Tytuł: Warblade

Autorzy: Natalia Pluta (NP), Hubert Mucha (HM)

Ostatnia modyfikacja: 13.09.2021

Spis treści

[1. Repozytorium git 1](#__RefHeading___Toc1311_2642343945)

[2. Wstęp 1](#__RefHeading___Toc375_832888478)

[3. Specyfikacja 1](#__RefHeading___Toc377_832888478)

[3.1. Opis ogólny algorytmu 1](#__RefHeading___Toc379_832888478)

[3.2. Tabela zdarzeń 2](#__RefHeading___Toc381_832888478)

[4. Architektura 2](#__RefHeading___Toc383_832888478)

[4.1. Moduł: top 2](#__RefHeading___Toc385_832888478)

[4.1.1. Schemat blokowy 2](#__RefHeading___Toc387_832888478)

[4.1.2. Porty 3](#__RefHeading___Toc389_832888478)

[a) mou – mouse\_ctl, input 3](#__RefHeading___Toc391_832888478)

[b) vga – vga\_ctl, output 3](#__RefHeading___Toc393_832888478)

[4.1.3. Interfejsy 3](#__RefHeading___Toc453_832888478)

[a) m2c – mouse\_ctl to core 3](#__RefHeading___Toc395_832888478)

[4.2. Rozprowadzenie sygnału zegara 3](#__RefHeading___Toc1150_832888478)

[5. Implementacja 4](#__RefHeading___Toc1313_2642343945)

[5.1. Lista zignorowanych ostrzeżeń Vivado. 4](#__RefHeading___Toc1315_2642343945)

[5.2. Wykorzystanie zasobów 4](#__RefHeading___Toc1317_2642343945)

[5.3. Marginesy czasowe 4](#__RefHeading___Toc1319_2642343945)

[6. Film. 4](#__RefHeading___Toc1249_1436539759)

# Repozytorium git

https://github.com/hubertmucha/Warblade

# Wstęp

*Skąd się wziął pomysł i co w ramach tego projektu robimy.*

*Gra Warblade bazuje na grze o tym samym tytule z 2003 roku wydanej przez* Edgara M. Wiecej informacji pod linkiem: <https://en.wikipedia.org/wiki/Warblade>.

# Specyfikacja

## 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 na pierwszym poziomie i w wersji obecnej 1.0 zaimplementowane są jeszcze drugi i trzeci. Celem graczy jest zestrzelenie statków przeciwnika jednocześnie samemu unikając zestrzelania przez wrogie pociski.*

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

## 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 dotyczący różnych stanów działania programu. Kategorie powinny odpowiadać stanom ze schematu z pkt. 2.1.*

|  |  |  |
| --- | --- | --- |
| **Zdarzenie** | **Kategoria** | **Reakcja systemu** |
| Przycisk „lewo” | Gra | Przemieszczenie statku gracza w lewo |
| Przycisk „prawo” | Gra | Przemieszczenie statku gracza w prawo |
| Przycisk „strzał” | Gra | Statek gracza wypuszcza pocisk |
| Trafienie pociskiem gracza statku przeciwnika | Gra | Przeciwnik znika i nie pojawia się dalej na danym poziomie |
| Trafienie pociskiem przeciwnika statku gracza | Gra | Gracz tarci jedno życie |
| Gracz traci ostanie życie | Gra | Gracz znika z pola gry, nie może strzelać. Drugi gracz nadal może kontynuować rozgrywkę. |
| Obaj gracze stracili wszystkie życia | Gra | Statku dwóch graczy nie są widocznie na ekranie.  Wyświetlany jest napis *game over.* Gra wymaga zresetowania. |
| Wszystkie statki przeciwnika zostały zestrzelone | Gra | Poziom gry zwiększa się o jeden, nowe statki przeciwnika pojawiają się, kolory tła zmieniają się. |

# 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.

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

## Moduł: top

Osoba odpowiedzialna: JK

### 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.

*Przykładowy 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).*

### Porty

#### mou – mouse\_ctl, input

|  |  |
| --- | --- |
| **nazwa portu** | **opis** |
| mou\_si | szeregowe wejście danych |
|  |  |

#### vga – vga\_ctl, output

|  |  |
| --- | --- |
| **nazwa portu** | **opis** |
| vga\_vs | sygnał synchronizacji pionowej VGA |
|  |  |

### Interfejsy

#### m2c – mouse\_ctl to core

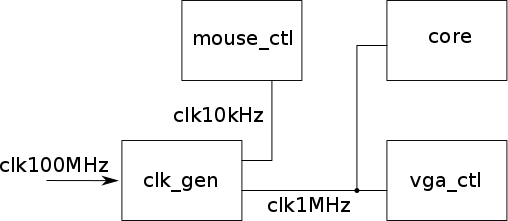
|  |  |
| --- | --- |
| **nazwa sygnału** | **opis** |
| m2c\_x[9:0] | horyzontalna pozycja kursora myszy na ekranie |
| m2c\_y[9:0] | wertykalna pozycja kursora myszy na ekranie |
|  |  |

## Rozprowadzenie sygnału zegara

Osoba odpowiedzialna: JK

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

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

*Uwaga: jeżeli używamy różnych częstotliwości zegara w układzie, to należy je tak dobrać, aby były wielokrotnościami siebie (umożliwia to wygenerowanie tych sygnałów z jednego IP core generatora zegara i zapobiega problemom z synchronizacją).*

# Implementacja

## Lista zignorowanych ostrzeżeń Vivado.

|  |  |  |
| --- | --- | --- |
| Identyfikator ostrzeżenia | Liczba wystąpień | Uzasadnienie |
|  |  |  |
|  |  |  |

## Wykorzystanie zasobów

Tabela z wykorzystaniem zasobów z Vivado

## Marginesy czasowe

Marginesy czasowe (WNS) dla setup i hold.

# Film.

Link do ściągnięcia filmu:

https://