodér hlavičky paketu

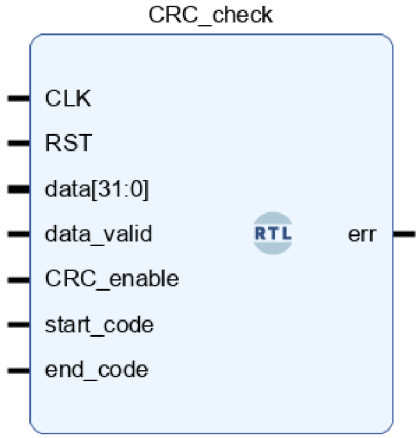
Pokud vyjde kontrolní součet správně, je hlavička rozložena na jednotlivé signály. Z pohledu navazujícího designu je nejdůležitější signál označující validní obrazová data LINE_VALID a signál označující konec snímku FRAME_END. Jelikož zpracování dat hlavičky je přesně jeden hodinový cyklus a obrazová data následují ihned za hlavičkou. Objeví se informace o validních obrazových datech na výstupu header dekodéru právě ve chvíli, kdy jsou již dostupná obrazová data. Signál informující o konci snímku je pak použit pro DMA, kde způsobí přeskočení na začátek další stránky v paměti. Každý snímek je totiž ukládán na samostatnou stránku.

Kromě zmíněných informací mohou být v hlavičce obsaženy informace o délce jednoho řádku. Tato informace není ovšem v hlavičce obsažena pokaždé. Záleží na nastavení obrazového senzoru. Pokud je délka řádku v hlavičce obsažena, promítne se její hodnota na výstupním portu LINE_LENGTH a signál LINE_LENGTH_VALID je nastaven do log 1. Dále je v hlavičce zapsáno ID dat v paketu. Toto rozlišování se používá, pokud je využita funkce multi-stream. Zde popisovaný design tuto funkci ovšem nepodporuje.

2.4.4 CRC

Kromě kontrolního součtu v hlavičce paketu je možné provést kontrolní součet i u obrazových dat v rámci jednoho paketu. Tato funkce je ovšem volitelná a je tedy nutné její nastavení jak v obrazovém senzoru, tak v kontrolním registru SLVS-EC přijímače. Kontrolní součet nikterak neovlivňuje zpracovávaná data a blok pro jeho kontrolu tedy může ležet mimo datovou cestu. Nezpůsobuje tedy žádné další zdržení při přenosu dat.

Pro správný výpočet kontrolního součtu je nutné znát z jakých dat má být počítán. Z tohoto důvodu je do CRC bloku přiveden signál nesoucí informaci o začátku paketu, konci paketu a o validitě obrazových dat. Pokud na konci paketu nevyjde požadovaný kontrolní součet, je tato informace pomocí signálu err přivedena do stavového registru. Zde se zapíše informace, ve kterém kanálu došlo k chybě kontrolního součtu obrazových dat a inkrementuje se čítač chyb obrazových dat.

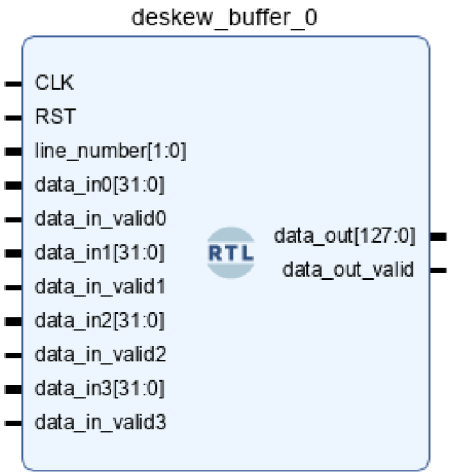


Obr. 2.12 Blok kontrolního součtu dat

2.4.5 Deskew buffer

Pokud jsou přenášena obrazová data s využitím více kanálů, je potřeba tato data spojit dohromady. V ideálním případě by synchronizace mezi jednotlivými kanály probíhala již v GTH transceiverech a následně by měly mít všechny kanály stejné zpoždění. Může se ovšem stát, že mezi daty budou přidány PAD kódy, které je potřeba odstranit a následně všechny kanály mezi sebou opět synchronizovat. Spojení dat z aktivních kanálů a jejich synchronizaci zajišťuje deskew buffer.

Pro správnou funkci deskew bufferu je kromě přivedení dat a jejich validace nutné znát také počet aktuálně aktivních kanálů. Tato informace se může měnit v závislosti na nastavení obrazového senzoru a její znalost je kritická pro správné zpracování obrazových dat. Počet aktuálně aktivních kanálů je uložen kontrolním registru a do deskew bufferu je přiveden na vstupní port line_number. Pokud by bylo aktivních méně, kanálů, než kolik je jich uvedeno v kontrolním registru, buffer by nečekal na přijetí dat ze všech kanálů. Kanály neuvedené v kontrolním registru by poté nebyly synchronizovány a nebyly by na výstupu bufferu zobrazeny. Pokud by bylo naopak v kontrolním registru zapsáno více aktivních kanálů, než přes které reálně probíhá komunikace, čekal by buffer na data z neaktivních kanálů a velmi brzy by došlo k jeho přetečení a ztrátě dat. V tomto případě by se na výstup bufferu nedostala žádný data. Aby k těmto situacím nedošlo, je možné zkontrolovat, zda se počet aktivních kanálů v kontrolním a stavovém registru shoduje.



