Tytuł: **Ogień i woda – Infinity Tower**

Autorzy: **Joanna Jaśkowiec (JJ), Aleksandra Gniadek**

Ostatnia modyfikacja: 01.09.2025

Spis treści

1. Repozytorium git

Adres repozytorium GITa:

https://github.com/jaskasia/UEC2_projekt_2025.git

W przypadku repozytorium prywatnego należy zaprosić użytkownika zewnętrznego o adresie mailowym: kaczmarczyk@agh.edu.pl

2. Wstęp

Gra inspirowana jest serią gier 'Fireboy & Watergirl'. W naszej wersji zamiast współpracy postawiłyśmy na rywalizację, ponieważ wygrywa gracz, który jako pierwszy dotrze do szczytu ekranu, omijając pola żywiołu przeciwnika. Ale tu zaczyna się prawdziwa rywalizacja, bo w naszej grze możesz zrobić podmiankę postaci i zamienić się z ognia w wodę i na odwrót zmieniając tym samym żywioł na ekranie przeciwnika.

3. Specyfikacja

3.1. Opis ogólny algorytmu

Uproszczony schemat blokowy działania implementowanego algorytmu. Co się dzieje po starcie, jak wygląda przebieg działania, kiedy i pod jakimi warunkami się kończy.

Gra rozpoczyna się ekranem startowym gdzie mamy wybór postaci: Woda – spacja, Ogień – enter. Po wyborze postaci wyświetlany jest ekran z planszą do gry. Po warunku końca gry wyświetla się ekran końcowy. Sterowanie postacią obsługują klawisze kolejno: W – góra, A – lewo, D - prawo

Schemat blokowy:

*- **Ekran startowy** – po uruchomieniu gry można dokonać wyboru postaci (ogień albo woda)*

***- Rozgrywa** – gra polega na skakaniu w górę na kolejne platformy tak by omijać wodę lub ogień (w zależności od postaci). Możliwa jest zmiana postaci w trakcie gry poprzez kliknięcie klawisza: enter albo spacja – utrudnia to przeciwnikowi rozgrywkę.*

***- Koniec gry** – koniec gry następuje w dwóch przypadkach:*

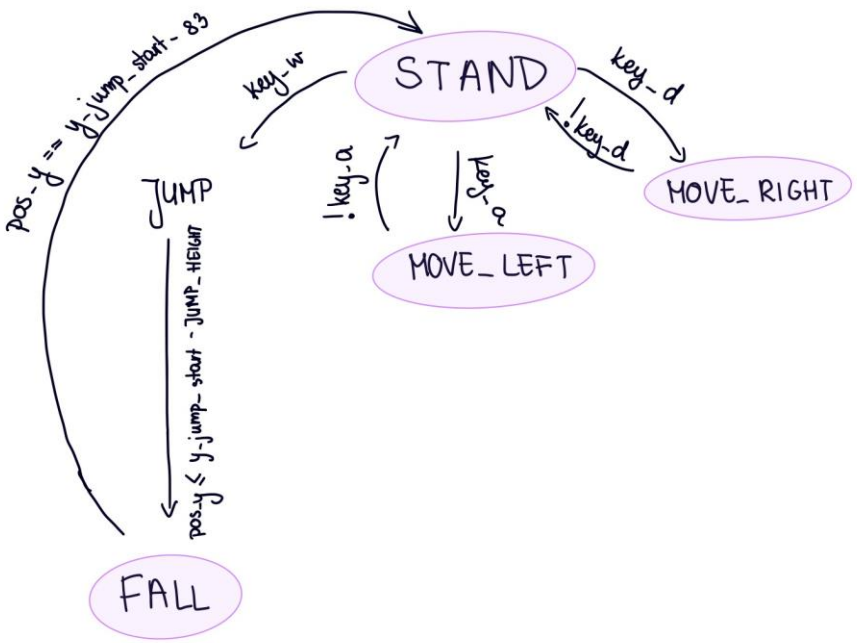
1) dotknięcie pola z przeciwnym żywiołem

2) zwycięstwo jednego z graczy: osiągnięcie ostatniej platformy

***- Ekran końcowy** – ekran informujący o porażce*

**bloki oznaczone na żółto nie są w pełni funkcjonujące*

Diagram stanu dla modułu ruchu:



Nasza gra jak mówi sam tytuł jest nieskończona. Innymi słowami nie kończy się (nie ma w nim warunku końcowego)

Ewentualnie przykładowe screen-shoty tego, co w przybliżeniu chcielibyśmy uzyskać.

