



AGH

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Fizyki i Informatyki Stosowanej

Języki opisu sprzętu

Dominik Dziuba

Miłosz Filus

kierunek studiów: **informatyka stosowana**

Licznik BCD

Spis treści

1 Wstęp.	3
2 Projekt.	3
2.1 Założenia projektowe.	3
2.2 Wymagana funkcjonalność.	3
3 Dokumentacja użytkownika.	3
3.1 Sprzętowe elementy interfejsu użytkownika.	4
3.2 Użycie urządzenia.	6
4 Dokumentacja techniczna.	7
4.1 Warstwa sprzętowa.	7
4.2 Warstwa programowa.	7
4.3 Struktura projektu.	8
4.4 Moduły projektu.	9
4.4.1 Moduł device.	9
4.4.2 Moduł clkdiv.	9
4.4.3 Moduł LicznikBCD.	9
4.4.4 Moduł licznikBCD1dekada.	10
4.4.5 Moduł oled_top.	10
4.4.6 Moduł fsm_init.	10
4.4.7 Moduł delay.	10
4.4.8 Moduł spi.	10
4.4.9 Moduł fsm_oper.	10
4.4.10 Moduł char2pixels.	10
4.4.11 Moduł update_page.	11
4.4.12 Moduł edge_detector.	11
4.4.13 Moduł filter.	11
5 Analiza syntezy.	11
5.1 Wykorzystanie zasobów.	11
6 Testy.	12
6.1 Zaimplementowane moduły testujące.	12
6.1.1 Testy modułu device.	12
6.1.2 Testy modułu licznikBCD.	13
6.1.3 Testy modułu clockdiv.	13
7 Podsumowanie i wnioski.	14

1 Wstęp.

Dokument ten stanowi dokumentację projektu **Licznik BCD** wykonanego w trakcie przedmiotu **Języki opisu sprzętu** (WFiIS AGH) przez Dominika Dziubę i Miłosza Filusa. Opisane zostały w nim założenia, opis zaimplementowanej funkcjonalności, dokumentacja techniczna czy użytkownika projektu. Opisano również analizę syntezy i procedury testowania poszczególnych modułów stworzonych w ramach projektu.

2 Projekt.

Ten rozdział opisuje założenia postawione przed projektem oraz oczekiwana funkcjonalność jaką miał osiągnąć.

2.1 Założenia projektowe.

Projekt miał dostarczyć funkcję elektronicznego licznika mającego wyświetlać obecny stan na wyświetlaczu OLED oraz zliczać impulsy pochodzące z generatora sygnału lub z enkodera obrotowego pozwalającego na zliczanie w górę jak i w dół. Realizacja projektu miała być przygotowana dla płytki **ZedBoard** (Xilinx Zynq-7000).

Do realizacji miały być wykorzystane języki Verilog lub SystemVerilog oraz środowisko programistyczne Xilinx Vivado. Licznik ma przechowywać liczby czterocyfrowe. Obrót enkodera obrotowego zgodnie ze wskaźówkami zegara ma zwiększać liczbę przechowywaną w liczniku o 1, a obrót przeciwnie do ruchu wskaźówek zegara ją zmniejszać o 1. Do płytki można również podłączyć generator sygnału, którego impulsy są zliczane w górę. Aktualny stan licznika ma być widoczny na wyświetlaczu OLED.