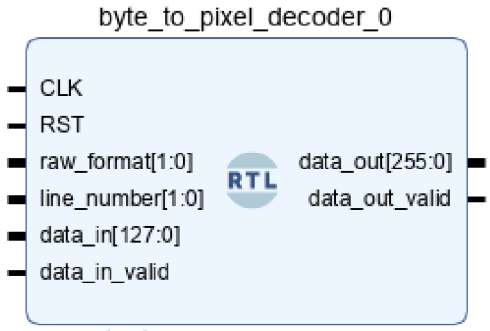
Obr. 2.13 Buffer spojující data z jednotlivých kanálů

2.4.6 Byte to pixel decoder

Po synchronizaci a spojení obrazových dat z jednotlivých kanálů je nutné jejich dekódování. Software firmy PSI zpracovávající obrazová data vyžaduje jednotlivé pixely seřazené za sebou v pořadí, v jakém se nacházejí na řádku výsledného snímku.

Standart SLVS-EC umožňuje přenášet obrazová data ve formátech RAW8, RAW10, RAW12, RAW14 a RAW16. tedy pixely s bitovou hloubkou 8, 10, 12, 14 a 16 bitů. U formátu RAW8 zabírá jeden pixel velikost přesně jednoho bajtu a ty to pixely jsou ukládány do paměti jeden za druhým. Obdobně formát RAW16, akorát s tím rozdílem, že jsou pro uložení jednoho pixelu využity dva bajty. Zbylé formáty RAW10, RAW12 a RAW14 jsou ukládány od LSB a zarovnány nulami do velikosti dvou bajtů.

Přenos dat z obrazového senzoru probíhá po celých bajtech. Respektive v desetibitových slovech dekódovaných do bajtů. Kromě formátu RAW8 a RAW16 nejsou bitové hloubky pixelů podporovaných formátů násobkem osmi. Tedy jednoho bajtu. Standard SLVS-EC místo neefektivního dorovnávání nulami každého pixelu do násobků bajtu, rozloží pixely na menší části a ty efektivně distribuuje po celých bajtech. I když by to bylo možné, nedochází zde k jednoduchému řazení pixelů za sebe, ale ke složitějšímu zamíchání na bitové úrovni. Kromě tohoto zamíchání jsou obrazová data ještě rovnoměrně rozprostřena mezi jednotlivé kanály. Díky tomuto systému skládání obrazových dat nejsou zbytečně přenášena redundantní data. Největší rozdíl efektivity je zde u formátu RAW10. Při využití rozložení bitů dle standardu SLVS-EC využijeme pro přenos čtyř pixelů pouze 5 bajtů. Pokud bychom dorovnávali každý pixel nulami do celých bajtů, bylo by pro přenos čtyř pixelů potřeba přenést 8 bajtů.

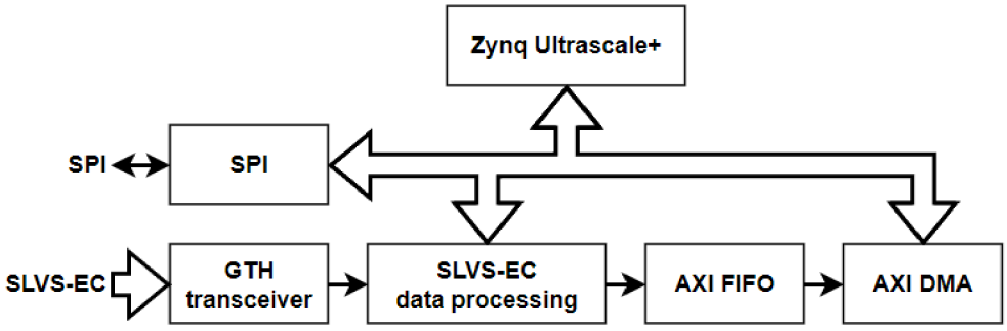


Obr. 2.14 Dekodér obrazových dat

Výstupem z byte to pixel dekodéru jsou již surová obrazová data ve formátu, který vyžaduje software tyto data zpracovávající. Výstupní data v této podobě mohou být tedy již přímo ukládány do RAM paměti, odkud budou vyčítána operačním systémem.

2.5 Komponenty připojené k AXI sběrnici

Pro zajištění konektivity mezi operačním systémem a SLVS-EC přijímačem je využito IP jádro Zynq Ultrascale+ firmy Xilinx, které je propojeno se všemi potřebnými komponentami pomocí AXI4 sběrnice. Toto propojení je znázorněno na Obr. 2.15.



Obr. 2.15 struktura AXI4 sběrnice

Pro komunikaci s obrazovým senzorem je zde využito SPI rozhraní, pomocí kterého je možné nastavovat a vyčítat parametry komunikace a snímání obrazu. Tato komunikace ani nastavení obrazového senzoru není cílem této práce. Pro nastavení parametrů senzoru je nutné nahlédnout do dokumentace konkrétního modelu.

Pro nastavení parametrů SLVS-EC přijímače jsou bloku SLVS-EC data processing zřízeny dva registry připojené k AXI4 sběrnici. Jeden registr je označen jako kontrolní a slouží k nastavování parametrů přijímače a dekodéru. V tomto registru se nachází

například kódy fyzické vrstvy nebo bitová hloubka pixelů. Druhým registrem je stavový registr. V tomto registru je zapsáno například které kanály byly detekovány jako aktivní nebo kolik se vyskytlo detekovaných chyb při příjmu dat.

