

Dokumentacja projektu - Technika cyfrowa (lab)

Zespół 3

Semestr 4 (letni 2021)

1 Podstawowe informacje o projekcie

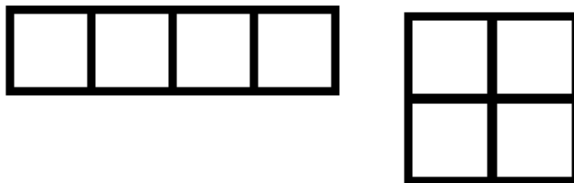
- Realizowany temat:
Realizacja gry "Tetris" w programie Logisim
- Pożądana ocena: 5
- Skład zespołu nr 3:
 - Filip Bochniak 144456
 - Maurycy Kujawski 144452
 - Daniel Różycki 144471
- Podział zadań:
 - Filip
 - * maszyna losująca, generacja poszczególnych klocków, input klawiatury, przesuwanie
 - Maurycy
 - * całe GPU, logika gry, łączenie wszystkich głównych podbloków
 - Daniel
 - * opadanie bloków, wykrywanie kolizji, zamrażanie bloków, wykrywanie podłogi

2 Funkcjonalności

2.1 Przewidywane

2.1.1 Bazowe

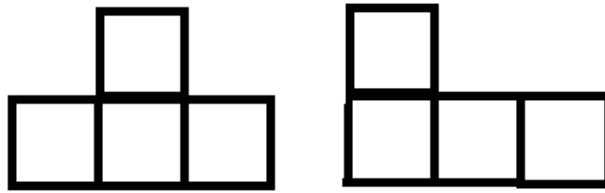
- Ideą projektu jest implementacja gry typu Tetris w środowisku Logisim na ekranie LED 16 na 8 diód. Gra polega na tym, że pojawiają się u góry ekranu klocki. Gracz kontroluje orientację klocka oraz jego pozycję na osi X ekranu. Zadaniem gracza będzie ustawienie klocków na ekranie o szerokości ośmiu klocków taki sposób, aby cały rząd był wypełniony klockami. Wypełniony rząd następnie znika, powodując opadnięcie klocków wyżej, o pozycję niżej. Planujemy zaimplementować w pierwszej kolejności następujące klocki.



Gracz będzie miał możliwość rotacji klocka. Do bazowej funkcjonalności planowana jest również przegrana, która nastąpi w momencie gdy na samej górze ekranu będzie kłoczek. Gra powinna wyświetlić na ekranie ilość rzędów które zostały wyeliminowane przez gracza, które będą wynikiem jaki osiągnął.

2.1.2 Dodatkowe

- Gdy po zaimplementowaniu funkcjonalności bazowej zostanie trochę czasu do zdania projektu planowana jest implementacja dodatkowych klocków.



Dodatkową funkcją będzie też stopniowe zwiększanie się prędkości opadania klocków co 10 klocków.

2.2 Uzgadnianie

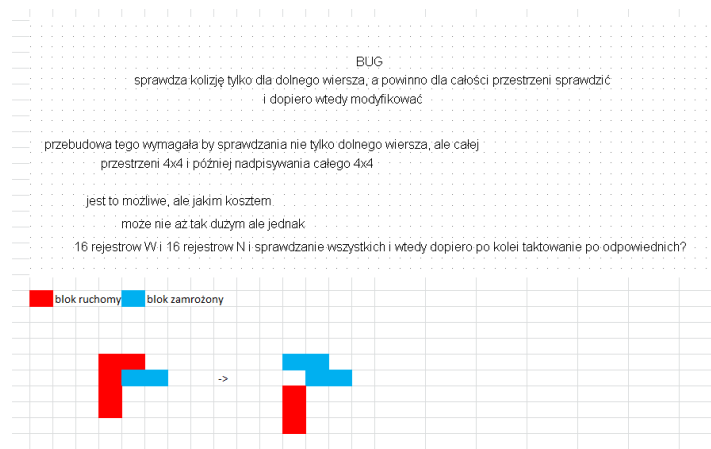
2.3 Zrealizowane

- GPU - czyli transfer pamięci logicznej na ekran
- Opadanie - ale sprawdza kolizję tylko wiersz po wierszu, więc jak zderzy się gdzieś od boku (niżej wyjaśnione) to się rozpadnie, więc nie zadziała dla kształtów z dodatkowej części
- Losowanie i generacja - losowanie kolejnego klocka, który ma się pojawić na ekranie, wraz z generacją