2.2 Wymagana funkcjonalność.

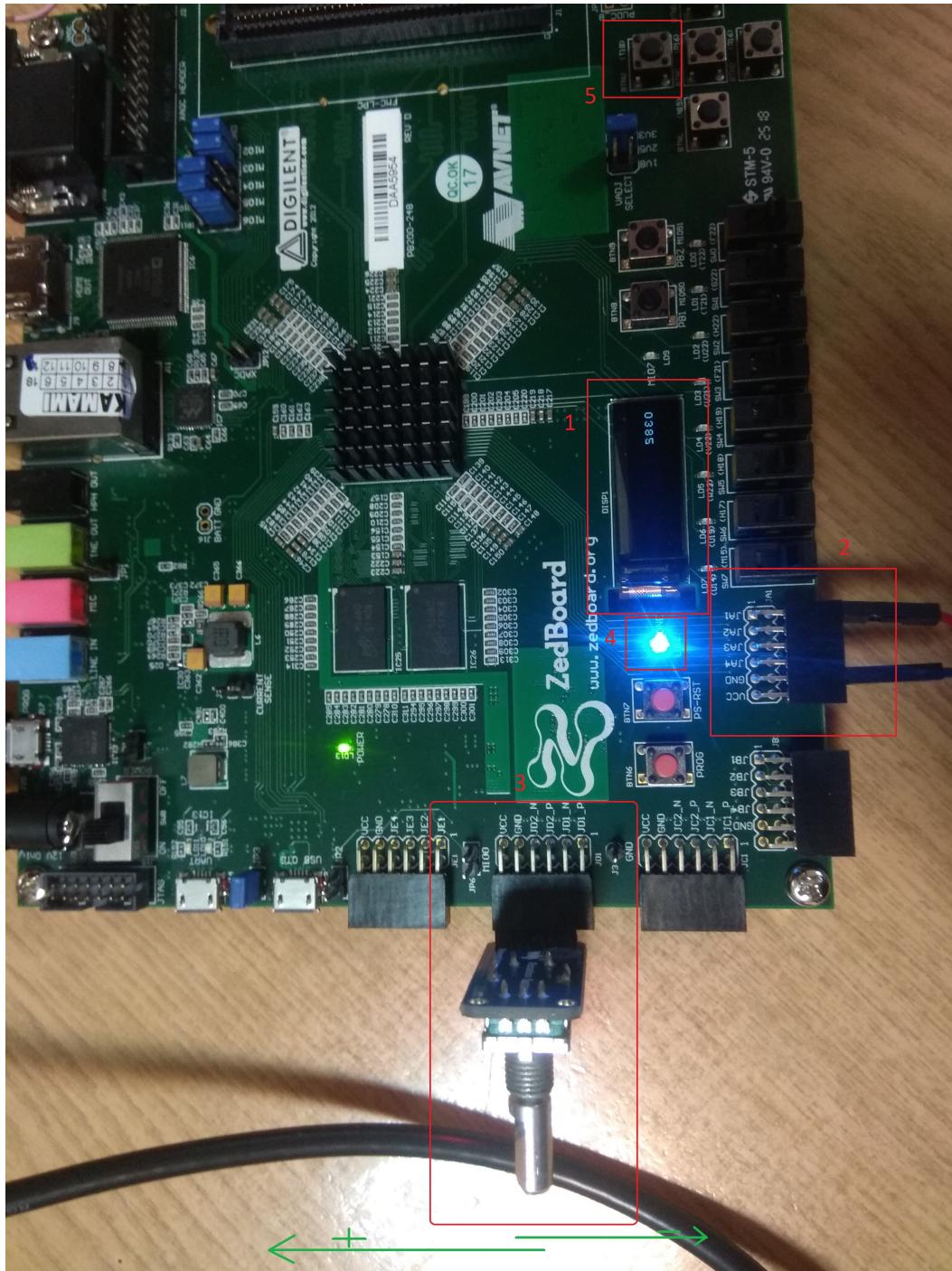
Następujące funkcjonalności były wymagane wobec projektu:

- wyświetlanie obecnej zawartości licznika na wyświetlaczu OLED
- możliwość podłączenia generatora sygnału oraz zliczanie impulsów przez niego generowanych
- możliwość podłączenia enkodera obrotowego i powiązane zliczanie impulsów przez niego generowanych
- w zależności od kierunku obrotu (oraz sygnału generowanego przez enkoder) licznik ma zliczać impulsy w górę lub w dół

3 Dokumentacja użytkownika.

Ten rozdział przedstawia jak poprawnie skorzystać z urządzenia oraz jakie elementy są wymagane do osiągnięcia działania opisanego we wstępie.

