**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 5](#_Toc154211210)

[2 Aplikovatelné a Odkazované dokumenty 6](#_Toc154211211)

[2.1 Seznam aplikovatelných dokumentů 6](#_Toc154211212)

[2.2 Seznam odkazovaných dokumentů 6](#_Toc154211213)

[3 Definice a seznam zkratek 7](#_Toc154211214)

[3.1 Definice 7](#_Toc154211215)

[3.2 Aktivní hodnota signálu 7](#_Toc154211216)

[3.3 Psaní čísel 7](#_Toc154211217)

[3.4 Jednotky 7](#_Toc154211218)

[3.5 Zkratky 7](#_Toc154211219)

[4 Představení projektu 8](#_Toc154211220)

[5 Plán vývoje 9](#_Toc154211221)

[6 Popis Návrhu 10](#_Toc154211222)

[7 Verifikační Plán 11](#_Toc154211223)

[7.1 Verifikační matice 11](#_Toc154211224)

[7.2 Popis verifikačního prostředí 12](#_Toc154211225)

[7.3 Verifikační testy 13](#_Toc154211226)

[8 Výsledky implementace 17](#_Toc154211227)

[8.1 Základní informace 17](#_Toc154211228)

[8.2 Nastavení pro implementaci 17](#_Toc154211229)

[8.3 Omezení implementace 19](#_Toc154211230)

[8.4 Stavový automat 19](#_Toc154211231)

1. Seznam Obrázků

[Obrázek 4‑1 Blokové schéma AAU 8](#_Toc154211232)

[Obrázek 5‑1 Vývojový diagram 9](#_Toc154211233)

[Obrázek 6‑1 Blokové schéma pomocné aritmetické jednotky 10](#_Toc154211234)

[Obrázek 7‑1 Blokové schéma verifikačního prostředí 12](#_Toc154211235)

1. Seznam Tabulek

[Table 2.1 Seznam aplikovatelných dokumentů 6](#_Toc154211236)

[Table 2.2 Seznam odkazovaných dokumentů 6](#_Toc154211237)

[Table 7.1 Verifikační matice 11](#_Toc154211238)

[Table 8.1 Použité nástroje 17](#_Toc154211239)

[Table 8.2 Využité zdroje 17](#_Toc154211240)

[Table 8.3 Maximální frekvence 17](#_Toc154211241)

[Table 8.4 Vstupní parametry 17](#_Toc154211242)

[Table 8.5 Parametry cílového zařízení 17](#_Toc154211243)

[Table 8.6 Vstupní nastavení 18](#_Toc154211244)

[Table 8.7 Výstupní nastavení 18](#_Toc154211245)

[Table 8.8 Obecná nastavení 19](#_Toc154211246)

[Table 8.9 Implementační strategie pro P&R 19](#_Toc154211247)

[Table 8.10 Omezení implementace 19](#_Toc154211248)

[Table 8.11 Statická časová analýza 19](#_Toc154211249)

[Table 8.12 Stavový automat (pkt\_ctrl) 20](#_Toc154211250)

# Úvod

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

Popis verifikace je největší částí tohoto dokumentu. Je představen verifikační plán neboli popis testů, kterými pokrýváme funkční požadavky návrhu. Na začátku je popsaná verifikační matice, tj. tabulka požadavků a odpovídající testy, které tyto požadavky pokrývají (dohromady to jsou 4 testy na pokrytí všech požadavků).

Na závěr jsou v kapitole 8 ukázány výsledky implementace včetně počtu stavů stavového automatu (buňka pkt\_ctrl), počtu jednotlivých hradel a výsledku STA.

# 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 | AAU-RS-BUT-0001 |  | 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), analýza (A) a simulace na RTL úrovni (S)

## Aktivní hodnota signálu

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

## Psaní čísel

Čísla, uvedená v dokumentu jsou napsána v následujících soustavách:

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

## Jednotky

Pouze jednotky ze soustavy SI jsou použity v textu.

## Zkratky

|  |  |
| --- | --- |
| AAU | Auxiliary Arithmetic Unit |
| SPI | Serial Peripheral Interface |
| MCU | Microcontroller unit – mikrokontroler |
| LSB | Least Significant Bit |
| MSB | Most Significant Bit |
| DUT | 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 specifikace projektu a představit, jak postupovat v návrhu.   
Dále byl vytvořen vývojový plán a v samotný návrh, tj. modelování ve VHDL. Poté bylo vytvořeno simulační prostředí včetně testbenche a jednotlivých procedur. Následně byla provedena implementace a všechny výsledky byly zkontrolovány, zda sedí se zadaným požadavkem.   
Pokud by v tom byl nějaký rozdíl, návrh by se opakoval z kroku č.3, což je návrh dílčích bloku.   
Nakonec byla udělána dokumentace s podrobným popisem testovaní a návrhu.   
Vývojový diagram je uveden níže:

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 s periferií. SPI rozhraní používá 4 signály pro přijetí dat z venku. Tyto signály jsou CS\_b – signál povolení s active LOW úrovní, definuje začátek a konec rámce, SCLK – hodinový signál komunikační linky, MOSI (Master Output Slave Input) – datový signál do řadičů paketu 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 bitů (s detekcí chybového rámce v případě odlišné šířky).