Poslední komponentou připojené k AXI4 sběrnici je DMA neboli Direct Memory Access. Tato komponenta obstarává streamování přijatých a dekódovaných dat skrze DDR4 řadič do externí RAM paměti. Před DMA je zařazen ještě FIFO buffer, který zajistí správné propojení DMA s blokem SLVS-EC data processing. Tento buffer je zde nutný z důvodu nesynchronizovaného čtení a zápisu a obou zmíněných bloků.

2.5.1 Registry k obsluze SLVS-EC

Aby bylo možné nastavovat parametry pro správný příjem a dekódování dat, je v designu implementován kontrolní registr. Tento registr je připojen pomocí AXI4 sběrnice, aby bylo možné do něj zapisovat přímo z operačního systému. Kontrolní registr má velikost 5 x 32 b a je v něm vyhrazeno místo pro 8 b podobu všech kontrolních kódů fyzické vrstvy. K 8 b podobě těchto kódů je zapotřebí také uvést, zda se jedná o bajt zakódovaný v 8b10b kódování jako datové nebo kontrolní slovo. K tomuto účelu je v kontrolním registru vyhrazené místo.

Například kód označující začátek paketu se skládá ze čtyř bajtů. Ve výchozím stavu je tato sekvence tvořena slovy z 8b10b kódování jako K.28.5 - K.27.7 - K.28.2 - K.27.7 [1]. V části PHY_START_CODE je uložena 8b podoba těchto slov a v části označené jako PHY_START_CODE_K je definováno, zda se jedná o kontrolní nebo datové slovo. V případě výchozí hodnoty start kódu jsou všechna tato slova kontrolní.

Všechny sekvence kontrolních kódů fyzické vrstvy nejsou ovšem definované jako 4 slova, přičemž každé může být jiné. Například synchronizační kód využívá pouze dvou slov, přičemž je druhé slovo třikrát zopakováno. Díky tomu je možné tento kód definovat pouze dvěma slovy. Podobný princip platí i pro některé další kódy, Jejich definice tedy v kontrolním registru zabírá pouze dva bajty pro definici 8 b podoby slov a dva bity pro označení, zda se jedná o datové nebo kontrolní slovo.

Všechny kódy fyzické vrstvy mají délku přesně čtyř slov. Speciální případ poté tvoří pouze idle kód, který je definován jako pouze jedno slovo. Obvykle je toto slovo odvysíláno čtyřikrát, aby byla zajištěna konzistence dat. Přesný počet opakování ovšem není pevně stanoven. Idle kód je navíc ve výchozím stavu nastaven jako datové slovo. Z těchto dvou důvodů se nevyužívá například uvnitř paketu.

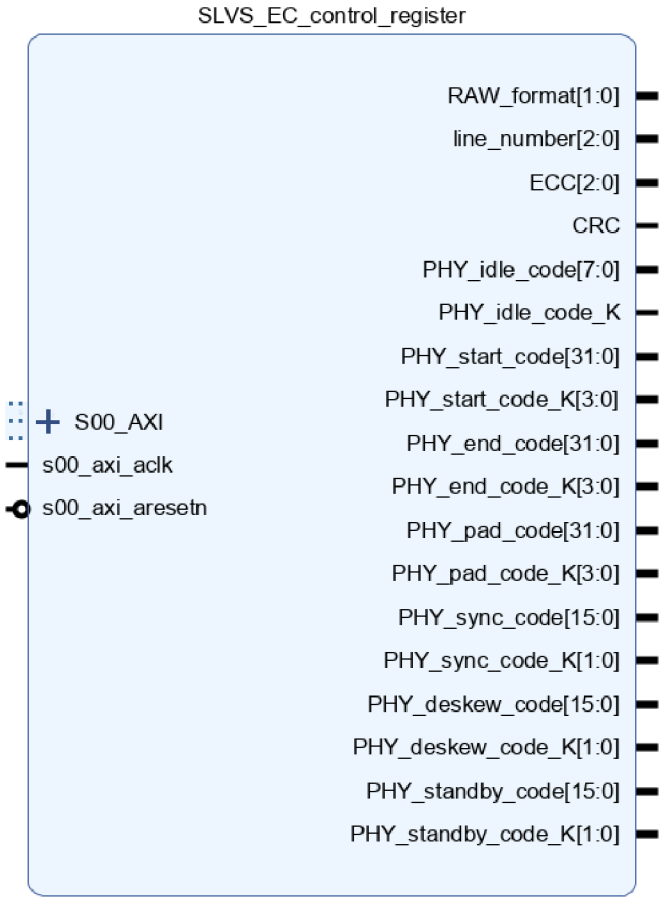
Kromě definice kódů fyzické vrstvy se v kontrolním registru nachází prostor označený jako CRC a ECC. V tomto prostoru je možné zapnout a nastavit kontrolu přijatých obrazových dat. Jedná se o kontrolu obrazových dat uvnitř paketu, ale také například v zápatí paketu. Obě tyto možnosti kontroly je nutné nastavit v obrazovém senzoru a v přijímači stejně. Pokud by se nastavení neshodovalo, byla by data chybně dekódována.

Všechny výše popsaná uložená data v kontrolním registru by bylo možné vypustit a uvažovat výchozí hodnoty. Pro širší možnosti nastavení a větší univerzálnost je ovšem implementována možnost tyto parametry měnit. To ovšem neplatí pro formát přenášených dat a počet kanálů, které jsou pro přenos využity. Uvažuje se s využitím tohoto designu pro příjem dat z různých senzorů. Převážně bude využíváno největší bitové hloubky pixelů a nevyššího možného počtu kanálů. Některé obrazové senzory vyráběné firmou SONY ovšem neumožňují přenos dat skrze 4 kanály a formát RAW16. Například obrazový senzor IMX433 disponuje pouze jedním SLVS-EC kanálem a bitovou hloubkou pouze 12 b [7]. Z tohoto důvodu je v kontrolním registru implementována možnost měnit tyto parametry.