3.1 Sprzętowe elementy interfejsu użytkownika.



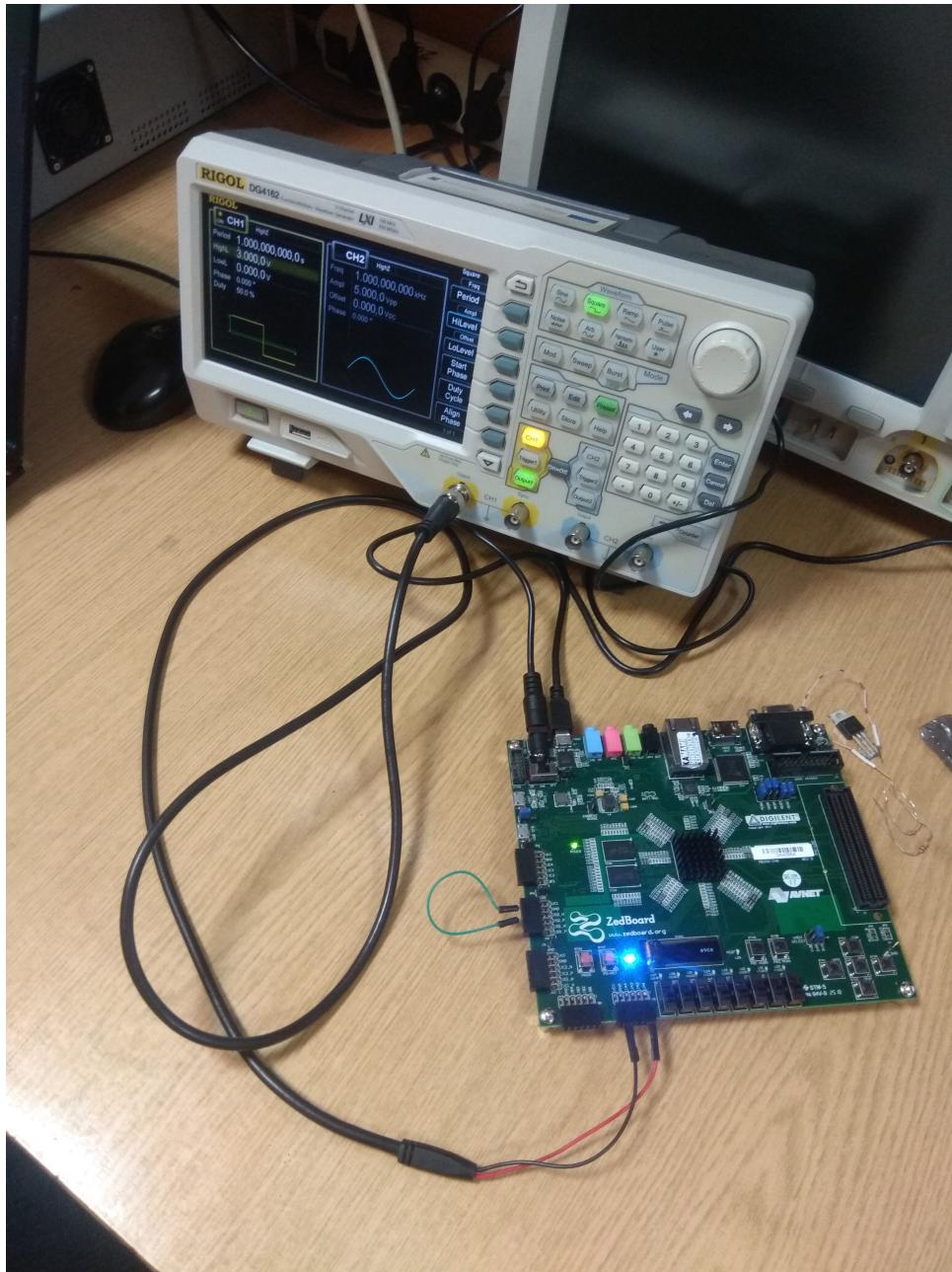
Rysunek 1: Zdjęcie płytka z zaznaczonymi elementami interfejsu użytkownika.

Na powyższym zdjęciu został również zaznaczony kierunek w którym enkoder dekrementuje i inkrementuje.

Elementy interfejsu w przygotowanym urządzeniu:

- enkoder obrotowy podpięty do złącza (JD1_N i JD2_P) [3]
- wyświetlacz OLED [1]
- przycisk resetu licznika [5]
- wejście na sygnał generatora podpięty do złącza (JA1) [2]
- dioda kontrolna wyświetlacza OLED [4]

Poniższe zdjęcia przedstawia cały zestaw z podłączonym generatorem.



Rysunek 2: Stanowisko z generatorem.

Ze względu na stan jałowy portu do którego jest podpinany enkoder, wejście odpowiadające za kierunek zliczania zostało uziemione. Gdy enkoder jest podłączony, licznik po otrzymaniu sygnału z generatora liczy dekrementując.

3.2 Użycie urządzenia.

Instrukcje użycia.

