**Závěrečná zpráva**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  | **Jméno** | | **Datum** |
| Autor: |  | Dmitrii Semenov | 22.12.2023 |
|  |  | Viktor Cejnek | 22.12.2023 |

1. document change log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Verze | Datum | Autor | Strany | Popis změn |
| 1.0 | 22.12.2023 | DSem, VCej | Všechny | První verze |

1. Obsah

[1 Úvod 4](#_Toc146618245)

[2 Aplikovatelné a Odkazované dokumenty 5](#_Toc146618246)

[2.1 Seznam aplikovatelných dokumentů 5](#_Toc146618247)

[2.2 Seznam odkazovaných dokumentů 5](#_Toc146618248)

[3 Definice a seznam zkratek 6](#_Toc146618249)

[3.1 Definice 6](#_Toc146618250)

[3.2 Psaní čísel 6](#_Toc146618251)

[3.3 Jednotky 6](#_Toc146618252)

[3.4 Zkratky 6](#_Toc146618253)

[4 Představení projektu 7](#_Toc146618254)

[5 Plán vývoje 8](#_Toc146618255)

[6 Popis Návrhu 9](#_Toc146618256)

[7 Verifikační Plán 10](#_Toc146618257)

[7.1 Verifikační matice 10](#_Toc146618258)

[7.2 Popis verifikačního prostředí 10](#_Toc146618259)

[7.3 Verifikační testy 11](#_Toc146618260)

[8 Výsledky implementace 13](#_Toc146618261)

1. Seznam Obrázků

[Obrázek 4‑1 Blokové schéma AAU 7](#_Toc154126393)

[Obrázek 5‑1 Vývojový diagram 8](#_Toc154126394)

[Obrázek 6‑1 Blokové schéma pomocné aritmetické jednotky 9](#_Toc154126395)

[Obrázek 7‑1 Blokové schéma verifikačního prostředí 11](#_Toc154126396)

1. Seznam Tabulek

[Table 2‑1 Seznam aplikovatelných dokumentů 5](#_Toc146618266)

[Table 2‑2 Seznam odkazovaných dokumentů 5](#_Toc146618267)

[Table 2‑2 Verifikační matice 10](#_Toc146618268)

# Úvod

Tento dokument zahrnuje v sobě plán vývoje návrhu, popis stadii návrhu, popis verifikačních požadavků a prostředí a výsledky implementace včetně statické časové analýzy (STA). Zadaní bylo navrhnout, realizovat a verifikovat design pomocné aritmetické jednotky (AAU) v VHDL s ověřením správnosti a funkčnosti návrhu.

Popis verifikaci je největší časti toho dokumentu. Je představen verifikační plán neboli popis testů, kterými pokrýváme funkční požadavky návrhu. Na začátku je popsaná verifikační matice, což je tabulka požadavků a odpovídající testy, které tyto požadavky pokrývají (dohromady jsou 4 testy na pokryti všech požadavku).

Nakonec, výsledky implementaci včetně počtu stavu stavového automatu (buňka pkt\_ctrl), počtu jednotlivých hradel a výsledku STA jsou ukázaný v kapitole 8.

# Aplikovatelné a Odkazované dokumenty

Všechny dokumenty sloužící jako zadávací dokumentace nebo odkazované v tomto textu jsou uvedeny v tabulkách dole. Pokud není známá přesná verze, je použito datum vydání dokumentu (měsíc/rok). V případě že není známé ani datum vydání, je jako verze uvedena hodnota 0.