3.2. Tabela zdarzeń

Opis zdarzeń występujących podczas działania programu/urządzenia, zarówno zewnętrznych (interakcje z użytkownikiem), jak i wewnętrznych (specyficzne stany w algorytmie). Zdarzenia podzielone są na kategorie dotyczące różnych stanów działania programu. Kategorie powinny odpowiadać stanom ze schematu z pkt. 2.1.

Zdarzenie	**Kategoria**	**Reakcja systemu**
Reset, uruchomienie sprzętu	Start	Początkowy stan gry – inicjalizacja rejestrów pamięci, wartości domyślnych, wyświetlenie ekranu początkowego gry.
Kliknięcie przycisku „spacja”	Wybór postaci	Użytkownik wybiera albo zmienia postać na wodę.
Kliknięcie przycisku „enter”	Wybór postaci	Użytkownik wybiera albo zmienia postać na ogień.
Kliknięcie przycisku „A”	Poruszanie postacią	Podczas klikania klawisza „A” postać porusza się w lewo ze stałą prędkością
Kliknięcie przycisku „D”	Poruszanie postacią	Podczas klikania klawisza „D” postać porusza się w prawo ze stałą prędkością
Kliknięcie przycisku „W”	Poruszanie postacią	Podczas klikania klawisza „W” postać porusza się w górę
Sygnał pozycji pos_y = y_jump_start - 83	Fizyka ruchu (pionowego)	Opadanie odbywa się do pozycji y, w której zaczął się skok pomniejszonej o odległość między platformami i grubość pojedynczej platformy
Sygnał pozycji pos_y = y_jump_start - JUMP HEIGHT	Fizyka ruchu (pionowego)	

4. Architektura

Uwaga: dobrze zrobiony projekt zawiera tylko moduły strukturalne (zbudowane z innych modułów) i funkcjonalne (zawierające bloki proceduralne always @). Staramy się nie generować bloków mieszających te dwa typy, o ile to możliwe (głównym kryterium powinna być czytelność kodu).

Uwaga: opisujemy architekturę **tylko głównego modułu oraz rozprowadzenie sygnału zegara**.

4.1. Moduł: top

Osoba odpowiedzialna: AG

4.1.1. Schemat blokowy

Uwaga: Schemat blokowy to nie jest schemat z Vivado! Nie zawiera on sygnałów, tylko interfejsy. Interfejs oznacza tutaj grupę sygnałów. Schemat blokowy pokazuje moduły składowe, oraz łączące je interfejsy.

Miejsce na schemat blokowy modułu głównego

Uwaga:

- *interfejsy dwukierunkowe rozbijamy na 2 interfejsy jednokierunkowe*
- *nazwa interfejsu stanowi prefiks nazwy sygnałów składowych*
- *w interfejsach nie uwzględniamy sygnałów globalnych (np. clk i rst).*

4.1.2. Porty

a) keyboard

nazwa portu	opis
ps2_data	szeregowe wejście danych z klawiatury
ps2_clk	zegar klawiatury

b) vga

nazwa portu	opis
vs	sygnał synchronizacji pionowej VGA
hs	sygnał synchronizacji poziomej VGA
r [3:0]	sygnał czerwonego koloru VGA
g [3:0]	sygnał zielonego koloru VGA
b [3:0]	sygnał niebieskiego koloru VGA

c) UART

data_in [7:0]	dane wejściowe z modułu komunikacyjnego
data_ready	flaga gotowości danych wyjściowych
data_out [7:0]	dane wyjściowe do modułu komunikacyjnego