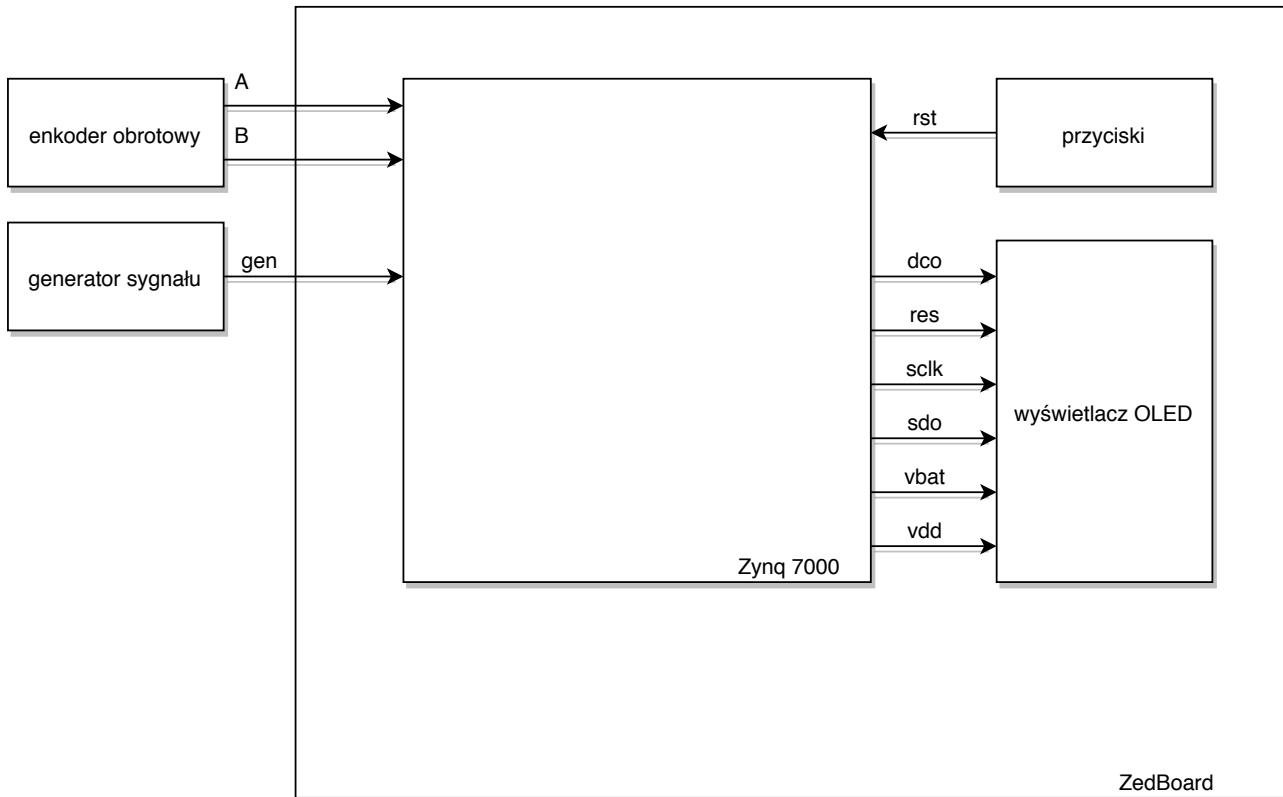
- Poprawne wgranie kodu na wspomnianą płytke jest widoczne po włączeniu się diody wyświetlacza OLED oraz pokazaniu się 4 cyfr na wyświetlaczu.
- Obrót pokrętłem enkodera przeciwnie do ruchu wskazówek zegara powoduje zwiększenie się liczby pokazywanej na wyświetlaczu, a obrót zgodnie z ruchem wskazówek zegara powoduje jej zmniejszenie. Należy zwrócić uwagę, że zwiększenie liczby o jeden w stanie 9999 powoduje przejście do stanu 0000, oraz zmniejszenie o jeden w stanie 0000 przejście w stan 9999.
- Generator podłączony jak wspomniano w poprzednim podrozdziale, powoduje natomiast zwiększenie liczby wyświetlnej przez wyświetlacz o jeden. Podobnie jak przy enkoderze, kiedy liczba osiągnie stan 9999, licznik przechodzi w stan 0000. Należy wtedy jednak pamiętać że trzeba zewrzeć złącze JD2_P z ziemią aby zaobserwować opisane działanie.

4 Dokumentacja techniczna.

Rozdział ten opisuje szczegóły implementacyjne projektu, takie jak: elementy sprzętowe wykorzystane w projekcie, strukturę programistyczną projektu (w środowisku Vivado oraz języku SystemVerilog) czy poszczególne moduły.

4.1 Warstwa sprzętowa.

Ten podrozdział zawiera opis generowanego przez projekt Vivado sprzętu.