Pro volbu formátu přijímaných dat je zde vyhrazen prostor o velikosti dvou bitů. Tento prostor umožňuje definovat pouze čtyři možnosti. Standard SLVE-EC ovšem umožňuje přenos obrazových dat v pěti různých formátech. Čtyři možnosti jsou ovšem dostatečné, protože dekódování formátu RAW8 a RAW16 probíhá, s ohledem na požadovaný výsledný formát dekódovaných dat, stejně. Správné nastavení formátu přijímaných dat je uvedeno v Tab. 5.

Druhou nepostradatelnou proměnou je počet kanálů, skrze které probíhá přenos obrazových dat. Zde se uvažuje při využití menšího počtu kanálů, než jsou čtyři, s jejich postupným zaplněním od kanálu 0. V případě designu implementovaného na zvolené MPSoC je možné využít maximálně 4 kanály. Standart SLVS-EC ovšem umožňuje využití až osmi kanálů pro jeden přenos. Z důvodu kompatibility kontrolního registru při přechodu na jinou verzi MPSoC je zde pro volbu počtu kanálů vyhrazen prostor tří bitů. V tomto prostoru je možné definovat využití všech osmi kanálů, i když v tomto designu nejsou implementovány.

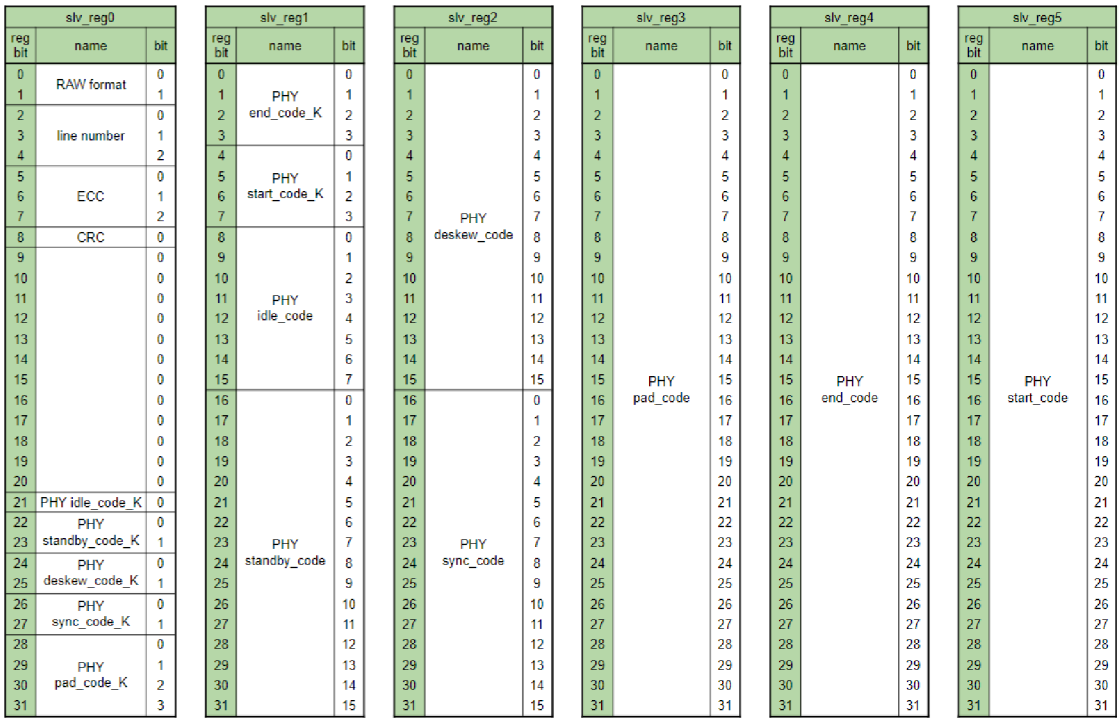
Zbylý prostor v kontrolním registru je možné využít například pro nastavení multi streamu. Multi stream není v tomto designu ovšem implementován a není zde tedy jeho nastavení tedy třeba.



Obr. 2.16 Kontrolní registr

Tab. 5 nastavení formátu obrazových dat v kontrolním registru

RAW formát	bit	
	0	1
RAW8	0	0
RAW10	0	1
RAW12	1	0
RAW14	1	1
RAW16	0	0



slv_reg0		
reg bit	name	bit
0	RAW format	0
1		1
2	line number	0
3		1
4		2
5	ECC	0
6		1
7		2
8	CRC	0
9		0
10		0
11		0
12		0
13		0
14		0
15		0
16		0
17		0
18		0
19		0
20		0
21	PHY idle_code_K	0
22	PHY standby_code_K	0
23		1
24	PHY deskew_code_K	0
25		1
26	PHY sync_code_K	0
27		1
28	PHY pad_code_K	0
29		1
30		2
31		3

slv_reg1		
reg bit	name	bit
0	PHY end_code_K	0
1		1
2		2
3		3
4	PHY start_code_K	0
5		1
6		2
7		3
8	PHY idle_code	0
9		1
10		2
11		3
12		4
13		5
14		6
15		7
16	PHY standby_code	0
17		1
18		2
19		3
20		4
21		5
22		6
23		7
24		8
25		9
26		10
27		11
28		12
29		13
30		14
31		15