3 Problemy

3.1 Bieżące

- Daniel
opadanie.circ: robiąc opadanie strasznie się męczyłem i zajęło to z 2-3 całe noce i w końcu się udało żeby porównywało cały wiersz na raz i na tej postawie nadpisywało, jeśli wykryje kolizję to wchodziło w tryb kolizji i zamiast opadać zamrażało blok ruchomy, niestety problem polega na tym, że sprawdza wiersz po wierszu i gdyby gdzieś od boku się pojawiło nagle pełny blok, to integralność bloczku by się rozpadła, wiem jak to zmienić, ale bym musiał zamiast 8 rejestrów i wiersz po wierszów bym musiał mieć chyba z 32 rejestrów może i je wszystkie najpierw porównać na check-zderzenie i wtedy dopiero iterować wiersz po wierszu? oczywiście da się to wydajniej zrobić, ale nieopłacalne to dla mnie od nowa budować wszystko



Rysunek 1: wiadomy bug

Drugi problem, zrobić sprawdzanie ścianek i podłogi i zależnie od tego różne sposoby latania po adresach i różne sposoby zapisywania do pamięci (4 - normal, lewo,prawo,podłoga, lewo podłoga, prawo podłoga)

update: 02.06.2021 nowy blok jest technicznie zrobiony, ale wciąż są błędy w środku trudne do wykrycia, więc prace trwają (nowy blok ma wykrywanie podłogi oraz sprawdza cały obszar 4x5 lub przy podłodze 4x4,4x3,4x2,4x1 i dopiero wtedy decyduje jak nadpisać, więc unikamy poprzedniego problemu)

- pewien pan na M do dokumentacji to nie zaglądał chyba ani razu, więc ciężko powiedzieć co robi, jakie ma problemy itp

3.2 Rozwiązane

- Maurycy: początkowo był problem z zegarem, gdy nieoptymalnie używaliśmy jego sygnału i zmiana wartości pixela w pamięci wymagała aż 4 taktów,
Rozwiązanie zmiana paru rzeczy żeby lepiej działało
- Daniel: Kompletnie zła logika gry, zasugerowałem się GPU i chciałem żeby logika się ustawiała w kolejności iterującej (ponadto pixel po pixelu, a nie jako obiekt), co prowadziło do tak wielu złych i nieoptymalnych rozwiązań że to się w głowie nie mieści
Rozwiązanie: zły pomysł do kosza, jednocześnie lepiej zawiązaliśmy współpracę nad pomysłem realizacji logiki podobnym do mojego nowego, ale Maurycego jednak
- Filip: Problemy z taktowaniem przy losowaniu, sygnał informujący o indeksie kolejnego klocka propagował się w tym samym momencie co ostatni takt generacji poprzedniego, a nie po nim, co powodowało rozjechanie się generacji w różnych momentach. Błąd nie występował w momencie generacji tego samego typu klocków pod rząd. Wystarczyło zmodyfikować licznik w maszynie losującej żeby reagował na zbocze opadające.
- Filip: Rozwiązano problem ograniczonej losowości poprzez odpowiednie rozmieszczenie wartości losowanych wchodzących na multiplexer. Była próba naprawienia tego poprzez rozpoczęcie losowania od 3-ciej próbki, jednak to sprawiało dodatkowe problemy, ponieważ losowanie odbywa się na tym samym zegarze co cały układ, co powodowało błędy w dalszych blokach wykorzystujących losowanie, ze względu na to, że na jego wyjściu RAND-a znajdował się stan błędny.

4 Opis każdej funkcji (bloku)

- obniżanie (Daniel) nowa wersja w trakcie pracy (debugowanie)
- RAND (Filip) - działając na zasadzie pociągu Turinga powoduje cykliczną zmianę słowa 5-bitowego, co daje na wyjściu liczbę z zakresu 0-31. Sam proces jest tak naprawdę pseudolosowy, ponieważ wylosowany numer powtórzy się co 32 przejścia zegara. Wyjście losowania podłączone jest do multiplexera, który do wartości 0-31 ma przypisane wartości 0-3, odpowiadające losowaniu odpowiednich bloków.
- RAND_BLC i bloki generacji poszczególnych klocków (Filip) - Bazując na blokach renderujących, które wysyłają sygnał potrzebny do generacji z pamięci, najpierw losuje numer generowanego klocka (0-3), a potem przez kolejne 16 sygnałów zegara wysyła informacje potrzebne do wyrenderowania wybranego klocka w dalszej części całego układu.