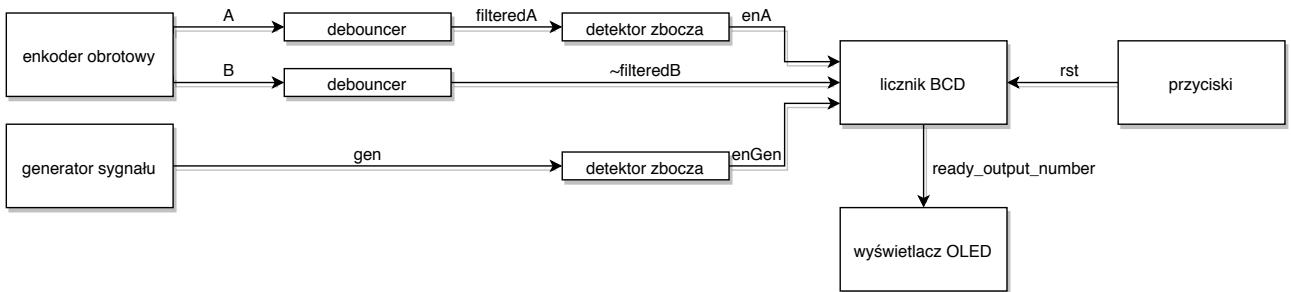
Rysunek 3: Diagram przedstawiający wykorzystane elementy płytki oraz elementy interfejsu użytkownika.

Gdzie poszczególne elementy:

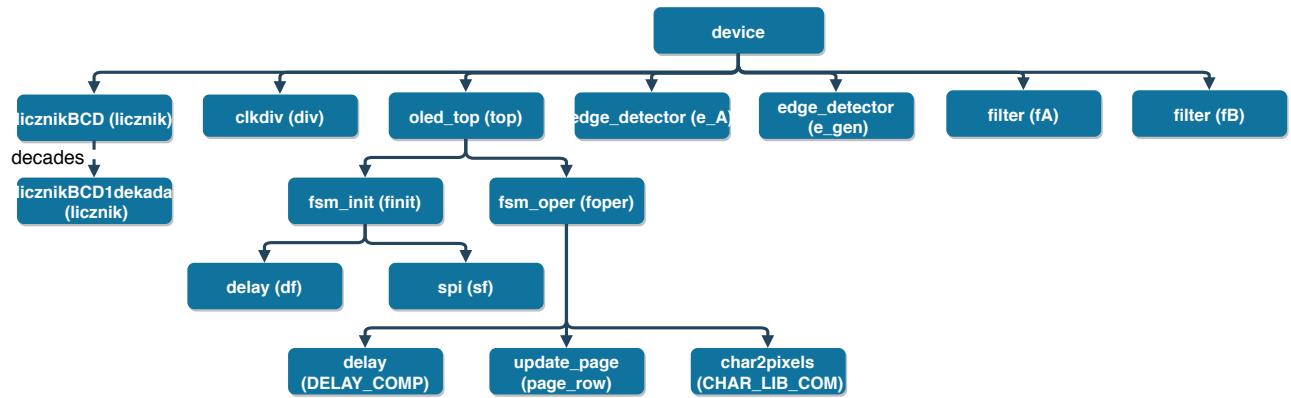
- enkoder obrotowy, element pozwalający na inkrementację lub dekrementację licznika poprzez generację odpowiedniego sygnału przekazywanego do płytki ZedBoard. Jest podłączony do złączy JD1_N i JD2_P
- Zynq 7000, czyli układ FPGA pozwalający na tworzenie własnego układu elektronicznego
- wyświetlacz OLED, znajduje się na płytce ZedBoard, wykorzystany do wyświetlania stanu licznika. Dane do licznika są wysyłane poprzez jednokierunkowy interfejs SPI
- przycisk BTNC, generuje sygnał resetujący, ustawiający licznik w stan początkowy

4.2 Warstwa programowa.

Ten podrozdział opisuje architekturę i przepływ sygnałów między modułami stworzonymi w projekcie. Omawia również ich hierarchię.



Rysunek 4: Diagram przedstawiający przepływ sygnałów od różnych elementów interfejsu użytkownika i modułów projektu.



Rysunek 5: Diagram hierarchii modułów w projekcie. Notacja *typ modułu (nazwa instancji)*. Słowo *decades* jest parametrem modułu *licznikBCD* i oznacza liczbę instancji modułu *licznikBCD1dekada* na przedstawionym diagramie.

4.3 Struktura projektu.

Struktura głównego katalogu ze źródłami.