slv_reg2		
reg bit	name	bit
0	PHY deskew_code	0
1		1
2		2
3		3
4		4
5		5
6		6
7		7
8		8
9		9
10		10
11		11
12		12
13		13
14		14
15		15
16	PHY sync_code	0
17		1
18		2
19		3
20		4
21		5
22		6
23		7
24		8
25		9
26		10
27		11
28		12
29		13
30		14
31		15

slv_reg3		
reg bit	name	bit
0		0
1		1
2		2
3		3
4		4
5		5
6		6
7		7
8		8
9		9
10		10
11		11
12		12
13		13
14		14
15	PHY pad_code	15
16		16
17		17
18		18
19		19
20		20
21		21
22		22
23		23
24		24
25		25
26		26
27		27
28		28
29		29
30		30
31		31

slv_reg4		
reg bit	name	bit
0		0
1		1
2		2
3		3
4		4
5		5
6		6
7		7
8		8
9		9
10		10
11		11
12		12
13		13
14		14
15	PHY end_code	15
16		16
17		17
18		18
19		19
20		20
21		21
22		22
23		23
24		24
25		25
26		26
27		27
28		28
29		29
30		30
31		31

slv_reg5		
reg bit	name	bit
0		0
1		1
2		2
3		3
4		4
5		5
6		6
7		7
8		8
9		9
10		10
11		11
12		12
13		13
14		14
15	PHY start_code	15
16		16
17		17
18		18
19		19
20		20
21		21
22		22
23		23
24		24
25		25
26		26
27		27
28		28
29		29
30		30
31		31

Obr. 2.17 Mapa kontrolního registru

Druhým registrem v designu je stavový registr. Tento registr je tvořen 32 b a stejně jako předchozí registr, je i tento připojen k AXI4 sběrnici. Je tedy možné k němu přistupovat přímo z operačního systému. Na rozdíl od předchozího registru slouží stavový registr primárně k předávání informací směrem z designu přijímače do operačního systému.

První čtyři bity v registru nesou informaci, zda jsou kanály k nim přiřazené aktivní. Ve výchozím stavu se nachází všechny kanály v neaktivním – standby stavu. Po přijetí a dekódování deskew sekvence, se zobrazí příslušné kanály jako aktivní. Zpět do standby módu budou v registru opět přepnuty až po přijetí standby sekvence na příslušném kanálu nebo po restartování a uvedení hodnot do výchozího stavu. Informace ze stavového registru o aktivních kanálech není v designu nikterak zpracovávána. Slouží pouze pro Případnou kontrolu operačnímu systému.

Zbylá část registru je zaplněna informacemi o detekovaných chybách způsobených během přenosu dat z obrazového senzoru. Ve standardu SLVS-EC se při přenosu dat nacházejí na několika místech kontrolní kódy. V hlavičce paketu se jedná o kontrolní redundantní součet CRC16. Tento kontrolní součet je v hlavičce paketu obsažen vždy. Zbylé kontrolní kódy se týkají už pouze obrazových dat a mohou být přenášeny nejen přímo mezi obrazovými daty, ale i na konci paketu. Tyto kontrolní kódy jsou volitelné a je možné je nastavit do vícero variant. Toto nastavení se ovšem musí shodovat jak v registrech obrazového snímače, tak v kontrolním registru SLVS-EC přijímače.

Pokud nastane nevyhovující výsledek kontrolního součtu v hlavičce paketu, je tato informace promítnuta do stavového registru na místě header err příslušícím konkrétnímu kanálu. Pokud jsou v přenášených obrazových datech zakomponovány kontrolní kódy a dojde zde ke zjištění nevyhovující hodnoty, je tato informace promítnuta obdobně jako v předchozím případě na místo náležící konkrétnímu kanálu označené jako data err. Tímto způsobem je přenášena informace, ve kterých kanálech a na jakém místě dat se vyskytla chyba. Není zde ale zohledněno, zda se vyskytla chyba pouze jednou nebo vícekrát. K tomuto účelu je zbylý prostor registru využit pro čítače. Jeden čítač slouží pro čítání chyb nalezených v hlavičkách paketů. Zatím co druhý čítač informuje o četnosti chyb v obrazových datech.

Informace o chybách v hlavičce či obrazových datech nejsou v designu nikterak využity. Mají pouze informativní charakter pro operační systém v případě řešení problémů s komunikací či při statistice. Oba čítače chyb mohou pouze zvyšovat svoji hodnotu až do maxima, tedy 1023 chyb. Poté se zastaví a k dalšímu přičítání či vynulování registru již nedochází. Hodnoty čítačů, stejně jako informace o umístění odhalených chyb je možné vynulovat pouze z operačního systému. Tímto je zajištěno, že žádná z těchto informací operačnímu systému neunikne.

slv_reg0		
reg bit	name	bit
0	ch0 standby	0
1	ch1 standby	0
2	ch2 standby	0
3	ch3 standby	0
4	ch0 header err	0
5	ch1 header err	0
6	ch2 header err	0
7	ch3 header err	0
8	ch0 data err	0
9	ch1 data err	0
10	ch2 data err	0
11	ch3 data err	0
12	header err counter	0
13		1
14		2
15		3
16		4
17		5
18		6
19		7
20		8
21		9
22	data err counter	0
23		1
24		2
25		3
26		4
27		5
28		6
29		7
30		8
31		9

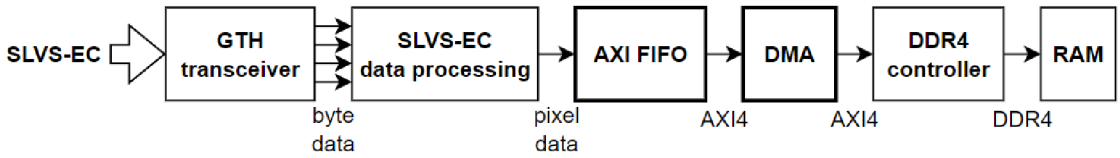
Obr. 2.18 Mapa stavového registru

2.5.2 FIFO buffer a DMA

Zpracovaná obrazová data je nutné uložit do externí RAM paměti. Z RAM paměti budou data vyčítána operačním systémem a přes ethernet odeslána na server k dalšímu zpracování softwarem firmy PSI.