4.1.3. Interfejsy

4.1.3. Interfejsy

vga_if

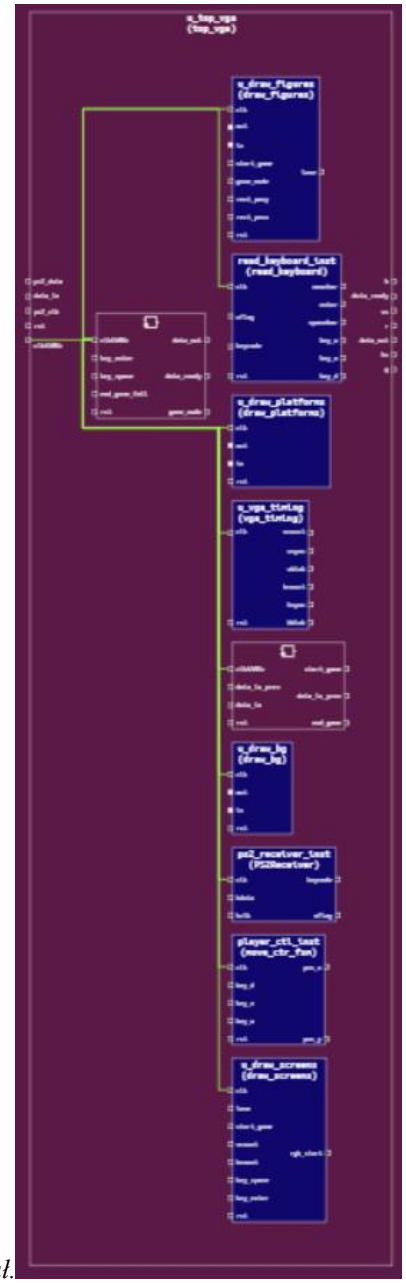
nazwa sygnału	opis
hcount [10:0]	poziomy licznik VGA – określenie pozycji x pixela
vcount [10:0]	pionowy licznik VGA – określenie pozycji x pixela
hsync	synchronizacja pozioma
vsync	synchronizacja pionowa
hblnk	*sygnał blankingu poziomego*
vblnk	*sygnał blankingu pionowego*
rgb	*sygnał koloru RGB*

4.2. Rozprowadzenie sygnału zegara

Osoba odpowiedzialna: JJ

Informacja na temat źródła sygnału zegarowego, używanych częstotliwości zegara w całym układzie. ***– 65MHz***

Moduł generatora zegara umieszczamy w module głównym projektu. W pozostałych modułach używamy tylko i wyłącznie sygnały zegara wygenerowane przez ten moduł.



5. Implementacja

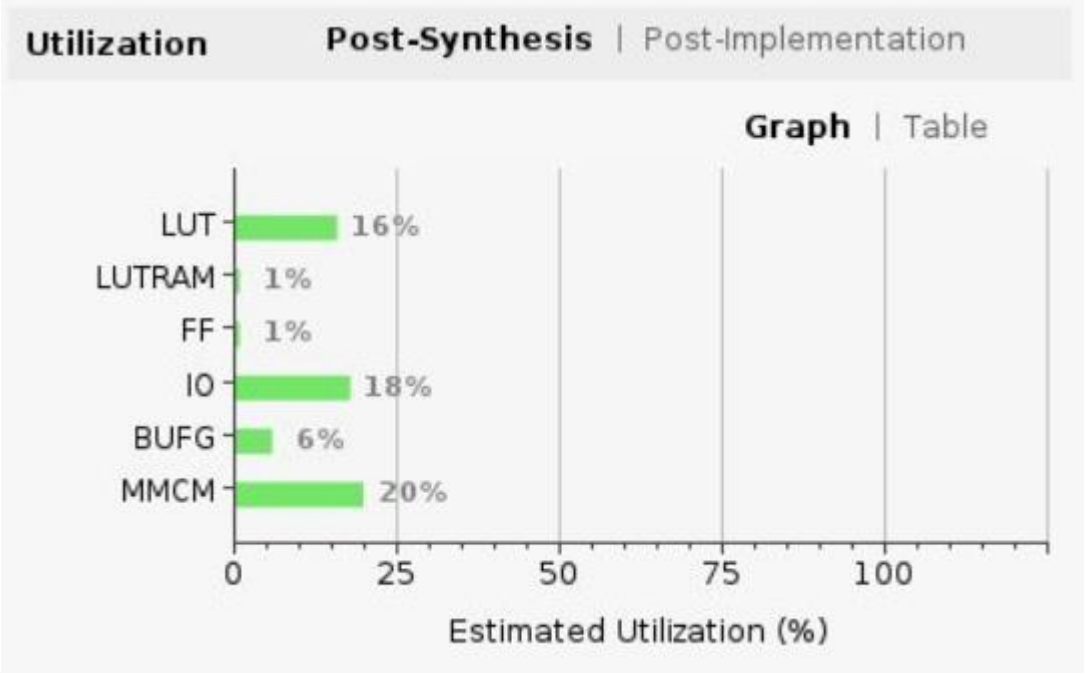
5.1. Lista zignorowanych ostrzeżeń Vivado.