LicznikBCD.srcs

```

└── constrs_1
    └── new
        └── test-oled.xdc
  └── sim_1
    └── new
      └── test-oled.xdc
  └── sources_1
    └── new
      └── test-oled.xdc
  
```

Struktura katalogu *constrs_1*

constrs_1

```

└── new
    └── test-oled.xdc
  
```

Plik *test-oled.xdc* zawiera wszystkie potrzebne ustawienia (ograniczenia), aby móc przekazać odpowiednie sygnały z elementów płytka, np. złącza JA1, do którego może być podłączony generator sygnału.

Struktura katalogu *sim_1*

```
sim_1
└── new
    ├── tb_clkdiv.sv
    ├── tb_licznik.sv
    └── tb_top.sv
```

Poszczególne pliki **.sv* zawierają definicje modułów testujących wybranych modułów wykorzystanych w projekcie.

Struktura katalogu *sources_1*

```
sources_1
└── new
    ├── char2pixels.sv
    ├── clkdiv.sv
    ├── debounce.sv
    ├── delay.sv
    ├── device.sv
    ├── edge_detector.sv
    ├── fsm_init.sv
    ├── fsm_oper.sv
    ├── licznikBCD.sv
    ├── licznikBCD1dekada.sv
    ├── oled_top.sv
    ├── screens.vh
    ├── spi.sv
    └── update_page.sv
```

Poszczególne pliki **.sv* zawierają definicje modułów implementujących żądane elementy elektroniczne, np. natywny licznik BCD (natwyna, gdyż przechowuje i operuje na dancyh w formacie BCD).

4.4 Moduły projektu.

Poszczególne moduły i ich znaczenie w projekcie oraz przyjmowane przez nie parametry.

4.4.1 Moduł device.

Jest to główny moduł, w którym zdefiniowane są pozostałe instancje modułów. Można w nim sparametryzować wybrane moduły, czyli na przykład liczbę dekad w liczniku BCD.

4.4.2 Moduł clkdiv.

Moduł, którego zadanie jest symulacja zegara o niższym taktowaniu. Jest on wymagany ze względu na potrzebę odfiltrowania fałszywych sygnałów (zakłóceń) pochodzących z enkodera obrotowego. Pobiera parametr *div* będący górną wartością, do której zlicza licznik zaimplementowany w tym module (licznie odbywa się zgodnie z zegarem systemowym).

4.4.3 Moduł LicznikBCD.

Jest to moduł, który posiada generyczną pętlę, która wykorzystywana jest do stworzenia zadanej przez parametr (*decades*) liczby instancji liczników pojedynczej dekady. Pełni również

rolę koordynatora przekazywań bitów przepełnienia.

Listing 1: Implementacja pętli genvar.

```
genvar i ;
generate for (i = 0; i < decades; i = i + 1) begin: dekada
    assign enn[i + 1] = ovll[i] & enn[i];
    licznikBCD1dekada licz(.clk(clk), .rst(rst), .signal(enn[i]),
                           .up(up), .ovl(ovll[i]), .out(cout[4 * i + 3:4 * i]));
end
endgenerate
```

4.4.4 Moduł licznikBCD1dekada.

Moduł pełni rolę operacji na pojedynczej dekadzie, wykonuje operację inkrementacji oraz dekrementacji. Na wyjściu przekazuje również bit przepełnienia. O kierunku zliczania decyduje sygnał *up*. Gdy sygnał *up* jest 1 oraz aktualne stan dekady to 9 następuje wyzerowanie oraz wpisanie do sygnału wyjściowego bitu przepełnienia na 1. Ta sama sytuacja występuje w przypadku, gdy stan *up* jest 0 oraz stan licznika to 0, następuje wtedy wpisanie 9 oraz bit przepełnienia zostaje ustawiony na 1.

4.4.5 Moduł oled_top.

Jest to moduł tworzący moduły *fsm_init* i *fsm_oper* z odpowiednimi parametrami. Zawiera też prostą maszynę stanów, która dba o uruchomienie wyświetlacza OLED i utrzymanie go w stanie działania.

4.4.6 Moduł fsm_init.

Jest to moduł zawierający instancje modułów *delay* i *spi*. Zawiera również maszynę stanów, odpowiedzialną za uruchomienie wyświetlacza OLED.

4.4.7 Moduł delay.

Jest to moduł generujący odpowiednie opóźnienia wymagane przy operowaniu wyświetlacza OLED.

4.4.8 Moduł spi.

Jest to moduł implementujący uproszczoną, jednokierunkową komunikację z wyświetlaczem OLED do przekazywania komend i danych.