## Seznam aplikovatelných dokumentů

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ref. | Název | | Číslo dokumentu |  | Verze |
|  | | AAU Requirement Specification | 1 |  | 2.1 |

Table 2‑1 Seznam aplikovatelných dokumentů

## Seznam odkazovaných dokumentů

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ref. | Název | Číslo dokumentu |  | Verze |

Table 2‑2 Seznam odkazovaných dokumentů

# Definice a seznam zkratek

## Definice

Každý požadavek (kapitola 7) se skládá z:

* Unikátní identifikace "REQ-AAU-x-yyy", kde 'x' souvisí se skupinou požadavku (G – obecné požadavky, F – funkční požadavky, I – požadavky pro interface); 'y' je unikátní 3-mistné číslo
* Unikátní jméno
* Verifikační metoda - určuje jednu(nebo i několik) z následujících metod: revize designu (R), analyza (A), simulace na RTL urovni (S)

## Aktivní hodnota signálu

Jeden z vstupních signálu povoleni (enable) CS\_b je active LOW (logická '0'), ostatní signály pro povolení jsou v active HIGH (logická '1')

## Psaní čísel

Čísla, uvedené v dokumentu jsou napsaný v následujících systémech:

* Decimální (integer) - desetinné číslo, například 123
* Hexadecimální – čísla, které začínají z 0x
* Binární – číslo je v uvozovkách (jeden bit ' ', vektor bitu " ")

## Jednotky

Pouze jednotky ze soustavy SI jsou použity v textu.

## Zkratky

|  |  |
| --- | --- |
| AAU  SPI  LSB  MSB  DUT | Auxiliary Arithmetic Unit  Serial Peripheral Interface  Least Significant Bit  Most Significant Bit  Device Under Test |
| TBC | To Be Confirmed |
| TBD | To Be Defined |
|  |  |
|  |  |

# Představení projektu

Cílem projektu je sestavení pomocné aritmetické jednotky (AAU – Auxiliary Arithmetic Unit) pro mikrokontroler (MCU). MCU komunikuje s AAU přes sériovou linku (SPI – Serial Peripheral Interface). MCU plní roli master a AAU roli slave. Úkolem aritmetické jednotky je provádět dvě aritmetické operace s čísli, sčítání a násobení. Výsledky jsou vraceny v následujícím paketu. Čísla jsou ve formátu se znaménkem a pevnou desetinou čárkou.

Datový tok je řízen masterem, ale slave musí kontrolovat úplnost paketů. Chybný paket AAU ignoruje a čeká na příjem dalšího.



Obrázek 4‑1 Blokové schéma AAU

Obsahem zprávy je krátké představení projektu, plán vývoje popisující výsledný postup práce, dále popis návrhu a blokové schéma. Následně obsahuje verifikační plán sestávající z verifikační matice, popisu verifikačního prostředí a popisů verifikačních testů. Závěrem je kapitola popisující výsledky implementace pro cílový FPGA obvod a cesta kterou bylo implementace dosaženo.

# Plán vývoje

První krokem bylo udělat analýzu specifikaci projektu a představit, jak postupovat v návrhu. Dále byl vytvořen vývojový plán a v samotný návrh, tj. psáni kódu ve VHDL. Dále bylo vytvořeno simulační prostředí včetně testbenchu a jednotlivých procedur. Nakonec byla udělaná implementace a všechny výsledky byly zkontrolovaný, jestli sedí s požadavkem. Pokud by v tom byl nějaký rozdíl, navrch by se opakoval z kroku č.3, co je návrhem dílčích bloku. Pote byla udělaná dokumentaci s podrobným popisem testovaní a navrchu. Vývojový diagram je uveden niž:

A diagram of a flowchart

Description automatically generated

Obrázek 5‑1 Vývojový diagram

# Popis Návrhu

A diagram of a computer program

Description automatically generated

Obrázek 6‑1 Blokové schéma pomocné aritmetické jednotky

**Rozhraní SPI** je sériový interface komunikace z periferií. SPI rozhraní používá 4 signály pro přijeti dat z venku. Této signály jsou CS\_b – signal povoleni s active LOW úrovni, definuje začátek a konec rámce, SCLK – hodinový signál komunikační linky, MOSI – datový signál do řadičů paketu (Master Output Slave Input) a MISO (Master Input Slave Output) – datový signál z řadiče paketu zpět do SPI. Rámec může mít různou šířku, v tomto projektu je definován na 16 bit (s detekcí chybového rámci v případě odlišné šířky).