Identyfikator ostrzeżenia	Liczba wystąpień	Uzasadnienie
[Synth 8-6014]	1	Syntezator usunął nieużywany przerzutnik
[Synth 8-3848]	1	Sygnał data_ jest zadeklarowany w module, ale nikt go nie przypisuje.
[Synth 8-7080]	1	Zbyt mały projekt by spełniał warunki równoległej syntezy.
[Synth 8-3332]	2	

		Automat nie ma użycia albo coś niepodłączone.

5.2. Wykorzystanie zasobów

Tabela z wykorzystaniem zasobów z Vivado



5.3. Marginesy czasowe

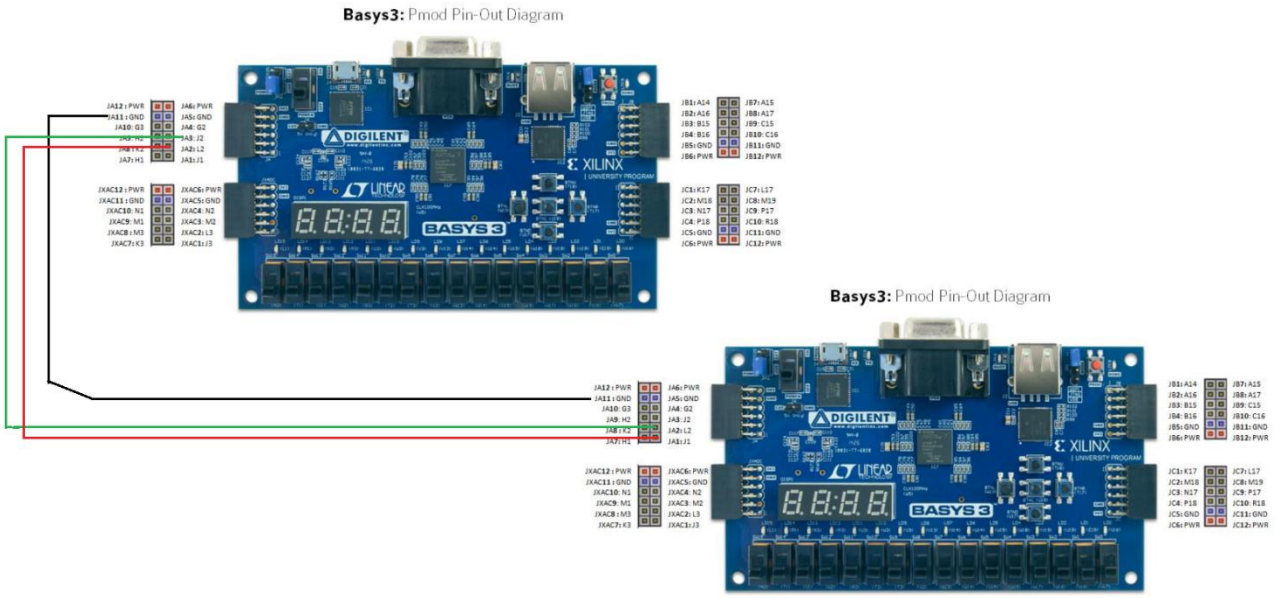
Marginesy czasowe (WNS) dla setup i hold.

Timing	Setup | Hold | Pulse Width
Worst Negative Slack (WNS):	1.979 ns
Total Negative Slack (TNS):	0 ns
Number of Failing Endpoints:	0
Total Number of Endpoints:	375
Implemented Timing Report	

Timing	Setup | Hold | Pulse Width
Worst Hold Slack (WHS):	0.101 ns
Total Hold Slack (THS):	0 ns
Number of Failing Endpoints:	0
Total Number of Endpoints:	375
Implemented Timing Report	

6. Konfiguracja sprzętu

Schemat połączenia ze sobą płytek Basys3 w trybie multiplayer.



Schematy podłączenia dodatkowych urządzeń peryferyjnych.

Konfiguracja zworek, przełączników, itp., jeśli inna niż domyślna.

7. Film.

Link do ściągnięcia filmu:

https://drive.google.com/file/d/1AElgnx3MNFLSNh5H4-gUzwj_YkHtyoTP/view?usp=sharing