4.4.9 Moduł fsm_oper.

Jest to moduł zawierający instancje modułów *delay*, *update_page* i *char2pixels*. Odpowiada za wysyłanie danych do wyświetlacza. W projekcie wysyła zawartość rejestru licznika.

4.4.10 Moduł char2pixels.

Jest to moduł odpowiadający za konwersję kodów ASCII na piksele, które mają być zapalone na ekranie.

4.4.11 Moduł update_page.

Moduł powiązany z *fsm_oper*, odpowiada za aktualizację wiersza pikseli w wyświetlaczu.

4.4.12 Moduł edge_detector.

Prosty moduł generujący sygnał przy wykryciu zbocza narastającego na danym porcie wejściowym.

4.4.13 Moduł filter.

Moduł ten implementuje ideę *debounce*'ra, i filtruje zakłócenia przychodzące z podłączonego portu.

5 Analiza syntezы.

Rozdział ten opisuje wykorzystanie zasobów płytki przez przygotowany projekt.

Do pracy z płytą oraz do przeprowadzenia symulacji *Zynq* zostało wykorzystane narzędzie **Xilinx Vivado 2019.1..**

5.1 Wykorzystanie zasobów.

- Do przechowywania informacji wczytanych z pliku *pixel_SSD1306.dat* został wykorzystany blok ROM.
- Zostało wykorzystane 537 komórek logicznych.

Poniżej znajduje się raport z syntezą, z którego można odczytać ile komórek zostało użytych w poszczególnych modułach.

	Instance	Module	Cells
1	top		537
2	div	clkdiv	46
3	e_A	edge_detector	4
4	e_gen	edge_detector_0	6
5	fA	filter	6
6	fB	filter_1	7
7	licznik	licznikBCD	38
8	\dekada [0].licz	licznikBCD1dekada	10
9	\dekada [1].licz	licznikBCD1dekada_2	10
10	\dekada [2].licz	licznikBCD1dekada_3	10
11	\dekada [3].licz	licznikBCD1dekada_4	8
12	top	oled_top	418
13	finit	fsm_init	172
14	df	delay	71
15	sf	spi	52
16	foper	fsm_oper	244
17	CHAR_LIB_COM	char2pixels	1
18	DELAY_COMP	delay_parameterized0	83
19	SPI_COMP	spi_parameterized0	47
20	page_row	update_page	26

Rysunek 6: Raport syntezy.

6 Testy.

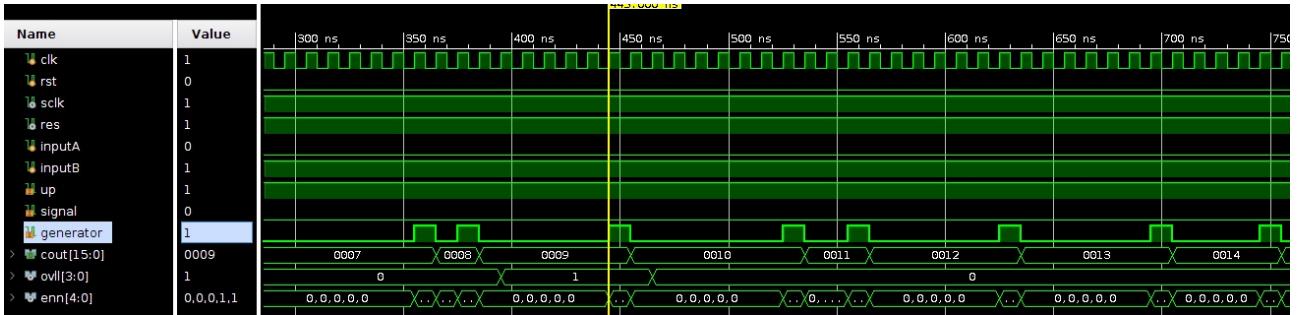
Rozdział ten opisuje moduły testujące zaimplementowane dla sprawdzania działania poszczególnych modułów stworzonych w ramach projektu. W ramach testowania układu zostały stworzone **trzy** moduły testowe. Umożliwiały symulacje działania poszczególnych modułów. Typy tych symulacji to *symulacja behawioralna*. Nie zostały przeprowadzone symulacje po syntezie.

6.1 Zaimplementowane moduły testujące.

Zostały przetestowane te moduły, które były odpowiadające tematowi projektu, zostały pominięte komponenty przygotowane wcześniej, czyli na przykład wyświetlacze.