Pro ukládání obrazových dat do RAM paměti je zde využito IP jádro DMA – Direct Memory Access, firmy Xilinx. Toto jádro obstarává, jak už název napovídá, přímý přístup do paměti a převádí AXI4 stream na memory map pro správné ukládání dat do paměti. Konfigurace tohoto IP jádra probíhá z operačního systému. Jádro samotné bylo navíc v potřebné konfiguraci využito již v předchozích projektech firmy PSI. Není tedy nutné se zde jeho konfigurací zabývat.

Jelikož DMA nevyžaduje nutně data ve stejný čas, ve který jsou produkována blokem SLVS-EC data processing, je mezi tyto dva bloky umístěn FIFO buffer. Ten zde na rozdíl od FIFO bufferu použitého výše neplní synchronizační funkci mezi dvěma časovými doménami, ale zajišťuje bezproblémové spojení částí s rozdílnou přenosovou rychlostí.



Obr. 2.19 Zařazení FIFO a DMA do datové cesty

2.5.3 SPI

Jelikož SLVS-EC komunikace slouží pouze pro zasílání dat směrem z obrazového senzoru do MPSoC, je potřeba zřídit ještě jiný komunikační kanál, aby bylo možné zasílat data směrem so senzoru. Do senzoru je nutné zasílat data pro správné nastavení komunikace, jako je například počet kanálů nebo využití kódů sloužících k detekci chyb způsobených při přenosu. V obrazovém senzoru je také nutné nastavit správné snímání obrazu. Zde se jedná například o bitovou hloubku, rozlišení nebo oblast snímání.

K nastavení senzorů je možné využít komunikaci pomocí SPI nebo I2C, přičemž díky rychlosti přenosu dat je preferované využití SPI rozhraní. Nastavení obrazového senzoru není součástí této práce a ke správnému nastavení jeho vnitřních registrů je nutné nahlédnout do neveřejné dokumentace konkrétního senzoru nebo modelové řady.

3. SIMULACE DESIGNU

Jelikož v době dokončení této diplomové práce není dostupný hardware pro otestování designu v reálných podmínkách. Byl design testován pouze v rámci simulací. K těmto simulacím byly využity implementované simulační nástroje v návrhovém prostředí Vivado 2021.

Testování designu proběhlo pro tři dílčí části, a pro celý design od vstupu SLVS-EC dat po výstup dekódovaných obrazových dat za byte to pixel dekodérem. Konfigurace následujícího FIFO bufferu a DMA, včetně ovladačů pro operační systém, je již otestovaná z předchozích projektů firmy PSI a není tedy nutné jejich opětovné testování v rámci této práce.

Mezi testované dílčí části designu patří GTH transceiver, u kterého bylo vzhledem k velkému množství parametrů, testováno jeho správné nastavení. Další testovanou dílčí částí je header dekodér. U header dekodéru bylo testováno správné vyhodnocení tří hlaviček paketu a následné rozložení do jednotlivých signálů označující například validní obrazová data nebo začátek a konec snímku.

Poslední dílčí testovanou částí byl byte to pixel dekodér. Vzhledem k jeho vnitřní struktuře, kde jsou nejdříve vstupní data poskládána do řady, není nutné jeho testování pro všechny varianty počtu kanálů. Z tohoto důvodu byl byte to pixel dekodér testován pouze pro dva aktivní kanály.

3.1 Postup testování

Pro testování header dekodéru bylo vzhledem k jeho jednoduchosti využito pouze jednoduchého test benche s pevně nastavenými vstupními daty. Pro ostatní testované dílčí části i testování celkového designu bylo ovšem potřeba velké množství vstupních dat. Jejich prostý zápis v rámci test benche by byl velice nepřehledný. Z toho důvodu bylo přistoupeno k vytvoření textového dokumentu se vstupními daty. Tento dokument je následně při simulaci otevřen a data v něm obsažená jsou promítána na vstupní porty testované části. Během simulace byte to pixel dekodéru a celého designu jsou výstupní data ukládána do druhého textového souboru pro pozdější kontrolu. K práci s textovými soubory při simulaci je využita knihovna std.textio. Díky této volbě zůstal test bench relativně krátký a přehledný.

Pro větší variabilitu řízení vstupních dat byl sestaven protokol pro textové soubory. Každý řádek začíná symbolem označujícím typ řádku. Pokud je na řádku napsaný komentář, je tento řádek označen #. Řádky komentářů nejsou nikterak promítány na vstupní porty testované komponenty. Jejich obsah je ovšem vypsán během testování do konzole, aby bylo možné sledovat průběh simulace. Tyto řádky jsou využívány například pro informování o začátku snímku nebo číslu řádku.

Při potřebě vyčkat určitý čas v nezměněném stavu je řádek označen písmenem W z anglického Wait. Za písmenem W následuje informace o délce čekání.