Zde můžete uvést ještě více podrobností – např. jak je implementována detekce neplatného rámce

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

Doplňte a popište stavový diagram

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

# Verifikační Plán

## Verifikační matice

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| REQ ID | Název | Ver. metoda | | | Verifikace požadavku | Splněno |
| **R** | **S** | **A** |
| REQ\_AAU\_G\_001 | Target technology | A | - | - | Kapitola 8  (Implementace 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 výsledných hodnot operací | 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

A diagram of a software process

Description automatically generated

Obrázek 7‑1 Blokové schéma verifikačního prostředí

**SPI BFM** je model, který v sobě zahrnuje dva bloky: Driver a Monitor. Driver přijímá data od buňky 'TestCase' a tyto data odesílá do DUT. Monitor naopak sleduje výstupní porty DUT a přijímá výstupní data, dekóduje je do čísel a porovnává se správnými výsledky. Tím pádem BFM zajištuje čtení a zápis 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 tyto části a funguje jako testovací prostředí přímo pro testovaní DUT.

**Testcase** je proces, který je součástí testbenche. Testcase obsahuje kroky, přes které musí design projít, aby splnil verifikační požadavky. Pomocí testcasu se testuje funkčnost designu a pokrytí požadavků.

## Verifikační testy

|  |  |  |  |
| --- | --- | --- | --- |
| **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. Test je opakován s různými frekvencemi signálu SCLK, a to 10 kHz, 100 kHz a 1 MHz, pro ověření kompatibility testované jednotky se všemi třemi frekvencemi. | | |
| **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 10 kHz. 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 platných rámců. 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 100 kHz a opakování kroků 4-9. 11. Nastavení frekvence signálu SCLK na 1 MHz 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 reprezentována s pevnou řádovou čárkou, 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 pak 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 platných rámců. 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 operací budou zaokrouhlené do maxima a do minima. | | |
| **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 maxima 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 minima 0x8000(10000000.00000000) a výsledek součinu do maxima 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 musí přetéct do minima 0x8000(10000000.00000000). | | |

# Výsledky implementace

## Základní informace

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

Table 8.1 Použité nástroje

|  |  |  |  |
| --- | --- | --- | --- |
| Název zdroje | Využito zdrojů | Maximum zdrojů | Využití zdrojů |
| Number of Slice Flip Flops | 174 | 3,840 | 4 % |
| Number of occupied Slices | 161 | 1920 | 8 % |
| Total Number of 4 input LUTs | 179 | 3840 | 7 % |
| - Number used as logic | 146 |  |  |
| - Number used as a route-thru | 33 |  |  |
| Number of bonded IOBs | 6 | 173 | 3 % |
| Number of MULT18X18s | 1 | 12 | 8 % |
| Number used as a route-thru | 1 | 8 | 12 % |

Table 8.2 Využité zdroje

|  |  |
| --- | --- |
| Maximum frequency | 161,917 MHz |
| Minimum period | 6,176 ns |

Table 8.3 Maximální frekvence

## Nastavení pro implementaci

|  |  |
| --- | --- |
| Source Parameters | Value |
| Input File Name | "dig\_top.prj" |
| Input Format | mixed |
| Ignore Synthesis Constraint File | NO |

Table 8.4 Vstupní parametry

|  |  |
| --- | --- |
| Target Parameters | Value |
| Output File Name | "dig\_top" |
| Output Format | NGC |
| Target Device | xc3s200-5-ft256 |

Table 8.5 Parametry cílového zařízení

|  |  |
| --- | --- |
| Source Options | Value |
| 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 |
| 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 Rese | Yes |
| Pack IO Registers into IOBs | Auto |
| Equivalent register Removal | YES |
| Automatic Register Balancing | No |

Table 8.6 Nastavení syntetizéru

|  |  |
| --- | --- |
| General Options | Value |
| 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 |

Table 8.7 Obecná nastavení

|  |  |
| --- | --- |
| Implementation Options | Value |
| Optimization Goal | Speed |
| Optimization Effort | Normal |

Table 8.8 Implementační strategie pro P&R

## Omezení implementace

|  |  |
| --- | --- |
| Timing constraint | Value |
| NET "CLK" TNM\_NET | CLK |
| TIMESPEC TS\_CLK | PERIOD "CLK" 50 MHz HIGH 50% |

Table 8.9 Omezení implementace

|  |  |
| --- | --- |
| Parameter | Time in ns |
| Slack (setup path) | 13,824 |
| Requirement | 20,000 |
| Data Path Delay | 6,176 |
| Clock Path Skew | 0,000 |
| Clock Uncertainty | 0,000 |

Table 8.10 Výsledky statické časové analýzy

## Stavový automat

|  |  |
| --- | --- |
| Parameter | Value |
| States | 4 |
| Transitions | 11 |
| Inputs | 4 |
| Outputs | 7 |
| Clock | clk (rising\_edge) |
| Reset | rst (positive) |
| Reset type | asynchronous |
| Reset State | s0 |
| Power Up State | s0 |
| Encoding | automatic |
| Implementation | LUT |

Table 8.11 Stavový automat (pkt\_ctrl)

|  |  |
| --- | --- |
| State | Encoding |
| S0 | 00 |
| S1 | 01 |
| S2 | 10 |
| S3 | 11 |

Table 8.12 FSM enkódování