**Řadič paketů** je modul, který odpovídá za označení posloupnosti rámce, detekci chybových rámce a případně dobu přetečení čekaní na další rámec, což je 1ms, když druhý rámec v paketě nepřichází dostatečně dlouho.

**Aritmetická jednotka** provádí aritmetické operace sčítání a násobení, z čísel, které přichází od řadiče paketů. Čísla jsou ve formátu pevné řádové čárky. Čísla mohou být kladné a záporné, závisí na MSB (sign bitu). Výsledky postupně odesílané zpět do SPI rozhrání přes řadič paketu ve formáte paketů.

# Verifikační Plán

*Při verifikaci splnění požadavku kontrolou dokumentace budou například uvedeny odkazy na jednotlivé kapitoly závěrečné zprávy, které splnění požadavku dokumentují. Při verifikaci pomocí simulace je pak třeba představit základní myšlenku simulace – tedy testování je prováděno pomocí testovacích vektorů, které jsou aplikovány na návrh a následně je vyhodnocena jeho odezva. V případě verifikace požadavku analýzou je pak třeba uvést, jakou analýzou je takový požadavek ověřen.*

## Verifikační matice

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| REQ ID | Název | Ver. metoda | | | Verifikace požadavku | Splněno |
| **R** | **S** | **A** |
| REQ\_AAU\_G\_001 | Target technology | A | - | - | Kapitola 8(Implenetace pro FPGA) | A |
| REQ\_AAU\_G\_002 | Synchronous design | A | - | - | Kapitola 8(Report review) | A |
| REQ\_AAU\_G\_003 | Output signals | A | A | - | Gate level analyza, netlist | A |
| REQ\_AAU\_G\_004 | Input Signals | A | A | - | Gate level analyza, netlist | A |
| REQ\_AAU\_G\_005 | FSM Safe Implementation | A | A | - | Advanced HDL Synthesis | A |
| REQ\_AAU\_G\_006 | Documentation | A | - | - |  | A |
| REQ\_AAU\_F\_010 | Auxiliary Arithmetic Unit | A | - | - | Test vyslednych hodnot operaci | A |
| REQ\_AAU\_F\_011 | Format of numbers | - | A | - | Test *tc\_au\_002* | A |
| REQ\_AAU\_F\_012 | Number rounding | - | A | - | Test *tc\_au\_002* | A |
| REQ\_AAU\_F\_013 | Overflow of arithmetic operations | - | A | - | Test *tc\_arit\_004* | A |
| REQ\_AAU\_I\_020 | SPI clock frequency | - | A | - | Test *tc\_au\_002* | A |
| REQ\_AAU\_I\_021 | Bit ordering | - | A | - | Test *tc\_spi\_001,* Test *tc\_au\_002* | A |
| REQ\_AAU\_I\_022 | Incomplete frame | - | A | - | Test *tc\_spi\_001* | A |
| REQ\_AAU\_I\_023 | Link reset | - | A | - | Test *tc\_rst\_003* | A |
| REQ\_AAU\_I\_024 | Packet format | - | A | - | Test *tc\_spi\_001* | A |

Table 7‑1 Verifikační matice

## Popis verifikačního prostředí

*Tato kapitola popisuje strukturu verifikačního prostředí a funkci jeho jednotlivých komponent. Podobně jako při popisu návrhu je nejprve představena celková struktura a popis předávání dat mezi jednotlivými komponentami. Následně jsou jednotlivé komponenty blíže představeny.*

A diagram of a software process

Description automatically generated

Obrázek 7‑1 Blokové schéma verifikačního prostředí

**SPI BFM** je model, který zahrnuje v sobě dva bloky: Driver a Monitor. Driver přijímá data od buňky 'TestCase' a této data odesila do DUT. Monitor naopak sleduje výstupné porty DUT a přijímá vystupni data, dekóduje je do čisel a porovnava se spravnymi vysledkami. Timto padem BFM zajištuje čteni a zapis dat do DUT.