6.1.1 Testy modułu device.

Jest to moduł, który został stworzony do przetestowania całości. Do przetestowania został wygenerowany sygnał generatora, który przyjmował losową wartość, co okres generowanego zegara.

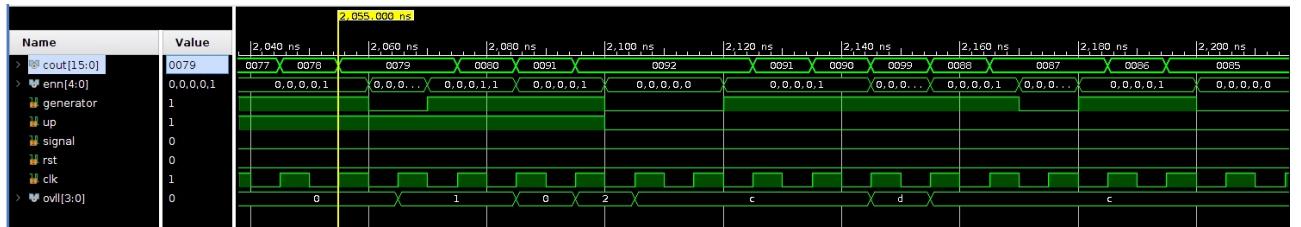


Rysunek 7: Przebieg symulacji behawioralnej modułu device.

InputA, który jest sygnałem, który generuje enkoder został ustawione stale na 0. Zaś *InputB*, który odpowiada za kierunek zliczania, tzn. decyduje czy inkrementować czy dekrementować stan licznika, w wyniku *up* jest stale 1.

6.1.2 Testy modułu licznikBCD.

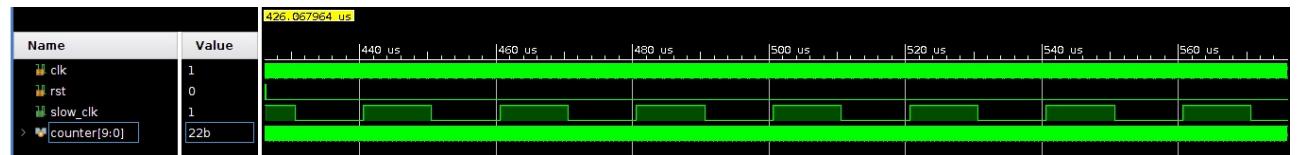
Moduł reaguje na sygnał z generatora jak i również enkodera. Został przetestowany ponownie przy użyciu sygnału generatora, gdzie zostało sprawdzone czy dobrze inkrementuje i dekrementuje aktualny stan. Jako, że sygnał *up* odpowiada za kierunek zliczania, to został on zmieniony w trakcie symulacji, by przetestować czy przejście z poprzedniego kierunku działa poprawnie.



Rysunek 8: Przebieg symulacji behawioralnej modułu licznikBCD.

6.1.3 Testy modułu clockdiv.

Test modułu clockdiv nie wymaga dodatkowo generowanych sygnałów na wejściu, wystarczają *rst* oraz *clk*. Zostało przedstawione jak prezentuje się spowolnione taktowanie nowego zegara, który jest wyjściem z testowanego modułu.



Rysunek 9: Przebieg symulacji behawioralnej modułu clockdiv.

7 Podsumowanie i wnioski.

Jednym z głównych mankamentów przygotowanego licznika jest wartość startowa. W przypadku gdy enkoder jest podłączony licznik zaczyna ze stanem *9998*, zaś gdy jedno z wejść enkodera decydujące o kierunku zliczania jest uziemione licznik zaczyna ze stanem *0002*. Nie do końca wiemy co jest przyczyną generowanego sygnału, który powoduje powyższy stan początkowy licznika. Kolejnym problemem jest przesuwanie się liczby na wyświetlaczu OLED, tj. w trakcie działania wyświetlacza, liczba na nim pokazana, przesuwa się cyklicznie o parę pikseli w prawo aż do pewnego momentu, po czym zaczyna się przesuwać w lewo aż do brzegu ekranu.

Pomimo powyższych, uważamy że cele postawione przed projektem zostały osiągnięte. I gotowe urządzenie spełnia wymagane założenia. Naturalnie projekt może zostać udoskonalony, czy to przez poprawkę nie do końca zdefiniowanych zachowań opisanych wcześniej lub przez dodanie nowych funkcjonalności. Na przykład, taką funkcjonalnością mogłyby być obsługa przełącznika dostępnego na płytce do wskazywania kierunku liczenia licznikowi.