Jelikož jsou data v SLVS-EC standardu přenášena po diferenciálních párech, mají během přenosu dat oba signály opačnou polaritu. To ovšem neplatí, pokud je konkrétní kanál ve stavu idle nebo před začátkem vysílání. Pro tyto účely jsou definovány další dva typy řádků. Jeden je označen písmenem Z a vstupní port testované komponenty je v tomto případě nastaven do stavu vysoké impedance. Druhý typ je označen číslicí 0. V tomto případě jsou oba signály diferenciálního páru nastaveny do log 0. Pro oba tyto typy platí, že za označením typu řádku následuje časový údaj informující o délce trvání požadovaného stavu.

Posledním typem je řádek nesoucí informace o datech. Tento řádek je označen písmenem D, za kterým následují binárně zapsaná data pro testované kanály. Na konci řádku je poté uvedena perioda, ve které mají být tyto data promítána na vstup testované komponenty.

Úryvek využitého kódu pro dva kanály poté může vypadat následovně:

```
# time when this file was generated: 03:26:56
Z 5000000000
0 500
D 1010000011 1010000011 200
D 0101101010 0101101010 200
…
```

3.2 Využití jazyka Python

Jelikož je při testování potřeba velké množství vstupních dat, byl pro generování textových souborů využit skriptovací jazyk python. Například pro odvysílání jednoho snímku s rozlišením 256 x 100 pixelů s využitím dvou kanálů je potřeba téměř 20 000 řádků textového souboru. Díky využití tohoto jazyka je velice snadné také zakódování obrazových dat dle standardu SLVS-EC a následně pomocí 8b10b kódování.

Pro testování nebyly využity skutečné obrázky, ale generované vzory. Například lineární gradient od 0 do 256. Úryvek kódu pro generování takovéhoto gradientu pro snímek o velikosti 100 řádků, 256 sloupců s využitím dvou kanálů vypadá následovně:

```
for r in range(100):
    write_comment("start of data line " + str(r))
    write_start_code()
    write_header(pixel_data=True, line_number=line_number)
    for i in range(0,256,2):
        write_word(byte_to_enc= i, byte_to_enc1= i+1)
    write_end_code()
    write_deskew_code()
    for i in range(16):
        write_idle_code()
    line_number = line_number +1
```

Výstupem simulace je kromě grafického průběhu jednotlivých signálů také textový soubor s výstupními daty. Tento soubor je následně možné snadno zkontrolovat pomocí dalšího skriptu napsaného v jazyce python.

4. ZÁVĚR

Cílem diplomové práce byla implementace SLVS-EC standardu do nových přístrojů firmy PSI, které využívají platformu Xilinx Zynq Ultrascale+. Díky této implementaci je možné využít moderní IMX CMOS obrazové senzory místo doposud využívaných zastaralých CCD senzorů. V rámci této implementace mělo být navrženo IP jádro pro příjem obrazových dat skrze rozhraní SLVS-EC a následné ukládání dekódovaných dat do externí RAM paměti. IP jádro mělo být přizpůsobeno architektuře Zynq Ultrascale+ a využívat maximální možnou rychlost přenosu dat s ohledem na zvolenou architekturu a limitace SLVS-EC standardu.

V první části této diplomové práce jsou shrnuty veřejně dostupně informace o standardu SLVS-EC a popsány jeho výhody oproti staršímu standardu sub LVDS. V rámci této kapitoly jsou také popsány topologie zapojení obrazových snímačů, které SLVS-EC standard nabízí. Dále je zde uvedeno, jaké principy SLVS-EC standard využívá pro správnou inicializaci přijímače, rekonstrukci hodinového signálu z přenášených dat a vzájemnou synchronizaci jednotlivých kanálů. Dále je zde také uvedena podoba přenášených dat, detekce chyb způsobených při přenosu a vlastnosti a výhody 8b10b kódování.

V druhé kapitole je přistoupeno k výběru cílového MPSoC s architekturou Xilinx Zynq Ultrascale+ s ohledem k jeho hardwarovým prostředkům umožňující přijímat elektrické signály definované SLVS-EC standardem. Vzhledem k aktuální situaci na trhu s polovodiči, byl vybrán model MPSoC řady ZU5EV v pouzdře SFVC784. Tento MPSoC má hardwarové prostředky pro příjem dat skrze pouze čtyři vysokorychlostní diferenciální páry. Ve zbytku práce je s tímto faktem počítáno a celkový design není tedy navržen pro 8 kanálů, které SLVS-EC standard umožňuje, ale pouze pro 4.

Následuje popis a nastavení GTH transceiveru, který je zde využit pro příjem dat z obrazového senzoru. V rámci tohoto popisu je uvedeno využití kontrolních kódů fyzické vrstvy. Dále je zde zvolena šířka datové sběrnice s ohledem na možnosti zarovnání přijatých dat a zvolena frekvence hodinového signálu pro navazující design. Dále také popis využití 8b10b dekodéru, který je hardwarově implementován v GTH transceiveru. V této části jsou dále popsány možnosti pro synchronizaci jednotlivých kanálů a jejich propojení. Na závěr je zde uvedeno nastavení obnovy hodinového signálu z přijatých dat.