**Testbench** je vrcholová jednotka, která má v sobě celý návrh, pomocné procesy, jako generovaní hodinového signálu a BFM. Testbench propojuje mezi sebou této části a funguje jako testovací prostředí přímo pro testovaní DUT.

**Testcase** je proces, který je součásti testbenchu. Tescase má v sobě kroky, přes které má design projít, aby splnil verifikační požadavky. Pomoci testcasu se testují funkčnost designu a pokryti požadavku.

## Verifikační testy

*V této kapitole jsou uvedeny všechny testy pro ověření funkčnosti návrhu (a splnění požadavků).*

|  |  |  |  |
| --- | --- | --- | --- |
| **Název testu** | Detekce chybného rámce na sběrnici SPI a kontrola pořadí výsledných paketů | **Číslo testu.:** | tc\_spi\_001 |
| **Popis testu** | Účelem tohoto testu je ověření, že chybný rámce není v DUT přijat. Testovány jsou dva scénáře, kdy nejprve je do DUT odeslán krátký rámec a následně dlouhý rámec. Pak se kontroluje kompatibilita z třemi frekvencí signálu SCLK a to jsou 10kHz, 100kHz a 1MHz. | | |
| **Reflektované požadavky** | REQ\_AAU\_I\_021  REQ\_AAU\_I\_022  REQ\_AAU\_I\_024 | | |
| **Postup testu** | 1. Reset DUT 2. Spuštění generování hodinového signálu s frekvencí 50 MHz. 3. Nastavení frekvence signálu SCLK na 10kHz. 4. BFM\_SPIM: Odeslán chybný rámec s hodnotou 000001110.000101101 (14.087890625). Tento rámec má 18 bitů a musí být považován za chybný. 5. BFM\_SPIM: Odeslán platný rámec s hodnotou 00001010.01110100 (10.453125). 6. BFM\_SPIM: Odeslán chybný rámec s hodnotou 0001110.0001010 (14.078125). Tento rámec má 14 bitů a musí být považován za chybný. 7. BFM\_SPIM: Odeslán platný rámec s hodnotou 00000101.10011111 (5.62109375). 8. BFM\_SPIM: Posílaní a přijetí dvou čísel. 9. BFM\_SPIM: Monitorovat výstup bloku (na výstupním pinu MISO), nejprve přijat rámec s výsledkem součtu a pak rámec s výsledkem součinu. Výsledek sčítání musí být 00010000.00010011 (16.07421875), výsledek násobení je 00111010.11000010 (58.7578125). 10. Nastavení frekvence signálu SCLK na 100kHz a opakování kroků 4-9. 11. Nastavení frekvence signálu SCLK na 1MHz a opakování kroků 4-9. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Název testu** | Kontrola formátu čísel, zaokrouhlování a pořadí bitů | **Číslo testu.:** | tc\_au\_002 |
| **Popis testu** | Účelem tohoto testu je ověření, že výstupní data jsou reprezentovaný v pevné řádové čárce, ověření správnosti výsledku aritmetických operací a zaokrouhlování | | |
| **Reflektované požadavky** | REQ\_AAU\_F\_011  REQ\_AAU\_F\_012  REQ\_AAU\_I\_020  REQ\_AAU\_I\_021 | | |
| **Postup testu** | 1. Reset DUT 2. Spuštění generování hodinového signálu s frekvencí 50 MHz. 3. BFM\_SPIM: Odeslán první rámec s hodnotou 00000111.10100111 (7.65234375). 4. BFM\_SPIM: Odeslán druhý rámec s hodnotou 00001001.10011001 (9.59765625). 5. BFM\_SPIM: Posílaní a přijetí dvou čísel. 6. BFM\_SPIM: Monitorovat výstup bloku (na výstupním pinu MISO). Výsledek sčítání musí být 00010001.01000000 (17.25), výsledek násobení je 01001001.01111001 (původně 01001001.0111000111001111) (73.44140625). | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Název testu** | Link reset test | **Číslo testu.:** | tc\_rst\_003 |
| **Popis testu** | Účelem tohoto testu je ověření, že první rámec bude považován za špatný, pokud druhý rámec nebude přijat do 1ms po prvním (testování 0.9ms, 1.1ms a 1.5ms). | | |
| **Reflektované požadavky** | REQ\_AAU\_I\_023 | | |
| **Postup testu** | 1. Reset DUT 2. Spuštění generování hodinového signálu s frekvencí 50 MHz. 3. BFM\_SPIM: Odeslán platný rámec s hodnotou 00000101.00100000 (5.125). 4. Čekání 1.5 ms pro reset linky 5. BFM\_SPIM: Odeslán platný rámec s hodnotou 00001011.11000000 (11.75). 6. BFM\_SPIM: Odeslán platný rámec s hodnotou 00000111.00100000 (7.125). 7. BFM\_SPIM: Posílaní a přijetí dvou čísel. 8. BFM\_SPIM: Monitorovat výstup bloku (na výstupním pinu MISO). Výsledek sčítání musí být 00010010.11100000 (18.875), výsledek násobení je 01010011.10111000 (83.71875). 9. BFM\_SPIM: Odeslán platný rámec s hodnotou 00000101.00100000 (5.125). 10. Čekání 0.9 ms pro kontrolu, že linka nepřejde do resetu 11. BFM\_SPIM: Odeslán platný rámec s hodnotou 00001011.11000000 (11.75). 12. BFM\_SPIM: Posílaní a přijetí dvou čísel. 13. BFM\_SPIM: Monitorovat výstup bloku (na výstupním pinu MISO). Výsledek sčítání musí být 00010000.11100000 (16.875), výsledek násobení je 00111100.00111000 (60.21875). 14. Opakování kroků 3-8 pro čekání 1.1 ms pro reset linky | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Název testu** | Přetečení aritmetických operaci | **Číslo testu.:** | tc\_arit\_004 |
| **Popis testu** | Účelem tohoto testu je ověření, že v případě přetečení, výsledky aritmetických operaci budou zaokrouhlené do maxima a do minimum. | | |
| **Reflektované požadavky** | REQ\_AAU\_F\_013 | | |
| **Postup testu** | 1. Reset DUT 2. Spuštění generování hodinového signálu s frekvencí 50 MHz. 3. BFM\_SPIM: Odeslán platný rámec s hodnotou 01101001.10000000 (105.5). 4. BFM\_SPIM: Odeslán platný rámec s hodnotou 00111101.01100000 (61.375). 5. BFM\_SPIM: Posílaní a přijetí dvou čísel. 6. BFM\_SPIM: Monitorovat výstup bloku (na výstupním pinu MISO). Výsledky součtu a součinu musí přetéct do maximuma 0x7FFF(01111111.11111111). 7. BFM\_SPIM: Odeslán platný rámec s hodnotou 10001001.10000000 (-119.5). 8. BFM\_SPIM: Odeslán platný rámec s hodnotou 11001111.01000000 (-49.75). 9. BFM\_SPIM: Posílaní a přijetí dvou čísel. 10. BFM\_SPIM: Monitorovat výstup bloku (na výstupním pinu MISO). Výsledek součtu musí přetéct do minimuma 0x8000(10000000.00000000) a výsledek součinu do maximuma 0x7FFF(01111111.11111111). 11. BFM\_SPIM: Odeslán platný rámec s hodnotou 00101111.01000000 (47.25). 12. BFM\_SPIM: Odeslán platný rámec s hodnotou 10001111.10000000 (-113.5). 13. BFM\_SPIM: Posílaní a přijetí dvou čísel. 14. BFM\_SPIM: Monitorovat výstup bloku (na výstupním pinu MISO). Výsledek součtu je 10111110.11000000 (-66.25) a výsledek součinu musi přetect do minimuma 0x8000(10000000.00000000). | | |