V další části je popsán navržený design a postup zpracování přijatých dat. Nejdříve jsou data zpracovávána samostatně pro každý kanál. Tato a další navazující části designu se nachází v jiné časové doméně. Pro bezproblémový přechod dat do jiné časové domény je zde tedy implementován FIFO buffer. Následně dochází k dekódování paketů a jejich hlavičky. Kromě toho je zde implementována kontrola chyb způsobených při přenosu dat mezi obrazovým senzorem MPSoC. Po dekódování paketů a extrakce obrazových dat jsou všechna obrazová data z aktivních kanálů opět synchronizována a spojena.

Do této chvíle byla data zakódována dle standardu SLVS-EC aby bylo dosaženo vyšší efektivity přenosu. Software firmy PSI pro zpracování obrazových dat ovšem vyžaduje jinou strukturu dat. Z toho důvodu byl navržen dekodér, který dokáže dekódovat data ze všech formátů podporovaných standardem SLVS-EC do podoby potřebné pro další zpracování. Informace o kódování bohužel firma SONY neposkytuje veřejně. Z toho důvodu není v této práci popsán konkrétní způsob zpracování těchto dat.

Jelikož SLVS-EC standard umožňuje konfiguraci kontrolních kódů fyzické vrstvy a kódů pro odhalování chyb způsobených při přenosu dat, jev designu implementován kontrolní registr. Této registr je připojen k AXI sběrnici, díky čemuž je pak možné z operačního systému nastavovat parametry pro správný příjem a dekódovaní dat z obrazového senzoru. Pro případnou diagnostiku je design opatřen i stavovým registrem, ve kterém jsou informace o aktivních kanálech a detekovaných chybách.

Jelikož nebyl v době dokončení této diplomové práce dostupný potřebný hardware, nebyl design otestován v reálných podmínkách. Pro testování bylo využito simulačních nástrojů v návrhovém prostředí Vivado 2021. Vzhledem k potřebě velkého množství vstupních dat a jejich zakódování bylo k ukládání těchto dat v externím textovém souboru. Pro jednodušší a přehlednější práci s těmito daty byl navržen také vlastní protokol. Pro generování potřebných dat bylo následně využito skriptovacího jazyka Python. Výstupní data ze simulací byla opět ukládána do textového souboru, aby byla možná jejich snadná kontrola pomocí vytvořeného skriptu.

Testování byly podrobeny tři dílčí části a následně i celý navržený design. Z dílčích částí bylo otestováno správné nastavení GTH transceiveru. Následně ověření funkce dekodéru hlavičky paketu a na závěr bloku dekódujícího obrazová data do požadované podoby. Při testování dekodéru obrazových dat byla otestována jeho funkce pouze při dvou aktivních kanálech. Vhledem ke struktuře kódování bylo toto otestování označeno za dostačující.

Po potřebných úpravách a odstranění chyb v kódech byl celý design úspěšně simulován s požadovanými výsledky. Pro otestování v reálných podmínkách je ovšem nutné vyčkat na dodání potřebného hardwaru. Následně bude možné design rozšířit na vyšší počet kanálů. Poté bude možné z designu vytvořit uživatelsky přívětivější IP jádro, které bude možné využít v budoucích projektech bez nutnosti studovat jeho strukturu.

LITERATURA

[1] UG0877. www.microchip.com [online]. [cit. 2022-5-23]. dostupné z: https://ww1.microchip.com/downloads/aemDocuments/documents/FPGA/ProductDocuments/SupportingCollateral/microsemi_polarfire_fpga_slvs_ec_user_guide_ug0877_v4.pdf

[2] Zynq Ultrascale+ MPSoC. www.xilinx.com [online]. [cit. 2022-05-23]. Dostupné z: https://www.xilinx.com/products/silicon-devices/soc/zynq-ultrascale-mpsoc.html

[3] ARM Cortex-A53. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-05-23]. Dostupné z: https://en.wikipedia.org/wiki/ARM_Cortex-A53

[4] SLVS-EC: What is SLVS-EC [online]. [cit. 2022-05-23]. Dostupné z: https://www.sony-semicon.co.jp/e/products/IS/industry/SLVS_EC_technology.html

[5] PG182: UltraScale FPGAs Transceivers Wizard v1.7. Docs.xilinx.com [online]. [cit. 2022-05-23]. Dostupné z: https://docs.xilinx.com/v/u/en-US/pg182-gtwizard-ultrascale

[6] UG576: UltraScale Architecture GTH Transceivers. Docs.xilinx.com [online]. [cit. 2022-05-23]. Dostupné z: https://docs.xilinx.com/v/u/en-US/ug576-ultrascale-gth-transceivers

[7] IMX433LLJ. Www.sony-semicon.co.jp [online]. [cit. 2022-05-23]. Dostupné z: https://www.sony-semicon.co.jp/products/common/pdf/IMX433LLJ_LQJ_Flyer02.pdf

[8] 8b10b Code. Newmascotcostumessupply.com [online]. [cit. 2022-01-03]. Dostupné z: https://newmascotcostumessupply.com/ru-decs/wiki/8b10b-Code

[9] 8b/10b encoding. Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 29.9.2021 [cit. 2022-01-03]. Dostupné z: https://en.wikipedia.org/wiki/8b/10b_encoding

SEZNAM ZKRATEK

AXI	Advanced eXtensible Interface
CCD	Charge Coupled Device
CMOS	Complementary Metal-Oxide-Semiconductor
CRC	Cyclic Redundancy Check
FIFO	First In, Firs Out
FPGA	Field Programmable Gate Array
LVDS	Low-Voltage Differential Signaling
MPSoC	Multiprocessor System on Chip
PSI	Photon Systems Instruments
SLVS-EC	Scalable Low-Voltage Signaling with Embedded Clock
RAM	Random Access Memory