# Výsledky implementace

*V závěrečné kapitole zprávy by měly být uvedeny výsledky implementace pro cílový obvod FPGA.*

*Kromě základních informací (využité zdroje, maximální pracovní frekvence) je nutné také uvést informace o tom, jakým způsobem byly výsledky získány. Jedná se především o použité nástroje včetně jejich verze, nastavení pro implementaci (konfigurace syntetizéru, implementační strategie pro P&R) a také omezení implementace (tzv. constraints), pokud jsou použity.*

*V tomto projektu jsou vstupní signály asynchronní, v takovém případě je žádoucí výslednou implementaci analyzovat z hlediska zpoždění na vstupních a výstupních signálech a dokázat dostatečnou časovou rezervu při zpracování signálů sběrnice SPI.*

|  |  |  |
| --- | --- | --- |
| Typ | Název | Verze |
| Vývojové studio | Xilinx ISE | 14.7 |
| Simulátor | Xilinx ISim | 14.7 |

Tabulka 8.1

využité zdroje

Number of Slice Flip Flops: 174 out of 3,840 4%

Number of occupied Slices: 161 out of 1,920 8%

Total Number of 4 input LUTs: 279 out of 3,840 7%

Number used as logic: 246

Number used as a route-thru: 33

Number of bonded IOBs: 6 out of 173 3%

Number of MULT18X18s: 1 out of 12 8%

Number of BUFGMUXs: 1 out of 8 12%

maximální pracovní frekvence

f = 50 MHz

konfigurace syntetizéru

---- Target device

xc3s200-ft256-5

---- Source Parameters

Input File Name : "dig\_top.prj"

Input Format : mixed

Ignore Synthesis Constraint File : NO

---- Target Parameters

Output File Name : "dig\_top"

Output Format : NGC

Target Device : xc3s200-5-ft256

---- Source Options

Top Module Name : dig\_top

Automatic FSM Extraction : YES

FSM Encoding Algorithm : Auto

Safe Implementation : No

FSM Style : LUT

RAM Extraction : Yes

RAM Style : Auto

ROM Extraction : Yes

Mux Style : Auto

Decoder Extraction : YES

Priority Encoder Extraction : Yes

Shift Register Extraction : YES

Logical Shifter Extraction : YES

XOR Collapsing : YES

ROM Style : Auto

Mux Extraction : Yes

Resource Sharing : YES

Asynchronous To Synchronous : NO

Multiplier Style : Auto

Automatic Register Balancing : No

---- Target Options

Add IO Buffers : YES

Global Maximum Fanout : 500

Add Generic Clock Buffer(BUFG) : 8

Register Duplication : YES

Slice Packing : YES

Optimize Instantiated Primitives : NO

Use Clock Enable : Yes

Use Synchronous Set : Yes

Use Synchronous Reset : Yes

Pack IO Registers into IOBs : Auto

Equivalent register Removal : YES

---- General Options

Optimization Goal : Speed

Optimization Effort : 1

Keep Hierarchy : No

Netlist Hierarchy : As\_Optimized

RTL Output : Yes

Global Optimization : AllClockNets

Read Cores : YES

Write Timing Constraints : NO

Cross Clock Analysis : NO

Hierarchy Separator : /

Bus Delimiter : <>

Case Specifier : Maintain

Slice Utilization Ratio : 100

BRAM Utilization Ratio : 100

Verilog 2001 : YES

Auto BRAM Packing : NO

Slice Utilization Ratio Delta : 5

implementační strategie pro P&R

Optimization Goal - Speed

Optimization Effort - Normal

constraints

NET "CLK" TNM\_NET = CLK;

TIMESPEC TS\_CLK = PERIOD "CLK" 50 MHz HIGH 50%;